DU-DARTS: Decreasing the Uncertainty of Differentiable Architecture Search

Shun Lu[1,2,3]
lushun19s@ict.ac.cn
Yu Hu*[1,2,3]
huyu@ict.ac.cn
Longxing Yang[1,2,3]
yanglongxing20b@ict.ac.cn
Zihao Sun[1,2,3]
sunzihao18z@ict.ac.cn
Jilin Mei[1,2,3]
meijilin@ict.ac.cn
Yiming Zeng[4]
zyms5244@gmail.com
Xiaowei Li[2,3]
lxw@ict.ac.cn

[1] Research Center for Intelligent Computing Systems, Institute of Computing Technology, Chinese Academy of Sciences
[2] State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences
[3] School of Computer Science and Technology, University of Chinese Academy of Sciences.
[4] Tecent ADlab

Abstract

Differentiable Neural Architecture Search (DARTS) recently attracts a lot of research attention because of its high efficiency. However, the competition of candidate operations in DARTS introduces high uncertainty for selecting the truly important operation, thus leading to serious performance collapse. In this work, we decrease the uncertainty of differentiable architecture search (DU-DARTS) by enforcing the distribution of architecture parameters to approach the one-hot categorical distribution and by replacing the *zero* operation with a gate switch. Without any extra search cost, our method achieves state-of-the-art performance with 2.32%, 16.74%, and 24.1% test error on CIFAR-10, CIFAR-100, and ImageNet datasets, respectively. Moreover, DU-DARTS can robustly find an excellent architecture on NAS-Bench-1Shot1 and NAS-Bench-201, which further demonstrates the effectiveness of our method. The source code is available at https://github.com/ShunLu91/DU-DARTS.

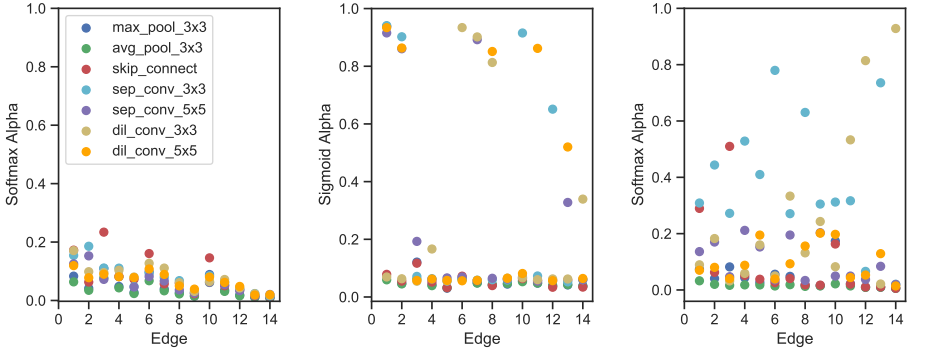
1 Introduction

In recent years, neural architecture search (NAS) has demonstrated competitive to or even better performance than hand-crafted neural architectures for various tasks, such as image classification, semantic segmentation, and natural language processing. The purpose of NAS is to automatically search high-performance network architectures in a pre-defined search space. As the search space is extraordinarily huge, e.g. 10^{18} in DARTS [18], the search process of NAS is very time- and resource-consuming.

*Corresponding Author

Lots of works have been proposed to improve the search efficiency for NAS. Among the existing works, gradient-based methods such as DARTS successfully reduce the search time to several GPU hours. DARTS[18] and its variants are very efficient because of end-to-end optimization of network architectures and weights. In specific, these methods adopt a mixed operation and utilize the Softmax function to relax architecture parameters from discrete to be continuous, and back-propagate the gradients with the chain rule, which enables architecture parameters to be efficiently trained by bi-level gradient descent optimization. However, the Softmax function of the mixed operations unavoidably introduces competitions among candidate operations, which results in high uncertainty for selecting the truly important operation and consequently leads to serious performance collapse.

Some works [5, 7, 16, 29] have proposed to resolve the competition problem. GDAS [7] proposed to sample a sub-graph at every training step to avoid the competition among candidate operations. FairDARTS [5] and PR-DARTS [29] both observed that the skip-connect has an unfair advantage over other operations. While FairDARTS replaced the Softmax function with a Sigmoid function, PR-DARTS utilized a binary gate to avoid unfair competition.



