EC-DARTS: Inducing Equalized and Consistent Optimization into DARTS

Qinqin Zhou[1], Xiawu Zheng[1], Liujuan Cao[1]*, Bineng Zhong[2], Teng Xi[3],
Gang Zhang[3], Errui Ding[3], Mingliang Xu[4], Rongrong Ji[1,5]
[1]MAC Lab, School of Informatics, Xiamen University,
[2]Guangxi Key Lab of Multi-Source Information Mining & Security, Guangxi Normal University,
[3]Department of Computer Vision Technology (VIS), Baidu Inc.,
[4]Zhengzhou University, [5]Institute of Artificial Intelligence, Xiamen University

Abstract

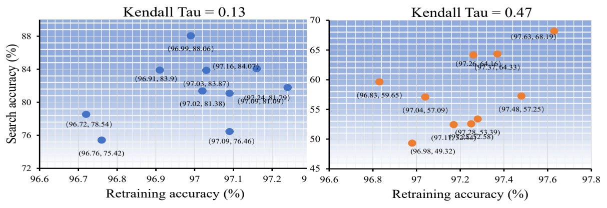
Based on the relaxed search space, differential architecture search (DARTS) is efficient in searching for a high-performance architecture. However, the unbalanced competition among operations that have different trainable parameters causes the model collapse. Besides, the inconsistent structures in the search and retraining stages causes cross-stage evaluation to be unstable. In this paper, we call these issues as an operation gap and a structure gap in DARTS. To shrink these gaps, we propose to induce equalized and consistent optimization in differentiable architecture search (EC-DARTS). EC-DARTS decouples different operations based on their categories to optimize the operation weights so that the operation gap between them is shrinked. Besides, we introduce an induced structural transition to bridge the structure gap between the model structures in the search and retraining stages. Extensive experiments on CIFAR10 and ImageNet demonstrate the effectiveness of our method. Specifically, on CIFAR10, we achieve a test error of 2.39%, while only 0.3 GPU days on NVIDIA TITAN V. On ImageNet, our method achieves a top-1 error of 23.6% under the mobile setting.

1. Introduction

The computer science community has witnessed the remarkable achievements of Deep Neural Networks (DNN), especially in the field of computer vision. However, the common DNNs are designed by human experts, which require a lot of computation resources and domain-specific knowledges. Recently, Neural Architecture Search (NAS) has emerged to search neural architectures in an automated way, which greatly eases the dependence on human experts and achieves a remarkable performance.

Among various NAS methods, Differentiable Neural Architecture Search (DNAS) [11, 37], *e.g.*, differentiable architecture search (DARTS) [26], has attracted a lot of attention because it improves the efficiency of searching a neural network by orders of magnitudes. Motivated by DARTS, there are many works [5, 43, 7, 6, 23, 44, 1, 46, 21, 2, 42, 10, 45, 36] followed the similar scheme in DARTS, which have achieved considerable performance gains. Despite these achievements, it remains a challenging problem to optimize the search process of DARTS due to the optimization gaps. The first gap is the operation gap caused by the different numbers of trainable parameters contained in different operations. The second gap is the structure gap caused by the inconsistent model structures adopted in the search and retraining stages. To explain the operation gap, different operations contain different numbers of trainable parameters. In this case, the costs of optimizing operations that have less trainable parameters will be smaller, which means that DARTS is biased towards the optimization of the parameter-free operations (*e.g.*, skip-connects and pooling layers). Therefore, the searched architecture might be dominated by parameter-free operations, which leads to a poor performance. To explain the structure gap, the operations that are mixed in the one-shot model during the search stage will

*Corresponding author: caoliujuan@xmu.edu.cn



(a) DARTS (2nd Order) (b) EC-DARTS(ours)

Figure 1. The correlation evaluation between the search rank and retraining rank of a single search. We summarize the results of 10 randomly selected architectures from a single run of DARTS and EC-DARTS in different colors.

be partially pruned in the retraining stage. The inconsistent structures in the search and retraining stages lead to the structure gap. These optimization gaps have been pointed out by several works [39, 43, 7, 6, 23, 1, 46, 21, 11, 37]. To avoid a dramatic performance degradation caused by these optimization gaps, several solutions have been put forward [43, 7, 6, 23]. DARTS+ [23] directly constrained the skip-connects to two per cell. Cyclic DARTS [43] introduced an evaluation network with 20 cells in the search stage. These solutions have alleviated the optimization gaps, while requiring strong prior information or additional computations. PR-DARTS [46] alleviates unfair competition only between skip connection and other categories of operations. Our CEN focuses on unfair competition among the whole operations and includes PR-DARTS as a special case.

In this paper, we propose to induce equalized and consistent optimization in differentiable architecture search (EC-DARTS). To shrink the operation gap, we devise a Cross-Edge Normalization (CEN) to equalize the dominance of each operation. Therefore, the operation weights can better reflect the importance of each operation. Specifically, CEN normalizes the weights of operations by categories, *e.g.*, normalize the weights of 3×3 separable convolutions from different edges. Besides, CEN eliminates the unbalanced competition between operations of different categories, while introducing a balanced competition between operations of the same category. To shrink the structure gap, a Induced Structural Transition (IST) is proposed to construct an auxiliary model to induce the model structure type in the search stage to transform into a similar one that is used in the retraining stage. In order to quantify the degree to which the optimization gaps are alleviated by our method, we adopt the Kendall Tau metric [17] to measure the correlations between different ranks. Please refer to supplementary materials for the 3 modes of the the Kendall Tau metric used in this paper. As shown in Figure 1, we conduct a correlation evaluation among 10 architectures randomly selected from a single search. Compared with DARTS, our method achieves a stronger correlation between the search rank and retraining rank during the search process.

