Variance-stationary Differentiable NAS

Hyeokjun Choe[1]
hyeokjun.choe@gmail.com
Byunggook Na[1]
byunggook.na@gmail.com
Jisoo Mok[1]
jmok908@gmail.com
Sungroh Yoon [*,1,2]
sryoon@snu.ac.kr

[1] Department of Electrical and Computer Engineering
Seoul National University
Seoul, South Korea
[2] Interdisciplinary Program in AI
Seoul National University
Seoul, South Korea
[*] Corresponding Author

Abstract

Differentiable architecture search (DARTS) has become the popular method of neural architecture search (NAS) due to its adaptability and low computational cost. However, following the publication of DARTS, it has been found that DARTS often yields a sub-optimal neural architecture because architecture parameters do not accurately represent operation strengths. Through extensive theoretical analysis and empirical observations, we reveal that this issue occurs as a result of the existence of unnormalized operations. Based on our finding, we propose a novel variance-stationary differentiable architecture search (VS-DARTS), which consists of node normalization, local adaptive learning rate, and $\sqrt{\beta}$-continuous relaxation. Comprehensively, VS-DARTS makes the architecture parameters a more reliable metric for deriving a desirable architecture without increasing the search cost. In addition to the theoretical motivation behind all components of VS-DARTS, we provide strong experimental results to demonstrate that they synergize to significantly improve the search performance. The architecture searched by VS-DARTS achieves the test error of 2.50% on CIFAR-10 and 24.7% on ImageNet.

1 Introduction

Neural architecture search (NAS) is drawing a considerable amount of attention from the machine learning society as an attractive alternative to hand-crafting a neural architecture [12, 14, 23, 24]. Among various NAS algorithms, DARTS [12], a gradient descent optimization-based search algorithm, has become the most widely benchmarked search algorithm due to its low computational cost and intuitive methodological approach. Despite the popularity of DARTS, it has been reported that DARTS often searches for a sub-optimal candidate architecture [3, 10, 16, 20].

Wang *et al*. [16] argue that the failure cases of DARTS occur because architecture parameters, on which the architecture selection rule of DARTS is based, do not faithfully reflect the true operation strengths. Thus, Wang *et al*. propose a new perturbation-based selection rule, which does not use architecture parameters. However, the perturbation-based selection requires progressive tuning, which results in a significant increase in the search cost. If the

© 2021. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.

search process can be fixed, so that the architecture parameters function as a reliable metric without additional computational cost, it becomes a more desirable solution for aligning the search process and the selection rule of DARTS.

In this work, we reveal an undiscovered factor that makes the architecture parameters unreliable for the selection rule: unnormalized outputs of intermediate nodes in the super-net. We provide extensive theoretical and empirical analyses to show how unnormalized outputs prevent the architecture parameters from accurately representing operation strengths. To address the problem of unnormalized outputs, we propose **node normalization**. Applying node normalization produces an undesirable side-effect, named the gradient imbalance problem, which is resolved with layer-wise adaptive control (LARC) [19]. We also introduce $\sqrt{\beta}$-continuous relaxation to improve the training stability of the proposed method. Our method that encompasses all of the amendments is named Variance-stationary DARTS (VS-DARTS); the differences between DARTS and VS-DARTS are illustrated in Figure 1. VS-DARTS focuses on making appropriate changes to DARTS, such that architecture parameters can function as intended.

VS-DARTS improves the search performance of DARTS by 0.57%p and that of DARTS +PT [16] by 0.18%p, achieving a competitive test accuracy on the CIFAR-10 dataset. Furthermore, the standard deviation of test errors is reduced in VS-DARTS, which indicates the robustness of the search performance. When evaluated across different datasets and search spaces, VS-DARTS once again consistently searches for a successful neural architecture.

2 Background and Related Works

2.1 Previous Works on Differentiable NAS

Automation of the neural architecture search process has continuously attracted interest, but wide search space and high evaluation cost require high computational resources[11, 15, 23, 24]. After the cell-based search space[14] and parameter sharing[14] are announced, one-shot neural architecture search (NAS) methods, which train supernet to search and evaluate candidate architectures simultaneously, are widely researched[6, 12, 17]. One-shot NAS can be categorized by two groups: 1) differentiable architecture search which train supernet with continuously relaxed seach space[1, 2, 3, 10, 12, 16, 18, 20] and 2) sampling-based methods which sample the candidate architecture and stochastically update the supernet[6, 17, 22].

