Probeable DARTS with Application to Computational Pathology

Sheyang Tang[1]*, Mahdi S. Hosseini[2]*, Lina Chen[3], Sonal Varma[4], Corwyn Rowsell [5],
Savvas Damaskinos [6], Konstantinos N. Plataniotis [7], and Zhou Wang[1]
[1]University of Waterloo, Canada [2]University of New Brunswick, Canada
[3]Sunnybrook health science center, Canada [4]Kingston Health Sciences Center, Canada
[5]St. Michaels Hospital, Canada [6]Huron Digital Pathology, Canada [7]University of Toronto, Canada
https://github.com/mahdihosseini/DARTS-ADP

Abstract

AI technology has made remarkable achievements in computational pathology (CPath), especially with the help of deep neural networks. However, the network performance is highly related to architecture design, which commonly requires human experts with domain knowledge. In this paper, we combat this challenge with the recent advance in neural architecture search (NAS) to find an optimal network for CPath applications. In particular, we use differentiable architecture search (DARTS) for its efficiency. We first adopt a probing metric to show that the original DARTS lacks proper hyperparameter tuning on the CIFAR dataset, and how the generalization issue can be addressed using an adaptive optimization strategy. We then apply our searching framework on CPath applications by searching for the optimum network architecture on a histological tissue type dataset (ADP). Results show that the searched network outperforms state-of-the-art networks in terms of prediction accuracy and computation complexity. We further conduct extensive experiments to demonstrate the transferability of the searched network to new CPath applications, the robustness against downscaled inputs, as well as the reliability of predictions.

1. Introduction

Recent years have witnessed great advances in AI-based Computational Pathology (CPath) [21, 15]. The emerging AI techniques have shown their superiority in more accurate, efficient, and large-scale medical diagnoses [4]. In particular, Convolutional Neural Networks (CNNs) have been widely employed to extract meaningful information from medical images for various pathology applications, including disease diagnoses [5, 37], medical image segmentation [26, 30], etc. Yet designing the network architectures has long been a manual process that requires adequate domain knowledge. As a result, it has become a common standard that architectures from CV applications (such as ResNet [8] and GoogLeNet [31]) are transferred for technical developments in other fields, including CPath [28, 33].

Table 1: Comparison between CV and CPath datasets.

General Stats	CV		CPath
	CIFAR-10	CIFAR-100	ADP
Training size	50000	50000	14134
Validation size	-	-	1767
Test size	10000	10000	1767
Resolution	32x32	32x32	272x272
# classes	10	100	33
Label form	single-label	single-label	multi-label
Background	various	various	white

(a) CIFAR (b) ADP

Figure 1: Sample images from CV and CPath datasets.

The ultimate question is whether transferring architectures between the two domains is an efficient strategy. To answer this question, we first demonstrate how CV and CPath datasets are different. Here we compare the CIFAR [18] and ADP [11] datasets. Besides different data structures shown in Table. 1, the nature of images from both sides is also different, which makes CV datasets more complicated. *First*, the pixel resolution in CPath is fixed, corresponding to a fixed field of view (FOV) size. The root cause

*Equal contribution

of such uniformity is the acquisition of whole slide images by a scanner in a much more controlled environment from both optics and illumination viewpoint [11]. In contrast, the pixel resolution in CV is randomly distributed across different images due to different image setup and configurations. CV images are captured in natural scenes where the distance has much variance. Examples from each imaging modality are shown in Figure 1(a), where the ship images are taken from a further distance than the dog ones, resulting in larger pixel size and lower resolution. *Second*, target objects in CV images only occupy part of the whole FOV and the rest are background which is irrelevant to the class label. Note that the diversity of the background in Figure 1(a) is very high. This is quite different in CPath where the background information is obtained from an empty area of the sample using uniform white light illumination [11] –leading to more uniform and homogeneous images. This is illustrated in Figure 1(b), where the white part denotes the background. In the light of this difference, we form a hypothesis that such simplified imaging modality in CPath translates to simpler network architecture compared to CV. To this end, new network architectures should be designed for CPath applications.

Neural architecture search (NAS) has recently been proposed to automate the design of neural networks by searching for the optimal network structure on a given dataset. In many CV applications, NAS has outperformed state-of-the-art manually designed networks in terms of prediction accuracy and computation complexity [7]. In medical image analysis, it has been utilized to find suitable networks for various applications, such as image segmentation for Magnetic Resonance Imaging (MRI) [17, 35, 3], ultrasound imaging [34], disease diagnoses from Computed Tomography (CT) scans [16, 9], etc. In pathology, however, NAS is not fully explored. There is a lack of a general framework that can be easily extended to various CPath applications.

In this work, we propose an architecture search platform based on differentiable architecture search (DARTS) [22]. We choose DARTS because it is gradient-based and thus much more efficient and computation-friendly than other searching strategies including reinforcement learning [38] and evolutionary algorithms [25]. DARTS achieves this by relaxing the search space to be continuous and dividing the whole pipeline into a search phase and an evaluation phase. However, in CV applications, it is reported that DARTS tends to exhibit overfitting issues, and the searched architecture does not generalize well in the evaluation phase [20, 36]. To combat these challenges, we first conduct searching on CIFAR [18] and utilize a probing metric *stable rank* [12] for each layer. In this way, we can better monitor the searching process and show that the overfitting issue comes from improper hyperparameter tuning. In addition, we use an adaptive optimizer *Adas* [12] that automatically tunes the learning rates for each layer based on their probing metrics, so that the generalization ability of the searched architecture is improved. We then apply this searching framework on ADP [11], which contains a great variety of histological tissue types that are representative enough, so that the searched architecture can generalize well in different CPath applications. The searched network outperforms the state-of-the-art architectures in the speed-accuracy trade-off, which is crucial for real-time high-throughput CPath applications. We further conduct extensive experiments to show the transferability of the searched architecture on new CPath datasets, demonstrate its robustness against decreased input images, and verify its superiority in extracting label-pertinent features. Our main contributions are listed below:

- We use a probing metric to show that the existing DARTS framework lacks proper hyperparameter tuning, and use an adaptive optimizer to improve the generalization ability of the searched model;
- We apply the proposed searching platform on CPath applications and show the superiority of the searched model in prediction accuracy and computation complexity;
- We demonstrate the transferability of the searched architecture in various CPath applications, show its robustness against decreased resolutions and its reliability in prediction.