Our contributions are summarized as follows.

- We propose to induce equalized and consistent optimization in differentiable architecture search (EC-DARTS) from the levels of operation and structure.
- We design a Cross-Edge Normalization (CEN) for NAS. By normalizing the operation weights in a balanced competition conditions, our methods exhibits a stronger correlation between the operation weight and model performance. Equipped with CEN, the operation gap is alleviated effectively.
- In order to improve the consistency between the search and retraining stages, we introduce the structure information of the retraining stage into the search stage by an Induced Structural Transition (IST).

We further conduct comprehensive experiments over four datasets to verify the effectiveness of our method. Specifically, our method achieves state-of-the-art performance on CIFAR10, CIFAR 100, Tiny-ImageNet-200, and ImageNet. Specifically, a test error of 2.39% on CIFAR10, and a top-1 error of 23.6% on ImageNet with a model size of 4.7M are reported by our method.

2. Related work

Based on the impressive progresses achieved by the great number of manually designed neural networks [34, 35, 13, 16, 15], an emerging search field named Neural Architecture Search (NAS) has been proposed to improve the efficiency of model design. According to the type of search strategy, the current NAS methods can be simply divided into: reinforcement-learning-based methods [24, 12, 48, 47], evolution-based methods [33, 40, 31, 25], and gradient-based methods [26, 5, 29, 44, 42, 3, 41, 27].

Reinforcement-learning-based NAS. For reinforcement-learning-based NAS, the reinforcement learning is used to train a controller that indicates the performances of neural networks. However, it has a high requirement for computation resources in training the intermediate architectures from scratch. For example, NASNet [47] needs 2,000 GPU-hours to search a state-of-the-art architecture.

Evolution-based NAS. Evolution-based NAS follows a different path to search architectures, which learns a probabilistic model to sample architectures and uses genetic operations to generate offspring in the search stage. In this way, a globally optimal architecture might be obtained. Similar to reinforcement-learning-based NAS, the sampled architectures in evolution-based NAS [40, 31] also need large amount of computation resources to train from scratch. Besides, the search spaces of these two types of NAS are non-differentiable, which are not efficient in optimization.

Gradient-based NAS. Gradient-based NAS relaxes the search space to a differentiable form to use gradient descent in optimizing the search process. DARTS is one of the typical gradient-based NAS methods, which adopts continuous weights for operations based on the relaxed search space. After that, the model weights and operation weights are iteratively optimized in a bi-level manner. As a result, the computation and time consumption in searching architectures are reduced by several orders of magnitude with DARTS. Despite the efficiencies achieved by DARTS, the downsides of DARTS have been revealed in recent years [39, 32]. [32] pointed out the gap between the model depths in the search and retraining stages, which leads to a sub-optimal performance in the retraining stage. In [39], Xie *et al.* argued that the major challenge faced by DARTS comes from the optimization gaps during the search and retraining stages. It remains an open issue to tackle the downsides of DARTS.

In this paper, we focus on two main optimization gaps: the operation gap and the structure gap in DARTS. To handle these gaps, we propose a new differential NAS framework based on DARTS namely EC-DARTS. In contrast with DARTS, EC-DARTS normalizes the operation weights under a equalized optimization manner and induces the model structure type in the search stage to transform into that in the retraining stage.

3. Preliminaries

To better comprehend our method, we first briefly review the search formulation of DARTS, and elaborate on the optimization gap problem of DARTS.

Given a search space $\mathcal{O}$, which includes M various candidate operations. Previous works [48] proposed to search for normal cell and reduction cell instead of directly searching the entire architecture. The cell is represented as a directed acyclic graph (DAG) of N sequential nodes. The nodes denote representations (*e.g.*, feature map in convolution neural networks), and the edge from node i to node j is associated with all operations in the search space, which transforms the node x_i to x_j. There are two inputs and one output node in a cell. We denote the architecture with a mixture of different operations in each edge as the one-shot model. Specifically, in the one-shot model, DARTS relaxes the search of each edge with a softmax over operations in the search space, the edge in the cell is formulated as

$$\bar{o}^{(i,j)}(x_i) = \sum_{o\in\mathcal{O}} \frac{\exp\left(\alpha_o^{(i,j)}\right)}{\sum_{o'\in\mathcal{O}}\exp\left(\alpha_{o'}^{(i,j)}\right)} o(x_i), \tag{1}$$

where $\alpha_o^{(i,j)}$ denotes the weight of operation o at the edge from node i to j. We denote $\alpha = \left\{\alpha_{o_1}^{(i,j)}, \alpha_{o_2}^{(i,j)}, \cdots, \alpha_{o_M}^{(i,j)}\right\}$ as the operation weight vector from node i to j after the softmax process. In other words, α is the probability of selecting the corresponding candidate operations to generate architectures. All inputs from incoming edges of the intermediate node j are gathered as

$$x_j = \sum_{i<j} \bar{o}_{i,j}\left(x_i\right), \tag{2}$$

After the relaxation, the search problem is turned to learn the optimal operation weight α and architecture weights ω by gradient descent. DARTS treats the search problem as a bi-level optimization problem as follows:

$$\begin{aligned} &\min{}_{\alpha}\mathcal{L}_{val}\left(\omega^*(\alpha), \alpha\right), \\ \text{s.t.}\quad &\omega^*(\alpha) = \arg\min{}_{\omega}\mathcal{L}_{train}(\omega, \alpha), \end{aligned} \tag{3}$$

where $\mathcal{L}_{val}$ and $\mathcal{L}_{train}$ denote the validation and training losses, respectively.