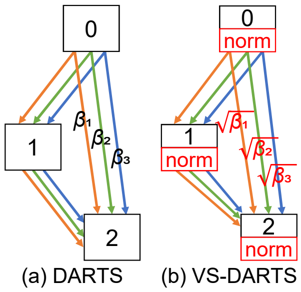


Figure 1: Overview of (a) DARTS and (b) VS-DARTS (proposed) which includes node normalization and $\sqrt{\beta}$-continuous relaxation, depicted in red colors.

DARTS [12],which is the mile-stone research in the first category, has become the most preferred baseline search algorithm because of its simple and intuitive methodological approach. Even though DARTS shows an impressive performance and computational efficiency, it has been reported that DARTS lacks stablility [2, 20] and often derives a sub-optimal architecture even in the search space where all possible architectures are evaluated [7]. By restricting the search space of DARTS, Zela *et al.* [20] empirically demonstrated the failure cases of DARTS. To make DARTS more robust against such failure cases, Zela *et al.* introduces a regularization term to the inner loop of the bi-level optimization. In the case of the study of Chen *et al.* [2], the instability issue of DARTS is addressed through the learning of weight parameters with respect to the perturbed architecture parameters.

Also, it has been empirically reported that the large number of skip connections in the final architecture derivation step results in a significant performance drop [3, 10, 20]. Chu *et al.* [4] argues that the skip connection operation in fact is performing two roles simultaneously: aiding the training of super-net and the candidate operation. Chu *et al.* separates these two roles of the skip connection operation from each other. However, they do not provide an additional analysis into the architecture selection rule based on the magnitude of α.

2.2 Preliminaries on DARTS

Differentiable architecture search (DARTS)[12] aims to search for a cell, which is repeatedly stacked to build the final architecture. In DARTS, two types of a cell are searched: a normal cell and a reduction cell. While the normal cell maintains the dimension of the input feature map, the reduction cell halves it and doubles the number of channels. These cells are represented as a directed acyclic graph (DAG) with N nodes and E edges, which are associated with feature maps and operations, respectively. The intermediate nodes of a cell are computed by summing up all preceding nodes: $x_j = \sum_{i<j} o^{(i,j)}(x_i)$. The output of a cell is computed by concatenating all intermediate nodes.

DARTS starts by constructing a super-net, in which each edge includes all candidate operations in the search space. To enable gradient-based optimization, DARTS continuously relaxes the discrete choice of an operation by applying the Softmax function over all candidate operations with the architecture parameters α:

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

where i and j indicate node indices. $\bar{o}$, the weighted summation of candidate operations, is referred to as a mixed operation. An example of a continuously relaxed cell in DARTS is illustrated in Figure 1(a). By applying the continuous relaxation scheme, the problem of architecture search can be equated with the following bi-level optimization problem:

$$\min_{\alpha} \mathcal{L}_{val}(w^*(\alpha), \alpha) \qquad \text{s.t.} \quad w^*(\alpha) = \text{argmin}_w \mathcal{L}_{train}(w, \alpha). \tag{2}$$

During the search process, architecture parameters α and operation weights w are updated alternatingly. After the search process, the operation associated with the maximum β is selected per edge. Afterwards, the final architecture is derived by selecting top-2 edges per intermediate node by ranking the edges according to β of the selected operation.

2.3 Issue of DARTS Architecture Parameter

Previous works mainly attributed the DARTS failure to the optimization of super-net, but the problem of the architecture selection process has largely been overlooked [16]. In this section, we discuss Wang *et al.* [16], which is the one of the first works to address the problem of architecture parameters . Wang *et al.* demonstrate why the magnitude of architecture parameters in the DARTS formulation is not suitable for indicating the contribution of each operation to the super-net's performance. According to Wang *et al.*, when the problem of

