Learning Latent Architectural Distribution in Differentiable Neural Architecture Search via Variational Information Maximization

Yaoming Wang[1†], Yuchen Liu[1†], Wenrui Dai[2*], Chenglin Li[1], Junni Zou[2], and Hongkai Xiong[1]
[1]Department of Electronic Engineering, Shanghai Jiao Tong University, China
[2]Department of Computer Science & Engineering, Shanghai Jiao Tong University, China
{wang_yaoming, liuyuchen6666, daiwenrui, lcl1985, zoujunni, xionghongkai}@sjtu.edu.cn

Abstract

Existing differentiable neural architecture search approaches simply assume the architectural distribution on each edge is independent of each other, which conflicts with the intrinsic properties of architecture. In this paper, we view the architectural distribution as the latent representation of specific data points. Then we propose Variational Information Maximization Neural Architecture Search (VIM-NAS) to leverage a simple yet effective convolutional neural network to model the latent representation, and optimize for a tractable variational lower bound to the mutual information between the data points and the latent representations. VIM-NAS automatically learns a nearly one-hot distribution from a continuous distribution with extremely fast convergence speed, e.g., converging with one epoch. Experimental results demonstrate VIM-NAS achieves state-of-the-art performance on various search spaces, including DARTS search space, NAS-Bench-1shot1, NAS-Bench-201, and simplified search spaces S1-S4. Specifically, VIM-NAS achieves a top-1 error rate of 2.45% and 15.80% within 10 minutes on CIFAR-10 and CIFAR-100, respectively, and a top-1 error rate of 24.0% when transferred to ImageNet.

1. Introduction

With the development of deep learning, various neural architectures are needed for specific tasks. Given a specific dataset, neural architecture search (NAS) frees researchers from cumbersome architecture design by exploring the search space automatically to search for the optimal network architecture. Since different datasets have their own preference for the architecture, for instance, biomedical images favor U-Net alike architectures with fully convolutional operations and symmetrical structure. NAS can be deemed as searching for the preference of a given dataset. Thus, the architectural parameters can be viewed as the latent representation of specific data points.

In the sense of methodology, NAS can be divided into three parts, *i.e.*, search space, search strategy, and performance estimation strategy [17, 38]. One-shot approaches [3, 4, 34] have been developed as a promising alternative to reduce the search time by finding the best sub-network in a super network through parameter sharing. Gradient-based approaches [51, 29, 37, 41], aka Differentiable NAS, further treat the super network as the whole search space, introduce architectural parameters, and obtain the sub-network by optimizing the architectural parameters and super network weights in a differentiable manner.

Despite the remarkable performance, existing differentiable NAS approaches make improper assumptions that the predefined architectural distribution on each edge is independent of each other, conflicting with intrinsic properties of architecture. Specifically, SNAS [41] and FBNet [39] utilize the *Concrete distribution* [22, 30] to approximate the discrete categorical distribution on architectural parameters. DATA [51] performs multiple sampling with replacement from the same *Concrete distribution* to expand the search space. SI-VDNAS [37] introduces Gaussian noise for variational dropout and samples the super network through the learned dropout rate. Isotropic Gaussian noise in SI-VDNAS as well as the factorizable distribution in SNAS, FBNet and DATA ignores the dependencies between architectural parameters.

In addition to the above improper assumptions, the search cost for differentiable NAS approaches is also prohibitive due to the numerous search spaces and tasks. Two-stage approaches [47, 36] decouple the model training and searching process for amortizing the training cost, but are still restricted by numerous search spaces. Therefore, an efficient search strategy with fast convergence speed is desirable for differentiable NAS.

In this paper, we propose a novel search strategy, namely Variational Information Maximization Neural Architecture Search (VIM-NAS), to maximize the mutual information

*Corresponding author: Wenrui Dai. †Equal contribution.

between the data points and the latent architectural representations. VIM-NAS leverages a simple yet effective convolutional neural network to model the latent architectural representation, and optimizes for a tractable variational lower bound to the mutual information. Our contributions are summarized below.

- We propose a novel perspective that architectural distribution can be deemed as the latent representation of a given dataset in NAS.
- We leverage a simple yet effective architectural neural network to model the dependencies among architectural distribution.
- We propose a novel search strategy to maximize the variational lower bound to the mutual information between the data points and the latent architectural representations.

Experimentally, VIM-NAS exhibits currently the fastest convergence speed within one epoch and learns a nearly one-hot distribution from a continuous distribution. Specifically, VIM-NAS achieves the state-of-the-art performance on DARTS search space with a top-1 error rate of 2.45% and 15.80% within 10 minutes on CIFAR-10/100 and a top-1 error rate of 24.0% when transferred to ImageNet. VIM-NAS also achieves the state-of-the-art performance on other search spaces, including NAS-Bench-1shot1, NAS-Bench-201 and simplified search spaces S1-S4.

2. Related Work

NAS approaches liberate researchers from tedious network architecture design on various tasks, including image classification [54], objection detection [19, 20, 45], image segmentation [18, 28], and pose estimation [2, 50] etc. Meanwhile, more efficient and stable neural architecture search strategies are still attracting a lot of research. Evolutionary algorithms [16, 31, 35, 44] encode network architectures into populations, optimize populations through continuous iterative mutation, and eventually translate populations back to networks. Reinforcement learning (RL)-based approaches [3, 34, 54] utilize a meta-controller to guide the search process in a large architectural space based on a reward function corresponding to the inference accuracy of the selected network. Despite the remarkable performance, evolutionary algorithms and RL-based methods suffer from high search cost due to repeated evaluations. One-shot approaches [1, 3, 4, 34] adopt parameter sharing to reduce the search cost. Gradient-based approaches [29, 41, 37] further introduce architectural parameters to optimize searched architectures in a differentiable fashion.

Though differentiable approaches can achieve fast search speed, the search process still encounters the unstable problem and tedious super network training for each search process. Recently, the collapse problem in DARTS attracts a lot of work. FairDARTS [11] utilizes the sigmoid function to avoid unfair exclusive competition. SGAS [25], instead, circumvents the problem with a greedy strategy. DARTS+ [27] and Progressive DARTS [8] employ early stopping to control the number of identity operations. These approaches involve too much human intervene, are not suitable for different search spaces, and are hard to transfer to different tasks. [48] designs indicators like Hessian eigenvalues for the collapse, and [7] adds perturbations to regularize such an indicator. Unsupervised representation [43] for neural architecture is also utilized to learn good architecture representation using architectures without accuracy. TE-NAS [6] proposes a training-free strategy to rank the candidate architectures with the spectrum of the neural tangent kernel (NTK) and the number of linear regions in the input space.