2. Related Works

Table 2: Summary of NAS applications in medical image analysis.

Task	Searching Strategy		
	Gradient-based	Reinforcement Learning	Evolutionary Algorithms
Segmentation	[34, 17, 6]	[3]	[35]
Classification	[24, 9]	[10]	[16]

As NAS has achieved promising results in many CV applications [7], several attempts are made to utilize NAS techniques to find optimum architectures for applications in medical image analysis. Based on the task and the searching strategy, these works can be categorized as in Table. 2. In applications of image segmentation, most works adopt a U-net structure, where detail configurations are searched in different manners. [34, 17] use differentiable architecture search to find cell structures as building blocks in the encoder and decoder. Bae et al. [3] utilize reinforcement learning to search for hyper-parameter configurations of the U-Net architecture. Yu et al. [35] first search for cell connections to form a U-Net topology using evolutionary algorithms, and then search for operations within each cell.

Dong et al. [6] extend the differentiable searching framework to work in adversarial training.

For classification applications, the searching is more task-specific. Using gradient-based searching, Peng et al. [24] develop a network to predict distant metastases on PET-CT images, and He et al. [9] design a network for COVID-19 detection with Chest CT Scans. Hosseini et al. [10] use a reinforcement learning-based controller to find the best parameter configuration of a CNN model for histological tissue type classification. Jiang et al. [16] search for a network to classify pulmonary nodules with evolutionary algorithms.

To the best of our knowledge, there hasn't been any work that fully explores the potentials of NAS in digital pathology applications.

3. Proposed Method

In this section, we introduce our searching algorithm. We first review the basic concepts of DARTS [22], then show how the existing DARTS framework can be improved using a probing metric and a new optimizer. Finally, a network size-based searching is proposed to seek a trade-off between prediction accuracy and model complexity.

3.1. Review on DARTS

The goal of DARTS [22] is to search for two types of cells (namely *normal* and *reduction*) as building blocks, which are stacked to form a full network. Each cell is represented as a directed acyclic graph with N nodes, including two input nodes, intermediate nodes and one output node. Every node x_i is a latent representation (*e.g.*, feature map in CNN) and every edge (i, j) is a mixture of weighted candidate operations in a pre-defined operation search space $\mathcal{O}$ (*e.g.*, convolution, skip-connection). The output $\bar{o}_{i,j}$ of an edge (i, j) is then a weighted sum of candidate operations [22]:

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

where $\alpha_{i,j}^{o}$ is an architecture parameter for weighting operation $o\left(x_i\right)$. The output of an intermediate node x_j is the sum of all input edges, *i.e.*, $x_j = \sum_{i<j} \bar{o}_{i,j}\left(x_j\right)$. The output node of a cell is the concatenation of all intermediate nodes. Normal cells keep the input resolution while reduction cells decrease resolutions with stride 2 in all candidate operations.

In the searching procedure, the network weight w and architecture parameter α are jointly learned via bi-level optimization [22]:

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

where $\mathcal{L}_{val}$ and $\mathcal{L}_{train}$ denote the validation and training datasets, respectively. Using gradient descent, w and α can be updated alternatively during each training iteration.

When the searching is finished, the discrete cell architecture is obtained by replacing each edge by the operation with the largest architecture weight, then selecting the two strongest input edges for each intermediate node. Fig. 2 (b) illustrates the evolution of cell structure during searching. The discrete network is then retrained from scratch for final evaluation. The whole process is shown in Fig. 2 (a).

3.2. Explainable Metrics for Probing

To monitor the searching process of DARTS, we adopt the explainability metric *stable rank* to probe the intermediate convolutional layers in different cells and quantify their learning quality as explained in [12]. Given a convolutional weight matrix, we first decompose it by low-rank factorization. This factors out the perturbation noise in the layer while keeping the most useful information in the low-rank component. The stable rank $\mathcal{S}$ is the normalized sum of the singular values of the low-rank matrix. It measures the norm energy of the convolutional weights and encodes the low-rank structure's space span of the output mapping. A higher value indicates better propagation of information through a convolutional layer [12].

Using this probing metric, we monitor the searching phase of the existing DARTS framework applied on the CIFAR100 dataset [18]. The left column of Fig. 3 shows an example of the stable rank evolution (top row) of the layers in one cell as well as the architecture weights evolution (bottom 2 rows) in two edges. We can see from the stable rank evolution that the convolutional layers are not learning well in the original DARTS with stochastic gradient descent (DARTS+SGD) and default initial learning rate 0.025 [22] – most layers generate zero stable rank through all training epochs. Recall the edge structure introduced in Sec. 3.1, each candidate operation is multiplied by a weight, which is between 0 and 1. This makes their gradients small during backpropagation. Therefore the convolutional layers are learning slowly. The bottom two images show that skip-connections (green curves) are preferred. The same phenomenon is reported in [20] that when searched on CV datasets, the original DARTS tends to select too many skip-connections, which is a kind of overfitting, resulting in a shallow network with poor representation ability.