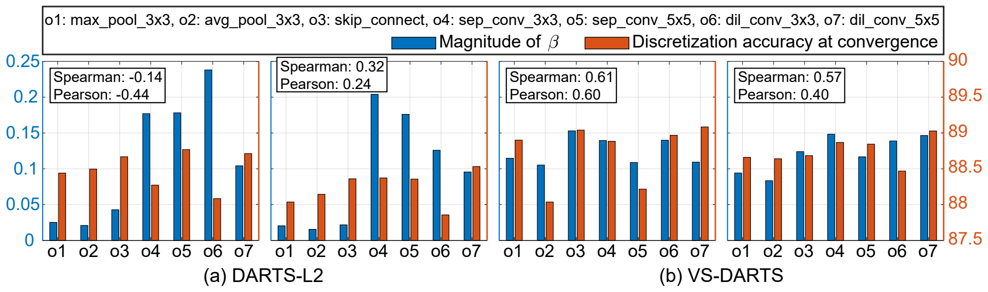


Figure 2: The magnitude of β vs. discretization accuracy (%) at convergence of 2 edges from a trained (a) DARTS-L2 supernet and (b) VS-DARTS supernet. While optimized β of the VS-DARTS supernet faithfully reflects the discretization accuracy of the corresponding operation, *i.e.*operation strength, that of the DARTS-L2 supernet fails to do so.

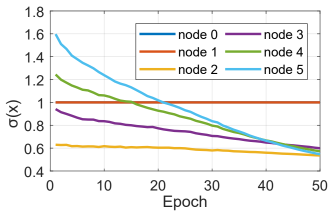


Figure 3: Change in the variance of the each node output feature map during the DARTS search process.

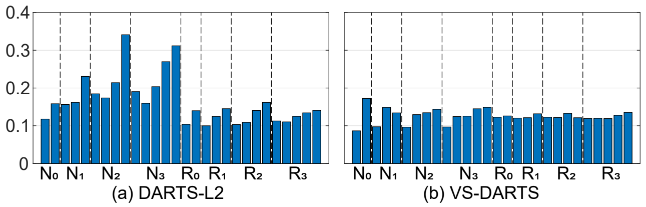


Figure 4: The magnitude of β_{zero} of (a) DARTS-L2 and (b) VS-DARTS. The bar graphs in N_i and R_i correspond to the incoming edges into the i-th node of the normal cell and of the reduction cell, respectively.

skip operation domination occurs in the hastily optimized super-net, the architecture parameters are not aligned with the discretization accuracy, *i.e.*the accuracy of the trained super-net when a single operation is selected for a randomly selected edge, while all other edges remain the same. Wang *et al.* thus proposed a new selection rule that re-evaluates the super-net after removing a single operation per edge and selects the operation that yields the largest drop in the super-net's validation accuracy upon removal; this selection rule is coined the perturbation-based selection rule (DARTS+PT) and does not directly utilize architecture parameters. However, to successfully execute the proposed perturbation-based selection rule, it requires the progressive tuning process that incurs a considerable amount of additional computation cost.

3 Lack of Reliability of β

From here on, we refer to the inability of β to appropriately reflect the operation strengths as *unreliability of* β. Eliminating the unreliability of β is crucial for improving the accuracy of the selection rule based on the magnitude of β. If we can improve the reliability of β, it becomes unnecessary to employ additional techniques, such as progressive tuning [16], which causes a non-negligible increase in the search cost, during the search process.

As mentioned in Section 2.3, if the input feature maps to each operation are close to being optimal, β_{skip} begins to grow larger than β of other operations. However, we demonstrate that the previously-observed dominance of skip connection caused by the optimized input feature maps is insufficient for explaining the unreliability of β. We train a DARTS super-net with strong regularization (DARTS-L2) by increasing the weight decay value from 3e-4 to 5e-3, such that the feature maps do not hastily converge to optimal in the middle of the search process. Figure 2 shows the magnitude of β and the operation strength in 2 randomly selected edges of the super-net, where the operation strength is represented with the discretization accuracy at convergence [16]. As can be seen from Figure 2(a), even after increasing the regularization effect, the magnitude of β and the operation strength in the DARTS-L2 super-net are not aligned, indicating that the unreliability of β still remains.

We further analyze the computational flow in the DARTS super-net to reveal another reason for **unreliability of β problem**. According to Eq.(1), the influence of an operation on the $\bar{o}$ can be estimated by $\beta_o o(x)$. Therefore, for β to be a trust-worthy measure of the operation strength, the outputs from $o(x)$ must be scaled to a similar range of values. However, as an example, when the scale of $o(x)$ is far smaller than 1, o has a smaller influence over $\bar{o}$, even if β_o is the highest among β of candidate operations.

