Operation and Topology Aware Fast Differentiable Architecture Search

Shahid Siddiqui[1,2], Christos Kyrkou[1], Theocharis Theocharides[1,2]
[1]*KIOS Research and Innovation Center of Excellence*
[2]*Department of Electrical and Computer Engineering*
University of Cyprus
Nicosia, Cyprus
{msiddi01,kyrkou.christos,ttheocharides}@ucy.ac.cy

***Abstract*—Differentiable architecture search (DARTS) has gained significant attention amongst neural architecture search approaches due to its effectiveness in finding competitive network architectures with affordable computational complexity. However, DARTS' search space is designed such that even a randomly sampled architecture performs reasonably well. Moreover, due to the complexity of search architectural building block or cell, it is unclear whether these are certain operations or the cell topology that contributes most to achieving higher final accuracy. In this work, we dissect the DARTS's search space to understand which components are most effective in producing better architectures. Our experiments show that: (1) Good architectures can be discovered regardless of the search network depth; (2) Seperable convolution with 3x3 kernel is the most effective operation in this search space; and (3) The cell topology also has substantial effect on the accuracy. Based on these insights, we propose an efficient search approach referred to as eDARTS, which searches on a pre-specified cell having good topology with increased attention to important operations, using a shallow search supernet. Moreover, we propose some optimizations for eDARTS that significantly speed up the search as well as alleviate the well known skip connection aggregation problem of DARTS. eDARTS achieves an error rate of 2.53% on CIFAR-10 using a 3.1M parameters model whereas the search cost is less than 30 minutes.**

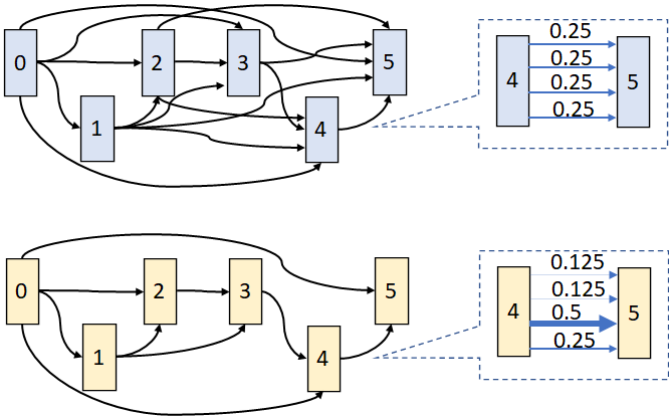

Fig. 1: DARTS' search cell with complex topology and equal attention to all operations (top) vs eDARTS' with operation attention (bottom). An increased attention to important operations with a simple, pre-specified cell topology allows searching with a shallow supernet yet discovering high quality cells.

I. Introduction

Deep neural networks have performed remarkably well across a variety of computer vision tasks such as image classification, object detection and semantic segmentation [1], [2], [3]. However, the complexity of such networks has increased dramatically over the past few years [4], [5] and therefore, manual network design demands significant engineering effort to be spent in exploring large spaces of hyper parameters. As a result, neural architecture search (NAS), as a means to automate the design of neural networks, has gained a lot of interest from both academia and industry. Early approaches for modern NAS used reinforcement learning [6], [7] or evolutionary algorithms [8], [9] and found competitive architectures. These methods however, were computationally too intensive i.e. requiring 2000 [7] or 3150 [8] GPU days. DARTS [10] formulated the architectural building block or a cell searching problem in a differentiable manner and brought down the computational cost to just one day.

The substantial reduction in the computational complexity has increased interest around differentiable architecture search framework with a number of potential improvements proposed [11], [12], [13], [14], [15], [16]. For example, P-DARTS [11] reports the depth gap problem between search and evaluation scenarios of DARTS and proposes to progressively increase the search network depth. Moreover, it identifies the skip connection bias problem, also noted by others [17], [18], and restricts the number of skip connections in the discovered cells. PC-DARTS [12] shows improved results by decreasing the memory and computation overhead of DARTS and reducing the information flow in the supernet to fewer channels. However, works such as [19], on the other hand, show that it is the expertly crafted DARTS' search space itself that even a randomly sampled architecture performs quite well. Consequently, it becomes unclear to know which factor contributes most to discovering better cells. Hence, DARTS framework can be improved by further investigating a number of directions i.e. the effect of different operations on the

This work was supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No 739551 (KIOS CoE) and from the Government of the Republic of Cyprus through the Directorate General for European Programmes, Coordination and Development.

resulting cells, the effect of information flow through different path possibilities, and the depth gap. Therefore, we motivate our study through the following scientific questions:

1) Which factor has a larger impact on accuracy and efficiency: the search network depth, the operation search space or the information flow path of the cells?
2) Can we use this information to enhance the resulting cell quality and speed up the search?