In light of this, we increase the initial learning rate from 0.025 to 0.05 and 0.175. The stable rank evolution of layers in the same cell, as well as the architecture weights in the same edges, are shown in Fig. 3 (b) and (c). With the increase in initial learning rates, more layers generate a higher stable rank, which means they're learning better. In the meantime, the architecture weights evolution reveals that the preference for skip-connections is suppressed. With

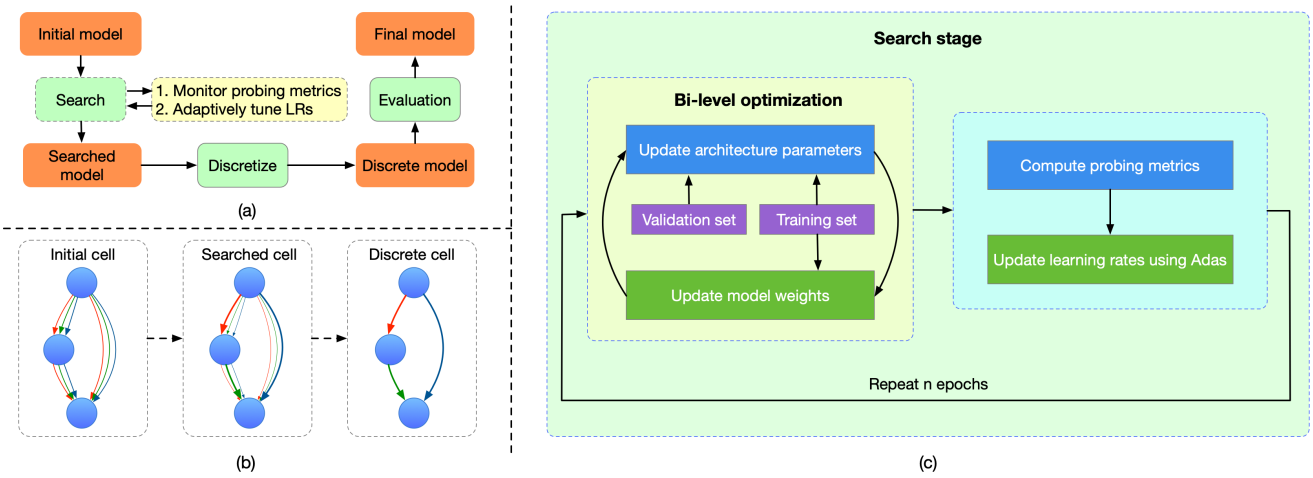


Figure 2: (a) Overview of the architecture searching pipeline. The network architecture is searched during the search stage and discretized when finished. It is then trained from scratch for final evaluation. (b) Illustration of the cell structure during searching. (c) Details of the search stage. During each searching epoch, the architecture parameters and model weights are updated alternately through bi-level optimization, then the learning rates for each convolutional layer are tuned based on probing metrics.

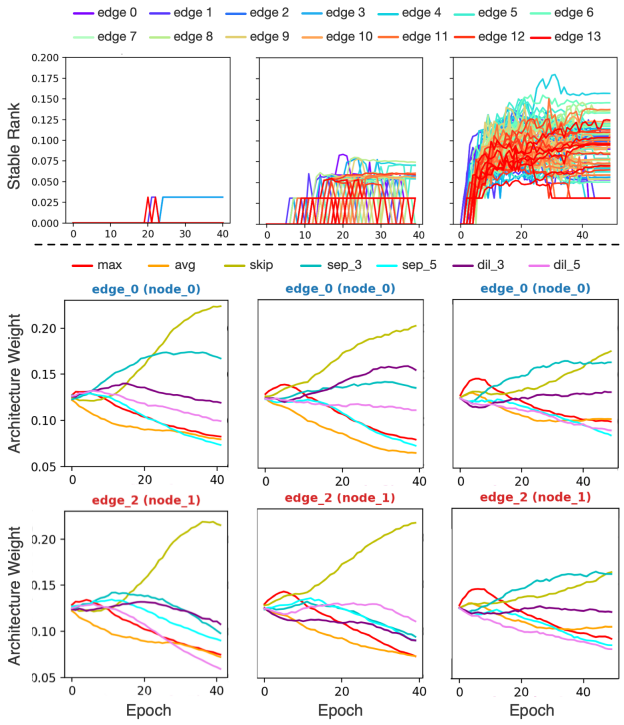


Figure 3: Comparison of the stable rank and architecture weight evolution on CIFAR100.

the help of the probing metric, we know that the existing DARTS framework lacks proper tuning in hyperparameters and how the searching can be improved.

3.3. Improving DARTS Performance

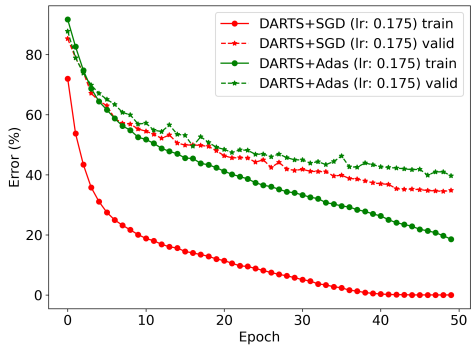


Figure 4: Training and validation error during searching on CIFAR-100.

We have shown in the previous section that the probing metric can be used to tune proper initial learning rates for DARTS. Then why not utilize an optimizer that incorporates such metric for learning rates adjustment during searching? To verify this, we adopt the Adas optimizer [12], which adaptively adjusts the learning rate for each layer based on their stable rank evolution. At the end of each searching epoch, it first computes the difference in stable rank over consecutive epochs for each convolutional layer, and then adds (with a weight) result to the learning rate momentum. A hyperparameter *scheduler beta* is used for weighting this term. This process is illustrated in Fig. 2 (c).

The Adas optimizer is aware of the learning quality of each layer and therefore tunes their learning rates accordingly.