At the end of the search stage, the searched cells are derived by preserving the two top operations among all non-zero candidate operations gathered from all connected nodes. In the retraining stage, DARTS stacks a certain number of cells to construct the final architecture.

The search paradigm of DARTS has exposed some limitations [39, 32]. As we mentioned in Section 1, the optimization gap problem has degraded the performance of DARTS. As shown in Figure 1 (a), the Kendall Tau value calculated between the 10 architectures selected from a single search of DARTS is 0.13, which means the search rank and retraining rank during the search process in DARTS are relatively independent.

4. Methodology

In this section, we introduce how to alleviate the aforementioned gaps by the proposed Cross-Edge Normalize and Induced Structural Transition in Section 4.1 and Section4.2, respectively.

4.1. Cross-Edge Normalization

As we discussed in Section 3, DARTS relaxes the operation weights with a continuous vector α named operation weights. The candidate operations that have different trainable parameters influence each other. In this case, there is an inherent operation gap that makes the parameter-free operations dominate in the searched architectures. Specifically, as illustrated in Equation 1, DARTS proposed to normalize the weights of different operations in the same edge, which leads to the operation gap.

In this paper, we introduce a simple yet effective strategy named Cross-Edge Normalization (CEN) to alleviate this issue. Based on the incoming edges of each node in the one-shot model, we group the operations in a cross-edge manner to decouple different operations in the search stage. As shown in Figure 2, CEN normalizes the weight cross-edge *according to the categories (a total of 8 categories) of the operations*. Formally, for a specific operation weight $\alpha_{o_k}^{(i,j)}$, we propose to normalize the weight by

$$\alpha_{o_k}^{(i,j)} = \frac{\hat{\alpha}_{o_k}^{(i,j)}}{\sum_{i'<j}\hat{\alpha}_{o_k}^{(i',j)}}, \tag{4}$$

where $\hat{\alpha}_{o_k}^{(i',j)}$ denotes the unnormalized operation weight of operation o_k in edge (i', j), $i' < j$ denotes the predecessors connected with node j. Then, given the normalized weight $\alpha_{o_k}^{(i,j)}$, the intermediate node x_j is obtained by

$$x_j = \sum_{i<j}\sum_{o_k} \alpha_{o_k}^{(i,j)} o_k(x_i), \tag{5}$$

where k denotes the k-th operation in the search space.

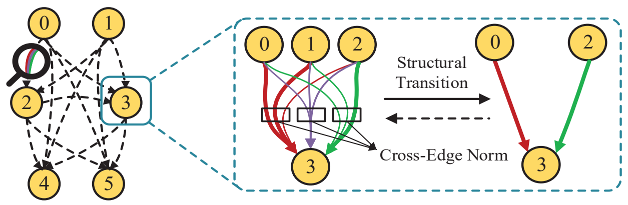


Figure 2. Illustration of EC-DARTS. Our EC-DARTS decouples different operations by employing softmax according to the categories of operations. Then, an auxiliary model is constructed to eliminate the structure gap.

As shown in Equation 4, $\alpha_{o_k}^{(i,j)}$ is computed among operations of the same category.The only difference between these operations in CEN is the incoming nodes connected by them. Therefore, CEN enables the search to focus on optimizing the importance of each operation without the impact of unbalanced competition.

There are several works [23, 6] that restricted or modified certain candidate operations to achieve a fair comparison among different operations. Intuitively, even with prior information, the degree of imbalance between these operations in the search stage is difficult to manually estimate. Comparing with these methods, there are two main benefits brings by our CEN. On the one hand, CEN does not involve additional calculations, and it is easy to be implemented in gradient-based NAS. On the other hand, CEN alleviates unbalanced optimization among different operations, while introducing equalized optimization among the same category operation.

4.2. Induced Structural Transition

It is impractical to completely train different sub-architectures in the search space due to limited computation resources. Therefore, DARTS proposed to share the model weights of the one-shot model among every sub-architecture. However, there is no theory to support weight-sharing yet. From the perspective of the structures, the one-shot model compounds all sub-architectures in the search space, which makes the structure of the one-shot model is very different from the final optimal architecture. For example, the 4-th node in a normal cell has (5×7) connections (includes 2 from the 2 input nodes) and each edge includes 7 categories of operations except zero operation.) in the search stage and only 2 connections in the retraining stage, which leads to another optimization gap for DARTS. We denote this optimization gap as the structure gap. To alleviate the structure gap, we design Induced Structural Transition (IST), which adds an auxiliary model in the search stage. The auxiliary model retains only two top operations for each node (*i.e.*, in the retraining stage, the one-shot model is pruned with two top operations for each node to generate the final architecture) and inherits the weights of the one-shot model, which is used to bridge the structure gap between search and retraining stages. Specifically, the structure of the auxiliary model in IST is more consistent with the structure in the retraining stage. Therefore, the verification result of the auxiliary model is more accurate compared to the original one-shot model in DARTS. And the auxiliary model dynamically gives a backward propagation to update the weights of the one-shot model. In this way, without coupling all candidate connections in the search space, the auxiliary model shares a similar model structure as in the retraining stage.