We observe that in DARTS, the scales of $o_{\text{skip}}(x)$ differs from those of the other operations. We use the variance of outputs from $o(x)$ to represent their scales, since they have the mean of zero. Figure 3 shows the output feature map variance for each node of a randomly selected cell during DARTS super-net training. We find that while the outputs of other operations are normalized to $(0,1)$ due to their last normalization layer, the output of the skip operation is not normalized. There exist two reasons why the output of the skip operation cannot normalized in DARTS: 1) the input feature map to the skip operation is the output of the previous node, which is unnormalized as shown in Figure 3 and 2) the skip operation itself does not have the ability to normalize it. If nodes remain unnormalized as in DARTS, to select an operation based solely upon the magnitude of β becomes unreliable and thus results in a sub-optimal architecture. Therefore, we proposed VS-DARTS, which is described in Section 4, and VS-DARTS makes β more reliable as shown in Figure 2(b).

4 Variance-stationary DARTS (VS-DARTS)

4.1 Node Normalization

To remedy the problems caused by the unnormalized nodes, we propose a simple yet effective method of normalizing the output of every intermediate node as follows:

$$\hat{x_j} = \frac{x_j - \mu_{x_{j,B}}}{\sigma_{x_{j,B}}}, \text{ where } x_j = \sum_{i<j} \bar{o}^{(i,j)}(x_i), \tag{3}$$

$\hat{x_j}$ is the normalized output feature map of node N_j, and $\mu_{x_{j,B}}$ and $\sigma^2_{x_{j,B}}$ are the mean and the variance of x_j within a single mini-batch, respectively. Applying the proposed node normalization to DARTS prevents the input to the skip operation from being unnormalized. Consequently, by scaling outputs of $o(x)$ to a similar range, node normalization contributes to amending the unreliability of β.

Node normalization additionally addresses the problem of suppression by zero operation. The zero operation was initially introduced to indicate the lack of connection between two nodes [12]. Against the initial expectation, the zero operation is in fact performing the role of scaling output feature maps in DARTS. As shown in Figure 4(a), without node normalization, the average β_{zero} of incoming edges grows large to make the variances of each intermediate

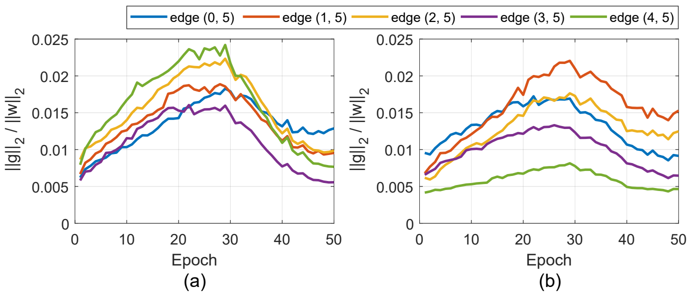


Figure 5: The change in the scale of gradient of 3x3 separable convolution operation at last intermediate node (a) before and (b) after applying node normalization.

node similar. The large value of β_{zero} in turn suppresses β of other operations, which we call suppression by zero. The unreliability of β exacerbated by this phenomenon. It can be clearly observed from Figure 4(b) that applying node normalization effectively prevents β_{zero} from exploding. Under the architecture selection rule based on the magnitude of β, we believe that node normalization is crucial for DARTS-based algorithms to function properly.

4.2 Remedying Gradient Imbalance

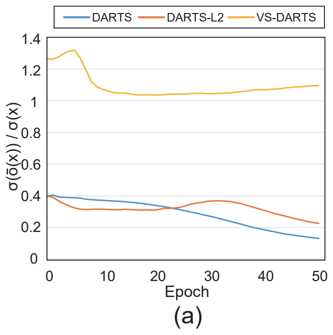
While node normalization is effective at making β reliable, from the perspective of supernet training, it has a minor side-effect: the gradient imbalance problem. In Figure 5(a) and (b), we visualize the gradients with respect to the weights of the 3x3 separable convolution operation before and after applying node normalization to empirically support the occurrence of the gradient imbalance problem. In Figure 5(a), the amount of gradient against weight of each edges are similar scale in early epochs, while not in Figure 5(b). If a different amount of gradient signal is delivered to each edge, operations in the edge with large gradient will likely converge faster than others. Therefore, the scale of gradient should be similar in the early stage of search for fair competition among different edges. The reason of the gradient imbalanced problem is that node normalization change the scale of node output feature maps.