Fig. 4 shows the training and validation errors when searching with different optimizers and initial learning rates. We can see that SGD with a 0.175 initial learning rate leads to overfitting during searching. The final gap between training and validation error is around 40%. While for Adas, the overfitting problem is reduced. The resulting final gap is around 20%. This indicates that Adas improves the searching process of DARTS with better generalization ability.

3.4. Network Size-based Searching

We investigate the trade-off between network size and test performance by searching for the optimal architectures with different numbers of cells and intermediate nodes. This trade-off is crucial for high-throughput CPath applications in real-time. On the ADP dataset, DARTS+Adas obtains the best-performing architecture. It consists of four cells, and each cell contains three intermediate nodes. Fig. 6 (c) and (d) show the snapshots of the cell structures.

4. Experiments and Results

Our experiments contain two stages. In the first stage, we search for the optimum architectures on CIFAR and ADP. In the second stage, we evaluate the transferability of the architecture searched on ADP, as well as its robustness and reliability in various cases.

4.1. Experimental Setup

5. Architecture Search

We carry out the searching on CIFAR and ADP datasets. The search space of candidate operations is the same as in [22], including 1) 3x3 separable convolution, 2) 5x5 separable convolution, 3) 3x3 dilated separable convolution, 4) 5x5 dilated separable convolution, 5) 3x3 max pooling, 6) 3x3 average pooling, 7) skip connection, and 8) zero operation. We stack the cells sequentially to build a network for searching and evaluation. The details of network structures can be found in Section Network Structures of the supplementary material. In both CIFAR and ADP experiments, we test two optimization strategies for optimizing model weights, *i.e.*, DARTS+SGD and DARTS+Adas. Detailed setup can be found in Section Hyperparameters of the Supplementary Material.

6. Architecture Evaluation

The searched network is discretized and then trained from scratch for final evaluation. Following [22], in each parameter setting, we conduct four independent runs of searching with different random seeds. We then perform a quick evaluation for each searched architecture by training them from scratch for 100 epochs and pick the best-performing one. The finalized architecture is trained from scratch for 600 epochs in three independent runs. We report the means and standard deviations of test accuracy. Training details can be found in Section Hyperparameters of the Supplementary Material.

6.1. Results on the CIFAR Dataset

On the CIFAR dataset, we search for the optimum optimizer and number of intermediate nodes. During searching, eight cells are stacked as in DARTS, while the number of nodes in each cell is tuned between 4, 5, and 6. During evaluation, to prevent the network size from being too large with more nodes, we also change the number of cells accordingly, *i.e.*, 20 cells for 4 nodes, 17 cells for 5 nodes, and 14 cells for 6 nodes. Table. 3 and Table. 4 show the test performance of architectures searched on CIFAR-10 and CIFAR-100. We can see that in each setting, DARTS+Adas outperforms the default DARTS+SGD in terms of test accuracy, while a cost is paid in parameter size. This is because the original DARTS+SGD tends to select skip-connections in the final architecture as described above in Sec. 3.2. The optimum number of nodes is 4.

Table 3: Performance of architectures searched with different number of intermediate nodes on CIFAR-10 (32x32 image resolution).

Number of nodes	DARTS+SGD			DARTS+Adas		
	Acc (%)	Prm (M)	MAC (G)	Acc (%)	Prm (M)	MAC (G)
4	$97.24_{0.09}$	3.30	0.55	$97.43_{0.05}$	4.14	0.66
5	$96.69_{0.07}$	3.46	0.57	$97.45_{0.06}$	4.23	0.67
6	$96.16_{0.05}$	2.02	0.34	$96.73_{0.07}$	4.35	0.69

Table 4: Performance of architectures searched with different number of intermediate nodes on CIFAR-100 (32x32 image resolution).

Number of nodes	DARTS+SGD			DARTS+Adas		
	Acc (%)	Prm (M)	MAC (G)	Acc (%)	Prm (M)	MAC (G)
4	$81.09_{0.24}$	2.42	0.38	$84.09_{0.18}$	4.24	0.68
5	$82.10_{0.25}$	2.92	0.46	$83.49_{0.23}$	4.41	0.70
6	$81.81_{0.24}$	3.25	0.52	$83.15_{0.27}$	4.11	0.66

6.2. Results on the ADP Dataset

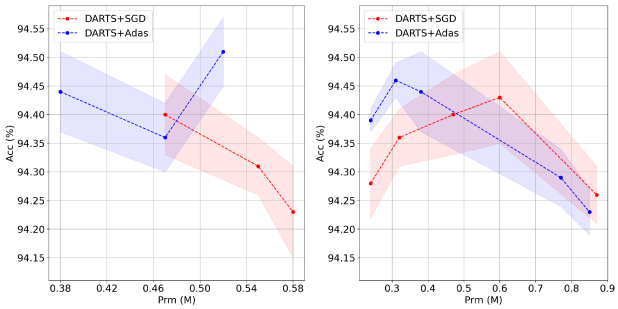
To find the optimum architecture on the ADP dataset, we run the architecture search in all different parameter settings, *i.e.*, different numbers of cells and nodes, different choices of optimizer. Note that in CIFAR experiments we search for a shallower network (with few cells) during searching but train a deeper one (with more cells) for evaluation due to the complexity of CV datasets. In ADP, however, we keep the number of cells the same in two stages.

Table 5: Performance of architectures searched with different optimization strategies and different number of cells on ADP (272x272 image resolution).