(a) DARTS (b) FairDARTS (c) DU-DARTS

Figure 1: Distribution of the architecture parameters in normal cells.

As architecture parameters for candidate operations are initialized to the same value, the selection of operations has a maximum uncertainty at the beginning. During the search process, architecture parameters are expected to be distinct for each candidate operation so that an optimal architecture can be constructed, indicating the decrease of uncertainty at the end of the search. However, as shown in Fig 1 (a), the values of architecture parameters of candidate operations in DARTS are still very close after search due to the competition. Under this circumstance, it is hard to distinguish which operation is the best for each edge.

Existing methods [5, 7, 16, 29] attempt to improve the fairness of the competition, which can be considered as implicitly decreasing the search uncertainty. In this paper, we propose to enforce the distribution of architecture parameters to approach the one-hot categorical distribution so that the most important operation can have clear superiority. Therefore, our method explicitly decreases the search uncertainty. Moreover, we substitute the *zero* operation with a searchable gate switch to further decrease the uncertainty caused by the disturbance of the *zero* operation. In this way, not only do we decrease the uncertainty of architecture parameters after the search, but also we can easily select the most important operation for each edge to construct a superb architecture.

Our contributions can be summarized as follows:

- We propose a new perspective of differentiable NAS that the search process is to decrease the uncertainty of architecture parameters so that the operation with clear superiority for each edge can be selected at the end of the search.
- We propose to enforce the distribution of architecture parameters to approach the one-hot categorical distribution by adding distance regularization or by information entropy in the loss function.
- A searchable gate switch is inserted behind each mixed operation to further eliminate the uncertainty caused by the disturbance of the *zero* operation.
- Extensive experiments are conducted on three datasets and the results demonstrate that our method achieves state-of-the-art performance on all of them. Furthermore, evaluating DU-DARTS on NAS-Bench-1Shot1 [28] further shows its effectiveness.

2 Related Work

Neural architecture search In the past few years, NAS has achieved state-of-the-art performance for many tasks. Reinforcement learning was firstly applied to NAS by [1, 30] but required thousands of GPU days to search for an excellent architecture. Afterwards, ENAS [20] and One-shot [2] proposed to share weights among child models, named weight sharing mechanism, dramatically reducing the search time to only one GPU day. Recently, gradient-based methods such as DARTS [18] and its variants apply the continuous-relaxation to optimize architecture parameters by gradient descent and are able to search architectures within several GPU hours.

DARTS variants Despite the high efficiency, DARTS[18] has difficulty in robustly finding a superb architecture from the enormous search space. According to previous works, we can summarize the main problems of DARTS as follows: the depth gap, the sharp minimum, the high memory consumption, and the operation competition problem.

P-DARTS [4] and StacNAS [13] pointed out *the depth gap* between the supernet and target model. P-DARTS progressively increased the depth of the supernet to mitigate the depth gap and StacNAS used gradient confusion to select a proper depth for the supernet. RobustDARTS [27] and SDARTS [3] observed that the search easily falls into *the sharp minimum* and thus is sensitive to the perturbations of architecture parameters. RobustDARTS leverages eigenvalues of architecture parameters to monitor the search process and applied stronger regularizations to improve the search performance. SDARTS enhanced the stability of DARTS by adding noise or by using adversarial training to architectural parameters. PC-DARTS [25] and SGAS [14] reduced *the memory consumption*, while the former employed the partial channel connections and the latter gradually pruned unnecessary candidate operations during the search process, both of which further boost the search efficiency and performance. Some works have focused on *the operation competition* problem. DARTS+ [16] adopted the early stop strategy to solve the cooperation and competition problem. Fair-DARTS [6] eliminated the unfair advantage of the skip-connect by a Sigmoid function. PR-DARTS [29] introduced a binary gate for each operation to avoid unfair competition.

Our method also attempts to resolve the operation competition problem. But different from previous methods, we consider the competition leads to large uncertainty, and the *zero* operation further disturbs the operation selection. Addressing the two underlying factors greatly improves the DARTS performance.

3 Method

We first briefly introduce the preliminary of DARTS. Then the uncertainty of architecture parameters is illustrated. Next, we propose our method to decrease the uncertainty and improve the stability of the search. Finally, we analyze the uncertainty caused by the disturbance of the *zero* operation and solve this problem by replacing it with a simple Sigmoid function.