To find the answers, we conduct extensive experimentation to study the effect of different operations, different cell topologies and the search network depth on the quality of discovered cells. Our contributions can be summarized as follows:

- Through extensive experimentation, we empirically show that good cells can be discovered independent of the search network depth.
- By dissecting DARTS' search space [10] multiple ways and searching in sub spaces, we show that among learnable operations of this search space (i.e. sep-3x3, sep-5x5, dil-3x3, dil-5x5), separable convolutions in general, are more effective operations in deciding the final performance of a discovered cell on CIFAR-10 while dilated convolutions tend to deteriorate it.
- Through the analysis of the different paths that can be used by a cell, we demonstrate that not all of them lead to good solutions, thus it is also critical to consider good information flow in the search space.
- We empirically show that search can be sped up drastically using only a subset of the target dataset and finetuning architectural parameter update rate, yet maintaining the discovered cell quality.
- Based on our findings, we propose eDARTS which searches along a simpler cell structure with increased attention to important operations, see Figure 1.

Our proposed optimisations enable eDARTS to: (1) carry search along a shallow, 5 layer search network; (2) bring down the computational cost from 1.5 to just 0.015 GPU days while maintaining the discovered cell quality; and (3) on average reduce the skip connections from 6 to 2 in the discovered cells thus combating the skip connection bias. eDARTS' discovered cell achieves an error rate of 2.53 on CIFAR-10 using only 3.1M parameters. On CIFAR-100, the discovered cell has an error rate of just 16.83% with a 3.5M parameter model.

II. Background and Related Work

Since AlexNet [20], there has been a wide interest in improving deep convolutional neural networks for image classification and other related tasks. However, early networks, following AlexNet, were hand crafted by the experts and demanded significant engineering efforts [21], [22], [23], [4], [24], [1]. More recently, there is an increased interest in automating the process of designing such networks for a given task. Some of the works try to search for an end to end network structure [6], [25], [26] while others focus on finding a network building block or a cell [7], [10]. Most of these works require enormous compute and are not feasible for many researchers. Therefore, what has sparked widespread interest in NAS is the low computational complexity of ENAS [27] and specifically DARTS [10].

A. Differentiable Architecture Search

There has been substantial research effort around improving DARTS i.e. P-DARTS [11] notices depth gap between search and evaluation networks and proposes to progressively increase the search network depth. PC-DARTS [12] works on decreasing the still large memory requirements of DARTS and reduces the search cost to 0.1 GPU days. GDAS [13] deploys a gradient architecture sampler to efficiently traverse the directed acyclic graph (DAG) search space. Fair DARTS [28] studies the skip connection bias problem of the DARTS and proposes collaborative competition as well as gaussian noise addition to differential process removing unfair advantage skip connections possess. EDAS [29] improves DARTS speed to 0.125 GPU days by sampling a single edge in the cell for parameter update. FBNet [30] is one of the differentiable search method that searches for mobile architectures.

B. Understanding Neural Architecture Seaarch

Another very interesting line of work is being about investigating the performance of one shot NAS approaches. Why certain algorithms work better than the others? Is it because of the search algorithm, the training pipeline or expertly crafted search space. [31] is an important recent work to fairly compare the weight sharing architecture search methods. [32] provides a fixed search space and a unified benchmark for a fair comparison of any recent NAS algorithm. A closely related work is that of [19] which shows that many search spaces are being crafted such that even randomly picked up architectures are good and the training protocol itself has a higher impact on the final accuracy. It also investigates DARTS' search space and shows that it has narrow accuracy range and the final network wiring or macro architecture has more effect than the operations themselves. Our work however, investigate DARTS' search space on a micro level, i.e. we study the properties of a cell itself and not the macro network architecture.

C. DARTS Preliminaries

We leverage DARTS [10] as our baseline framework which searches for an architectural cell. The discovered cell is eventually stacked up to form a network of desired complexity. A cell is a directed acyclic graph consisting of an ordered sequence of *N* nodes, $\{x_0, x_1, \cdots, x_{N-1}\}$. Each node $x^{(i)}$ is a feature map and each edge $E_{(i,j)}$ is associated with some operation $o^{(i,j)}$ being applied on $x^{(i)}$. Each cell has two input nodes and a single output node where the input nodes are defined as the cell outputs from the previous two layers and the output is obtained by applying concatenation operator to all the intermediate nodes i.e. $x_{N-1} = concat(x_0, x_1, \cdots, x_{N-2})$. Each intermediate node is represented as $x^{(j)} = \sum_{i<j} o^{(i,j)} x^{(i)}$. The main idea of DARTS is to place a mixture of candidate operations *O* (e.g. convolutions, residual and pooling operations) at each edge and eventually learn the best one. To make the search

TABLE I: Sub Search Spaces Splits

Search Space	Operations
SS1	sep-3x3, dil-3x3
SS2	sep-5x5, dil-5x5
SS3	sep-3x3, sep-5x5
SS4	dil-3x3, dil-5x5
SS5	sep-3x3, dil-5x5
SS6	sep-5x5, dil-3x3
SS7	sep-3x3, sep-5x5, dil-3x3
SS8	sep-3x3, sep-5x5, dil-5x5

space differentiable, DARTS relaxes the categorical choice of a particular operation to a softmax over all possible operations:

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

where the operation mixing weights for a pair of nodes (i, j) are parameterized by a vector $\alpha^{(i,j)}$ of dimension $|O|$. For more details, please refer to the original paper [10].