Number of cells	DARTS+SGD			DARTS+Adas		
	Acc (%)	Prm (M)	MAC (G)	Acc (%)	Prm (M)	MAC (G)
4	$94.40_{0.07}$	0.47	0.35	$94.44_{0.07}$	0.38	0.30
5	$94.31_{0.05}$	0.55	0.42	$94.36_{0.06}$	0.47	0.43
6	$94.23_{0.08}$	0.58	0.50	$94.51_{0.06}$	0.52	0.51

Table 6: Performance of architectures searched with different number of intermediate nodes on ADP (272x272 image resolution).

Number of nodes	DARTS+SGD			DARTS+Adas		
	Acc (%)	Prm (M)	MAC (G)	Acc (%)	Prm (M)	MAC (G)
2	$94.28_{0.06}$	0.24	0.20	$94.39_{0.02}$	0.24	0.21
3	$94.36_{0.05}$	0.32	0.26	$94.46_{0.03}$	0.31	0.27
4	$94.40_{0.07}$	0.47	0.35	$94.44_{0.07}$	0.38	0.30
5	$94.43_{0.08}$	0.60	0.43	$94.29_{0.05}$	0.77	0.52
6	$94.26_{0.05}$	0.87	0.57	$94.23_{0.04}$	0.85	0.57



(a) Different number of cells (b) Different number of nodes

Figure 5: (a) Performance of different numbers of cells. Each dot denotes a different cell number in 4, 5, and 6. (b) Performance of different numbers of nodes. Each dot represents a different node number in 2, 3, 4, 5, and 6.

This is because ADP is a simpler dataset so we don't need to increase the model complexity during evaluation. This also brings more consistency to the search-evaluation pipeline.

Optimum optimizer and number of cells. We first search for the optimum optimizer and the number of cells. The test results of the searched architectures after final evaluation are shown in Table. 5. We also plot the accuracy versus parameter size in Fig. 5 (a), where each dot represents a different choice of cell number. We can see that as the number of cells increases the accuracy of DARTS+SGD drops while its parameter size increases. When using DARTS+Adas, the test accuracy remains the highest across different numbers of cells, and the parameter size remains the smallest. The highest accuracy is achieved with 6 cells, while for 4 cells, the searched architecture has the smallest size but still obtains the second highest accuracy.

Optimum number of intermediate nodes. We then fix the number of cells as four and search for the optimum number of intermediate nodes. The test performance are shown in Tabel.6 and in Fig. 5 (b). We can see that DARTS+Adas achieves higher accuracy than DARTS+SGD with fewer nodes, hence less computation complexity. The highest accuracy 94.46% is achieved with 3 nodes, leading to 0.31M parameters and 0.27G MAC operations. Fig. 6 shows the cell architectures searched with 2, 3, and 4 nodes using DARTS+Adas.

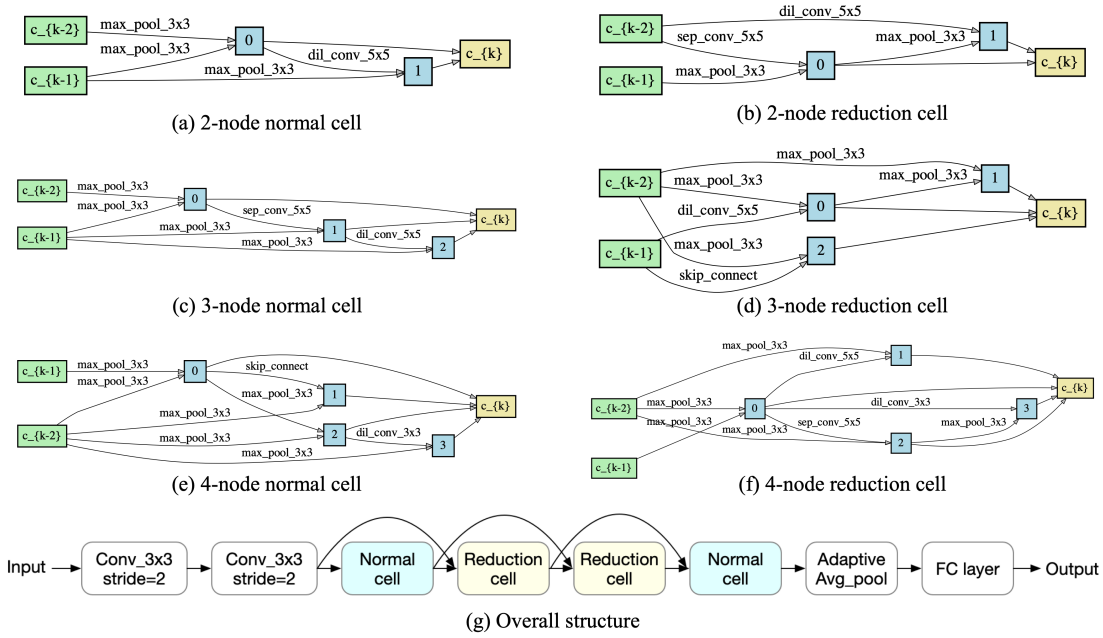
6.3. Architecture Transferabilty

We select the searched architectures with 2, 3 and 4 nodes (namely DARTS-ADP-N2, -N3 and -N4), and train them on three more datasets: BCSS [1], BACH [2] and Osteosarcoma [19]. Details including data augmentations can be found in Section Datasets of the Supplementary Material. The goal is to evaluate how the searched architectures perform when transferred to different CPath datasets that cover different variations of single- vs multi-labels, multi-class problems, data samples, and organs. We also train several mobile-friendly architectures on them for comparison, including a 4-cell DARTS [22]. All networks are trained for 600 epochs with batch size 96, using the SGD optimizer with a 0.025 initial learning rate and cosine annealing scheduler.