Node normalization scales the output feature maps using their *sigma* as in Eq.(3), and scaling the feature maps also scales the gradient with respect to the corresponding feature maps. For instance, as in Figure 3, when σ of the unnormalized feature map is 1.6, the gradient is divided by 1.6 and becomes smaller; on the contrary, when σ is 0.6, the gradient becomes larger. This results in the gradient becoming unbalanced across incoming edges of a single intermediate node.

We deal with the gradient imbalance problem by applying the operation-wise adaptive learning rate. This adaptive learning rate scheme is realized by Layer-wise Adaptive Rate Control (LARC) [19]. Through LARC, the local learning rate for each operation is computed by multiplying the global learning rate by $||w||/||g||$, eliminating the above issue entirely. Therefore, LARC resolves the gradient imbalance problem caused by node normalization, and applying node normalization and LARC together is important for improving the reliability of β.

4.3 $\sqrt{\beta}$-Continuous Relaxation

Aside from improving the reliability of β, we propose an additional method to improve the search performance of DARTS: $\sqrt{\beta}$-continuous relaxation. Figure 6(a) shows the change



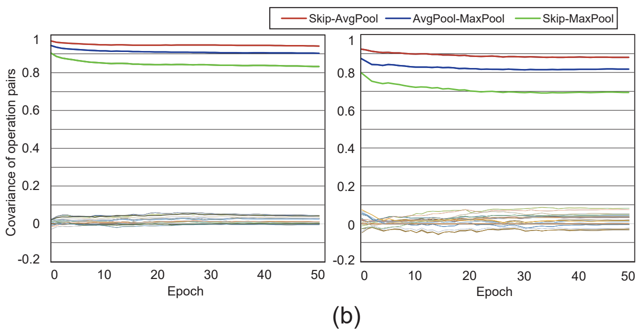


Figure 6: (a) The ratio of variance of the output of the mixed operation to that of the input of the mixed operation in a randomly selected edge. (b) Covariances between each pair of operations in the search process using DARTS with node normalization.

in $\sigma(\bar{o}(x))/\sigma(x)$, where $\sigma^2(\bar{o}(x))$ and $\sigma^2(x)$ denote the variance of the output and input of the mixed operation respectively, as the search process proceeds. We observe that the ratio starts with 0.4 and decreases monotonically in DARTS. Through this observation, we want to answer the following questions: 1) why the scale of the output feature map is smaller than that of the input and 2) why the ratio of the output to the input varies in the mixed operation with the conventional DARTS continuous relaxation.

To address the first question, let us denote the variance of a feature map generated by the mixed operation as $\sigma^2(\bar{o}(x))$ which is expressed as:

$$\sigma^2(\bar{o}(x)) = \sum_{o \in O} \beta_o^2 \sigma^2(o(x)) + \sum_i \sum_{j \neq i} \beta_{o_i} \beta_{o_j} Cov(o_i(x), o_j(x)), \tag{4}$$

where *Cov* denotes the covariance between the feature maps generated by two different operations within the mixed operation. We assume that every output feature map of the operations is normalized. The ratio of the variance of the output to that of the input of the mixed operation is 1 if and only if all of the covariances are equal to 1. We empirically show that this condition is never satisfied by plotting the covariances between each pair of operations in DARTS with node normalization in Figure 6(b). The covariances between the unparametrized operations stay above 0.7 throughout the search process, because the feature maps of unparametrized operations are highly correlated. On the contrary, covariances between all operations and parametrized operations remain close to zero as a result of the negligible correlation between them. This observation provides an answer to the first question.

Based on Figure 6(b), we also find that the covariances between unparametrized operations become stationary approximately after the first 10 epochs. Once the covariances between unparametrized operations become stabilized, the covariance term in Eq.(4) can be treated as a constant C. Therefore, with the proposed node normalization, $\sigma^2(\bar{o}(x))$ can now be approximated as:

$$\sigma^2(\bar{o}(x)) \approx \sum_{o \in O} \beta_o^2 \sigma^2(o(x)) + C \approx \sum_{o \in O} \beta_o^2 + C. \tag{5}$$

During the search process, $\sigma^2(\bar{o}(x))$ is bound to change because values of β no longer follow the uniform distribution and start to vary. Consequently, in the mixed operation with node normalization, although the variance of the input feature map $\sigma^2(x)$ is equal to 1 because of node normalization, the variance of the output feature map $\sigma^2(\bar{o}(x))$ changes during the search process. This observation explains why the ratio of the output to the input varies in the mixed operation, providing an answer to the second question.

