Shapley-NAS: Discovering Operation Contribution for Neural Architecture Search

Han Xiao[1,2], Ziwei Wang[1,2], Zheng Zhu[1,2], Jie Zhou[1,2], Jiwen Lu[1,2]*
[1]Department of Automation, Tsinghua University, China
[2]Beijing National Research Center for Information Science and Technology, China
{h-xiao20,wang-zw18}@mails.tsinghua.edu.cn; zhengzhu@ieee.org; {jzhou,lujiwen}@tsinghua.edu.cn

A. Detailed Algorithms

A.1. The Monte-Carlo sampling algorithm for Shapley value estimation

The complete algorithm of Monte-Carlo sampling with early truncation for Shapley value estimation is illustrated in Algorithm 1. As mentioned in Section 3.3 (main body), Eq. (3) can be equivalently formulated as Eq. (4) if we consider all the possible joining orders of the operations [2]. In each joining order, when an operation is added to the supernet, it makes a marginal contribution compared with the operations that have been added before it. Viewing this process as a random variable, we can utilize the Monte-Carlo method by sampling from the uniform distribution over the set of all $N!$ permutations $\pi(N)$, where each permutation represents a distinct joining order. For each new operation in the permutation, we remove it from the supernet and calculate the performance drop caused by only removing this operation. This marginal contribution of the operation is one Monte-Carlo sample of its Shapley value. We repeat the same process over multiple Monte Carlo permutations and take the average of all marginal contributions to derive the final estimation of the Shapley values.

A.2. The full algorithm of Shapley-NAS

The iterative procedure of our Shapley-NAS is shown in Algorithm 2. We first pre-train a supernet by only fine-tuning its network weight $\boldsymbol{w}$ while keeping architecture parameters $\boldsymbol{\alpha}$ frozen. This warm-up process is essential for the initialized Shapley estimation and we keep $\boldsymbol{\alpha}$ frozen to ensure a reasonable computation. After that, we iteratively optimize the network weight $\boldsymbol{w}$ and the mixing operation weight $\boldsymbol{\alpha}$ according to its Shapley value estimated by Algorithm 1. The momentum update is introduced to reduce the fluctuation of the Monte-Carlo sampling. After the search stage, we select the operation with the largest accumulated Shapley value on each edge to get the final architecture.

*Corresponding author

Algorithm 1: Shapley value Estimation

Input: Supernet components $N = \mathcal{O} \times \mathcal{E} = \{o^{(i,j)}\}_{o\in\mathcal{O},(i,j)\in\mathcal{E}}$, performance evaluation metric V, sampling times M, early truncation threshold η

Output: Shapley value of operations $\{\phi_o^{(i,j)}\}_{o\in\mathcal{O},(i,j)\in\mathcal{E}}$

Initialization: Shapley of operation $\{\phi_o^{(i,j)}\} = 0$, $t = 0$.

```
while t < M do
    Randomly generate a permutation R of N;
    v_0 = V(N);
    for k = 1, 2, ..., |N| do
        if v_{k-1} > η · V(N) and R[k] ≠ zero then
            mask out operation R[k], re-evaluate the
              validation accuracy V and update v_k:
              v_k = V(R[k+1], R[k+2], ..., R[n]);
        end
        else
            v_k = v_{k-1}
        end
        φ_{R[k]} = φ_{R[k]} + (v_{k-1} − v_k)
    end
end
```

Return $\phi_o^{(i,j)} = \phi_o^{(i,j)}/M$, for $o^{(i,j)} \in N$.

B. Experimental Results

B.1. Training Details

CIFAR-10: The CIFAR-10 dataset [7] includes 60K images equally divided into 10 classes, all of which are with the size of 32×32. We keep the same operation space $\mathcal{O}$ as DARTS, including 3×3 and 5×5 separable convolutions, 3×3 and 5×5 dilated separable convolutions, 3×3 max pooling, 3×3 average pooling, skip connect (i.e., identity) and zero (i.e., none). At the search phase, we construct

Algorithm 2: Shapley-NAS

Input: Initialized supernet weights $\boldsymbol{w}_0$ and architecture parameters $\boldsymbol{\alpha}_0$, warm-up epochs $\mathcal{T}_1$, search epochs $\mathcal{T}_2$, momentum efficient μ, step size ϵ.

Output: The final architecture with chosen operation on every edge $\{o_{(i,j)}\}$.

Initialization:accumulated Shapley $\boldsymbol{s}_0 = 0$, $t = 0$.