As shown in Tabel.7, across all datasets, the group of DARTS-ADP networks achieves higher or comparable test accuracy than state-of-the-art networks, but with smaller parameter sizes. The 2-node version contains only 0.24M parameters but still ranks high in test accuracy. As for computation complexity, though MobileNetV2 0.35 [27], MobileNetV3-small [13], and MNASNet-small [32] achieve fewer MAC operations, their accuracies are two percent lower. This shows the superiority of DARTS-ADP in the speed-accuracy trade-off, which is desirable for CPath applications of high-throughput image analysis.

6.4. Performance of different image resolution

Another way to meet the needs of high-throughput applications is to decrease the image resolution. To evaluate the robustness of the architectures against downscaled inputs, we retrain several models with different resolutions (272, 136, 68, 34) in three datasets. Fig. 7 shows the performance of different network selection. Each line represents a network and each dot represents a specific resolution. The DARTS-based networks consistently achieve the highest accuracy with the lowest computation complexity in all resolutions and across all datasets. As the resolution decreases, their test accuracy exhibits a much less drop compared to ResNet18 [8] and MobileNetV2 [27], which shows the robustness of the DARTS-based networks. Such robustness is also illustrated in the standard deviation (denoted by shades). Compared to 4-cell DARTS [22], the two DARTS-ADP networks obtain higher or comparable test

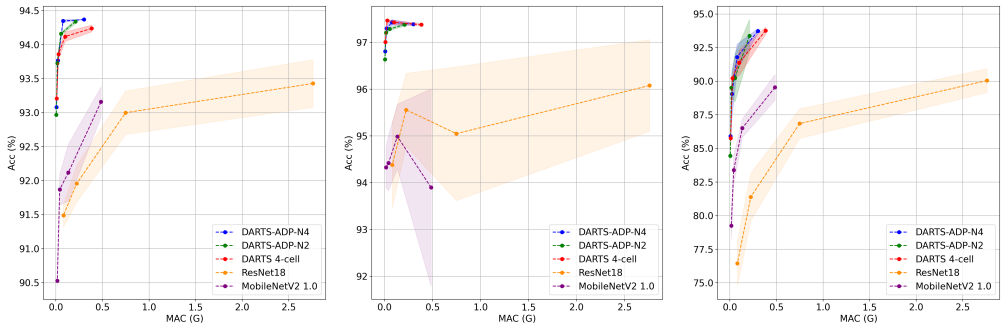


(a) 2-node normal cell (b) 2-node reduction cell (c) 3-node normal cell (d) 3-node reduction cell (e) 4-node normal cell (f) 4-node reduction cell (g) Overall structure

Figure 6: (a)-(f) Best cell structures found in different cases. (g) Overall network structure.

Table 7: Performance of different networks on different datasets. Note that **green** indicates the best, and **orange** indicates within standard deviation from the best.

Network	ADP [11]			BCSS [1]			BACH [2]			Osteosarcoma [19]		
	Acc (%)	Prm (M)	MAC (G)	Acc (%)	Prm (M)	MAC (G)	Acc (%)	Prm (M)	MAC (G)	Acc (%)	Prm (M)	MAC (G)
ResNet18 [8]	$93.43_{0.35}$	11.19	2.76	$96.13_{0.80}$	11.18	2.76	$90.05_{0.89}$	11.18	2.76	$93.23_{0.30}$	11.18	2.76
MobileNetV2 1.0 [27]	$93.16_{0.22}$	2.27	0.48	$93.90_{2.11}$	2.24	0.48	$89.55_{0.95}$	2.23	0.48	$90.73_{0.46}$	2.23	0.48
MobileNetV2 0.35 [27]	$91.73_{0.79}$	0.44	0.10	$93.84_{1.02}$	0.41	0.10	$87.18_{0.46}$	0.40	0.10	$90.59_{0.50}$	0.40	0.10
MobileNetV3-large [13]	$91.43_{0.82}$	4.24	0.34	$94.40_{0.62}$	4.21	0.34	$86.98_{1.61}$	4.21	0.34	$89.60_{0.69}$	4.21	0.34
MobileNetV3-small [13]	$92.15_{0.21}$	1.55	**0.09**	$93.70_{0.39}$	1.53	**0.09**	$86.32_{1.16}$	1.52	**0.09**	$89.11_{0.89}$	1.52	**0.09**
ShuffleNetV2 1.0 [23]	$92.18_{1.62}$	1.29	0.23	$96.14_{0.73}$	1.26	0.23	$91.41_{0.69}$	1.26	0.23	$91.67_{0.57}$	1.26	0.23
SENet18 [14]	$93.33_{0.18}$	11.28	2.76	$96.63_{0.27}$	11.27	2.76	$90.93_{0.78}$	11.27	2.76	$92.10_{1.14}$	11.27	2.76
MNASNet-A1 [32]	$91.98_{0.21}$	2.65	0.50	$95.84_{0.11}$	2.62	0.50	$83.63_{2.40}$	2.61	0.50	$90.26_{0.64}$	2.61	0.50
MNASNet-small [32]	$92.44_{0.17}$	0.79	0.11	$94.63_{1.51}$	0.76	0.11	$83.54_{2.15}$	0.75	0.11	$91.47_{0.64}$	0.75	0.11
DARTS 4-cell[22]	$94.24_{0.05}$	0.49	0.38	**$97.38_{0.03}$**	0.48	0.38	**$93.77_{0.26}$**	0.48	0.38	**$95.04_{0.27}$**	0.48	0.38
DARTS-ADP-N4	**$94.37_{0.00}$**	0.38	0.30	**$97.39_{0.02}$**	0.36	0.30	**$93.74_{0.18}$**	0.36	0.30	$93.99_{0.06}$	0.36	0.30
DARTS-ADP-N3	**$94.41_{0.04}$**	0.31	0.27	$97.34_{0.07}$	0.30	0.27	$92.07_{2.15}$	0.30	0.27	$94.52_{0.24}$	0.30	0.27
DARTS-ADP-N2	$94.34_{0.03}$	**0.24**	0.21	**$97.38_{0.03}$**	**0.24**	0.21	$93.38_{1.21}$	**0.23**	0.21	$94.54_{0.35}$	**0.23**	0.21



(a) ADP (b) BCSS (c) BACH

Figure 7: Performance of different image resolution (denoted by different dots) on different dataset and network selection.

accuracy with lower computation complexity, which again shows their superiority in the speed-accuracy trade-off.

