DARTS-PD: DIFFERENTIABLE ARCHITECTURE SEARCH WITH PATH-WISE WEIGHT SHARING DERIVATION

He Cai* *Zhaokai Zhang** *Tianpeng Feng* *Yandong Guo*†

OPPO Research Institute, Guangdong OPPO Mobile Telecommunications Corp., Ltd.

2022 IEEE International Conference on Image Processing (ICIP) | 978-1-6654-9620-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICIP46576.2022.9897275

ABSTRACT

With the advent of Neural Architecture Search (NAS), auto-designing of artificial neural networks has been made possible. Among various NAS methods, Differentiable Architecture Search (DARTS) has achieved significant progress due to its high calculating efficiency. However, it suffers from poor stability and obvious performance drop because of bi-level optimization and hard pruning. Besides, it only generates one best architecture at once. To alleviate the problems above, we design a three-stage framework with a path-wise weight sharing derivation. We first prune the supernet with differentiable methods to keep top-k operations on each edge instead of one. Then the pruned supernet is trained with our path-wise weight sharing method. At the derivation stage, the best candidate operations are selected with Evolutionary Search based on the validation accuracy of paths. Our weight sharing derivation is proved effective in improving searching stability as well as alleviating the performance drop. Furthermore, it also allows us to search for a large number of architectures with different parameter sizes at one time. Comprehensive experiments on CIFAR-10 and ImageNet show that we manage to find a group of state-of-the-art architectures (97.61% on CIFAR-10 and 76.4% on ImageNet).

Index Terms— Neural Architecture Search, path-wise weight sharing derivation

1. INTRODUCTION

Differentiable architecture search methods have been widely accepted in recent years, including DARTS[1] and its related algorithms. Many of these methods are designed on two mainstream search spaces, namely, cell-based[1, 2, 3] and block-based[4, 5] search space. They have all been proved effective in speeding up the training process and achieving convincing results on CIFAR-10[6] and ImageNet[7].

However, the performance of searched architecture is often unstable and dramatically degraded compared with the supernet. Fair DARTS[8] believes that the performance collapse can be attributed to exclusive competition and thus they make each operation's architectural weights independent of others and also propose a zero-one loss to separate them. DARTS+[9] introduces several 'early stopping' criteria to avoid performance collapse. P-DARTS[3] allows the depth of searched architectures to grow gradually to bridge the gap between search and evaluation accuracy.

Unfortunately, the performance collapse and unfairness at search and derivation stages have not been eliminated. We believe the performance collapse and instability of searching results can be attributed to the improper optimization objective. Besides, the dominant advantage of operations, bi-level optimization[10] during search and hard pruning[11] during derivation also aggravate the problem.

In this paper, we propose a novel three-stage framework combined with path-wise weight sharing derivation to alleviate these problems. It has been proved effective in improving both the performance and stability of chosen architectures. Furthermore, it also allows deriving dozens of architectures with different parameter sizes at one time to satisfy specific hardware requirements.

As is illustrated in Fig.1, a supernet S with parameterized operations is first pruned with the differentiable method. We keep k operations with the highest architectural weights. The network with k operations on each edge is defined as S'. Second, we train S' in a path-wise way. Inspired by path-wise One-shot Architecture Search methods[12, 13], we fairly select training paths. Only model weights are optimized during this phase and architectural weights are frozen. The trained S' is remarked as $S'_{trained}$. Finally, we employ Evolutionary Search (ES)[12] to derive best architectures from $S'_{trained}$. ES helps us to evaluate a large number of derived paths with the limited computational resource.

In summary, the key contributions of this paper are:

- Firstly, we prove the existence of the main factors that lead to the instability and performance drop of traditional differentiable search methods and design a general framework, named DARTS-PD, with path-wise weight sharing derivation that improves the overall performance of gradient-based architecture search methods.
- Secondly, DARTS-PD allows us to find dozens of architectures with different parameter sizes to fulfill various hardware requirements at one time.