3. Methodology

3.1. Preliminary: Differentiable NAS

Differentiable NAS decomposes the searched architecture into stacking cells. The search space is defined as a directed acyclic graph (DAG) over cells with N nodes. Each node is a latent representation and there is a predefined operation set denoted by $\mathcal{O}$ on each edge (i, j) connecting node i and j. The core idea is to relax the discrete operation selection to be continuous and obtain a weighted sum over all $|\mathcal{O}|$ operations on each edge as

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

where x_i is the output of the i-th node and $\alpha_{i,j}^{o}$ is the architectural parameter. The output of a node is the sum over all input edges as $\mathbf{x}_j = \sum_{i<j} f_{i,j}\left(\mathbf{x}_i\right)$, and the output of the total cell is the concatenation of the output of nodes as $\operatorname{concat}\left(\mathbf{x}_2, \mathbf{x}_3, \ldots, \mathbf{x}_{N-1}\right)$, where $\mathbf{x}_0$ and $\mathbf{x}_1$ are the fixed input nodes. Based on the cell search space, Differentiable NAS relaxes the architecture search to learn the architectural parameter $A = \left[\alpha^{(i,j)}\right]$. With the target dataset and optimization objective, the architectural parameter A and the network weight w are optimized via gradient descent alternatively [29] or jointly [41].

3.2. VIM-NAS

We deem the architecture as the latent variable for the observed data. Given a dataset $\mathcal{D}$ consisting of N pairs of observations and labels $\{(x_n, y_n)\}_{n=1}^{N}$, we consider a joint distribution $p_\phi(\mathcal{D}, A) = p(\mathcal{D})p_\phi(A|\mathcal{D})$ between the dataset $\mathcal{D}$ and architecture A, where $p(\mathcal{D})$ is the distribution of dataset $\mathcal{D}$ and $p_\phi(A|\mathcal{D})$ is the posterior architectural distribution parameterized by ϕ. In differentiable NAS, we learn the parameter ϕ that accurately predicts the specific dataset $\mathcal{D}$ using the architecture A. To achieve this goal, we propose to maximize the mutual information $I_\phi(\mathcal{D}, A)$ between the dataset $\mathcal{D}$ and architecture A as:

$$\begin{aligned}\max_\phi I_\phi(\mathcal{D}, A) &= \mathbb{E}_{p_\phi(\mathcal{D},A)}\left[\log \frac{p_\phi(\mathcal{D}, A)}{p(\mathcal{D})p_\phi(A)}\right] \\ &= H(\mathcal{D}) - H_\phi(\mathcal{D}|A), \end{aligned} \quad (2)$$

where $H(\cdot)$ denotes information entropy. Since the data entropy $H(\mathcal{D})$ is constant for a given dataset $\mathcal{D}$ and independent of ϕ, we can omit $H(\mathcal{D})$ in Equation (2).

$$\max_\phi -H_\phi(\mathcal{D}|A) = \mathbb{E}_{p_\phi(\mathcal{D},A)}\left[\log p_\phi(\mathcal{D}|A)\right] \quad (3)$$

It can be extremely challenging to compute the mutual information between high-dimensional random variables. Thus, we obtain the lower bound of the mutual information by introducing a variational approximation $q_\theta(\mathcal{D}|A)$ to the true posterior distribution $p_\phi(\mathcal{D}|A)$.

$$\begin{aligned} I_\phi(\mathcal{D}, A) = H(\mathcal{D}) &+ \mathbb{E}_{p_\phi(\mathcal{D},A)}\left[\log q_\theta(\mathcal{D}|A)\right] \\ &+ D_{KL}(p_\phi(\mathcal{D}|A)||q_\theta(\mathcal{D}|A)) \\ \geq H(\mathcal{D}) &+ \mathbb{E}_{p_\phi(\mathcal{D},A)}\left[\log q_\theta(\mathcal{D}|A)\right] \end{aligned} \quad (4)$$

Equation (4) suggests that the lower bound is tight when $q_\theta(\mathcal{D}|A)$ matches $p_\phi(\mathcal{D}|A)$. In the view of NAS, the true posterior $p_\phi(\mathcal{D}|A)$ is the distribution of sub-network and is approximated with the distribution of super network $q_\theta(\mathcal{D}|A)$. Formally, the objective for differentiable NAS is

$$\max_{\theta,\phi} \mathbb{E}_{p_\phi(\mathcal{D},A)}\left[\log q_\theta(\mathcal{D}|A)\right], \quad (5)$$

where ϕ is the architectural parameter and θ is the network weight. Given arbitrary finite dataset $\mathcal{D}$, we estimate the expectations with respect to $p(\mathcal{D})$ and its gradients via Monte Carlo methods. The objective $\mathcal{L}(\phi, \theta; \mathcal{D})$ is formulated as

$$\max_{\theta,\phi} \mathcal{L}(\phi, \theta; \mathcal{D}) = \sum_{d \in \mathcal{D}} \mathbb{E}_{p_\phi(A|\mathcal{D})}\left[\log q_\theta(\mathcal{D}|A)\right]. \quad (6)$$

Existing differentiable NAS approaches [29, 39, 41] assume that architectural distribution on each edge is independent of each other. Thus, the true posterior $p_\phi(A|\mathcal{D})$ can be factorized in the form of $\prod_i p_{\phi_i}(A_i|\mathcal{D})$ to simplify the search process. A simple Monte Carlo sampling is leveraged to compute the expectation in Equation (6). However, this assumption obscures the natural dependencies among architectural parameters. As convolutional neural networks exhibit powerful ability in fitting arbitrary functions, we can leverage it to model the implicit distribution [33, 46] and achieve the simple sampling by forward propagation. Instead of fully factorizable assumption, we leverage convolutional neural network to model the architectural distribution $p_\phi(A|\mathcal{D})$. Since the single convolutional neural network $\phi(\cdot)$ can only conduct point estimation for architectural distribution, we further introduce an additive Gaussian noise $\xi \sim \mathcal{N}(0, 1)$, and reformulate the architectural distribution $p_\phi(A|\mathcal{D}) = \mathcal{N}(\mu_\phi, 1)$, where μ is parameterized by the convolutional neural network $\phi(\cdot)$. Algorithm 1 elaborates the implementation of VIM-NAS.