III. Methodology

In this section, we provide the methodology for assessing how search network depth, operations and edges effect the discovered cells, and the main components of eDARTS, which searches along a simpler pre-determined path in the DAG with operation attention, Figure 1. We also describe optimizations that help alleviating the skip connection bias problem of DARTS as well as speed up the search significantly.

A. Effect of Search Depth and Operations

DARTS [10] works by searching for a cell on a network of a given depth. However, searching with deeper networks can be prohibitive due to memory limitations. DARTS tries to search on a network of increased depth i.e. 20 layers, by decreasing the initial number of channels e.g., from 16 to 6, which however, results in worst cells. Potential reasons suggested for worst performance could either be the enlarged discrepancy of the number of channels between search and evaluation scenarios or the fact that searching deeper might require a different set of hyperparameters. To reduce the search depth discrepancy, P-DARTS [11] proposes to progressively reduce the search space while increasing the search network depth. It starts the search with a shallower network (i.e., 5 layers) but with the full search space (i.e., 8 operations), and then progressively increases search depth while pruning the operation search space. When it reaches a depth that is sufficiently close to the evaluation depth, its search space is limited to only a fraction of the original operations. In this way, it is possible to search for an increased depth. However, during the search space approximation, operations important for the deeper networks may already have been dropped down whilst searching within the shallower network. Essentially decisions on which operations to keep are based on the shallower network search process rather than the deeper network. Therefore, there is limited understanding still of how the depth of a network impacts the resulting cell quality.

Fig. 2: Test Accuracy for each split search space under shallow (5 layers) and deep (16 layers) search configurations.

To understand the effect of depth better, it is necessary to perform search for networks with varying number of layers e.g. 5 and 20. However searching for deeper networks can be problematic due to potential GPU memory limitations (DARTS' search settings consume almost entire 11GB memory for one 1080Ti). Hence, we cannot simultaneously search deeper and within full search space using a reasonable number of channels and batch size. For example, to fit a (20-16)[1] search supernet into memory, we decrease the batch size to 16 but the resulting cells perform worse than shallower-searched cells. We want to come up with a way to search deeper and on all possible operations in the search space without added memory cost. Since we cannot search on all the operations simultaneously, the proposed methodology involves splitting the search space and search within each subspace with shallower as well as deeper networks. In this way, it is possible to explicitly and effectively study the effect of depth on the search process with respect to all operations and without running into memory issues.

There are 4 learnable operations in the DARTS' search space i.e. seperable and dilated convolutions with filters 3x3 and 5x5 filters, whereas 3 operations are parameter free. The idea is to keep the parameter free operations in all the sub search spaces while only a subset of learnable operations as these actually incur increased memory cost. Essentially there are six possible combinations for learnable operations, if we choose only 2 out of 4 for each sub search space. In Table I, we show how we create sub search spaces where the last two search spaces are for additional experiments to investigate the effect of dilated convolutions in detail. Also note that each sub search space contains skip, none and either max or average pooling operation. Experimental details are discussed in section IV-C, but Figure 2 reveals some interesting insights, i.e. with the same number of initial channels, searching deeper yields only marginally better cells than searching shallower. However, the search cost increases drastically with increased depth therefore,

[1]Notation: L-C where L is the number of layers and C is the channels in the first layer.

much faster search can be performed with shallow networks without significant accuracy drops.

An added benefit of searching within split search spaces is that we are able to have a closer look at which operations can lead to better cells. Therefore, another interesting insight from Figure 2 is that on average, an all separable convolution search space yields the best cells while an all dilated convolution search space produces worst, see for example SS3 i.e. all separable and SS4 i.e. all dilated convolution search space. Moreover, in general, 3x3 kernel operations are better than 5x5 kernel operations and in all cases, substituting a separable convolution within a search space by a dilated convolution results in worse performance. Therefore, among operations with learnable parameters, the best operation is sep-3x3, followed by sep-5x5 and then dil-3x3 followed by dil-5x5. This indicates that separable convolutions might provide more meaningful features towards classification. We can exploit this information within the search process by adding weight constant w_o to the architectural parameters of operations $\overline{o}^{(i,j)}(x)$ that are found to lead to better cells and subtract from the rest. Please note that putting more emphasis on the best performing operations such as separable convolutions does not mean we entirely delete the possibility of other operations, however, we adjust the weights such that important operations have higher probability of getting selected. Experimental details are covered in section IV-C.