Algorithm 1: EC-DARTS

```
Input: The search space $\mathcal{O}$, the training and
       validation datasets, operation weights $\alpha$ and
       model weights $\omega$, and search epochs $S$;
Output: The searched optimal architecture;
1  Randomly initialize $\alpha$ and $\omega$;
2  for search step $i$ in $i \in [0, S]$ do
3      Calculate $\mathcal{L}_{train}(\omega, \alpha)$ and update $\omega^*$;
4      Prune $\alpha$ to $\alpha^*$ by retaining the top-k operations;
5      Construct auxiliary model by $\alpha^*$ with the
         weights of one-shot model;
6      Calculate $\mathcal{L}_{train}(\omega^*, \alpha^*)$ and update $\omega^*_{aux}$;
7      Compute Acc = Eval($\omega^*_{aux}, \alpha^*$);
8      if Acc surpasses the best seen value then
9          Update $\omega$ with $\omega^* + \omega^*_{aux}$;
10     end
11     Update $\alpha$ by descending $\mathcal{L}_{val}\left(\omega^*_{aux}(\alpha^*), \alpha\right)$;
12 end
```

Formally, by applying the IST, the optimization problem in Equation 3 is reformulated as

$$\begin{aligned} &\min_{\alpha} \mathcal{L}_{val}\left(\langle \omega^*, \omega^*_{aux} \rangle, \alpha\right), \\ \text{s.t.} \quad &\omega^* = \arg\min{}_{\omega} \mathcal{L}_{train}(\omega, \alpha), \\ &\omega^*_{aux}(\alpha^*) = \arg\min{}_{\omega_{aux}(\alpha)} \mathcal{L}_{train}(\omega^*, \alpha^*), \end{aligned} \tag{6}$$

where α^* indicates the operation weights after pruning α. ω^*_{aux} is the model weights of the auxiliary model, which is part of ω^*. The one-shot model weights might be updated based on the backward propagation from the auxiliary model. In other words, the training of the one-shot model and auxiliary model dynamically optimize each other. We depict the dynamic procedure in Algorithm 1. Specifically, the discretized sub-architecture is sampled from the one-shot model according to the operation weights α. Then, an auxiliary model is constructed with the sub-architecture where the model weights are inherited from the one-shot model. We train the auxiliary model by one epoch and validate it on the validation set. Finally, the one-shot model might be updated by the auxiliary model and trained in the next iteration to sample a new auxiliary model. These processes are iterated until the search stage is completed.

As shown in Figure 2, the one-shot model includes all sub-architectures by embedding different operations with corresponding operation weights α in each edge. Our IST makes use of the operation weights α and the model weights of the one-shot model to construct an auxiliary model. Extensive experiments in Section 5.6 provide evidence that IST induces the structural transition in the search stage, thus shrink the structure gap and improve the final performance.

5. Experiments

5.1. Datasets and implementation details

We conduct experiments on CIFAR10/100 [19], Tiny-ImageNet-200[1], and ImageNet [8] to verify the effectiveness of our method. **CIFAR** is a basic image classification benchmark that includes CIFAR10 and CIFAR100. CIFAR10 contains 10 classes, and each class has 6,000 images. CIFAR100 contains 100 classes, and each class has 600 images. The training and testing sets in CIFAR10/100 consist of 50,000 and 10,000 images, respectively. **Tiny-ImageNet-200** includes 200 classes, and each class has 500 and 50 images in the training set and testing set, respectively. **ImageNet** is a large-scale image classification benchmark, which contains 1,000 classes. There are 1.2 million training images and 50,000 validation images in this benchmark.

5.2. Implementation details

We adopt the search space used in most NAS methods [26, 3, 5, 42, 7], which contains 3×3 and 5×5 separable convolution, 3×3 and 5×5 dilated separable convolution, skip-connect (*i.e.*, identity), 3×3 max-pooling, 3×3 average-pooling, and zero (*i.e.*, none). It should be noted that EC-DARTS is orthogonal to the search space, thus can be combined with different search spaces. Following the settings in [26], EC-DARTS includes a search stage to search basic cell structures, and a retraining stage to train the architecture that consists of a certain number cells from scratch. In the search stage, the training data is randomly separated into two parts equally, where the model weights (*i.e.*, ω^*, ω^*_{aux}) and operation weights (*i.e.*, α) are optimized on these two parts, respectively. In the retraining stage, we follow the procedure in conventional image classification [13, 20] to train the target architecture. In our experiments, the image sizes are unified to 32×32, 64×64, and 224×224 on CIFAR10/100, Tiny-ImageNet-200, and ImageNet, respectively. All experiments are completed on NVIDIA TITAN V. Code will be released in PaddleSlim[2].

[1]http://tiny-imagenet.herokuapp.com/

[2]https://github.com/PaddlePaddle/PaddleSlim

5.3. Results on CIFAR

Following the search settings in [26], the one-shot model contains 6 normal cells and 2 reduction cells. We use SGD with an initial learning rate of 0.025, a momentum of 0.9, weight decay 3×10^{-4} to optimize the model weights of the one-shot model and the auxiliary model. The operation weights are optimized by the Adam optimizer [18] with an initial learning rate of 3×10^{-4}, weight decay 10^{-3} for 25 epochs with a batch size of 64. We use less training epochs in our search stage because there are two update steps in the main loop of our algorithm, and our method tends to convergence in about 25 epoch. The two update steps are for the one-shot model and auxiliary model. Due to the less computation of auxiliary model, our method is more efficient.