[$\star$] Equal contribution

[$\dagger$] Corresponding Author

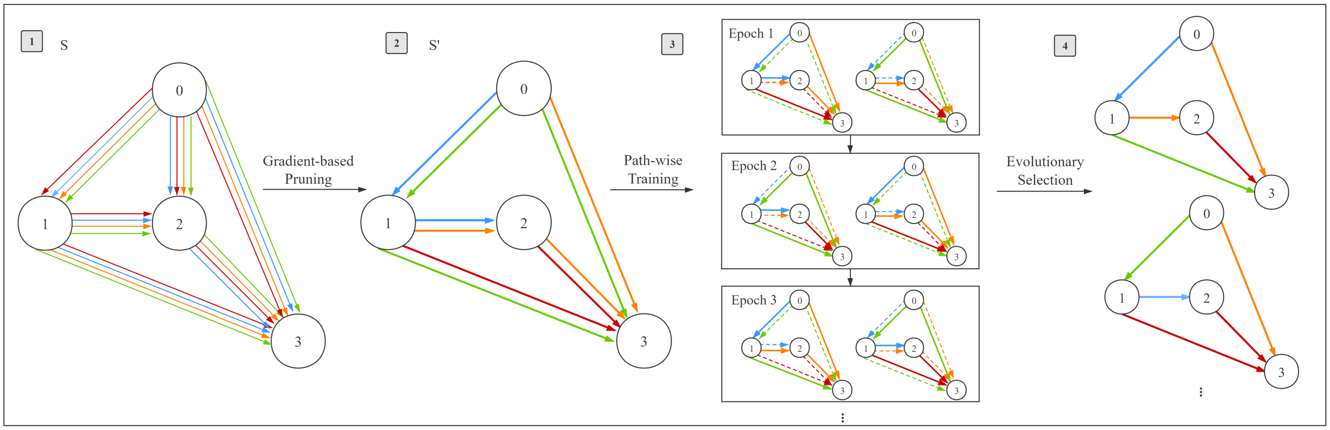

Fig. 1. Path-wise weight sharing derivation in cells: (1)-(2) The supernet S is pruned with gradient-based method. (3) The pruned supernet S' is optimized with our path-wise weight sharing method. (4) Paths are sampled and assessed using ES. Final Best architectures are selected according to their validation performance.

- Thirdly, we demonstrate the improvement of accuracy and stability of DARTS-PD based on comprehensive experiments and find state-of-the-art architectures on both CIFAR-10 (97.61%) and ImangeNet (76.4%).

2. RELATED WORKS

NAS and its related methods aim at looking for the best neural architectures. Up till now, RL-based NAS[14], evolution-based NAS[15] and differentiable NAS[1] are three main branches of NAS methods. It is noteworthy that recent differentiable methods have achieved higher accuracy with less search cost. DARTS[1] and its relative works are among the most representative ones.

DARTS is designed on the cell search space based on the continuous relaxation of the architecture representation. DARTS updates the architectural parameters and weights with joint optimization and the discrete architecture is derived according to the largest architectural weights. PC-DARTS[2], designed based on DARTS, reduces the redundancy by sampling a small part of super-net and largely eliminates the uncertainty of edge selecting by introducing a new set of edge-level parameters. Such differntiable methods can speed up the search process but the bi-level optimization inevitably leads to the instability of architecture search.

Inspired by path-wise search methods, we try to alleviate the problems above by combining path-wise weight sharing training with differentiable methods. Both Single Path One-shot NAS (SPOS)[12] and FairNAS[13] aim to fairly train the supernet by updating a single path at a time. The difference lies in that SPOS selects operations from each layer by uniform sampling while FairNAS follows a strict fair principle to update each operation an equal number of times per training step. However, it is extremely demanding to adopt these methods on our search space[1]. To strictly promise training fairness, we have to train 8^{16} and $(8!)^{16}$ paths respectively. SPOS and FairNAS alleviate the calculating burden by randomly selecting paths. In our work, we reduce our search space to $(k!)^{16}$ with soft pruning. Then, we adopt a novel principle **Path-wise Strict Fairness** to further promise the **fairness of path sampling** and select best-performed paths according to their validation performance with ES.

3. METHODOLOGY

3.1. Rethinking Upon Current Methods

Instability of architecture search in differentiable methods can be attributed to the dominant advantage of operations, bi-level optimization during search, and hard pruning during architecture derivation.

In DARTS, each cell can be represented as a directed acyclic graph (DAG). During the training process, bi-level optimization couples architectural parameters and weights optimizing within one stage:

$$\begin{aligned} \arg\min_{\alpha} \quad & \mathcal{L}_{val}(\omega'(\alpha), \alpha) \\ \boldsymbol{s.t.} \quad & \omega'(\alpha) = \arg\min_{\omega} \mathcal{L}_{train}(\omega, \alpha) \end{aligned} \tag{1}$$

where $\mathcal{L}_{train}$ and $\mathcal{L}_{val}$ denote the training and the validation loss. ω represents the network weights and α serves as the architectural weights. The bi-level optimization inevitably introduces disturbance into architecture search.