3.1 Preliminary of DARTS

Since NASNet [31] proposed to build the architecture with repeated cells, many follow-up works adopt the same paradigm. DARTS [18] was the first to utilize the gradient descent for cell-based NAS. Specifically, each cell contains several nodes and each edge between every node is regarded as a mixed operation $\overline{o}^{(i,j)}$ in DARTS, which can be formulated as:

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

where $\mathcal{O}$ denotes the candidate operation corpus and $\alpha_o^{(i,j)}$ is the corresponding architecture parameter for the operation o between nodes (i,j). By replacing the categorical representation with continuous architecture parameters, DARTS is able to efficiently search the architecture in a differentiable method. During the search, it adopts a bi-level optimization to update the network weights and architecture parameters alternately as below:

$$\begin{aligned} \min_{\alpha} \quad & \mathcal{L}_{val}(w^*(\alpha), \alpha) \\ s.t. \quad & w^*(\alpha) = \arg\min_w \mathcal{L}_{train}(w, \alpha) \end{aligned} \tag{2}$$

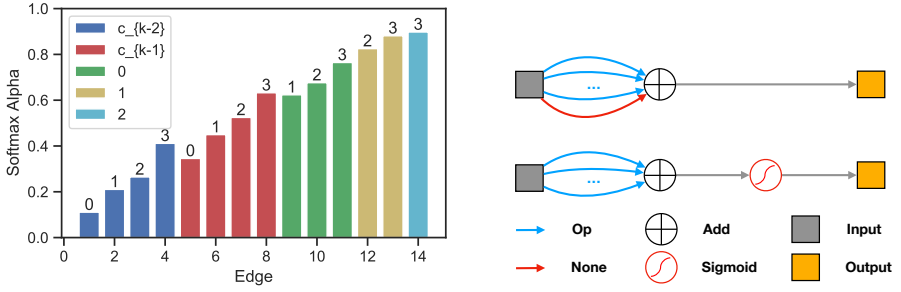
where $\mathcal{L}_{train}$ and $\mathcal{L}_{val}$ are the training and validation loss, respectively. The goal of the search is to find the best architecture parameters α^* by minimizing the validation loss $\mathcal{L}_{val}(w^*(\alpha), \alpha)$.

3.2 The uncertainty of architecture parameters

The differentiable NAS method enjoys a substantially efficient search, but the uncertainty of architecture parameters is still large after the search. As shown in Fig. 1 (a), we visualize the architecture parameters of the normal cells searched by DARTS [18] and it is obvious that most architecture parameters of candidate operations are very close. Since architecture parameters are too close to distinguish, the selection of the candidate operation with the largest architecture parameter becomes uncertain. FairDARTS [8] eliminated the unfair advantage of the skip-connect (as shown in Fig. 1(b), the number of dominant red dots are significantly reduced), and improved the fairness of competition by adopting a Sigmoid function for each operation. As shown in the figure, the dots are clearly separated into the high Sigmoid alpha group and the low Sigmoid alpha group. However, using Sigmoid function can not prevent architecture parameters from being close to each other, FairDARTS still has the uncertainty issue. As shown in the figure, at Edge 1, 5, and 6, the dots with high Sigmoid alphas are still entangled, indicating difficulties in distinguishing the most important operation.

3.3 Decrease the uncertainty of NAS

Based on the analysis above, we propose to decrease the uncertainty of the searched architecture parameters for differentiable NAS by adding a regularization term to the loss function.



(a) Architecture parameters of the *zero* operation (b) Comparison of the mixed operation

Figure 2: (a) Architecture parameters of the *zero* operation on all edges obtained from DARTS [18]. The legend denotes indies of start nodes of edges and the start node gradually becomes deeper from left to right. We place indies of target nodes of edges on the top of each bar. (b) Comparison of the mixed operation from DARTS[18] and our method. The original mixed operation in DARTS[18] is shown on the top of this figure and the *zero* operation is emphasized with red color. We replace the *zero* operation with a gate switch, specifically a Sigmoid function.

The regularization term can either be the distance between the continuous architecture parameters and their one-hot counterpart or be the information entropy. Both regularization techniques are effective for decreasing the uncertainty of NAS according to our experiments.