Algorithm 1 VIM-NAS

```
Input: Data $\mathcal{D} = \{x_n, y_n\}_{1:N}$, initialized network weights
    $\theta$, initialized architectural neural network parameters $\boldsymbol{\phi}$,
    and input Gaussian noise $\epsilon$.
Output: The searched final architecture.
1: while not converged do
2:     Sample Gaussian noise $\xi \sim \mathcal{N}(0, 1)$.
3:     Sample architecture $A = \boldsymbol{\phi}(\epsilon) + \xi$, $A \sim p_\phi(A|\mathcal{D})$.
4:     Update weights $\theta$ by descending $\nabla_\theta \mathcal{L}(\phi, \theta; \mathcal{D})$.
5:     Update network $\boldsymbol{\phi}$ by descending $\nabla_{\boldsymbol{\phi}} \mathcal{L}(\phi, \theta; \mathcal{D})$.
6: end while
7: Derive the final architecture based on the learned $\boldsymbol{\phi}$.
```

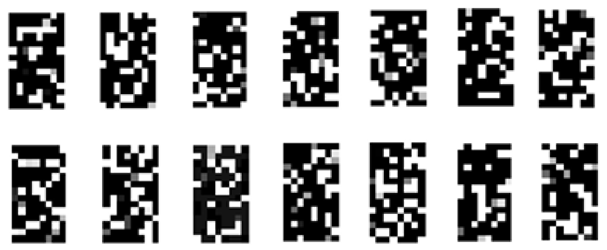
3.3. Architectural Neural Network

VIM-NAS leverages an architectural neural network ConvReLUBN(3,14,3)-ConvReLUBN(14,1,3) to model the architectural distribution, where ConvReLUBN denotes the module stacked with convolution, relu and batch normalization, and the following three numbers denote the input channel number, output channel number and kernel size, respectively. The details of architectural neural network can be referred to supplementary materials. We further take a microscopic view for this architectural neural network, and figure out what is learned by the simple convolutional neural network. We visualize the feature map of the ConvReLUBN(3,14,3) module in Figure 1. As we can see from Figure 1, the initialized feature map is dense and random, while a sparse connection is learned from convolutional neural network after training one epoch. This phenomenon indicates that the proposed simple convolutional neural network captures the dependencies among architectural distribution.

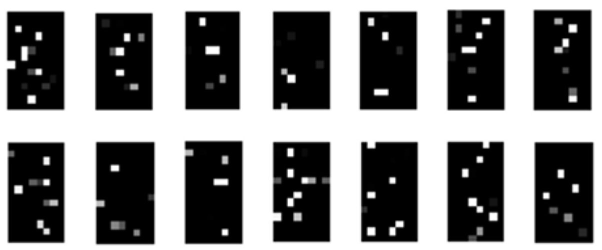
3.4. Convergence to Nearly One-hot Within 1 Epoch

To our best knowledge, VIM-NAS is the fastest differentiable NAS approaches up to now. VIM-NAS can reach the convergent result by only training one epoch in DARTS search space. Figures 2 and 3 demonstrate that VIM-NAS learns a nearly one-hot architectural distribution from the random initialized continuous distribution automatically after one epoch training.

Compared with DARTS, as shown in Figure 2, which exhibits oscillation and homogeneity among candidate operations during the search period, VIM-NAS approaches to the true posterior distribution of sub-network with a nearly one-hot architectural distribution. To verify the impact of initialization imposed on the convergent nearly one-hot architectural distribution, Figure 3 utilizes different colors for

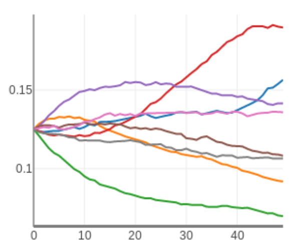


(a) Initialized feature map

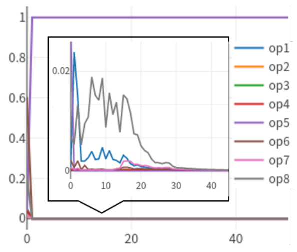


(b) Convergent feature map after one epoch

Figure 1. Comparison with initialized feature map and convergent feature map (one epoch training) of intermediate layer. In each sub figure, 14 channels of feature maps and each feature map shares the same size as candidate operations and edges.

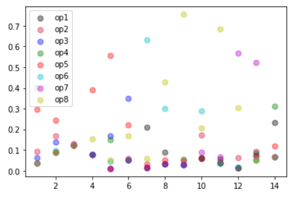


(a) DARTS-N

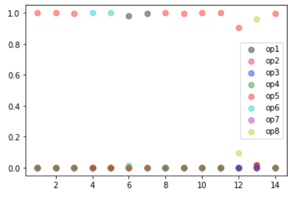


(b) VIM-NAS-N

Figure 2. Anytime architectural weights on the first edge of normal cell on DARTS search space for DARTS and VIM-NAS. 'N' denotes the searched normal cell(best viewed in color).



(a) Initialized architectural distribution



(b) Convergent architectural distribution after one epoch training

Figure 3. Contrast of architectural distribution between initialization and convergence after training one epoch.

a brief instruction that the information is learned from the dataset instead of random initialization.

3.5. Discussion

Understanding VIM-NAS in a GAN-alike Way. In generative models, a convolutional neural network is leveraged to learn the good representation from the noise for downstream tasks. Similarly, VIM-NAS utilizes a simple convolutional neural network ϕ to map the input noise to the target architectural distribution.

When considering to utilize the network ϕ for the modeling of architectural distribution, the optimization objective can be deemed as adversarial optimization of two networks. Specifically, the architectural neural network serves as a generator to learn a good representation of architectural distribution, while the super network acts as a discriminator to distinguish the architecture with good performance. In contrast to vanilla GAN, which utilizes the real data as the ground truth to optimize the discriminator, our VIM-NAS has no explicit good architectures as reference. Consequently, VIM-NAS leverages the training accuracy of a given dataset to optimize the discriminator.

Reformulation of Variational Dropout NAS. Furthermore, we also leverage the convolutional neural network $\varphi(\cdot)$ to parameterize the variance σ of Gaussian architectural distribution. Specifically, we achieve the parameterized architectural distribution as $p_{\phi,\varphi}(A|\mathcal{D}) = \mathcal{N}(\mu_\phi, \sigma_\varphi)$. We can utilize the reparameterization trick to sample from the distribution as $A = \mu_\phi + \sigma_\varphi \cdot \xi, \xi \sim \mathcal{N}(0, 1)$. Moreover, we can reinterpret the architectural distribution with variational dropout [23, 32] as $\mathcal{N}(\mu, \mu^2\delta)$, where $\delta = \sigma/\mu^2 = p/(1-p)$, and p denotes the Gaussian dropout rate. Following [37], a sparse constraint is also leveraged in our experiments. We name the reformulated variational dropout NAS as VIM-NAS-Dropout and report the results in Section 4.