An architecture with 20 cells (18 normal cells and 2 reduction cells) is retrained from scratch on CIFAR10/100. The architecture is optimized by the SGD optimizer with an initial learning rate of 0.025, a momentum of 0.9, weight decay 3×10^{-4}, and gradient clipping at 5 for 600 epochs. To prevent overfitting, the retraining stage is regularized by cutout [9] and drop path with a rate of 0.2.

We compare EC-DARTS with other state-of-the-art methods[16, 25, 31, 30, 48, 24, 26, 41, 3, 5, 42, 10, 7, 4] on CIFAR10 and CIFAR100. The results of EC-DARTS on CIFAR10 and CIFAR100 are collected from 10 and 3 individual runs, respectively. As shown in Table 1, EC-DARTS achieves test errors of 2.39%/16.13% on CIFAR10/100, and when transferring the architecture from CIFAR10 to CIFAR100, EC-DARTS achieves a test error of 16.13%. These results are competitive among the compared methods. EC-DARTS surpasses the manually designed DenseNet-BC by a large margin, and the architecture searched by EC-DARTS is more lightweight. EC-DARTS also outperforms DARTS in the comparison of the test error and search cost. Note that the search space adopted in ProxylessNAS [3] is different from EC-DARTS. Although the test error achieved by ProxylessNAS is smaller than that of EC-DARTS, the model size and search cost of ProxylessNAS are much larger. For a comprehensive comparison, we also list the transferred results of architectures searched on CIFAR10/100 by EC-DARTS in Table 1. Note that the transferred results deteriorate a bit. A similar phenomenon is reported by [3]. Intuitively, it demonstrates that NAS is affected by the dataset bias.

5.4. Results on Tiny-ImageNet-200

We adopt similar search and retraining settings as on CIFAR. The only difference is that the stride of the first 3×3 convolution layer in the architecture is set to 2 to obtain 32×32 features, which constrains the model size at the same level as that in CIFAR.

To compare with other state-of-the-art NAS methods [26, 23, 42] in a fair way, we also present the results of

Table 1. Quantitative comparison results with state-of-the-art NAS methods on CIFAR10 and CIFAR100.

Architectures	Test Err. (%)		Params	Search Cost	Type
	CIFAR 10	CIFAR 100	(M)	(GPU-days)	
DenseNet-BC [16]	3.46	17.18	25.6	-	manual
Hireachical Evolution [25]	3.75±0.12	-	15.7	300	evolution
AmoebaNet-B [31]	2.55±0.05	-	2.8	3150	evolution
ENAS [30]	2.89	-	4.6	0.5	RL
NASNet-A [48]	2.65	-	3.3	1800	RL
PNAS [24]	3.41±0.09	-	3.2	225	SMBO
DARTS (1st order) [26]	3.00±0.14	-	3.3	0.4	gradient-based
DARTS (2nd order) [26]	2.76±0.09	-	3.3	1	gradient-based
SNAS (moderate) [41]	2.85±0.02	-	2.8	1.5	gradient-based
ProxylessNAS [3]	2.08	-	5.7	4.0	gradient-based
P-DARTS (CIFAR10) [5]	2.50	16.55	3.4	0.3	gradient-based
P-DARTS (CIFAR100) [5]	2.62	15.92	3.6	0.3	gradient-based
PC-DARTS [42]	2.57±0.07	-	3.6	0.1	gradient-based
GDAS [10]	2.82	18.13	2.5	0.17	gradient-based
FairDARTS [7]	2.54	-	2.8	0.4	gradient-based
SDARTS-ADV [4]	2.61±0.02	-	3.3	1.3	gradient-based
Ours (CIFAR10)	2.39±0.08	16.13±0.12	3.2	0.3	gradient-based
Ours (CIFAR100)	2.44±0.10	15.72±0.11	3.5	0.3	gradient-based

transferring the architectures searched on CIFAR10 to Tiny-ImageNet-200. As demonstrated in Table 2, among the architectures searched on Tiny-ImageNet-200, EC-DARTS achieves a test error of 28.1% with a smaller model size. Note that the model size of the architecture searched by EC-DARTS on Tiny-ImageNet-200 is larger than that on CIFAR10. One possible reason is that the larger dataset needs more trainable parameters to learn the diversity of samples. When transferring the searched architecture from CIFAR10 to Tiny-ImageNet-200, EC-DARTS shows the state-of-the-art performance at a test error of 28.7%.

5.5. Results on ImageNet

We follow the ImageNet search settings in PC-DARTS[42], which are different from the settings on CIFAR. The training and validation sets in the search stage are randomly sampled 10% and 2.5% from the training set of ImageNet, respectively. The first three layers of the one-shot model are 3×3 convolution layers with a stride of 2, which reduce the input size from 224×224 to 28×28. The one-shot model in the search stage is stacked with 8 cells (6 normal cells and 2 reduction cells). The model weights are optimized by the SGD with an initial learning rate of 0.5, a momentum of 0.9, and a weight decay of 3×10^{-4}. We adopt an Adam optimizer with an initial learning rate of 6×10^{-3} and a weight decay of 10^{-3} to optimize the operation weights. We follow the mobile setting to restrict the total number of multiply-add operations in the model to be less than 600M. A total of 30 epoch with 1024 batch size is adopted to ensure the convergence of the search stage.