Distance regularization On the one hand, we can explicitly enforce the continuous distribution to approximate the one-hot categorical distribution. The motivation of continuous relaxation in DARTS [18] is to approximate the one-hot representation with continuous values and thus can be optimized in an end-to-end differentiable manner. However, as the architecture parameters of candidate operations are close to each other, the search result has deviated from the original motivation. We add an additional loss term $\mathcal{L}_\alpha$ to explicitly enforce architecture parameters towards what we expect by measuring the distance between the current architecture parameters and its one-hot counterpart, which can be formulated as:

$$\mathcal{L}_\alpha = Distance(\alpha, \hat{\alpha}) \tag{3}$$

where the $\hat{\alpha}$ is obtained according to the location of the largest value of each edge by $\arg\max(\alpha)$. As for measuring the distance, there are many options for us to choose from, such as MAE, MSE, RMSE, and amount of information (AoI). We adopt the widely used metrics in our experiments and further analyze its effect in Section 4.3.

Information entropy regularization On the other hand, the large uncertainty of architecture parameters after search indicates that the information entropy of the result is also great. Hence, a more straightforward method is to directly reduce the information entropy of architecture parameters during the search by adding the additional loss term $\mathcal{L}_\alpha$ as follows:

$$\mathcal{L}_\alpha = -\alpha \log_2 \alpha \tag{4}$$

When the search converges, the information entropy term included in the loss function will gradually decline. Moreover, the Softmax function limits the sum of architecture parameters of each edge to be 1, resulting in only one operation close to 1 and the others will be close to 0. As a result, only one operation of each edge obviously surpasses others and shows clear superiority, so the uncertainty of architecture parameters will undoubtedly be minimized.

Algorithm 1: Training of DU-DARTS using first-order approximation

Initialize a mixed operation $\overline{o}^{(i,j)}$ parameterized by $\alpha^{(i,j)}$ for each edge (i,j)
while *not converged* **do**
1. Update parameters α and β by descending $\nabla_{\alpha,\beta}(\mathcal{L}_{val}(w,\alpha,\beta)+\mathcal{L}_{\alpha}*\mathcal{L}_{\beta})$
2. Update architecture weights w by descending $\nabla_{w}\mathcal{L}_{train}(w,\alpha,\beta)$

Derive the searched architecture according to the optimized architecture parameters α and β

3.4 Decrease the disturbance of *zero* operation

From the experiments, we notice that the *zero* operation dominates most edges, making rest candidate operations share the remaining small weight portion. Consequently, the weights of the 7 candidate operations are all small and close to each other again. In fact, as the output of the mixed operation is from the other candidate operations and the *zero* operation does not convey any information at each forward pass, it is irrational to let the *zero* operation compete with others. As the *zero* operation denotes whether there is an edge between two nodes, so we can convert it to a soft gate switch to control whether or not to output the mixed operation. Specifically, we propose to replace the *zero* operation with a Sigmoid function which is located behind each mixed operation, as shown in Fig. 2(b). We denote the architecture parameter of the *zero* operation as β. In this way, we can eliminate the uncertainty caused by the disturbance of the *zero* operation and directly represent the importance of each edge with β. We reformulate the output of the mixed operation between nodes (i,j) as:

$$\overline{o}^{(i,j)}(x)=\sigma(\beta^{(i,j)})*\sum_{o\in\mathcal{O}'}\frac{\exp(\alpha_{o}^{(i,j)})}{\sum_{o'\in\mathcal{O}'}\exp(\alpha_{o'}^{(i,j)})}o(x) \tag{5}$$

where σ denotes the Sigmoid function and $\mathcal{O}'$ represents the candidate operation corpus without the *zero* operation.

Furthermore, we observe that the dominance of the *zero* operation becomes more prominent along with increasing the indexes of the target nodes, as shown in Fig. 2(a), which implies that DARTS prefers to select the edge between shallow nodes, as also claimed in [22]. Therefore, we propose to enforce the edge importance β to become larger by formulating its reciprocal as another loss term:

$$\mathcal{L}_{\beta}=\frac{1}{\beta} \tag{6}$$

In this way, as the *zero* operations at deeper edges have higher values, so the deeper β has a smaller value, corresponding to a larger loss $\mathcal{L}_{\beta}$, which helps to search deeper cells.