4. Experiments

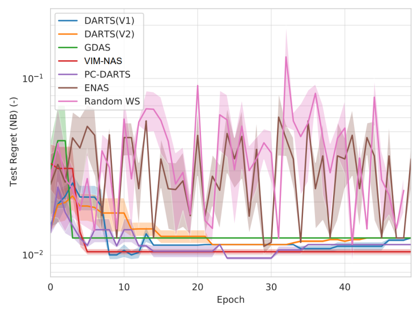
4.1. Datasets

We conduct experiments on CIFAR-10/100 [24] and ImageNet [12], three most popular datasets for evaluating NAS. CIFAR-10/100 consist of 60K images, 50K training images and 10K test images. ImageNet is a large-scale benchmark for image classification that contains 1.3M training images and 50K test images.

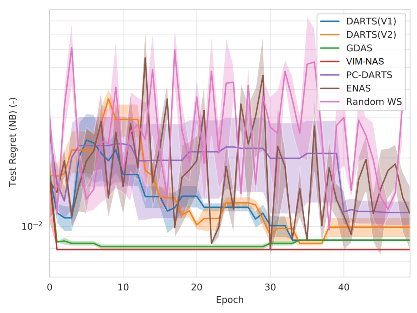
4.2. Experiments on DARTS Search Space

Search Space. Following DARTS [29], the macro architecture is constructed by stacking 6 normal cells and 2 reduction cells. Each cell contains seven nodes, including two input nodes, four intermediate nodes and one output node. The outputs of four intermediate nodes are concatenated as the input to the output node. Each cell has 14 candidate edges, 8 candidate operations for each edge, 'zero' operation is utilized to select candidate edges implicitly.

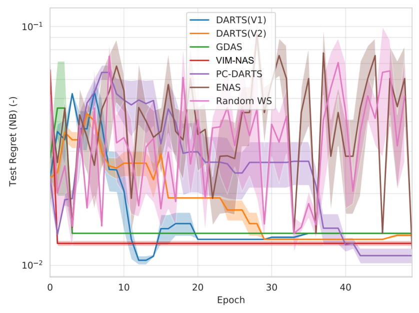
Search Settings. During the search process, DARTS directly utilizes 14×8 architectural parameters to describe the micro DAG. Instead, VIM-NAS leverages an architectural neural network to model the architectural distribution. The



(a) Search space 1

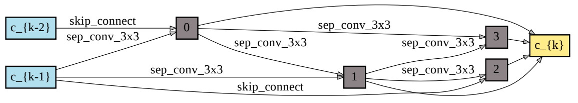


(b) Search space 2

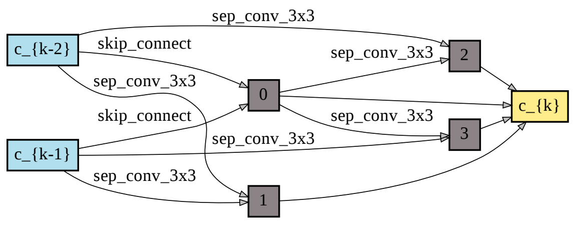


(c) Search space 3

Figure 4. Anytime test regret on NAS-Bench-1Shot1 (best viewed in color).



(a) Normal cell



(b) Reduction cell

Figure 5. Searched normal cell and reduction cell of VIM-NAS on DARTS search space on CIFAR-10.

Architecture	Top-1 (Test) Error (%) CIFAR-10	Top-1 (Test) Error (%) CIFAR-100	Params (M)	Cost (days)
DARTS-V1 [29]	3.00 ± 0.14	17.76*	3.3	0.4
DARTS-V2 [29]	2.76 ± 0.09	17.54*	3.3	1
P-DARTS [8]	2.50	16.55*	3.4	0.3
SNAS [41]	2.85 ± 0.02	-	2.8	1.5
PARSEC [5]	2.81 ± 0.03	-	3.7	1
BayesNAS [53]	2.81 ± 0.04	-	3.4	0.2
DATA (M = 7) [51]	2.79	-	2.9	1
PC-DARTS [42]	2.57 ± 0.07	-	3.6	0.1
ASNG-NAS [1]	2.83 ± 0.14	-	3.9	0.11
SI-VDNAS-C† [37]	2.60 ± 0.05	16.20	2.7	0.8
GDAS [15]	2.93	18.38	3.4	0.21
SDARTS-ADV [7]	2.61 ± 0.02	-	3.3	1.3
SGAS [25]	2.66 ± 0.24	-	3.7	0.25
DARTS- [10]	2.59 ± 0.08	-	3.5	0.4
TE-NAS [6]	2.63 ± 0.06	-	3.8	0.05
VIM-NAS	**2.45 ± 0.04**	**15.80**	3.9	**0.007**

Table 1. Comparison with state-of-the-art gradient-based NAS methods for image classification on CIFAR-10/100. For each method, top-1 test error (%), number of parameters (M) and search cost (GPU-days) are evaluated. Here, lower error rate stands for better performance and * indicates that the experiments are conducted by P-DARTS. † denotes that SI-VDNAS-C are the searched convergent cell.

architectural neural network parameters and conventional network weights are optimized with two separate momentum SGD optimizers, respectively. Since the architectural neural network is quite small, the extra computation cost is negligible. Other detailed settings for searching and evaluation are the same as DARTS (please refer to supplementary materials).

Results on CIFAR-10. Our proposed VIM-NAS reaches the convergent result after training only one epoch (within 10 minutes on a single NVIDIA GTX 1080 Ti GPU). The architectural distribution reaches almost one-hot and remains unchanged during further training. The searched normal cell and reduction cell on CIFAR-10 are shown in Figure 5. As in Table 1, VIM-NAS can achieve state-of-the-art performance with the top-1 test error of 2.45% and 15.80% on CIFAR-10 and CIFAR-100, respectively.

Results on ImageNet. We further transfer the searched architectures on CIFAR-10 to ImageNet to evaluate the generalization ability. As shown in Table 2, VIM-NAS achieves state-of-the-art performance with a top-1 error rate of 24.0% and a top-5 error rate of 7.2% compared with other popular gradient-based NAS approaches. Notice that we can outperform the directly searched architecture on ImageNet by PC-DARTS. Moreover, we conduct a direct search on ImageNet and achieve even better performance with a top-1 error rate of 23.8% and a top-5 error rate of 7.1%.

4.3. Experiments on NAS-Bench-201

Settings. Based on the reduced DARTS-like cell search space, NAS-Bench-201 [13] has 4 internal nodes with 5 operations per node, which constructs 15,625 architectures in total. All the architecture performance on three datasets (CIFAR-10, CIFAR-100, ImageNet-16-120 [9]) can be directly obtained by querying in the database as ground-truth. We use the same search setting with Section 4.2, and keep the hyperparameters for all compared methods the same as [13]. We run every method 4 independent times with different random seeds and report the results in Table 3.