Table 2. Quantitative comparison results with state-of-the-art NAS methods on Tiny-ImageNet-200, * denotes the architecture is searched on Tiny-ImageNet-200

Architectures	Test Err.(%)	Params (M)	Type
ResNet-34 [13]	42.5	20.4	manual
DenseNet-BC [16]	37.1	-	manual
DARTS [26]	30.4	3.8	gradient-based
PC-DARTS [42]	28.9	4.2	gradient-based
DARTS+ [23]	29.1	4.2	gradient-based
DARTS+* [23]	28.3	3.8	gradient-based
EC-DARTS	28.7	3.3	gradient-based
EC-DARTS*	28.1	3.7	gradient-based

In the retraining stage, the architecture is stacked with 14 cells. The cells at 1/3 and 2/3 of the architecture are reduction cells, and the others are normal cells. The other settings follow that of DARTS except that the first three layers of the stacked architecture are 3×3 convolution layers with a stride of 2. The architecture is optimized by the SGD optimizer with an initial learning rate of 0.5, and a weight decay of 3×10^{-6}. The initial channel number is set to 48, and the architecture is trained from scratch for 250 epochs.

We summarize the results and comparison to recent state-of-the-art methods [35, 14, 28, 48, 31, 24, 22, 26, 41, 5, 10, 38, 7, 42, 3] in Table 3. As demonstrated in

Table 3. Quantitative comparison results with state-of-the-art NAS methods on ImageNet.

Architectures	Test Err. (%)		Params	Multi-Add	Search Cost	Type
	top-1	top-5	(M)	(M)	(GPU-days)	
Inception-v1 [35]	30.2	10.1	6.6	1448	-	manual
MobileNet [14]	29.4	10.5	4.2	569	-	manual
ShuffleNet 2$\times$ (v2) [28]	26.4	10.2	5	524	-	manual
NASNet-A [48]	26.0	8.4	5.3	564	1800	RL
AmoebaNet-C [31]	24.3	7.6	6.4	570	3150	evolution
PNAS [24]	25.8	8.1	5.1	588	225	SMBO
PC-NAS-S [22]	23.9	8.5	5.1	-	-	heuristic
DARTS [26] (2nd order, CIFAR10)	26.7	8.7	4.7	574	4.0	gradient-based
SNAS (mild) [41]	27.3	9.2	4.3	522	1.5	gradient-based
P-DARTS (CIFAR10) [5]	24.4	7.4	4.9	557	0.3	gradient-based
GDAS [10]	25.1	-	5.5	375	9.0	gradient-based
FBNet-C [38]	26.0	8.5	5.3	581	0.2	gradient-based
FairDARTS (CIFAR10) [7]	24.9	7.5	4.8	541	0.4	gradient-based
FairDARTS (ImageNet) [7]	24.4	7.4	4.3	440	3	gradient-based
PC-DARTS (CIFAR10) [42]	25.1	7.8	5.3	586	0.1	gradient-based
PC-DARTS (ImageNet) [42]	24.2	7.3	5.3	597	3.8	gradient-based
ProxylessNAS (ImageNet) [3]	24.9	7.5	7.1	465	8.3	gradient-based
Ours(CIFAR10)	24.3	7.3	4.5	549	0.3	gradient-based
Ours(ImageNet)	23.6	6.9	4.7	572	3.6	gradient-based

Table 3, the architecture searched by EC-DARTS on ImageNet achieves 23.6%/6.9% top-1/5 test errors, which is competitive among the other gradient-based methods. EC-DARTS outperforms manually designed architectures by a clear margin, with a smaller model size. In comparison, PC-NAS-S [22], a heuristic-based method, reports a top-1 test error comparable to that achieved by EC-DARTS, but its model size is larger. To comprehensively analyze EC-DARTS with the other NAS methods, we further transfer the searched architecture from CIFAR10 to ImageNet. The comparison results are listed in Table 3, and EC-DARTS achieves state-of-the-art top-1/5 test errors of 24.3%/7.3%. This result verifies the robustness of the architecture searched by our EC-DARTS.

5.6. Ablation study

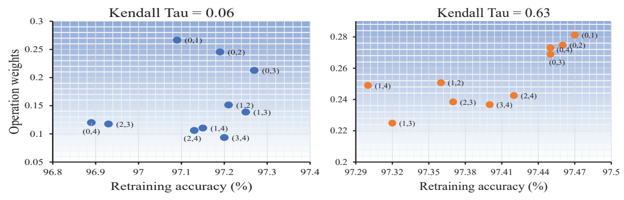
We conduct the ablation studies on CIFAR10 to verify the effectiveness of different components in EC-DARTS. All experiments listed in Table 4 are under the same search and retraining settings.

Effectiveness of Cross-edge Normalization. The operation gap makes the coupled operations in the one-shot model to be optimized to different degrees. As a result, the optimization of operation weights is biased, that is, the correlation between operation weights and model performances becomes weak. In EC-DARTS, we propose CEN to decouple different operations based on their categories. As summarized in Table 4, the test error is reduced from 2.76% to 2.60% with CEN, and the search cost remains the same level as the baseline (DARTS). To further verify the effectiveness of CEN in alleviating the operation gap problem, we adopt the Kendall Tau Metric [17] to measure the correlation between operation weights and model performances. We use 10 architectures which are generated by replacing the pairwise edges, that are connected with the last intermediate node in the searched architecture, with different combinations. Specifically, the last intermediate node can have an edge connected to each of the remaining 5 nodes, and only 2 of these edge connections remain in the final architecture. Therefore, a total of 10 architectures can be obtained through all possible pairwise combinations of these edge connections. As shown in Figure 3 (a), the Kendall Tau value of DARTS [26] is 0.06, which indicates a weak correlation between operation weights and model performances. In contrast, CEN is able to shrink the operation gap, which significantly improves the Kendall Tau value to 0.63.