The feature map on each node is denoted as:

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

[1] Our search space is made up of 2 kinds of cells (normal cell and reduction cell), two edges on each intermediate node and 8 operations on each edge.

where $x^{(i)}$ is a feature map in convolutional networks and $\bar{o}^{(i,j)}$ denotes candidate operations relaxed to a softmax within the whole operation space O. As a result, the supernet output only serves as a measurement of the performance produced by a mixture of operations.

3.2. Differentiable Architecture Search with Path-wise Weight Sharing Derivation

To overcome the shortcomings mentioned in Section 3.1, we propose a three-stage search framework:

First, we conduct soft pruning on the original supernet S: we select top-k operations instead of one. In DARTS and many of its related methods, operations are parameterized according to their architectural weights α:

$$\bar{o}^{(i,j)}(x) = \sum_{o \in O} \frac{exp(\alpha_o^{(i,j)})}{\sum_{o' \in O} exp(\alpha_{o'}^{(i,j)})} o(x). \tag{3}$$

Each directed edge (i,j) is associated with some operation $\bar{o}^{(i,j)}$ that transforms $x^{(i)}$. If $p(o^{(i,j)} = o^{(i,j)}_{optimal})$ represents the probability of selecting the best-performed operations on an edge between node i and j, to preserve top-k operations (α_1,α_2...α_k) of k largest weights helps to raise p from $p_{top1} = p(o^{(i,j)} = o^{(i,j)}_{\alpha_1})$ to $p_{top1} + \sum_{r=2}^{k} p(o^{(i,j)} = o^{(i,j)}_{\alpha_r})$.

Second, we introduce path-wise weight sharing derivation into pruned supernet S' to observe the k operations on each edge. FairNAS proposed Strict Fairness to promise that each operation is observed for an equal number of times over n trials. To verify the effectiveness of Strict Fairness, we run FairNAS on a 21-layer supernet with 6 choice blocks on each layer. After 250 training epochs, 44939 paths are selected more than once, among which 47 paths are even sampled more than 8 times. To overcome the drawback above, we propose a novel concept **Path-wise Strict Fairness** to ensure that every single path sampled from the supernet is updated the same number of times over N trials. $\{P_1,\ P_2...P_m\}$ denotes m paths from the pruned supernet S'. $T(P)$ denotes the number of times path P is updated during training:

$$p(T(P_1) = T(P_2) = ... = T(P_m)) = 1. \tag{4}$$

Then, S' is updated with accumulated gradients from all paths once during each step:

$$\mathcal{W}_{S'} = \arg\min_{\omega'} \sum_{i=1}^{m} \mathcal{L}_{train}(\mathcal{P}_i(\mathcal{O}_i(x), \omega')), \tag{5}$$

where $m = k^{16}$ denotes the total number of paths [2]. $\mathcal{P}_i$ represents a single path with one operation on each edge. $\mathcal{O}_i$ represents operations selected on each edge on the i^{th} path. ω' denotes model weights of S'.

We finally derive the best architectures according to evaluation accuracy on the validation set with ES:

$$\left[\ o^1_{k_1}, o^2_{k_2}...o^l_{k_l}\ \right] = \arg\max_{o^1,o^2,...o^l \in O} \mathcal{ACC}_{val}(P_{o^1 o^2 ... o^l}(x)) \tag{6}$$

where $\{o^1_{k1}, o^2_{k2}...o^l_{kl}\}$ represents the best operations selected on l edges (one operation on an edge). $P_{o^1 o^2 ... o^l}$ denotes a single path containing l edges.

Our framework works as Algorithm 1.

Algorithm 1 Differentiable NAS with weight sharing derivation

```
Input: supernet S with weights $\omega$, operations with architectural parameters $\mathcal{A}(\alpha)$, training data $\mathcal{D}_{train}$, validatation data $\mathcal{D}_{val}$, test data $\mathcal{D}_{test}$, and k
1.Supernet pruning:
while not terminated do
    update $\omega$ on $\mathcal{D}_{train}$ by descending: $\nabla\omega\mathcal{L}_{train}(\omega(\alpha), \alpha)$
    update $\alpha$ on $\mathcal{D}_{val}$ by descending: $\nabla\alpha\mathcal{L}_{val}(\omega(\alpha), \alpha)$
end
Get $S'$ with top-k operations on each edge.
2.Training of $\omega'$(weights of $S'$):
while not converge do
    sample k paths according to Path-wise Strict Fairness and calculate their gradients on $\mathcal{D}_{train}$: $\nabla\omega' \sum_{i=1}^{k} \mathcal{L}_{train}(\mathcal{P}_i(\mathcal{A}_i, \omega'_i))$
    update $\omega'$ with accumulated gradients of k paths
end
3.Evaluate each path $\mathcal{P}_i$ generated with ES on $\mathcal{D}_{test}$
Output: the best paths $\{\mathcal{P}_1, \mathcal{P}_2...\mathcal{P}_n\}$
```

4. EXPERIMENTS AND RESULTS

We embed our weight sharing derivation in the cell-based search space of PC-DARTS and verify its effectiveness on two mainstream datasets. On both datasets, we manage to derive state-of-the-art architectures.