Architecture	Test Error (%)		FLOPS	Search Cost
	Top-1	Top-5	(M)	(GPU-days)
DARTS (2nd) [29]	26.7	8.7	574	1
GDAS [15]	26.0	8.5	581	0.21
PARSEC [5]	26.0	8.4	-	1
PC-DARTS [42]	25.1	7.8	586	0.1
PC-DARTS† [42]	24.2	7.3	597	3.8
P-DARTS [8]	24.4	7.4	557	0.3
DARTS+† [27]	23.9	7.4	582	6.8
DARTS-† [10]	23.8	7.0	467	4.5
FairDARTS-B [11]	24.9	7.5	541	0.4
DSO-NAS-share [52]	25.4	8.4	586	6
SDARTS-ADV [7]	25.2	7.8	-	1.3
SGAS [25]	24.2	7.2	585	0.25
SparseNAS [40]	24.7	7.6	-	1
BayesNAS [53]	26.5	8.9	-	0.2
DATA (M = 7) [51]	24.9	8.1	-	1.5
SI-VDNAS-B [37]	25.3	8.0	577	0.3
TE-NAS [6]	26.2	8.3	-	0.05
TE-NAS† [6]	24.5	7.5	-	0.17
VIM-NAS	**24.0**	**7.2**	627	**0.007**
VIM-NAS†	**23.8**	**7.1**	660	**0.26**

Table 2. Comparison with state-of-the-art gradient-based NAS methods on ImageNet. For each method, top-1 and top-5 test errors (%), FLOPS (M) and search cost (GPU-days) are evaluated. Here, lower error rate stands for better performance and † indicates that the architecture is directly searched on ImageNet.

Results. During the search process on CIFAR-10, our proposed VIM-NAS can reach the convergent result after training only one epoch (232.51 seconds on a single NVIDIA GTX 1080 Ti GPU). Compared with Random baseline, our VIM-NAS shows better performance on all three datasets. Compared with other differentiable algorithms DARTS-V1 [29], DARTS-V2 [29], GDAS [15], SETN [14], DARTS- [10] and TE-NAS [6], our proposed VIM-NAS achieves a new state-of-the-art, the best of which almost touches the optimal. Moreover, we achieve state-of-the-art performance by training one epoch with much reduced search time, which is minimal among differentiable approaches. Though TE-NAS proposes a training-free strategy, the search cost is six times as large as VIM-NAS. Consequently, our VIM-NAS demonstrates more stable and superior performance on NAS-Bench-201.

4.4. Experiments on NAS-Bench-1Shot1

Settings. NAS-Bench-1Shot1 [49] consists of 3 search spaces based on CIFAR-10, which contains 6,240, 29,160 and 363,648 architectures, respectively. The macro architecture is constructed by stacking 3 blocks and each block contains 3 stacked cells. The micro architecture of each cell is represented as a DAG. The search algorithm needs to determine the operation on every edge, as well as the topology of edges connecting input, output nodes and the choice blocks. We leverage the architectural neural network (same as Section 4.2) to model the operation distribution and keep the other parameters the same as [49] to determine the topology. We compare our proposed methods with other popular NAS algorithms on all 3 search spaces. Each algorithm is trained for 50 epochs with three independent times, and the hyperparameters are set as defaults [49].

Results. The anytime test regret averaged from three independent runs are exhibited in Figure 4. Our proposed VIM-NAS can reach the convergent result after training for several epochs (2-5 epochs) in all three search spaces. Though the operation distribution parameterized by the architectural neural network reaches convergence in one epoch, the other topology parameters require several epochs to converge. Random Search with Weight Sharing [26] and ENAS [34] mainly find some poor performance architectures. GDAS [15] turns to converge prematurely to a sub-optimal local minimum. DARTS and PC-DARTS explore some better architecture as the search process goes gradually. Our proposed VIM-NAS achieves state-of-the-art performance after training several epochs in search spaces 1 and 2, respectively. For search space 3, VIM-NAS can also achieve a satisfactory result after training two epochs. Compared with GDAS, which demonstrates the premature convergence due to the temperature annealing of the Gumbel Softmax, our VIM-NAS consistently converges to a better local minimum across all three search spaces.

4.5. Experiments on Simplified Search Spaces S1-S4

Settings. RobustDARTS (R-DARTS) [48] proposed four simplified search spaces (S1-S4), which keep the same macro architecture as DARTS but only contain a portion of candidate operations (please refer to details in R-DARTS [48]). We search on CIFAR-10 with the same architectural neural network to model the architectural distribution as Section 4.2. There are two different evaluation settings from R-DARTS [48], SDARTS [7] and DARTS- [10]. Following R-DARTS [48], we use 20 cells with 36 initial channels for CIFAR-10 in S1 and S3, 20 cells with 16 initial channels for CIFAR-10 in S2 and S4, and 8 cells with 16 initial channels for CIFAR-100 in all four search spaces. Moreover, following SDARTS [7] and DARTS- [10], we evaluate the architecture performance using 20 cells with 36 initial channels for CIFAR-10 in S2 and S4, and 20 cells with 36 initial channels for CIFAR-100 in all four search spaces. We run every method 4 independent times and pick the final best architecture performance reported in Table 4.

Results. Our proposed VIM-NAS can converge after training one epoch (within 10 minutes), and the searched architecture performance outperforms the recent SOTAs across several spaces and datasets, which further demonstrate the robustness of VIM-NAS. Specifically, we find a good architecture in S1 with the lowest top-1 test error of 2.61% and 16.12% on CIFAR-10 and CIFAR-100, respectively. Be-