B. Exploring Information Flow Paths

With respect to edges in DARTS' search space, each intermediate node is connected to all its predecessors during the search process, Figure 1 (top). At the end of search, top 8 out of 14 edges are chosen on the basis of the highest weights learned for corresponding operations. This process is referred to as discretization, where for each intermediate node, only the top two incoming edges are preserved. In addition to the operations chosen during the search process, the information flow paths are crucial to deciding the effectiveness of a certain cell. There can be numerous cells with the same number of learnable parameters and even the same operations but the performance difference can vary significantly. The differentiating factor amongst these architectures is the flow of information. Hence, it is important to consider which paths are important and whether they can be identified. By doing so, we can effectively prune the search space, both reducing computation and restricting the search within high-quality information flow paths.

We develop a methodology to answer if some edges are really more effective than others. Specifically, we first fix the type of operations such as all separable convolutions for the normal and all max pooling operations for the reduction cell (DARTS searches for a normal and a reduction cell). Then, we train a network for each possible cell topology and rank each path along the graph (cell) with respect to the resulting test accuracy. This approach highlights edge configurations that can potentially lead to high-quality cells.

As evidenced by the surface accuracy map in Figure 3, even with the same number of parameters and exactly similar

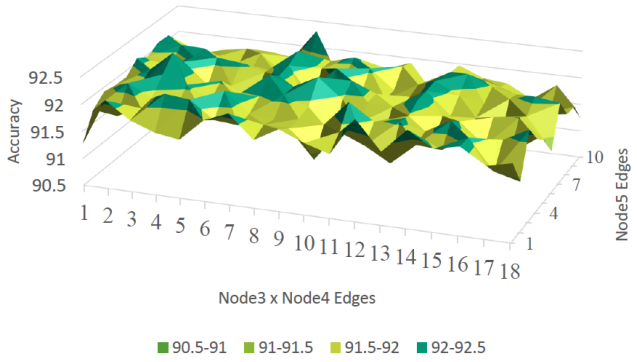

Fig. 3: Accuracy Surface Map of all possible 180 path combinations with their respective accuracies.

operations, there indeed exist good and bad performing edges to choose from. Therefore, we exploit this phenomenon by considering only the best topology with top edges and search only along this pre-specified path. In addition, we can also add more attention to the better performing operations such as separable convolutions. Figure 1 (bottom) shows an example of a pre-specfied path with operation attention. Please note that eDARTS' discretization step only discretizes operations after search and does not select edges as they are pre-specified.

C. Unbiased and Fast Search

Skip connection bias is a well known problem of DARTS [11], [17], [18], [15]. While attempting to reproduce DARTS' search results, we notice that when the supernet training is making faster progress, there are lesser skip connections, but when the training loss is close to saturation, skip connections start to emerge in both normal and the reduction cells. Some recent works provide an in depth analysis on why DARTS faces this problem [28], [33] proposing feasible solutions.

We take insights from these works and propose improvements that simultaneously reduce skip bias and allow for faster search. First we refer to the "early stopping" paradigm proposed by DARTS+ [33] which proposes two criterion for stopping the search: (1) The search procedure stops when there are two or more than two skip-connects in one cell, and (2) The search procedure stops when the ranking of architecture parameters α for learnable operations becomes stable for a determined number of epochs (e.g., 10 epochs). We notice that following either of the criteria is actually equivalent to making fewer architectural parameter updates throughout the search, and another way of achieving similar results is by using only a small subset of the target dataset, however, with the added advantage of much faster search.

Secondly, we note that both P-DARTS [11] and PC-DARTS [12] use the "warm-up" mechanism i.e. first train the supernet for 10 epochs and only then start architectural parameter updates for the next 15 epochs. However, they still update architecture parameters at every training step within an epoch which makes their search still computationally intensive. Therefore, we suggest a "sparse-update" mechanism specifically

for improving the learning of the architectural parameters and to speed up the search further. Since the architecture parameters are dependent on the search network weights, therefore, within each epoch, we let the model weights to improve significantly before we make any adjustments to architecture weights. An intuitive explanation for doing that is we make architecture adjustments only after the search network has seen a sufficient number of samples and the underlying model weights' direction becomes stable. Based on the above, we summarize our improvements as:

- Searching with a subset of the target dataset for full 50 epochs instead of early stopping reduces the skip connection bias, yet significantly speeds up the search, see Table II.
- Using "sparse-update" through out the search instead of traditional warm-up is also effective at reducing the skip connection aggregation problem, with a much reduced search cost, Table II.