6.5. Grad-CAM analysis

To better understand and interpret the performance of feature representation of different networks, we apply Grad-CAM [29] on the last convolutional layers of different networks to obtain the heatmaps for predicting ground truth labels. This visualization technique allows us to evaluate the reliability of different networks in label prediction. We randomly select five patches that contain different ground truth labels from the test set of ADP, BCSS, and BACH, and feed them into three networks for comparison. The selected networks are DARTS-ADP-N4, MobileNetV3-large [13], and MNASNet-A1 [32]. Results are shown in Fig. 8, where the heatmap indicates pixel-level confidence of pertinent labels of the image patch.

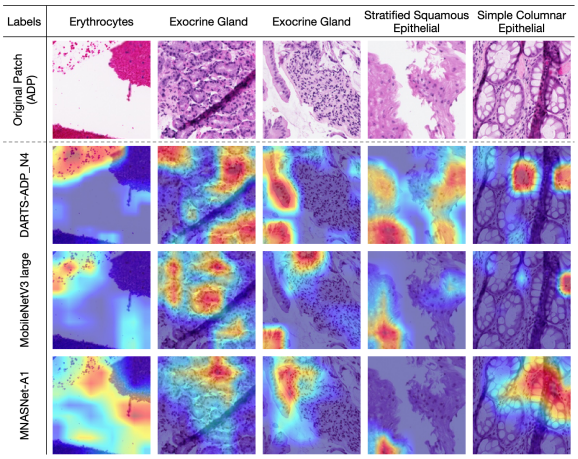
According to pathologists' assessment, the overall performance of DARTS-ADP-N4 is the best. Examples are shown in the first column of Fig. 8(a), where DARTS-ADP-N4 successfully highlights the region of *Erythrocytes*. Either MobileNetV3-large or MNASNet-A1 discovers incomplete or false regions. This demonstrates the superiority of DARTS-ADP in extracting label-pertinent features from image patches, and hence more reliable predictions.

7. Conclusion

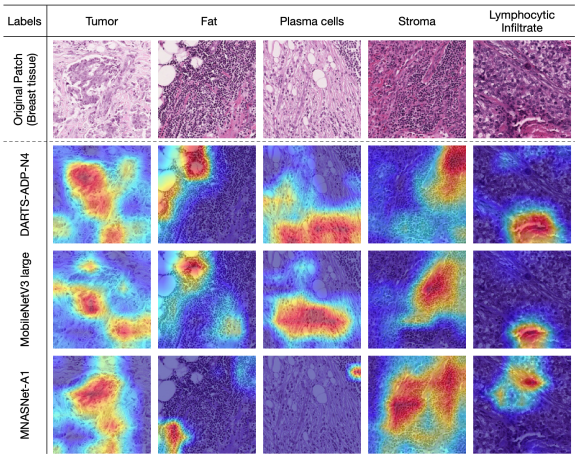
In this paper, we propose a general DARTS-based searching framework for CPath applications. We first use a probing metric to show that the existing DARTS lacks proper hyperparameter tuning, and how the generalization performance of the searched model can be improved with an adaptive optimization strategy. We then apply this searching framework on a histological tissue type dataset ADP and develop architectures that outperform the state-of-the-art networks with higher prediction accuracy and lower computation complexity. We transfer the searched architectures to other CPath datasets including BCSS, BACH, and Osteosarcoma, and conduct extensive experiments to demonstrate the robustness and reliability of the networks in various cases.

References

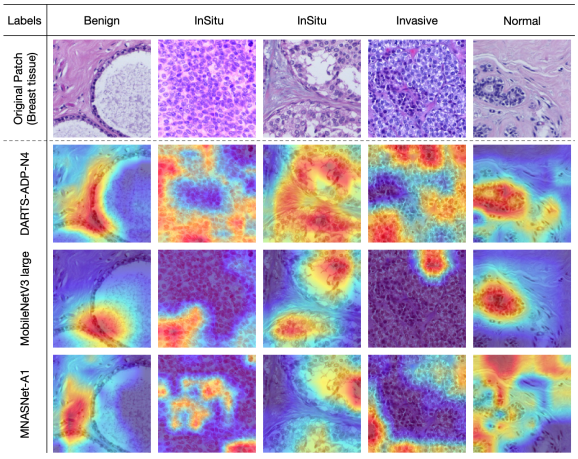
[1] Mohamed Amgad, Habiba Elfandy, Hagar Hussein, Lamees A Atteya, Mai A T Elsebaie, Lamia S Abo Elnasr, Rokia A Sakr, Hazem S E Salem, Ahmed F Ismail, Anas M Saad, Joumana Ahmed, Maha A T Elsebaie, Mustafijur Rahman, Inas A Ruhban, Nada M Elgazar, Yahya Alagha, Mohamed H Osman, Ahmed M Alhusseiny, Mariam M Khalaf, Abo-Alela F Younes, Ali Abdulkarim, Duaa M Younes, Ahmed M Gadallah, Ahmad M Elkashash, Salma Y Fala, Basma M Zaki, Jonathan Beezley, Deepak R Chittajallu, David Manthey, David A Gutman, and Lee A D Cooper.



(a) ADP



(b) BCSS



(c) BACH

Figure 8: Grad-CAM performance of different network selection on different datasets.