Overall, the training objective of our method can be reformulated as:

$$\begin{aligned}&\min_{\alpha,\beta}\quad \mathcal{L}_{val}(w^{*}(\alpha,\beta),\alpha,\beta)+\mathcal{L}_{\alpha}*\mathcal{L}_{\beta}\\&s.t.\quad w^{*}(\alpha,\beta)=\arg\min_{w}\mathcal{L}_{train}(w,\alpha,\beta)\end{aligned} \tag{7}$$

The optimization procedures are summarized in Alg. 1 and the visualization of architecture parameters of the normal cell searched by DU-DARTS are shown in Fig. 1(c). We can conclude that architecture parameters of operations with clear superiority are significantly larger than the others, manifesting that DU-DARTS manages to decrease the uncertainty and can easily pick out truly important operations as expected.

4 Experiments

To validate the effectiveness of our method, we conduct extensive experiments on three benchmark datasets of classification. In this section, we first briefly introduce each dataset and provide the implementation details. Then we report and analyze the evaluation results. Finally, we adopt our method to search on NAS-Bench-1Shot1 [28] and NAS-Bench-201 [8] to further demonstrate its effectiveness.

4.1 Datasets

CIFAR-10 and CIFAR-100 Both datasets contain 60000 color images in total with 50000 training images and 10000 test images, and are widely used in image classification. CIFAR-10 is a 10-classes dataset with 6000 images for each class while CIFAR-100 is divided into 100 classes, each of which only has 600 samples. During the search phase, we take half of the training set as the validation set to optimize the architecture parameters. However, we utilize the whole training set to retrain our searched models and evaluate them on the test set.

ImageNet ImageNet [12] is a widely used large-scale dataset for image classification. There are 128 million training images and 50000 validation images, which are divided into 1000 classes. We adopt the total training images to retrain our searched architectures and use the validation accuracy to measure the model performance.

4.2 Implementation details

We conduct architecture search on CIFAR-10 and CIFAR-100 datasets by constructing the supernet with 8 and 5 cells respectively and adopt the first-order DARTS algorithm. We employ the SGD optimizer and the initial learning rate is 0.025 for network parameters with a cosine decay strategy, while architecture parameters are optimized by an Adam optimizer with a fixed learning rate of 3e-4. The batch size is 64 and we search for 50 epochs in total.

As for evaluation, we adopt the same tricks as DARTS such as cutout and auxiliary loss. The seed is fixed as 0 for all models retraining and other hyper-parameters are also kept the same as DARTS. Moreover, we directly retrain the searched models in the last epoch without any selection like DARTS, which further improves our search efficiency.

To test the transferability, we transfer our best searched architectures on CIFAR-10 to ImageNet. We adopt the SGD optimizer with weight decay 3e-5 to retrain the model, and the initial learning rate is 0.5 and decayed to 0 within 250 epochs. More experimental details can be found in the supplementary material.

4.3 Results on CIFAR-10 and CIFAR-100

To explore which kind of regularization is the most effective for our method, we conduct comparative experiments according to Eq. (3) and Eq. (4). To exclude the influence of accidental factors, we search with three different random seeds (0, 1, 2 in our experiments) and retrain the searched architectures with the same random seed=0 to get a fair comparison.

As shown in Tab. 1, all distance regularization and information entropy regularization can achieve lower average test error and standard deviation than DARTS, which validates the necessity of decreasing the uncertainty of NAS. Moreover, applying the entropy regularization outperforms the other regularization by a large margin, showing that measuring

Regularization	Test Error (%)		
	#0	#1	#2
MAE	2.77	2.73	2.52
MSE	2.71	2.91	2.84
RMSE	2.76	2.93	2.66
AoI	2.69	2.81	2.68
Entropy	**2.32**	**2.47**	**2.36**

Table 1: Results on CIFAR-10 using various regularization types. DARTS [18] has no additional regularization for architecture parameters and achieves 3.00±0.14 test error. However, from above results of different seeds, we can get better average results of 2.67±0.11, 2.82±0.08, 2.78±0.11, 2.73±0.06, **2.38±0.06** by above regularization types respectively.