D. Shallow Network Evaluation

Evaluating the cells discovered by NAS approaches is computationally expensive since training results from CIFAR10 (which is the most common dataset for NAS) are subject to high variance, hence requiring multiple evaluation runs [34]. And without training, there is no guarantee that the discovered cells will yield good final test accuracy. For example, DARTS [10] and PC-DARTS [12] discover cells by searching for a shallower network of 8 layers and 16 initial channels $(8-16)$ and evaluate on a deeper network of $20-36$. Moreover, evaluating a cell with training on a $20-36$ network until convergence is too compute intensive i.e. around 30 hours on a 2080ti GPU. Further, DARTS runs its search algorithm 4 times, tests the discovered cells by training a $20-36$ network from scratch for 100 epochs to get performance estimate of each cell, and chooses one with the highest validation accuracy. Therefore, it is necessary to have a faster evaluation criteria to speed up the overall experimentation. Earlier works have also used various lower fidelity estimates such as reducing training time by training; for a fewer epochs, on a subset of data, and with a downscaled model [7], [35], [36]. Therefore, we use an approximate evaluation of a discovered cell to speed up the experimentation by evaluating on a shallower network for only 100 epochs. To get an estimate of the performance of the cells discovered by DARTS and related works, we evaluate them on shallower network, see Table III. The averaged test accuracies of these cells across multiple runs set a baseline on what to expect from a good cell.

IV. Experiments

We evaluate the proposed methodology to measure its effectiveness in producing high quality cells much faster than the previous methods. Based on the results of existing approaches, we first setup a faster evaluation process relying on shallow networks. Then, we show the impact of searching deeper and shallower using the sub search spaces. Next, we evaluate the possible information flow paths within the supernet to identify good candidate edges to search along. Finally, we evaluate our proposed optimizations for skip connection bias and search speed. We use both CIFAR-10 and CIFAR-100 for architecture search as well as evaluation scenarios. Both of these datasets contain 60K, 32×32 RGB images. 50K samples belong to the training while 10K are reserved for testing purposes.

TABLE II: Comparison of searching with 1) original DARTS, 2) with only a subset of data, 3) by reducing architectural parameter updates and 4) combining 2 and 3. Accuracy and Parameters are averaged while Search Cost is the minimum across five runs.

Method	**Skip Connections**		**Shallow Network Accuracy**	**Parameters (M)**	**Search Cost (GPU Hours)**
	Normal	**Reduction**			
DARTS	14	19	90.25	0.11	36
DARTS + Data Subset	4	14	91.27	0.15	6
DARTS + Epoch Update	5	11	91.33	0.16	4.6
DARTS + Data Subset + Epoch Update	5	8	92.15	0.2	0.85

A. Evaluation of Existing Approaches

Existing works often show results using different hyper-parameters which makes comparison of different cells rather challenging. There is a possibility that it is not actually the discovered cell but the training hyper-parameters which lead to one discovered cell's superiority over another. For example, P-DARTS [11] and PC-DARTS [12], as compared to DARTS, train their final networks with a slightly bigger batch size i.e. 128 and a drop path probability of 0.3 instead of 0.2. To evaluate the performance of the discovered cells, we need to keep the same training hyper-parameters and run across multiple seeds. Therefore, we first evaluate the cells discovered by DARTSV1, DARTSV2, P-DARTS and PC-DARTS, using a shallow network to set a baseline on the expected performance. We create a small network of 5 layers (cells) and 16 initial channels and train it for 100 epochs. We use a batch size of 64, drop path probability of 0.2, learning rate of 0.025 that drops to 0 using cosine annealing, momentum of 0.9, and weight decay of 0.0003. We apply gradient clipping at 5 and do not use cutout or auxiliary tower as in the full evaluation of DARTS. We train networks using each of the discovered cells with 10 different seeds and record the average test accuracies on CIFAR-10.

Mean evaluation accuracies from Table III show that cells which perform better for deeper networks of 20-36 layers

also perform better on shallower networks of 5-16 layers. For example, cells discovered with DARTSV1 and DARTSV2 perform worse than PDARTS and PCDARTS. Therefore, as a rough approximation, cells that perform well on full 20-36 architectures seem to be performing better for smaller architectures of 5-16 too. Unless otherwise stated, we use these shallow network settings to evaluate the effectiveness of a discovered cell in all of our experiments, hence reducing the evaluation time from 30 hours to < 3 hours. Moreover, we train the final networks only when our cell performs well enough, i.e. test accuracy of $> 93\%$ on a shallow network.

B. Unbiased Search via Reduction Techniques

In this section, we show experiments towards reducing the skip connections whereas simultaneously speeding up the search. First, we show that it is possible to achieve similar performance with the improved search, and later we use this improved search as basis to evaluate the rest of the approaches from Section III. To build a baseline, we run DARTS with its original search settings 5 times. The average number of skip connection for normal and reduction cell combined turn out to be 7 while the average accuracy is just 90%. We rerun search with the exact same settings as the original DARTS (2nd order; please refer to equation 7 in DARTS paper) with only a subset of data and notice a performance increase of 1%. This also gives significant speed up, i.e. from 36 hours down to 6 hours and the average number of skip connections went down to 4, as shown in Table II.