Structured crowdsourcing enables convolutional segmentation of histology images. *Bioinformatics*, 35(18):3461–3467, 02 2019.

[2] Guilherme Aresta, Teresa Araújo, Scotty Kwok, Sai Saketh Chennamsetty, Mohammed Safwan, Varghese Alex, Bahram Marami, Marcel Prastawa, Monica Chan, Michael Donovan, et al. Bach: Grand challenge on breast cancer histology images. *Medical image analysis*, 56:122–139, 2019.

[3] Woong Bae, Seungho Lee, Yeha Lee, Beomhee Park, Minki Chung, and Kyu-Hwan Jung. Resource optimized neural architecture search for 3d medical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 228–236. Springer, 2019.

[4] Kaustav Bera, Kurt A Schalper, David L Rimm, Vamsidhar Velcheti, and Anant Madabhushi. Artificial intelligence in digital pathology—new tools for diagnosis and precision oncology. *Nature reviews Clinical oncology*, 16(11):703–715, 2019.

[5] Dan C Cireşan, Alessandro Giusti, Luca M Gambardella, and Jürgen Schmidhuber. Mitosis detection in breast cancer histology images with deep neural networks. In *International conference on medical image computing and computer-assisted intervention*, pages 411–418. Springer, 2013.

[6] Nanqing Dong, Min Xu, Xiaodan Liang, Yiliang Jiang, Wei Dai, and Eric Xing. Neural architecture search for adversarial medical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 828–836. Springer, 2019.

[7] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *The Journal of Machine Learning Research*, 20(1):1997–2017, 2019.

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

[9] Xin He, Shihao Wang, Xiaowen Chu, Shaohuai Shi, Jiangping Tang, Xin Liu, Chenggang Yan, Jiyong Zhang, and Guiguang Ding. Automated model design and benchmarking of 3d deep learning models for covid-19 detection with chest ct scans. *arXiv preprint arXiv:2101.05442*, 2021.

[10] Mahdi S Hosseini, Lyndon Chan, Weimin Huang, Yichen Wang, Danial Hasan, Corwyn Rowsell, Savvas Damaskinos, and Konstantinos N Plataniotis. On transferability of histological tissue labels in computational pathology. In *European Conference on Computer Vision*, pages 453–469. Springer, 2020.

[11] Mahdi S Hosseini, Lyndon Chan, Gabriel Tse, Michael Tang, Jun Deng, Sajad Norouzi, Corwyn Rowsell, Konstantinos N Plataniotis, and Savvas Damaskinos. Atlas of digital pathology: A generalized hierarchical histological tissue type-annotated database for deep learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11747–11756, 2019.

[12] Mahdi S Hosseini and Konstantinos N Plataniotis. Adas: Adaptive scheduling of stochastic gradients. *arXiv preprint arXiv:2006.06587*, 2020.

[13] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1314–1324, 2019.

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

[15] Andrew Janowczyk and Anant Madabhushi. Deep learning for digital pathology image analysis: A comprehensive tutorial with selected use cases. *Journal of pathology informatics*, 7, 2016.

[16] Hanliang Jiang, Fuhao Shen, Fei Gao, and Weidong Han. Learning efficient, explainable and discriminative representations for pulmonary nodules classification. *Pattern Recognition*, 113:107825, 2021.

[17] Sungwoong Kim, Ildoo Kim, Sungbin Lim, Woonhyuk Baek, Chiheon Kim, Hyungjoo Cho, Boogeon Yoon, and Taesup Kim. Scalable neural architecture search for 3d medical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 220–228. Springer, 2019.

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

[19] P Leavey, A Sengupta, D Rakheja, O Daescu, HB Arunachalam, and R Mishra. Osteosarcoma data from ut southwestern/ut dallas for viable and necrotic tumor assessment [data set]. *The Cancer Imaging Archive*, 14, 2019. `https://doi.org/10.7937/tcia.2019.bvhjhdas`.

[20] 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.

[21] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017.

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

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

[24] Yige Peng, Lei Bi, Michael Fulham, Dagan Feng, and Jinman Kim. Multi-modality information fusion for radiomics-based neural architecture search. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 763–771. Springer, 2020.

[25] 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.

[26] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmen-

tation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[27] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[28] Andrew J Schaumberg, Mark A Rubin, and Thomas J Fuchs. H&e-stained whole slide image deep learning predicts spop mutation state in prostate cancer. *BioRxiv*, page 064279, 2017.

[29] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[30] Youyi Song, Ee-Leng Tan, Xudong Jiang, Jie-Zhi Cheng, Dong Ni, Siping Chen, Baiying Lei, and Tianfu Wang. Accurate cervical cell segmentation from overlapping clumps in pap smear images. *IEEE transactions on medical imaging*, 36(1):288–300, 2016.

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

[32] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019.

[33] Dayong Wang, Aditya Khosla, Rishab Gargeya, Humayun Irshad, and Andrew H Beck. Deep learning for identifying metastatic breast cancer. *arXiv preprint arXiv:1606.05718*, 2016.

[34] Yu Weng, Tianbao Zhou, Yujie Li, and Xiaoyu Qiu. Nas-unet: Neural architecture search for medical image segmentation. *IEEE Access*, 7:44247–44257, 2019.

[35] Qihang Yu, Dong Yang, Holger Roth, Yutong Bai, Yixiao Zhang, Alan L Yuille, and Daguang Xu. C2fnas: Coarse-to-fine neural architecture search for 3d medical image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4126–4135, 2020.

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

[37] Jianwei Zhao, Minshu Zhang, Zhenghua Zhou, Jianjun Chu, and Feilong Cao. Automatic detection and classification of leukocytes using convolutional neural networks. *Medical & biological engineering & computing*, 55(8):1287–1301, 2017.

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