Batch normalization is a commonly adopted technique to make the ratio between the input and the output stable, and it also keeps the input and the output at a similar level. However, directly applying batch normalization causes the unreliability of β because of re-scaling; the detailed discussion is provided in the Supplementary Materials. To aid training of the super-net by mimicking the effect of batch normalization, we instead propose $\sqrt{\beta}$-continuous relaxation that substitutes β in the continuous relaxation with $\sqrt{\beta}$. Let us denote the mixed operation with $\sqrt{\beta}$-continuous relaxation as $\tilde{o}(x)$. Then, Eq.(5) is modified as:

$$\sigma^2(\tilde{o}(x)) \approx \sum_{o\in O} \sqrt{\beta_o}^2 \sigma^2(o(x)) + C \approx \sum_{o\in O} \sqrt{\beta_o}^2 + C = \sum_{o\in O} \beta_o + C = 1 + C. \tag{6}$$

Eq.(6) indicates that $\sigma^2(\tilde{o}(x))$ is maintained as a constant 1+C, while $\sigma^2(x)$ is 1. If C is kept small, then $\sqrt{\beta}$-continuous relaxation can obtain the effect of batch normalization while avoiding another cause of the unreliability of β problem. This property is empirically supported by Figure 6(a). In VS-DARTS that includes $\sqrt{\beta}$-continuous relaxation, this ratio remains fairly stationary after the early stages in the search process, while in others, it becomes quite volatile.

5 Experiments

5.1 Results in DARTS Search Space

5.1.1 Settings

All of our experiments are conducted using a single NVIDIA Tesla V100 GPU. For the search process, following Zela *et al.* [20], we use a larger weight decay factor than default DARTS [12] to account for the significant performance drop caused by the increase in the number of skip connections. The remaining hyperparameters of the search process are kept the same as those of default DARTS [12]. The searched architecture is retrained for final evaluation, which is the standard practice in NAS. We follow the experimental protocols of default DARTS [12] for the hyperparameters of the retrain process. More detailed hyperparameter settings are included in the Supplementary Materials. The cells searched by VS-DARTS are also visualized in the Supplementary Materials.

5.1.2 Results

In Table 1, we compare the search performance of VS-DARTS on the CIFAR-10 dataset against that of various NAS Algorithms. The search performance comparison for the ImageNet dataset is provided in Table 2.

CIFAR-10: To validate the stability of the search performance, we repeat the search and retrain processes three times with different seeds; hence, we report the average performance of our searched architectures with its standard deviation. The architectures searched by VS-DARTS are depicted in the Supplementary Materials. As shown in Table 1, the performance of VS-DARTS architectures is comparable to the state-of-the-art performance.

ImageNet: To evaluate the performance of the VS-DARTS architecture on ImageNet, we transfer the architecture that yields the lowest test error on CIFAR-10 to ImageNet. As

Table 1: Comparison of architectures searched by various NAS algorithms on CIFAR-10. Cost refers the search cost with GPU days.

Method	Params (M)	Test Err. (%)	Cost
NASNet-A[24]	3.3	2.65	2000
ENAS[14]	4.6	2.89	0.5
AmoebaNet-B[15]	2.8	2.55	3150
PNAS[11]	4.6	3.41	225
DARTS^{1st}[12]	3.3	3.00	1.5
DARTS^{2nd}[12]	3.3	2.76	4.0
SNAS[17]	2.8	2.85 ± 0.30	1.5
GDAS[6]	3.4	2.93	0.2
P-DARTS[1]	3.4	2.50	0.2
PC-DARTS[18]	3.6	2.57	0.1
R-DARTS[20]	-	2.95 ± 0.21	1.6
FairDARTS[3]	3.3 ± 0.5	2.54 ± 0.05	0.4
DARTS-[4]	3.5	2.59 ± 0.08	0.4
PR-DARTS[22]	3.4	2.32	0.17
DARTS+PT[16]	3.0	2.61 ± 0.08	0.8
VS-DARTS (avg)	3.4 ± 0.3	2.50 ± 0.05	0.4
VS-DARTS (best)	3.38	2.43	0.4

Table 2: Performance comparison of architectures on ImageNet, where the all architectures are searched on CIFAR-10 and then transferred to ImageNet classification task.