Next, we evaluate the rate of the architectural parameters update. DARTS itself updates the architectural parameters for operations at every step of the training while the supernet weights themselves are changing rapidly. Using the same DARTS search settings, we reduce the architectural parameters update rate to only once per epoch. This improves the resulting cells by more than 1% on average and the search takes only 4.6 hours. Moreover, the average number of skip connections is down to 3.

By simultaneously reducing the search data, i.e. using only 2500 samples for training and 2500 for validation (instead of 25K each in the DARTS), and decreasing the architecture parameter update frequency we observe an improvement of almost 2%. Moreover, it brings down the search cost to less than an hour (0.85 GPU hours or 0.035 GPU days), and reduces the number of skip connections to 2. Please note that other than reducing the dataset for search and reducing architectural parameter updates, all of the search settings are exactly that of the search scenario from DARTS [10]. Using the above two techniques combined gives the least number of skip connections (although not zero) and the fastest gradient based search method i.e. 11x faster than DARTS+ [33], 8.5x faster than P-DARTS [11] and 2.8x faster than PC-DARTS [12].

C. Searching Shallow and Deep

To investigate the depth gap and whether some operations are more effective than others, we perform various experiments on the search spaces introduced in Section III-A. Initially, for SS1, we run search for 4 different depths i.e. 5, 10, 16, 20 (10 times each) and keep the initial number of channels to 16. We notice that the resulting cell quality is similar when searching for depths of 16 and 20 but was superior to when searching on depth 5. Therefore, for the rest of the spaces, the search is focused on either 5 (shallow) or 16 (deeper) layer networks. For each search space; we run the search 10 times each for a shallow and a deep network (using seeds 20 to 29), train each discovered cell's network, and take average of the test accuracy. Figure 2 shows that searching in a 16-16 deep network yields only marginally better cells than searching in a 5-16 shallow network. That is except in one case where the search space contains only separable 3x3 and 5x5 operations. Since the accuracy gap is not significant, it means that one can search using shallow networks, which is faster and can still get good cells. A key observation is that from the resulting cells, the worst performing ones all have dilated convolution in their search space. Specifically, dilated convolution 5x5 is the worst operation followed by dilated convolution 3x3. Overall, the best performing cells are found in an all separable convolution search space. This leads us to deduce that separable convolutions are much more likely to lead to high quality cells. Therefore, in our final search settings for architectural parameters, we add the following constant weight factors: 0.0006 to *sep3x3* and 0.0004 to *sep5x5* while subtracting the same from *dil5x5* and *dil3x3*, which are empirically found to work best.

TABLE III: Evaluation of cells discovered by different NAS approaches. Cells which perform better for deep networks tend to perform better for shallow ones too.

Network	DARTS_V1	DARTS_V2	P-DARTS	PC-DARTS
Shallow	90.14	91.37	92.76	93.00
Final	97.00	97.24	97.50	97.43

D. Searching along the best Paths

Considering the possible graph topologies within the original DARTS search space, there are 180 possible discrete paths for each of the normal and reduction cells. Hence, the total number of possible graph topology combinations for both normal and reduction cell is $32,400$. Now, considering that the reduction cell has to appear only twice in a 20 layer network [10], we can focus on just the normal cell by fixing the reduction cell. Therefore, we evaluate 180 different, path based shallow networks, where normal cell comprises of only 3x3 separable convolutions, with a fixed reduction cell containing only max pooling. The idea is to discover the best search path by figuring out the optimal flow of information. Within the best path, the algorithm then has to search for the right operations only. This evaluation scheme reveals that different topologies have different capacities to learn good features and there are indeed, both good and bad information flow paths. Although all the operations are similar, the highest accuracy along a certain path is as high as 92.44% while the lowest, along a different path was 91.24%. This certainly provides evidence that in addition

TABLE IV: Comparison with state-of-the-art architectures on CIFAR-10 and CIFAR-100 datasets. † As reported by P-DARTS [11]. ‡ This time is recorded on a GTX 1070 GPU.

Architecture	**Test Err. (%)**		**Params**	**Search Cost**	**Search Method**
	C10	**C100**	**(M)**	**(GPU-days)**	
DenseNet-BC [24]	3.46	17.18	25.6	-	manual
NASNet-A + cutout [7]	2.65	-	3.3	1800	RL
AmoebaNet-A + cutout [8]	3.34	-	3.2	3150	evolution
AmoebaNet-B + cutout [8]	2.55	-	2.8	3150	evolution
Hierarchical evolution [9]	3.75	-	15.7	300	evolution
PNAS [37]	3.41	-	3.2	225	SMBO
ENAS + cutout [27]	2.89	-	4.6	0.5	RL
DARTS (first order) + cutout [10]	3.00	17.76†	3.3	1.5	gradient-based
DARTS (second order) + cutout [10]	2.76	17.54†	3.3	4	gradient-based
SNAS + mild constraint + cutout [38]	2.98	-	2.9	1.5	gradient-based
SNAS + moderate constraint + cutout [38]	2.85	-	2.8	1.5	gradient-based
SNAS + aggressive constraint + cutout [38]	3.10	-	2.3	1.5	gradient-based
ProxylessNAS + cutout [39]	2.08	-	5.7	4	gradient-based
PC-DARTS + cutout [12]	2.57	-	3.6	0.1	gradient-based
Fair DARTS [28]	2.54	-	2.8	0.3	gradient-based
P-DARTS CIFAR10 + cutout [11]	2.50	16.55	3.4	0.3	gradient-based
P-DARTS CIFAR100 + cutout [11]	2.62	15.92	3.6	0.3	gradient-based
eDARTS CIFAR10 + cutout	2.53	17.00	3.1	**0.015**‡	gradient-based
eDARTS CIFAR100 + cutout	2.72	16.83	3.5	0.016	gradient-based