Method	Test Error(%)		#P
	Top-1	Top-5	(M)
MobileNet [10]	29.4	10.5	4.2
ShuffleNet V2 [19]	25.1	-	-
NASNet-A [31]	26.0	8.4	5.3
PNAS [17]	25.8	8.1	5.1
MnasNet-92 [23]	25.2	8.0	4.4
DARTS [18]	26.7	8.7	4.7
SNAS [24]	27.3	9.2	4.3
P-DARTS [4]	24.4	7.4	4.9
PC-DARTS [25]	25.1	7.8	5.3
GDAS [7]	26.0	8.5	5.3
SGAS [14]	24.1	7.3	5.4
FairDARTS [5]	24.9	7.5	4.8
DU-DARTS	**24.1**	**7.3**	5.3

Table 2: Comparison with other NAS methods on ImageNet. #P: The number of model parameters.

the amount of information of architecture parameters is more effective than calculating the distance between architecture parameters and their one-hot counterpart.

With the entropy regularization, we further compare our method with other state-of-the-arts in Tab. 3. We report the best results and the average results by 3 independent runs with different random seeds to test the effectiveness and stability of our method. Using the same search cost as DARTS, our approach achieves the lowest test error 2.32% and 16.74% on CIFAR-10 and CIFAR-100 datasets respectively, outperforming most previous methods. Besides, through 3 independent runs, our method is very stable with the average error rate 2.38±0.06 and 17.04±0.23 on both datasets. We provide the visualization of our searched cells and detailed explanations of the abbreviation operations in the supplementary material.

4.4 Results on ImageNet

Like previous methods, we directly retrain our best searched architecture on the large-scale image classification dataset ImageNet [12] to test the transferability of our method. As shown in Tab. 2, NAS methods generally perform better than the hand-crafted models such as MobileNet [10] and ShuffleNet V2 [19]. Moreover, DU-DARTS achieves the lowest top-1 error 24.1% and top-5 error 7.3%, outperforming all the previous methods by a large margin. The result shows the superior transferability of DU-DARTS.

4.5 Results on NAS-Bench-1Shot1

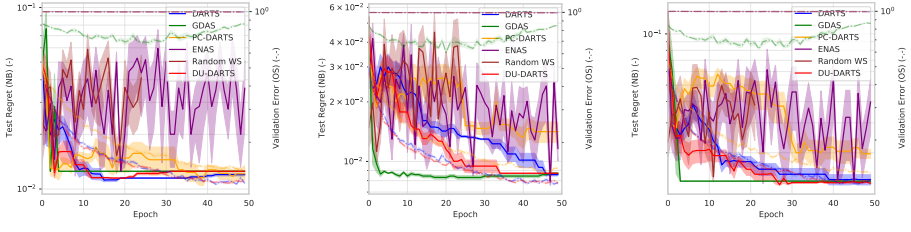
To verify the robustness of our method, we further conduct experiments on NAS-Bench-1Shot1 [28] which splits the original search space in NAS-Bench-101 [26] into three parts, named search space 1, search space 2 and search space 3. Architectures in various search spaces consist of 9 cells and other basic layers. Each cell includes several nodes and NAS-Bench-1Shot1 constrains the total number of parents of all nodes to be 9 in all search spaces.

Method	CIFAR-10		CIFAR-100		GPU
	Test Err.(%)	Params(M)	Test Err.(%)	Params(M)	Days
NASNet [31]	2.65	3.3	-	-	1800
AmoebaNet [21]	2.55±0.05	2.8	18.93*	3.1	3150
PNAS [17]	3.41±0.09	3.2	19.53*	3.2	225
ENAS [20]	2.89	4.6	19.43*	4.6	0.5
DARTS(1st order) [18]	3.00±0.14	3.3	20.58±0.44†	-	1.5
DARTS(2nd order) [18]	2.76±0.09	3.3	-	-	4
SNAS [24]	2.85±0.02	2.8	-	-	1.5
P-DARTS [4]	2.5	3.4	**15.92**	3.6	0.3
PC-DARTS [25]	2.57±0.07	3.6	-	-	0.1
GDAS [7]	2.93	3.4	18.38	3.4	0.17
RobustDARTS [27]	2.95±0.21	-	18.01±0.26	-	1.6
SGAS [14]	2.39	3.7	-	-	0.25
SDARTS [3]	2.61±0.02	3.3	-	-	1.3
CNAS[9]	2.60±0.06	3.7	-	-	0.3
FairDARTS [5]	2.54±0.05	3.32±0.46	-	-	-
DU-DARTS (avg.)	**2.38±0.06**	3.69±0.06	17.04±0.23	3.14±0.14	0.4
DU-DARTS (best)	**2.32**	3.8	16.74	3.1	0.4