Method	Search Cost	CIFAR-10 (%)		CIFAR-100 (%)		ImageNet-16-120 (%)	
	(seconds)	validation	test	validation	test	validation	test
ResNet [21]	N/A	90.83	93.97	70.42	70.86	44.53	43.63
Random	0.01	90.93 ± 0.36	93.70 ± 0.36	70.60 ± 1.37	70.65 ± 1.38	42.92 ± 2.00	42.96 ± 2.15
Reinforce [54]	0.12	91.09 ± 0.37	93.85 ± 0.37	70.05 ± 1.67	70.17 ± 1.61	43.04 ± 2.18	43.16 ± 2.28
ENAS [34]	14058.80	39.77 ± 0.00	54.30 ± 0.00	10.23 ± 0.12	10.62 ± 0.27	16.43 ± 0.00	16.32 ± 0.00
DARTS (1st) [29]	11625.77	39.77 ± 0.00	54.30 ± 0.00	38.57 ± 0.00	38.97 ± 0.00	18.87 ± 0.00	18.41 ± 0.00
DARTS (2nd) [29]	35781.80	39.77 ± 0.00	54.30 ± 0.00	38.57 ± 0.00	38.97 ± 0.00	18.87 ± 0.00	18.41 ± 0.00
GDAS [15]	31609.80	89.89 ± 0.08	93.61 ± 0.09	71.34 ± 0.04	70.70 ± 0.30	41.59 ± 1.33	41.71 ± 0.98
SETN [14]	34139.53	84.04 ± 0.28	87.64 ± 0.00	58.86 ± 0.06	59.05 ± 0.24	33.06 ± 0.02	32.52 ± 0.21
DARTS- [10]	11625.77	91.03 ± 0.44	93.80 ± 0.40	71.36 ± 1.51	71.53 ± 1.51	44.87 ± 1.46	45.12 ± 0.82
TE-NAS [6]	1558	-	93.90 ± 0.47	-	71.24 ± 0.56	-	42.38 ± 0.46
Ours	232.51	$\mathbf{91.48 \pm 0.09}$	$\mathbf{94.31 \pm 0.11}$	$\mathbf{73.12 \pm 0.51}$	$\mathbf{73.07 \pm 0.58}$	$\mathbf{45.92 \pm 0.51}$	$\mathbf{46.27 \pm 0.17}$
Ours (best)	232.51	91.55	94.36	73.49	73.51	46.37	46.34
Optimal	N/A	91.61	94.37	73.49	73.51	46.77	47.31

Table 3. Comparison with state-of-the-art NAS methods on NAS-Bench-201 [13]. Averaged on 4 runs of searching.

Benchmark		DARTS†	R-DARTS†		DARTS†		DARTS-†	Ours†	PC-DARTS‡	SDARTS‡		DARTS-‡	Ours‡
			DP	L2	ES	ADA				RS	ADV		
C10	S1	3.84	3.11	2.78	3.01	3.10	2.68	**2.61**	3.11	2.78	2.73	2.68	**2.61**
	S2	4.85	3.48	3.31	3.26	3.35	3.71	**3.22**	3.02	2.75	2.65	2.63	**2.53**
	S3	3.34	2.93	2.51	2.74	2.59	**2.42**	**2.42**	2.51	2.53	2.49	**2.42**	**2.42**
	S4	7.20	3.58	3.56	3.71	4.84	3.88	**3.55**	3.02	2.93	2.87	2.86	**2.85**
C100	S1	29.46	25.93	24.25	28.37	24.03	22.41	**22.07**	18.87	17.02	16.88	16.92	**16.12**
	S2	26.05	22.30	22.24	23.25	23.52	21.61	**20.90**	18.23	17.56	17.24	**16.14**	16.35
	S3	28.90	22.36	23.99	23.73	23.37	21.13	**21.11**	18.05	17.73	17.12	**15.86**	15.94
	S4	22.85	22.18	21.94	21.26	23.20	21.55	**21.01**	17.16	17.17	**15.46**	17.48	17.39

Table 4. Comparison in various search spaces. We report the lowest error rate (%) of 4 found architectures. ‡: under [7, 10] evaluation settings where all models have 20 layers and 36 initial channels. †: under [48] settings where CIFAR-10 models in S2 and S4 have 20 layers and 16 initial channels, and CIFAR-100 models have 8 layers and 16 initial channels.

sides, VIM-NAS achieves state-of-the-art performance in S2 with the top-1 test error of 2.53% on CIFAR-10 and 20.90% on CIFAR-100 under small evaluation settings. In S3, we obtain a well-performed architecture with state-of-the-art top-1 test error of 2.42% on CIFAR-10 and 21.11% on CIFAR-100. Moreover, VIM-NAS achieves a top-1 error rate of 2.85% on CIFAR-10 and 21.01% on CIFAR-100 in S4. The architectures of all these models can be found in the supplementary material.

4.6. Extensive Experiments

Since the DARTS search space is the most popular search space, we conduct extensive experiments on it.

Ablation Study on Architectural Network. To demonstrate the effectiveness of our proposed architectural neural network, we further evaluate DARTS† with the vanilla 14×8 architectural parameters, but with the same pretraining stage, posterior Gaussian distribution with covariance as 1 and the same learning rate (0.025) for the architectural neural network as VIM-NAS. For a better understanding of the convergence of search algorithms, we visualize the architectural weights for the first edge of normal cell and reduction cell.

As seen in Figure 6, vanilla DARTS converges slowly and suffers from high uncertainty lying in the homogenization of architectural parameters. Though with the same large learning rate, DARTS† fails to converge quickly and turns to behave like noise. As seen in Figure 8, Our proposed VIM-NAS can achieve fast convergence to a well-performed local minimum after training one epoch. Consequently, it is our architectural neural network design that contributes to the fast and stable convergence.

To verify the design of the architectural network, we further implement a small architectural neural network with a single ConvReLUBN module and a large architectural neural network with five modules, which are denoted as VIM-NAS-Small and VIM-NAS-Large, respectively. Please refer to the supplementary material for the detailed network structure. The architectural weights for the first edge of the normal cell are visualized in Figure 7, and the evaluation performance is listed in Table 5. VIM-NAS-Small exhibits slightly poor performance and unstable convergence with large fluctuation due to the limited capacity of a small network. VIM-NAS-Large turns to gradually converge after training several epochs, since the large architectural neural network needs a longer period of training.

Extensive Experiments on Approximating Posterior Architectural Distribution. Instead of leveraging the proba-

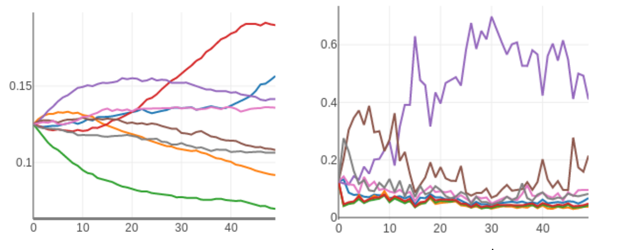

(a) DARTS-N (b) DARTS†-N

Figure 6. Anytime architectural weights on DARTS search space. 'N' denotes the searched normal cell. DARTS†: DARTS is implemented with high learning rate (0.025) and added noise.

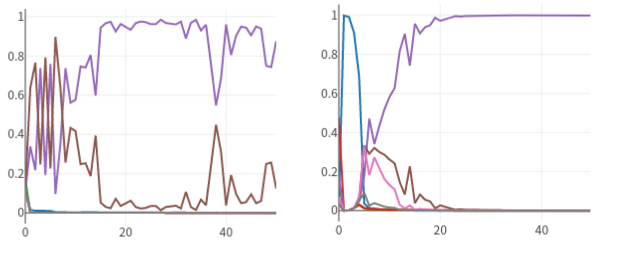

(a) VIM-NAS-S-N (b) VIM-NAS-L-N

Figure 7. Anytime architectural weights on DARTS search space. 'N' denotes the searched normal cell. VIM-NAS-S: VIM-NAS implemented with a small architectural network. VIM-NAS-L: VIM-NAS implemented with a large architectural network.