Method	Params	FLOPs	Test Err. (%)	
	(M)	(M)	Top1	Top5
MobileNet[8]	4.2	569	29.4	10.5
ShuffleNet2×(v2)[21]	∼5	524	25.1	-
NASNet-A[24]	5.3	564	26.0	8.4
PNAS[11]	5.1	588	25.8	8.1
DARTS^{2nd}[12]	4.7	574	26.7	8.7
SNAS (mild)[17]	4.3	522	27.3	9.2
PC-DARTS[18]	5.3	586	25.1	7.8
GDAS[6]	5.3	581	26.0	8.5
P-DARTS[1]	4.9	557	24.4	7.4
FairDARTS-B[3]	4.8	541	24.9	7.5
PR-DARTS[22]	5.0	543	24.1	7.3
DARTS+PT[16]	4.6	-	25.5	8.0
VS-DARTS	5.3	589	24.8	7.5
VS-DARTS +Proxy[13]	5.7	640	24.7	7.7

Table 3: Performance comparison (test error (%)) across three datasets and four search spaces, which are constrained from the cell-based search space [20]. †: evaluated ourselves while settings are the same as cited paper.

Benchmark		DARTS	R-DARTS(L2)	DARTS-ES	DARTS-ADA	PR-DARTS† [22]	DARTS+PT [16]	VS-DARTS (ours)
C10	S1	3.84	**2.78**	3.01	3.10	3.26	3.50	**2.58**
	S2	4.85	3.31	3.26	3.35	3.63	**2.79**	**2.57**
	S3	3.34	**2.51**	2.74	2.59	3.99	**2.49**	2.52
	S4	7.20	3.56	3.71	4.84	**2.59**	2.64	**2.49**
C100	S1	29.46	24.25	28.37	**24.03**	32.74	24.48	**23.87**
	S2	26.05	**22.24**	23.25	23.52	32.52	23.16	**21.49**
	S3	28.90	23.99	23.73	23.37	26.96	**22.03**	**22.03**
	S4	28.85	21.94	21.26	23.20	**20.80**	**20.80**	**20.69**
SVHN	S1	4.58	4.79	2.72	**2.53**	3.33	2.62	**2.37**
	S2	3.53	**2.51**	2.60	2.54	4.48	2.53	**2.37**
	S3	3.41	2.48	2.50	2.50	2.75	**2.42**	**2.37**
	S4	3.05	2.50	2.51	2.46	2.95	**2.42**	**2.32**

shown in Table 2, the performance of the searched architecture achieves is comparable to that of the state-of-the-art architecture. We highlight that VS-DARTS achieves 1.9%p improvement from DARTS, the baseline for our method, and 0.7%p improvement from DARTS+PT [16]. When directly searching an architecture on ImageNet using proxy dataset [13], VS-DARTS yields a test error of 24.7%, which is comparable to the recently searched architectures.

5.2 Results in RobustDARTS Search Space

We further evaluate the search performance of VS-DARTS across three datasets, CIFAR-10, CIFAR-100, and SVHN, and four search spaces denoted by S1∼S4 [20]. These search spaces are designed to evaluate the robustness of the search algorithm against changes in candidate operations. We followed the experimental protocols of Zela *et al.* [20]. The evaluation process for the CIFAR-10 dataset is identical to the one described in Section 5.1. For CIFAR-100 and SVHN datasets, we use the architecture that consists of eight cells, *i.e.*, six normal cells and two reudction cells.

The search results are reported in Table 3. VS-DARTS clearly outperforms recent DARTS variants, indicating that in VS-DARTS, the selection rule based on the magnitude of β can function properly to derive the optimal architecture. In particular, on S4, which includes the meaningless noise operation as a candidate operation, the cells searched by VS-DARTS do not contain any noise operation at all, while those of other NAS algorithms [4, 20] do contain some amount of noise operations.

5.3 Ablation Study

In this section, we provide a comprehensive ablation study of VS-DARTS to demonstrate the contribution of each added component to the increase in search performance. The experimental settings are as same as Section 5.1. In Table 4, we report the search performance in test accuracy after applying node normalization, LARC, and $\sqrt{\beta}$. Just by adding node normalization, the test accuracy is improved by 0.2%p (97.09% $\rightarrow$ 97.29%). However, because the problem of imbalanced gradient remains unaddressed, the search performance only with adding node normalization has room for improvement. Applying LARC and introducing $\sqrt{\beta}$ each brings upon 0.31%p (97.09% $\rightarrow$ 97.40%) and 0.48%p (97.09% $\rightarrow$ 97.57%) improvement in test accuracy. Our ablation results support that all of the applied techniques are crucial for substantially improving the search performance.