to the operations, cell topology can be a decisive factor in the resulting final evaluation of a network, see Figure 3.

For exploring different paths, we initially fix the reduction cell with all max pooling operations to speed up the evaluation. In the next phase, we take top 20 best performing paths from the initial evaluation and apply these to both normal and reduction cells (i.e., same path and all separable 3x3 operations for both normal and reduction cells) and re-evaluate the unified paths. One of the best paths results in an accuracy of 93.66% which is better than our initial shallow baseline from Table III. Therefore, we create a supernet where the edges are already fixed with this pre-specified path, and the search is focused towards learning the right operations only. Moreover, searching along a smaller super network, i.e. with 8 instead of 14 edges, further brings down to cost to 0.015 GPU days.

E. Final Architecture Search and Evaluation

For the final architecture search, we incorporate all suggested improvements; the faster unbiased search from section IV-B, the focus on separable convolution based on section IV-C, and run the search along one of the best paths we discover in section IV-D. We use a shallow supernet of 5 layers with 16 initial channels for the search, where we give slightly higher weights to choosing separable convolution operations. Overall, eDARTS' approach reduces the search cost down to just 0.015 GPU days. For the final evaluation of our discovered cells, we deploy the exact training pipeline as that of P-DARTS and PC-DARTS except that we use a smaller batch size of 96 due to GPU memory limitations, and set the initial number of channels to 32 so that the total number of parameters are always less than or roughly equal to 3.3M. As shown in Table IV, on CIFAR-10 we achieve an error of 2.53% with a model of 3.1M parameters. For CIFAR-100, the error is 16.84% which is better than DARTS. However, the search cost is the lowest among all the existing works and we run search only once for each respective dataset.

F. Ablation Studies

We run search on a 16 layer deep network with an all separable search space and all 14 edges, and the resulting best cell gives an error of 2.56%. However, this cell is found amongst 10 runs of the search, and the cost of each individual run is roughly around one hour. Next, we run search 10 times along the pre-specified path but with equal attention to all operations of the search space, and end up with lower accuracy cells, i.e. less than 92.5% on shallow network. However with eDARTS, we need to run search only once for each CIFAR-10 and CIFAR-100 and still get competitive cells.

V. Conclusions

We propose a search strategy eDARTS that leverages analysis of the DARTS search space on a micro-level leading to more informed search and improved search times. Through extensive experimentation, we show that as compared to search network depth, operations and cell topologies are more important factors in deciding the performance of a discovered cell. As demonstrated by the experiments, the improved search also alleviates the skip connection bias problem of DARTS. Consequently, eDARTS can produce competitive architectures at only a fraction of search cost by putting more attention to separable convolution operations and searching along a strong information flow path. Overall, we bring down the

computational cost from 1.5 to just 0.015 GPU days while maintaining the discovered cell quality.

REFERENCES

[1] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018.

[2] B. Singh, M. Najibi, and L. S. Davis, "Sniper: Efficient multi-scale training," in *Advances in neural information processing systems*, pp. 9310–9320, 2018.

[3] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 801–818, 2018.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[5] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *European conference on computer vision*, pp. 646–661, Springer, 2016.

[6] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *arXiv preprint arXiv:1611.01578*, 2016.

[7] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8697–8710, 2018.

[8] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proceedings of the aaai conference on artificial intelligence*, vol. 33, pp. 4780–4789, 2019.

[9] H. Liu, K. Simonyan, O. Vinyals, C. Fernando, and K. Kavukcuoglu, "Hierarchical representations for efficient architecture search," *arXiv preprint arXiv:1711.00436*, 2017.

[10] H. Liu, K. Simonyan, and Y. Yang, "Darts: Differentiable architecture search," *arXiv preprint arXiv:1806.09055*, 2018.

[11] X. Chen, L. Xie, J. Wu, and Q. Tian, "Progressive differentiable architecture search: Bridging the depth gap between search and evaluation," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1294–1303, 2019.