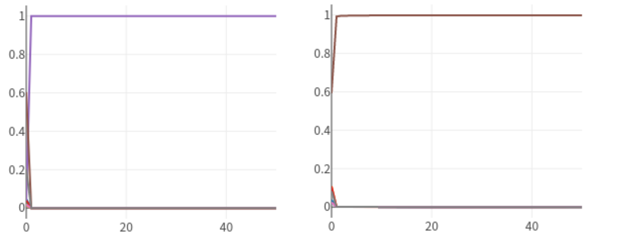

(a) VIM-NAS-N (b) VIM-NAS-R

Figure 8. Anytime architectural weights on DARTS search space. 'N' and 'R' denote normal cell and reduction cell, respectively.

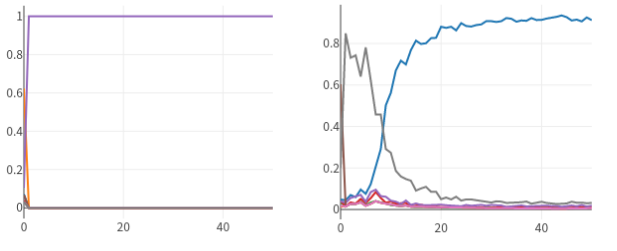

(a) VIM-NAS-D-N (b) VIM-NAS-P-N

Figure 9. Anytime architectural weights on DARTS search space. 'N' denotes the searched normal cell. VIM-NAS-D: reformulation of variational dropout NAS with VIM. VIM-NAS-P-N: VIM-NAS implemented with point estimation.

bility estimation to model the architectural distribution in Section 3.2, we directly utilize the convolutional neural network as a point estimation to approximate the archi-

Method	DARTS	DARTS†	VIM-NAS
Top-1 Error (%)	3.00	2.65	2.45
Params (M)	3.3	3.6	3.9
Method	VIM-NAS-S	VIM-NAS-L	VIM-NAS-D
Top-1 Error (%)	2.51	2.58	2.58
Params (M)	3.6	3.7	3.3

Table 5. Extensive experiments on CIFAR10. DARTS†: DARTS implemented with high learning rate (0.025) and added noise. VIM-NAS-S: VIM-NAS implemented with a small architectural network. VIM-NAS-L: VIM-NAS implemented with a large architectural network. VIM-NAS-D: reformulation of variational dropout NAS with variational information maximization.

tectural distribution, namely VIM-NAS-P. Furthermore, we also leverage the convolutional neural network to parameterize the variance of the posterior architectural distribution, which is deemed as a reformulation of variational dropout NAS in Section 3.5, namely VIM-NAS-Dropout.

The architectural weights for the first edge of searched normal cells are visualized in Figure 9. The point estimation is less effective, and VIM-NAS-P turns to converge slowly with large fluctuation. VIM-NAS-Dropout can also achieve fast and stable convergence as VIM-NAS.

5. Conclusion

In this paper, we provide new insights into NAS that the architectural distribution is the latent representation of a given dataset. Then, we leverage a simple yet effective convolutional neural network to model the dependencies among architectural distribution. Moreover, we propose a novel search strategy to maximize the variational lower bound to the mutual information between the data points and the latent architectural representations. Experimental results demonstrate VIM-NAS exhibits extremely fast convergence speed within one epoch, and achieves state-of-the-art performance on various search spaces, including DARTS search space, NAS-Bench-1shot1, NAS-Bench-201 and simplified search spaces S1-S4. Specifically, VIM-NAS achieves a top-1 error rate of 2.45% and 15.80% within 10 minutes on CIFAR-10 and CIFAR-100, respectively. When transferred to ImageNet, VIM-NAS reaches a 24.0% top-1 error rate. Moreover, a direct search on ImageNet achieves even better performance with a top-1 error rate of 23.8% and a top-5 error rate of 7.1%.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China under Grants 61932022, 61720106001, 61931023, 61831018, 61971285, 61871267, 61972256, 61838303, and in part by the Program of Shanghai Science and Technology Innovation Project under Grant 20511100100.

References

[1] Youhei Akimoto, Shinichi Shirakawa, Nozomu Yoshinari, Kento Uchida, Shota Saito, and Kouhei Nishida. Adaptive stochastic natural gradient method for one-shot neural architecture search. In *Proceedings of the 36th International Conference on Machine Learning*, pages 171–180, 2019.

[2] Qian Bao, Wu Liu, Jun Hong, Lingyu Duan, and Tao Mei. Pose-native network architecture search for multi-person human pose estimation. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 592–600, 2020.

[3] Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc Le. Understanding and simplifying one-shot architecture search. In *Proceedings of the 35th International Conference on Machine Learning*, pages 549–558, 2018.

[4] Andrew Brock, Theo Lim, J.M. Ritchie, and Nick Weston. SMASH: One-shot model architecture search through hypernetworks. In *6th International Conference on Learning Representations*, 2018.

[5] Francesco Paolo Casale, Jonathan Gordon, and Nicolo Fusi. Probabilistic neural architecture search. *arXiv preprint arXiv:1902.05116*, 2019.

[6] Wuyang Chen, Xinyu Gong, and Zhangyang Wang. Neural architecture search on ImageNet in four GPU hours: A theoretically inspired perspective. In *9th International Conference on Learning Representations*, 2021.

[7] Xiangning Chen and Cho-Jui Hsieh. Stabilizing differentiable architecture search via perturbation-based regularization. In *Proceedings of the 37th International Conference on Machine Learning*, pages 1554–1565, 2020.

[8] Xin Chen, Lingxi Xie, Jun Wu, and Qi Tian. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1294–1303, 2019.

[9] Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. A downsampled variant of ImageNet as an alternative to the CIFAR datasets. *arXiv preprint arXiv:1707.08819*, 2017.

[10] Xiangxiang Chu, Xiaoxing Wang, Bo Zhang, Shun Lu, Xiaolin Wei, and Junchi Yan. DARTS-: Robustly stepping out of performance collapse without indicators. In *9th International Conference on Learning Representations*, 2021.

[11] Xiangxiang Chu, Tianbao Zhou, Bo Zhang, and Jixiang Li. Fair DARTS: Eliminating unfair advantages in differentiable architecture search. In *16th European Conference on Computer Vision*, pages 465–480, 2020.

[12] 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*, pages 248–255, 2009.

[13] Xuanyi Dong and Yi Yang. NAS-Bench-201: Extending the scope of reproducible neural architecture search. In *7th International Conference on Learning Representations*, 2019.