Table 4. Ablation study of CEN and IST on CIFAR10.

Method	Test Err.(%)
Baseline(DARTS)	2.76±0.09
Baseline + CEN	2.60±0.13
Baseline + IST	2.47±0.06
EC-DARTS	2.39±0.08

Effectiveness of Induced Structural Transition. The in-



(a) DARTS (2nd Order) (b) DARTS with CEN

Figure 3. The correlation evaluation between operation weights and model performance for CEN.

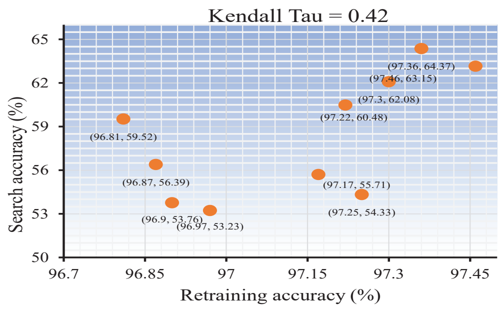
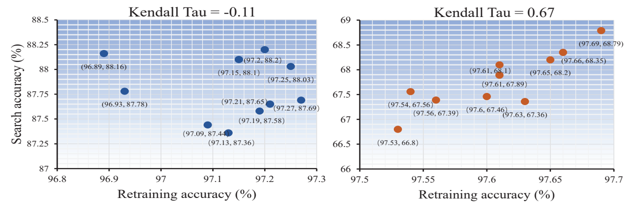


Figure 4. The correlation evaluation between the search rank and retraining rank with our IST.

consistent structures in search and retraining stages bring the structure gap in DARTS [26], which can be interpreted as a correlation between the search rank and retraining rank. We use the Kendall Tau Metric [17] to measure this correlation in a single run of DARTS. As shown in Figure 1 (a), DARTS reports a Kendall Tau value of 0.13, which means that the search rank and retraining rank during the search process in DARTS have a weak correlation. We summarize 10 randomly selected results in a single run of the baseline (DARTS) with IST in Figure 4. The baseline with IST achieves a higher Kendall Tau value than that in DARTS, which means a stronger correlation between the search rank and retraining rank during the search process. The high Kendall Tau value illustrates that the structure gap can be shrunken by IST. Results shown in Table 4 demonstrate that the baseline with IST achieves a test error of 2.47%, which surpasses the baseline by a clear margin. The effectiveness of IST is further verified.

Additional Analysis. In this part, we analyze the correlation of the searched results from 10 independent searches. As shown in Figure 5, the Kendall Tau value is calculated between the search rank and retraining rank of 10 independent searched architectures, and the average retraining accuracies of DARTS and EC-DARTS are 97.13% and 97.61%, respectively. The lower search accuracy in Figure 1 (b), Figure 4 and Figure 5 (b) is caused by IST, because the auxiliary model in IST has a much smaller parameter quantity than the one-shot model. Compared with DARTS, EC-DARTS achieves higher Kendall Tau and average retraining accuracy, which further indicates that the correlation of the search rank and retraining rank is improved in EC-DARTS. These results also reflects that EC-DARTS is relatively stable than DARTS.



(a) DARTS (2nd Order) (b) EC-DARTS(ours)

Figure 5. The correlation evaluation between the search rank and retraining rank of 10 independent searches.

6. Conclusion

In this work, we analyze the optimization gaps lie in DARTS, and conduct experiments to show the negative impact caused by the optimization gaps. As shown in Figure 3 (a), the operation weights and model performance in DRATS are almost irrelevant, Moreover, DARTS shows a weak correlation between the search rank and retraining rank in Figure 1 (a) and Figure 5 (a). Then, we propose EC-DARTS, which includes CEN and IST to shrink these optimization gaps in DARTS. CEN decouples operations to optimize operation weights of the same categories in a equalized manner, and IST contains an auxiliary model to induce the model structure type transforms between the search and retraining stages. The experimental results demonstrate that EC-DARTS achieves competitive performances on both the proxy dataset and target dataset. The experimental analysis of the correlation further verified that EC-DARTS is effective in shrinking the operation and structure gaps.

Acknowledgments

This work is supported by the National Science Fund for Distinguished Young Scholars (No.62025603), the National Natural Science Foundation of China (No.U1705262, No. 62072386, No. 62072387, No. 62072389, No. 62002305, No.61772443, No.61802324 and No.61702136), Guangdong Basic and Applied Basic Research FoundationNo.2019B1515120049) and the Fundamental Research Funds for the central universities (No. 20720200077, No. 20720200090 and No. 20720200091).

References

[1] Kaifeng Bi, Changping Hu, Lingxi Xie, Xin Chen, Longhui Wei, and Qi Tian. Stabilizing darts with amended gradient estimation on architectural parameters. *arXiv preprint arXiv:1910.11831*, 2019.

[2] Kaifeng Bi, Lingxi Xie, Xin Chen, Longhui Wei, and Qi Tian. Gold-nas: Gradual, one-level, differentiable. *arXiv preprint arXiv:2007.03331*, 2020.

[3] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. *arXiv preprint arXiv:1812.00332*, 2018.

[4] Xiangning Chen and Cho-Jui Hsieh. Stabilizing differentiable architecture search via perturbation-based regularization. *arXiv preprint arXiv:2002.05283*, 2020.