Stage 1 (Warm-up)

while $t < \mathcal{T}_1$ **do**

  Update supernet weights $\boldsymbol{w}_t$ by descending $\nabla_{\boldsymbol{w}} \mathcal{L}_{train}(\boldsymbol{w}_{t-1}, \boldsymbol{\alpha}_{t-1})$;

  $t = t + 1$;

end

Stage 2 (Architecture Search)

while $t < \mathcal{T}_2$ **do**

  Update supernet weights $\boldsymbol{w}_t$ by descending $\nabla_{\boldsymbol{w}} \mathcal{L}_{train}(\boldsymbol{w}_{t-1}, \boldsymbol{\alpha}_{t-1})$;

  Estimate the Shapley value $\boldsymbol{\phi}(Acc_{val}(\boldsymbol{w}_{t-1}, \boldsymbol{\alpha}_{t-1}))$ by Monte-Carlo sampling according to Algorithm 1;

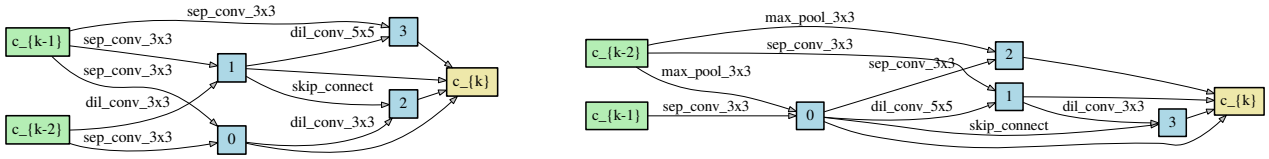
  Compute the accumulated Shapley value: $\boldsymbol{s}_t = \mu \cdot \boldsymbol{s}_{t-1} + (1-\mu) \cdot \frac{\boldsymbol{\phi}(Acc_{val}(\boldsymbol{w}_{t-1}, \boldsymbol{\alpha}_{t-1}))}{||\boldsymbol{\phi}(Acc_{val}(\boldsymbol{w}_{t-1}, \boldsymbol{\alpha}_{t-1}))||_2}$;

  Update architecture parameters: $\boldsymbol{\alpha}_t = \boldsymbol{\alpha}_{t-1} + \epsilon \cdot \frac{\boldsymbol{s}_t}{||\boldsymbol{s}_t||_2}$;

  $t = t + 1$;

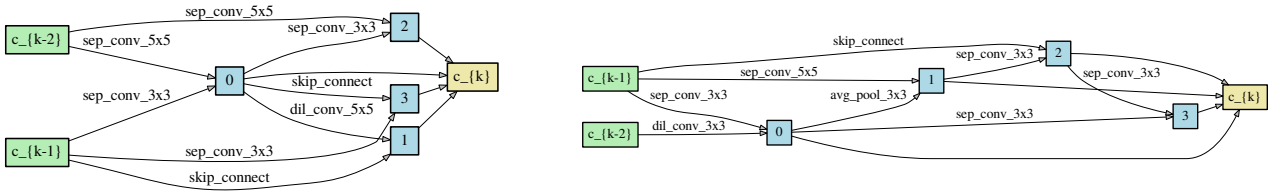
end

Derive the final architecture through argmax: $o^{(i,j)} = \arg\max_{o \in \mathcal{O}} \alpha_o^{(i,j)}$.



(a) Normal Cell (b) Reduction Cell

Figure 6. Normal and Reduction cells discovered by Shapley-NAS on CIFAR-10



(a) Normal Cell (b) Reduction Cell

Figure 7. Normal and Reduction cells discovered by Shapley-NAS on ImageNet

the supernet by stacking 8 cells (6 normal cells and 2 reduction cells) and set the initial channel number as 16. Each cell has $N = 7$ nodes (2 input nodes, 4 intermediate nodes, and 1 output node). The reduction cells are placed at the $1/3$ and $2/3$ of the total depth of the network. At the evaluation phase, we stack 20 cells including 18 normal cells and 2 reduction cells with the initial channel number being 36 to form the architecture. Then we retrain the network from scratch for 600 epochs on the entire 50K training set. We employ the SGD optimizer with a cosine annealing learning rate initialized as 0.025, a momentum of 0.9, and a weight decay of 3×10^{-4}. We also use the cutout with length 16 [5] and drop-path [9] with a rate of 0.3 for regularization.

ImageNet: Different from the architecture for CIFAR-10, the network for ImageNet [4] starts with three convolution layers with stride of 2 which reduce the input resolution from 224×224 to 28×28 following previous works [3, 8]. At the evaluation stage, the network is composed of 14 cells (18 normal cells and 2 reduction cells) and the initial channel number is 48. We train the network from scratch for 250

Table 6. Ablation study on PPI dataset.