[14] Xuanyi Dong and Yi Yang. One-shot neural architecture search via self-evaluated template network. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3681–3690, 2019.

[15] Xuanyi Dong and Yi Yang. Searching for a robust neural architecture in four GPU hours. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1761–1770, 2019.

[16] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Efficient multi-objective neural architecture search via Lamarckian evolution. In *7th International Conference on Learning Representations*, 2019.

[17] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *Journal of Machine Learning Research*, 20:1–21, 2019.

[18] Jiemin Fang, Yuzhu Sun, Qian Zhang, Kangjian Peng, Yuan Li, Wenyu Liu, and Xinggang Wang. FNA++: Fast network adaptation via parameter remapping and architecture search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(9):2990–3004, 2020.

[19] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V. Le. NAS-FPN: Learning scalable feature pyramid architecture for object detection. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7036–7045, 2019.

[20] Jianyuan Guo, Kai Han, Yunhe Wang, Chao Zhang, Zhaohui Yang, Han Wu, Xinghao Chen, and Chang Xu. Hit-Detector: Hierarchical trinity architecture search for object detection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11405–11414, 2020.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[22] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-Softmax. In *Workshop on Bayesian Deep Learning, NIPS 2016*, 2016.

[23] Durk P. Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems 28*, pages 2575–2583, 2015.

[24] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[25] Guohao Li, Guocheng Qian, Itzel C. Delgadillo, Matthias Muller, Ali Thabet, and Bernard Ghanem. SGAS: Sequential greedy architecture search. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1620–1630, 2020.

[26] Liam Li and Ameet Talwalkar. Random search and reproducibility for neural architecture search. In *Uncertainty in Artificial Intelligence*, pages 367–377, 2020.

[27] 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.

[28] Chenxi Liu, Liang-Chieh Chen, Florian Schroff, Hartwig Adam, Wei Hua, Alan L. Yuille, and Li Fei-Fei. Auto-DeepLab: Hierarchical neural architecture search for semantic image segmentation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 82–92, 2019.

[29] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. In *7th International Conference on Learning Representations*, 2019.

[30] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *Workshop on Bayesian Deep Learning, NIPS 2016*, 2016.

[31] Risto Miikkulainen, Jason Liang, Elliot Meyerson, Aditya Rawal, Daniel Fink, Olivier Francon, Bala Raju, Hormoz Shahrzad, Arshak Navruzyan, Nigel Duffy, and Babak Hodjat. Evolving deep neural networks. In *Artificial Intelligence in the Age of Neural Networks and Brain Computing*, pages 293–312. Elsevier, 2019.

[32] Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. Variational dropout sparsifies deep neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 2498–2507, 2017.

[33] Dmitry Molchanov, Valery Kharitonov, Artem Sobolev, and Dmitry Vetrov. Doubly semi-implicit variational inference. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, pages 2593–2602, 2019.

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

[35] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. Regularized evolution for image classifier architecture search. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 4780–4789, 2019.

[36] Dilin Wang, Meng Li, Chengyue Gong, and Vikas Chandra. AttentiveNAS: Improving neural architecture search via attentive sampling. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6418–6427, 2021.

[37] Yaoming Wang, Wenrui Dai, Chenglin Li, Junni Zou, and Hongkai Xiong. SI-VDNAS: Semi-implicit variational dropout for hierarchical one-shot neural architecture search. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, pages 2088–2095, 2020.

[38] Martin Wistuba, Ambrish Rawat, and Tejaswini Pedapati. A survey on neural architecture search. *CoRR*, abs/1905.01392, 2019.

[39] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. FBNet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10734–10742, 2019.

[40] Yan Wu, Aoming Liu, Zhiwu Huang, Siwei Zhang, and Luc Van Gool. Neural architecture search as sparse supernet. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pages 10379–10387, 2021.

[41] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. SNAS: stochastic neural architecture search. In *6th International Conference on Learning Representations*, 2018.

[42] 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 *8th International Conference on Learning Representations*, 2020.

[43] Shen Yan, Yu Zheng, Wei Ao, Xiao Zeng, and Mi Zhang. Does unsupervised architecture representation learning help neural architecture search? In *Advances in Neural Information Processing Systems 32*, pages 12486–12498, 2020.

[44] Zhaohui Yang, Yunhe Wang, Xinghao Chen, Boxin Shi, Chao Xu, Chunjing Xu, Qi Tian, and Chang Xu. CARS: Continuous evolution for efficient neural architecture search. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1829–1838, 2020.

[45] Lewei Yao, Hang Xu, Wei Zhang, Xiaodan Liang, and Zhenguo Li. SM-NAS: Structural-to-modular neural architecture search for object detection. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pages 12661–12668, 2020.

[46] Mingzhang Yin and Mingyuan Zhou. Semi-implicit variational inference. In *Proceedings of the 35th International Conference on Machine Learning*, pages 5660–5669, 2018.

[47] Jiahui Yu, Pengchong Jin, Hanxiao Liu, Gabriel Bender, Pieter-Jan Kindermans, Mingxing Tan, Thomas Huang, Xiaodan Song, Ruoming Pang, and Quoc Le. BigNAS: Scaling up neural architecture search with big single-stage models. In *16th European Conference on Computer Vision*, pages 702–717, 2020.

[48] Arber Zela, Thomas Elsken, Tonmoy Saikia, Yassine Marrakchi, Thomas Brox, and Frank Hutter. Understanding and robustifying differentiable architecture search. In *7th International Conference on Learning Representations*, 2019.

[49] Arber Zela, Julien Siems, and Frank Hutter. NAS-Bench-1Shot1: Benchmarking and dissecting one-shot neural architecture search. In *7th International Conference on Learning Representations*, 2019.

[50] Wenqiang Zhang, Jiemin Fang, Xinggang Wang, and Wenyu Liu. EfficientPose: Efficient human pose estimation with neural architecture search. *Computational Visual Media*, 7:335–347, 2021.

[51] Xinbang Zhang, Jianlong Chang, Yiwen Guo, Meng Gaofeng, Shiming Xiang, Zhouchen Lin, and Chunhong Pan. DATA: Differentiable ArchiTecture Approximation with distribution guided sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(9):2905–2920, 2020.

[52] Xinbang Zhang, Zehao Huang, Naiyan Wang, Shiming Xiang, and Chunhong Pan. You only search once: Single shot neural architecture search via direct sparse optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(9):2891–2904, 2020.

[53] Hongpeng Zhou, Minghao Yang, Jun Wang, and Wei Pan. BayesNAS: A Bayesian approach for neural architecture search. In *Proceedings of the 36th International Conference on Machine Learning*, pages 7603–7613, 2019.

[54] 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*, pages 8697–8710, 2018.