[5] Xin Chen, Lingxi Xie, Jun Wu, and Qi Tian. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1294–1303, 2019.

[6] Xiangxiang Chu, Bo Zhang, and Xudong Li. Noisy differentiable architecture search. *arXiv preprint arXiv:2005.03566*, 2020.

[7] Xiangxiang Chu, Tianbao Zhou, Bo Zhang, and Jixiang Li. Fair darts: Eliminating unfair advantages in differentiable architecture search. *arXiv preprint arXiv:1911.12126*, 2019.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[9] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[10] Xuanyi Dong and Yi Yang. Searching for a robust neural architecture in four gpu hours. In *Proceedings of the IEEE Conference on computer vision and pattern recognition*, pages 1761–1770, 2019.

[11] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *arXiv preprint arXiv:1808.05377*, 2018.

[12] Minghao Guo, Zhao Zhong, Wei Wu, Dahua Lin, and Junjie Yan. Irlas: Inverse reinforcement learning for architecture search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9021–9029, 2019.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[14] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[15] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[16] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[17] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[21] Guilin Li, Xing Zhang, Zitong Wang, Zhenguo Li, and Tong Zhang. Stacnas: Towards stable and consistent optimization for differentiable neural architecture search. *arXiv preprint arXiv:1909.11926*, 2019.

[22] Xiang Li, Chen Lin, Chuming Li, Ming Sun, Wei Wu, Junjie Yan, and Wanli Ouyang. Improving one-shot nas by suppressing the posterior fading. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13836–13845, 2020.

[23] Hanwen Liang, Shifeng Zhang, Jiacheng Sun, Xingqiu He, Weiran Huang, Kechen Zhuang, and Zhenguo Li. Darts+: Improved differentiable architecture search with early stopping. *arXiv preprint arXiv:1909.06035*, 2019.

[24] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 19–34, 2018.

[25] Hanxiao Liu, Karen Simonyan, Oriol Vinyals, Chrisantha Fernando, and Koray Kavukcuoglu. Hierarchical representations for efficient architecture search. *arXiv preprint arXiv:1711.00436*, 2017.

[26] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[27] Renqian Luo, Fei Tian, Tao Qin, Enhong Chen, and Tie-Yan Liu. Neural architecture optimization. In *Advances in neural information processing systems*, pages 7816–7827, 2018.

[28] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, pages 116–131, 2018.

[29] Niv Nayman, Asaf Noy, Tal Ridnik, Itamar Friedman, Rong Jin, and Lihi Zelnik. Xnas: Neural architecture search with expert advice. In *Advances in Neural Information Processing Systems*, pages 1977–1987, 2019.

[30] Hieu Pham, Melody Y Guan, Barret Zoph, Quoc V Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. *arXiv preprint arXiv:1802.03268*, 2018.

[31] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pages 4780–4789, 2019.

[32] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. A comprehensive survey of neural architecture search: Challenges and solutions. *arXiv preprint arXiv:2006.02903*, 2020.

[33] Cristiano Saltori, Subhankar Roy, Nicu Sebe, and Giovanni Iacca. Regularized evolutionary algorithm for dynamic neural topology search. In *International Conference on Image Analysis and Processing*, pages 219–230. Springer, 2019.

[34] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[35] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[36] Alvin Wan, Xiaoliang Dai, Peizhao Zhang, Zijian He, Yuandong Tian, Saining Xie, Bichen Wu, Matthew Yu, Tao Xu, Kan Chen, et al. Fbnetv2: Differentiable neural architecture search for spatial and channel dimensions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12965–12974, 2020.

[37] Martin Wistuba, Ambrish Rawat, and Tejaswini Pedapati. A survey on neural architecture search. *arXiv preprint arXiv:1905.01392*, 2019.

[38] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10734–10742, 2019.

[39] Lingxi Xie, Xin Chen, Kaifeng Bi, Longhui Wei, Yuhui Xu, Zhengsu Chen, Lanfei Wang, An Xiao, Jianlong Chang, Xiaopeng Zhang, et al. Weight-sharing neural architecture search: A battle to shrink the optimization gap. *arXiv preprint arXiv:2008.01475*, 2020.

[40] Lingxi Xie and Alan Yuille. Genetic cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1379–1388, 2017.

[41] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. Snas: stochastic neural architecture search. *arXiv preprint arXiv:1812.09926*, 2018.

[42] Yuhui Xu, Lingxi Xie, Xiaopeng Zhang, Xin Chen, Guo-Jun Qi, Qi Tian, and Hongkai Xiong. Pc-darts: Partial channel connections for memory-efficient differentiable architecture search. *arXiv preprint arXiv:1907.05737*, 2019.

[43] Hongyuan Yu and Houwen Peng. Cyclic differentiable architecture search. *arXiv preprint arXiv:2006.10724*, 2020.

[44] Arber Zela, Thomas Elsken, Tonmoy Saikia, Yassine Marrakchi, Thomas Brox, and Frank Hutter. Understanding and robustifying differentiable architecture search. *arXiv preprint arXiv:1909.09656*, 2019.

[45] Xiawu Zheng, Rongrong Ji, Lang Tang, Baochang Zhang, Jianzhuang Liu, and Qi Tian. Multinomial distribution learning for effective neural architecture search. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1304–1313, 2019.

[46] Pan Zhou, Caiming Xiong, Richard Socher, and Steven CH Hoi. Theory-inspired path-regularized differential network architecture search. *arXiv preprint arXiv:2006.16537*, 2020.

[47] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

[48] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018.