Datasets. CIFAR-10[6] is made up of 60000 images, all of which are of a spatial resolution of 32 × 32. All 60000 images are distributed over 10 classes, with 50000 in the training set and 10000 in the testing set. ImageNet[7] is a high-resolution dataset with 1000 categories. It is separated into 1300K training images and 50K validation images.

4.1. Experimental Setup on CIFAR-10

Our cell-based search space follows that of PC-DARTS. To make our results comparable with previous works, we adopt preprocessing and training tricks like cutout[22] and dropout[23]. We first conduct supernet pruning with the same setting as the search stage in PC-DARTS except that we keep k=2 operations on each edge instead of one. We include the following operations: 3×3 and 5×5 separable convolutions, 3×3 and 5×5 dilated separable convolutions, 3×3 max pooling, 3×3 average pooling, skip connection and zero.

[2] 8 edges in the normal and reduction cell respectively, with k operations on an edge

Table 1. Comparison of state-of-the-art architectures on CIFAR-10. †: Averaged on models from 3 runs of DARTS-PD

Architecture	Test Acc.(%)	Params(M)	$\times+$(M)	Search Space	Search Method
NASNet-A[14]	97.35	3.3	608	cell	RL
ENAS[16]	97.11	4.6	626	cell	RL
DARTS(2nd order)[1]	97.24±0.09	3.3	528	cell	gradient-based
GDAS[17]	97.07	3.37	519	cell	gradient-based
SNAS[18]	97.15±0.02	2.8	422	cell	gradient-based
SGAS[19]	97.33±0.21	3.9±0.22	640±39	cell	gradient-based
P-DARTS[3]	97.50	3.4	532	cell	gradient-based
PC-DARTS[2]	97.43±0.07	3.6	558	cell	gradient-based
TE-NAS[20]	97.37±0.064	3.8	618	cell	gradient-based
DARTS-PD†(ours)	97.52±0.09	3.7±0.2	580±14	cell	gradient-based
DARTS-PD-a(ours)	**97.61**	3.6	576	cell	gradient-based

Table 2. Comparison of state-of-the-art architectures on ImageNet.

Architecture	Test Acc.(%)		Params	$\times+$	Search
	top-1	top-5	(M)	(M)	Space
MobileNetV2[21]	74.7	92.2	6.9	585	block
MnasNet-92[5]	74.8	92.1	3.9	388	block
FairNAS-A[13]	75.3	92.4	4.6	388	block
SPOS[12]	74.4	91.8	4.5	323	block
ProxylessGPU[4]	75.1	92.4	3.5	465	block
NASNet-A[14]	74.0	91.6	5.3	564	cell
AmoebaNet-A[15]	74.5	92.0	5.1	555	cell
DARTS[1]	73.3	91.3	4.7	574	cell
P-DARTS[3]	75.6	92.6	4.9	557	cell
PC-DARTS[2]	75.8	92.7	5.3	597	cell
TE-NAS[20]	75.5	92.5	5.4	599	cell
DARTS-PD-A(ours)	**76.4**	**92.9**	6.2	707	cell

Then, the pruned supernet S' is trained based on our weight sharing method with a stochastic gradient descent optimizer with a momentum of 0.9 for 600 epochs, with a batch size of 256, a dropout rate of 0.3, and a cutout length of 16. A cosine learning rate decay strategy is employed with an initial learning rate of 0.0025. We also regularize the training with L2 weight decay of 0.0003. To be consistent with the previous works, our setup of stand-alone model training follows that of PC-DARTS. Our final experimental results on CIFAR-10 are shown in Table.1.

4.2. Experimental Setup on ImageNet

First, operations and cells are built and pruned in the same way as the search stage in PC-DARTS on ImageNet except that we keep k=2 operations on each edge instead of one. Second, S' is trained with a stochastic gradient descent optimizer with a momentum of 0.9 for 250 epochs with a batch size of 2048. A cosine learning rate decay strategy is employed with an initial learning rate of 0.045. We also regularize the training with L2 weight decay of 0.00004. Our setup of stand-alone model training on ImageNet also follows that of PC-DARTS. The performance of final stand-alone models on ImageNet is shown in Table.2.