[12] Y. Xu, L. Xie, X. Zhang, X. Chen, G.-J. Qi, Q. Tian, and H. Xiong, "Pc-darts: Partial channel connections for memory-efficient differentiable architecture search," *arXiv preprint arXiv:1907.05737*, 2019.

[13] X. Dong and Y. Yang, "Searching for a robust neural architecture in four gpu hours," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1761–1770, 2019.

[14] X. Jin, J. Wang, J. Slocum, M.-H. Yang, S. Dai, S. Yan, and J. Feng, "Rc-darts: Resource constrained differentiable architecture search," *arXiv preprint arXiv:1912.12814*, 2019.

[15] A. Zela, T. Elsken, T. Saikia, Y. Marrakchi, T. Brox, and F. Hutter, "Understanding and robustifying differentiable architecture search," *arXiv preprint arXiv:1909.09656*, 2019.

[16] J. Chang, Y. Guo, G. MENG, S. XIANG, C. Pan, *et al.*, "Data: Differentiable architecture approximation," in *Advances in Neural Information Processing Systems*, pp. 874–884, 2019.

[17] H. Liang, S. Zhang, J. Sun, X. He, W. Huang, K. Zhuang, and Z. Li, "Darts+: Improved differentiable architecture search with early stopping," *arXiv preprint arXiv:1909.06035*, 2019.

[18] K. Bi, C. Hu, L. Xie, X. Chen, L. Wei, and Q. Tian, "Stabilizing darts with amended gradient estimation on architectural parameters," *arXiv preprint arXiv:1910.11831*, 2019.

[19] A. Yang, P. M. Esperança, and F. M. Carlucci, "Nas evaluation is frustratingly hard," in *International Conference on Learning Representations*, 2020.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[21] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*, pp. 818–833, Springer, 2014.

[22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2015.

[23] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[24] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

[25] B. Baker, O. Gupta, N. Naik, and R. Raskar, "Designing neural network architectures using reinforcement learning," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, OpenReview.net, 2017.

[26] M. Suganuma, S. Shirakawa, and T. Nagao, "A genetic programming approach to designing convolutional neural network architectures," in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 497–504, 2017.

[27] H. Pham, M. Guan, B. Zoph, Q. Le, and J. Dean, "Efficient neural architecture search via parameters sharing," in *Proceedings of the 35th International Conference on Machine Learning* (J. Dy and A. Krause, eds.), vol. 80 of *Proceedings of Machine Learning Research*, (Stockholmsmässan, Stockholm Sweden), pp. 4095–4104, PMLR, 10–15 Jul 2018.

[28] X. Chu, T. Zhou, B. Zhang, and J. Li, "Fair darts: Eliminating unfair advantages in differentiable architecture search," *arXiv preprint arXiv:1911.12126*, 2019.

[29] H. Gwon Hong, P. Ahn, and J. Kim, "Edas: Efficient and differentiable architecture search," *arXiv*, pp. arXiv–1912, 2019.

[30] B. Wu, X. Dai, P. Zhang, Y. Wang, F. Sun, Y. Wu, Y. Tian, P. Vajda, Y. Jia, and K. Keutzer, "Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 10734–10742, 2019.

[31] A. Zela, J. Siems, and F. Hutter, "Nas-bench-1shot1: Benchmarking and dissecting one-shot neural architecture search," in *International Conference on Learning Representations*, 2020.

[32] X. Dong and Y. Yang, "Nas-bench-201: Extending the scope of reproducible neural architecture search," in *International Conference on Learning Representations (ICLR)*, 2020.

[33] H. Liang, S. Zhang, J. Sun, X. He, W. Huang, K. Zhuang, and Z. Li, "Darts+: Improved differentiable architecture search with early stopping," 2019.

[34] M. Lindauer and F. Hutter, "Best practices for scientific research on neural architecture search," 2019.

[35] A. Zela, A. Klein, S. Falkner, and F. Hutter, "Towards automated deep learning: Efficient joint neural architecture and hyperparameter search," *arXiv preprint arXiv:1807.06906*, 2018.

[36] S. Falkner, A. Klein, and F. Hutter, "BOHB: Robust and efficient hyperparameter optimization at scale," in *Proceedings of the 35th International Conference on Machine Learning* (J. Dy and A. Krause, eds.), vol. 80 of *Proceedings of Machine Learning Research*, (Stockholmsmässan, Stockholm Sweden), pp. 1437–1446, PMLR, 10–15 Jul 2018.

[37] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L.-J. Li, L. Fei-Fei, A. Yuille, J. Huang, and K. Murphy, "Progressive neural architecture search," in *The European Conference on Computer Vision (ECCV)*, September 2018.

[38] S. Xie, H. Zheng, C. Liu, and L. Lin, "SNAS: stochastic neural architecture search," in *International Conference on Learning Representations*, 2019.

[39] H. Cai, L. Zhu, and S. Han, "ProxylessNAS: Direct neural architecture search on target task and hardware," in *International Conference on Learning Representations*, 2019.