NodeNorm	LARC	$\sqrt{\beta}$	Test Acc.
✗	✗	✗	97.09%
✓	✗	✗	97.29%
✓	✓	✗	97.40%
✓	✓	✓	97.57%

Table 4: Ablation studies

6 Conclusion

In this study, we revealed that unnormalized node in a continuously-relaxed cell leads DARTS to yield a sub-optimal neural architecture because the architecture parameters do not accurately represent operation strengths. Node normalization was proposed as a simple yet effective solution to address this issue. After applying node normalization, we found that the gradient imbalance problem becomes prominent, and thus, to remedy this problem, the local adaptive learning rate strategy was utilized. Lastly, to further stabilize training of the super-net, we newly introduced $\sqrt{\beta}$-continuous relaxation, which makes the scales of the input and the output feature maps to be similar. We provided through theoretical analysis and empirical results to support the effectiveness of each component. By combining all the components, VS-DARTS successfully searched for a competitive architecture on CIFAR-10. This study alerts that when constructing a search space, the influence of normalization or lack there of must be carefully considered.

Acknowledgements

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) [NO.2021-0-01343, Artificial Intelligence Graduate School Program (Seoul National University)], National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) [2018R1A2B3001628], Samsung Electronics Co., Ltd, Samsung Research Funding & Incubation Center of Samsung Electronics under Project Number SRFC-IT1901-12, and the Brain Korea 21 Plus Project in 2021.

References

[1] X. Chen, L. Xie, J. Wu, and Q. Tian. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1294–1303, 2019. doi: 10.1109/ICCV.2019.00138.

[2] Xiangning Chen and Cho-Jui Hsieh. Stabilizing differentiable architecture search via perturbation-based regularization. 2020.

[3] Xiangxiang Chu, Tianbao Zhou, Bo Zhang, and Jixiang Li. Fair DARTS: Eliminating Unfair Advantages in Differentiable Architecture Search. In *16th Europoean Conference On Computer Vision*, 2020. URL https://arxiv.org/abs/1911.12126.pdf.

[4] Xiangxiang Chu, Xiaoxing Wang, Bo Zhang, Shun Lu, Xiaolin Wei, and Junchi Yan. Darts-: Robustly stepping out of performance collapse without indicators. 2021.

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

[6] 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.

[7] Xuanyi Dong and Yi Yang. Nas-bench-201: Extending the scope of reproducible neural architecture search. In *International Conference on Learning Representations (ICLR)*, 2020. URL https://openreview.net/forum?id=HJxyZkBKDr.

[8] 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.

[9] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015.

[10] 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.

[11] 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.

[12] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *International Conference on Learning Representations*, 2019.

[13] Byunggook Na, Jisoo Mok, Hyeokjun Choe, and Sungroh Yoon. Accelerating neural architecture search via proxy data. In *International Joint Conference on Artificial Intelligence*, 2021.

[14] Hieu Pham, Melody Y. Guan, Barret Zoph, Quoc V. Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4095–4104, 2018.

[15] 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.

[16] Ruochen Wang, Minhao Cheng, Xiangning Chen, Xiaocheng Tang, and Cho-Jui Hsieh. Rethinking architecture selection in differentiable nas. In *International Conference on Learning Representations*, 2021.

[17] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. Snas: Stochastic neural architecture search. In *International Conference on Learning Representations*, 2019.

[18] 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 *International Conference on Learning Representations*, 2020.

[19] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

[20] Arber Zela, Thomas Elsken, Tonmoy Saikia, Yassine Marrakchi, Thomas Brox, and Frank Hutter. Understanding and robustifying differentiable architecture search. In *International Conference on Learning Representations*, 2020.

[21] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6848–6856, 2018.

[22] Pan Zhou, Caiming Xiong, Richard Socher, and Steven Chu Hong Hoi. Theory-inspired path-regularized differential network architecture search. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 8296–8307, 2020.

[23] Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. In *International Conference on Learning Representations*, 2017.

[24] 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.