Shapley value evaluation	MC sampling	Optimizer	Micro-F1 (%)	Search Cost (GPU days)
✘	✘	✘	98.58± 0.38	0.002
✔	✘	✘	99.39± 0.11	0.9
✔	✔	✘	99.34± 0.15	0.002
✔	✘	✔	99.45± 0.06	0.9
✔	✔	✔	99.43± 0.08	0.002

epochs by an SGD optimizer with a linearly decayed learning rate initialized as 0.5, a momentum of 0.9, and a weight decay of 3×10^{-5}. Similar to previous works [3, 8], label smoothing and an auxiliary loss tower are employed during the training.

NAS-Bench-201: In the search space of NAS-Bench-201 [6], the operation set $\mathcal{O}$ has 5 elements (zero, skip connection, 1×1 and 3×3 convolution, and 3×3 average pooling) and each cell contains 4 nodes, which results in a total search space of 15,625 architectures. NAS-Bench-201 supports three datasets, CIFAR-10, CIFAR-100, and ImageNet-16-120, and we use the results obtained by training 12 epochs on CIFAR-10, and 200 epochs on CIFAR-100 and ImageNet-16-120. Specifically, we evaluate the task-specific performance by directly searching on the evaluation dataset. We keep the hyper-parameters in the search and evaluation phase the same as CIFAR-10 and report the mean and standard deviation for the best architecture from 4 independent runs with different random seeds.

B.2. Visualization of searched cells

We visualize the best normal and reduction cells discovered by our Shapley-NAS. Fig. 6 shows the best cells found on CIFAR-10 [7], which yield 2.43% test error using 3.6M parameters. Fig. 7 shows the best cells found on ImageNet [4], which achieve 23.9% top-1 test error with 5.3M parameters and 582M multiply-add operations. The best reduction cells are deeper than the normal cells found on both datasets, which indicates reduction cells might need more complex architectures compared with normal cells since the purpose of reduction cells is to reduce the feature map dimensions. Meanwhile, both types of the found cells on ImageNet are deeper than on CIFAR-10, which suggests the optimal architecture for ImageNet is more sophisticated since the image classification task on ImageNet is much more challenging than on CIFAR-10.

B.3. Additional Experimental Results

Contributions of different components in Shapley-NAS. We conduct a systematic ablation study to evaluate the contributions of three components in Shapley-NAS, i.e. Shapley value as the optimization objective, Monte-Carlo sampling for approximation, and the specific optimizer in section 3.4. Since the exact computation of Shapley value on large datasets may by expensive, we search for the GCN architecture for node classification on a small dataset PPI.

Table 7. Results on ImageNet using MobileNet-like search space.

Architecture	Top-1 Acc (%)	Params (M)	×+ (M)	Search Cost (GPU days)
FairNAS-A	75.3	4.6	388	12
Proxyless	75.1	7.1	465	8.3
FairDARTS-D	75.6	4.3	440	3
SPOS	74.8	5.4	472	-
Shapley-NAS	76.1	4.8	468	3.2

As shown in Tab. 6, only Shapley value evaluation improves the Micro-F1 score by 0.81% but leads to heavy search cost. Monte-Carlo sampling significantly reduces that cost while maintaining comparable performance. The optimizer can further improve the result of architecture search in both accuracy and stability.

Experiments on MobileNet-like search space. We conduct further experiments on ImageNet using MobileNet-like search space proposed in ProxylessNAS [1]. For fair comparison, we restrict the FLOPs of the searched model to be less than 470M. As shown in Tab. 7, our Shapley-NAS obtains 76.1% top-1 accuracy with only 468M FLOPs, verifying that our method can achieve effective results over different search spaces.

References

[1] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. *arXiv preprint arXiv:1812.00332*, 2018. 3

[2] Javier Castro, Daniel Gómez, and Juan Tejada. Polynomial calculation of the shapley value based on sampling. *Comput. Oper. Res.*, 36(5):1726–1730, 2009. 1

[3] Xin Chen, Lingxi Xie, Jun Wu, and Qi Tian. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. In *ICCV*, pages 1294–1303, 2019. 2, 3

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009. 2, 3

[5] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 2

[6] Xuanyi Dong and Yi Yang. Nas-bench-201: Extending the scope of reproducible neural architecture search. *arXiv preprint arXiv:2001.00326*, 2020. 3

[7] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 1, 3

[8] 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. *arXiv preprint arXiv:1907.05737*, 2019. 2, 3

[9] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *CVPR*, pages 8697–8710, 2018. 2