Table 3: Search results on CIFAR-10 and CIFAR-100. We report the average and the best performance of DU-DARTS. *: Reported by GDAS [7]. †: Reported by RobustDARTS [27].

Since NAS-Bench-1Shot1 provides a novel mapping between different search space representations, we can easily query the performance of the searched architecture at each epoch.

We compare our method with others in Fig 3. We can see that DU-DARTS not only robustly achieves comparable results but also converges faster than the other methods and has a smaller variance after convergence, indicating that DU-DARTS is effective and robust.



(a) Search space 1 (b) Search space 2 (c) Search space 3

Figure 3: Comparison of DU-DARTS with different NAS methods on three search spaces defined in NAS-Bench-1Shot1 [28]. The solid lines show the anytime test regret (mean ± std), while the dashed blurred lines represent validation error (Best viewed in color).

4.6 Results on NAS-Bench-201

Architectures in NAS-Bench-201 [8] are built with repeated cells, which comprise 4 nodes and 6 edges. With 5 candidate operations for each edge, there are 15,625 distinct structures in total. We further deployed our method on NAS-Bench-201 [8] and the results are shown in Tab.4. Our method achieved the highest accuracy in different datasets and outperformed others by a large margin, clearly demonstrating the effectiveness of DU-DARTS.

Method	CIFAR-10		CIFAR-100		ImageNet-16-120	
	validation	test	validation	test	validation	test
Optimal	91.61	94.37	73.49	73.51	46.77	47.31
RSPS	80.42±3.58	84.07±3.61	52.12±5.55	52.31±5.77	27.22±3.24	26.28±3.09
DARTS	39.77±0.00	54.30±0.00	15.03±0.00	15.61±0.00	16.43±0.00	16.32±0.00
GDAS	89.89±0.08	93.61±0.09	71.34±0.04	70.70±0.30	41.59±1.33	41.71±0.98
SETN	84.04±0.28	87.64±0.00	58.86±0.06	59.05±0.24	33.06±0.02	32.52±0.21
ENAS	37.51±3.19	53.89±0.58	13.37±2.35	13.96±2.33	15.06±1.95	14.84±2.10
Ours	**91.21±0.11**	**93.86±0.05**	**71.88±0.53**	**71.84±0.39**	**45.65±0.75**	**45.94±0.93**

Table 4: The average search results from three independent runs on NAS-bench-201 [8]. Results of RSPS [15], DARTS [18], GDAS [7], SETN [6] and ENAS [20] are also listed for comparison.

4.7 Discussion

Uncertainty In general, uncertainty means lack of sureness about something and here we use it to represent the probability of selecting a candidate operation. We adopt the modified version of the Shannon entropy $H_{S'} = -\sum_j c_j \log c_j^2$ from [11] to measure the uncertainty of searched architecture parameters in Tab.5. Some methods got even higher uncertainty than DARTS, but DU-DARTS achieved significant uncertainty decrease as expected.

Ablation study We conducted detailed component analysis in Tab.6. Note that directly applying the regularization to DARTS [18] managed to decrease the uncertainty and bring the performance gain. Though deleting the candidate *zero* operation got worse results, it can be improved a lot when equipped with a gate switch.

Method	Normal	Reduce
DARTS [18]	57.09	81.77
SNAS [24]	60.90	60.92
PC-DARTS [25]	82.22	83.44
GDAS [7]	83.89	83.92
SDARTS [3]	75.51	77.57
DU-DARTS	**43.90**	**22.86**

Table 5: Uncertainty measure.

DEL Zero	REG	Gate	Test Err.(%)
×	×	×	3.00±0.14
×	√	×	2.62±0.05
√	√	×	2.75±0.09
√	√	√	**2.38±0.06**

Table 6: Ablation study. DEL Zero: delete the candidate *zero* operation. REG: the entropy regularization. Gate: the soft gate switch.