4.3. Results Analysis

Results of our proposed method and other state-of-the-art architectures on CIFAR-10 are listed in Table.1. Our best-performed model is named DARTS-PD-a. Compared to current architectures, we have achieved better overall performance with an average accuracy of 97.52%, 0.28% higher than DARTS and 0.15% higher than TE-NAS. Notably, DARTS-PD-a has achieved an accuracy of 97.61%, 0.11% higher than P-DARTS.

Results on ImageNet are listed in Table.2. Our best-performed architecture DARTS-PD-A has reached the highest top-1 accuracy of 76.4%, 3.1% higher than DARTS, 0.8% higher than P-DARTS, 0.6% higher than PC-DARTS and 0.9% higher than TE-NAS. It is worth noticing that we have also achieved 2.0% higher top-1 accuracy than SPOS and 1.1% higher than FairNAS. We have effectively raised model accuracy and also bridged the performance gap.

5. CONCLUSION

In this work, we prove the existence of the main factors that result in the instability and performance decay of most differentiable search methods. Then, we propose a three-stage framework, including differentiable soft pruning, path-wise weight sharing derivation, and path evaluation with ES. Finally, comprehensive experiments are conducted on the cell-based search space. Experimental results show that our framework is effective in finding state-of-the-art architectures as well as alleviating the instability and performance collapse. In the future, we will work on and transfer our framework to gradient-based methods on other search spaces.

6. REFERENCES

[1] Hanxiao Liu, Karen Simonyan, and Yiming Yang, "Darts: Differentiable architecture search," in *International Conference on Learning Representations*, 2018.

[2] 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*, 2019.

[3] Xin Chen, Lingxi Xie, Jun Wu, and Qi Tian, "Progressive darts: Bridging the optimization gap for nas in the wild," *International Journal of Computer Vision*, vol. 129, no. 3, pp. 638–655, 2021.

[4] Han Cai, Ligeng Zhu, and Song Han, "Proxylessnas: Direct neural architecture search on target task and hardware," in *International Conference on Learning Representations*, 2018.

[5] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le, "Mnasnet: Platform-aware neural architecture search for mobile.," in *CVPR*, 2019.

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

[7] 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*. Ieee, 2009, pp. 248–255.

[8] Xiangxiang Chu, Tianbao Zhou, Bo Zhang, and Jixiang Li, "Fair darts: Eliminating unfair advantages in differentiable architecture search," in *European conference on computer vision*. Springer, 2020, pp. 465–480.

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

[10] Benoît Colson, Patrice Marcotte, and Gilles Savard, "An overview of bilevel optimization," *Annals of operations research*, vol. 153, no. 1, pp. 235–256, 2007.

[11] Asaf Noy, Niv Nayman, Tal Ridnik, Nadav Zamir, Sivan Doveh, Itamar Friedman, Raja Giryes, and Lihi Zelnik, "Asap: Architecture search, anneal and prune," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 493–503.

[12] Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, and Jian Sun, "Single path one-shot neural architecture search with uniform sampling," in *European Conference on Computer Vision*. Springer, 2020, pp. 544–560.

[13] Xiangxiang Chu, Bo Zhang, and Ruijun Xu, "Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12239–12248.

[14] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le, "Learning transferable architectures for scalable image recognition," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, 2018, pp. 8697–8710.

[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*, 2019, vol. 33, pp. 4780–4789.

[16] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean, "Efficient neural architecture search via parameters sharing," in *International Conference on Machine Learning*. PMLR, 2018, pp. 4095–4104.

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

[18] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin, "Snas: stochastic neural architecture search," in *International Conference on Learning Representations*, 2018.

[19] Guohao Li, Guocheng Qian, Itzel C Delgadillo, Matthias Muller, Ali Thabet, and Bernard Ghanem, "Sgas: Sequential greedy architecture search," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1620–1630.

[20] Wuyang Chen, Xinyu Gong, and Zhangyang Wang, "Neural architecture search on imagenet in four gpu hours: A theoretically inspired perspective," in *International Conference on Learning Representations (ICLR)*, 2021.

[21] Mark Sandler, Andrew G Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *CVPR*, 2018.

[22] Terrance DeVries and Graham W Taylor, "Improved regularization of convolutional neural networks with cutout," 2017.

[23] Yarin Gal and Zoubin Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," *Advances in neural information processing systems*, vol. 29, 2016.