5 Conclusion

In this paper, we discover that the competition of candidate operations leads to a large uncertainty for the differentiable NAS, greatly disturbing the operation selection. We propose to explicitly decrease the uncertainty by adding distance regularization or information entropy loss to enforce the distribution of architecture parameters to approach the one-hot categorical distribution and measure it by the Shannon entropy. Moreover, we replace the *zero* operation with a searchable gate switch to further eliminate the disturbance. Extensive experiments on benchmark datasets demonstrate the effectiveness and stability of our method.

6 Acknowledgements

This work is supported in part by the National Key R&D Program of China under Grant No. 2018AAA0102701, and in part by the National Natural Science Foundation of China under Grant No. 6217023268.

References

[1] Bowen Baker, Otkrist Gupta, Nikhil Naik, and Ramesh Raskar. Designing neural network architectures using reinforcement learning. In *ICLR*, 2017.

[2] Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc Le. Understanding and simplifying one-shot architecture search. In *ICML*, 2018.

[3] Xiangning Chen and Cho-Jui Hsieh. Stabilizing differentiable architecture search via perturbation-based regularization. In *ICML*, 2020.

[4] Xin Chen, Lingxi Xie, Jun Wu, and Qi Tian. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. In *ICCV*, 2019.

[5] Xiangxiang Chu, Tianbao Zhou, Bo Zhang, and Jixiang Li. Fair DARTS: eliminating unfair advantages in differentiable architecture search. In *ECCV*, 2020.

[6] Xuanyi Dong and Yi Yang. One-shot neural architecture search via self-evaluated template network. In *ICCV*, 2019.

[7] Xuanyi Dong and Yi Yang. Searching for a robust neural architecture in four gpu hours. In *CVPR*, 2019.

[8] Xuanyi Dong and Yi Yang. Nas-bench-201: Extending the scope of reproducible neural architecture search. In *ICLR*, 2020.

[9] Yong Guo, Yaofo Chen, Yin Zheng, Peilin Zhao, Jian Chen, Junzhou Huang, and Mingkui Tan. Breaking the curse of space explosion: Towards efficient nas with curriculum search. In *ICML*, 2020.

[10] 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.

[11] Niall Hurley and Scott Rickard. Comparing measures of sparsity. *IEEE Transactions on Information Theory*, 2009.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

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

[14] Guohao Li, Guocheng Qian, Itzel C Delgadillo, Matthias Muller, Ali Thabet, and Bernard Ghanem. Sgas: Sequential greedy architecture search. In *CVPR*, 2020.

[15] Liam Li and Ameet Talwalkar. Random search and reproducibility for neural architecture search. In *UAI*, 2020.

[16] 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.

[17] 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 *ECCV*, 2018.

[18] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. In *ICLR*, 2019.

[19] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *ECCV*, 2018.

[20] Hieu Pham, Melody Y. Guan, Barret Zoph, Quoc V. Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. In *ICML*, 2020.

[21] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *AAAI*, 2019.

[22] Yao Shu, Wei Wang, and Shaofeng Cai. Understanding architectures learnt by cell-based neural architecture search. In *ICLR*, 2019.

[23] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *CVPR*, 2019.

[24] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. SNAS: Stochastic neural architecture search. In *ICLR*, 2019.

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

[26] Chris Ying, Aaron Klein, Eric Christiansen, Esteban Real, Kevin Murphy, and Frank Hutter. Nas-bench-101: Towards reproducible neural architecture search. In *ICML*, 2019.

[27] Arber Zela, Thomas Elsken, Tonmoy Saikia, Yassine Marrakchi, Thomas Brox, and Frank Hutter. Understanding and robustifying differentiable architecture search. In *ICLR*, 2020.

[28] Arber Zela, Julien Siems, and Frank Hutter. Nas-bench-1shot1: Benchmarking and dissecting one-shot neural architecture search. In *ICLR*, 2020.

[29] Pan Zhou, Caiming Xiong, Richard Socher, and Steven Chu-Hong Hoi. Theory-inspired path-regularized differential network architecture search. In *NeurIPS*, 2020.

[30] Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. In *ICLR*, 2017.

[31] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *CVPR*, 2018.