SHAPLEY-NAS: DISCOVERING OPERATION CONTRIBUTION FOR NEURAL ARCHITECTURE SEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper, we propose a Shapley value based operation contribution evaluation method (Shapley-NAS) for neural architecture search. Differentiable architecture search (DARTS) acquires the expected architectures by optimizing the architecture parameters with gradient descent, which benefits from the high efficiency due to the significantly reduced search cost. However, DARTS leverages the learnable architecture parameters of the supernet to represent the operation importance during the search process, which fails to reveal the actual impacts of operations on the task performance and therefore harms the effectiveness of obtained architectures. On the contrary, we evaluate the direct influence of operations on accuracy via Shapley value for supernet optimization and architecture discretization, so that the optimal architectures are acquired by selecting the operations that contribute significantly to the tasks. Specifically, we iteratively employ Monte-Carlo sampling based algorithm with early truncation to efficiently approximate the Shapley value of operations, and update weights of the supernet whose architecture parameters are assigned with the operation contribution evaluated by Shapley value. At the end of the search process, operations with the largest Shapley value are preserved to form the final architecture. Extensive experiments on CIFAR-10 and ImageNet for image classification and on NAS-Bench-201 for optimal architecture search show that our Shapley-NAS outperforms the state-of-the-art methods by a sizable margin with light search cost.

1 INTRODUCTION

Neural architecture search (NAS) has attracted great interest in deep learning since it discovers the optimal structure from a large search space of network components according to task performance and hardware configurations. However, pioneering works applied reinforcement learning (Zoph & Le, 2016), evolutionary algorithms (Real et al., 2019; Wang et al., 2020) and Bayesian optimization (Liu et al., 2018a) for the architecture search, and the large computational overhead causes heavy search burden that prohibits practical deployment of NAS algorithms. Therefore, it is desirable to design highly efficient search strategies without performance degradation.

To reduce the search cost of architecture search, several efficient search strategies have been presented including one-shot NAS (Pham et al., 2018), network transformation (Cai et al., 2018a) and architecture optimization (Luo et al., 2018). Among these approaches, one-shot NAS preserves the optimal sub-networks from the over-parameterized supernet with weight sharing, which prevents the time-consuming exhaustive training for model evaluation. In particular, DARTS (Liu et al., 2018b) converted the discrete operation selection into continuous mixing weights, and utilized the gradient descent to simultaneously optimize the architecture parameters and supernet weights with significantly reduced search cost. However, DARTS methods leverage the learnable architecture parameters to represent the operation importance during search process, which fails to reflect the actual contribution of operations to task performance (Wang et al., 2021b) and degrades the effectiveness of acquired architectures.

In this paper, we present a Shapley-NAS method to evaluate the operation contribution via the Shapley value of supernet components for neural architecture search. Unlike existing methods which leverage the learnable architecture parameters to represent the operation importance in joint optimization of supernet weights and architecture parameters, we directly evaluate operation influence

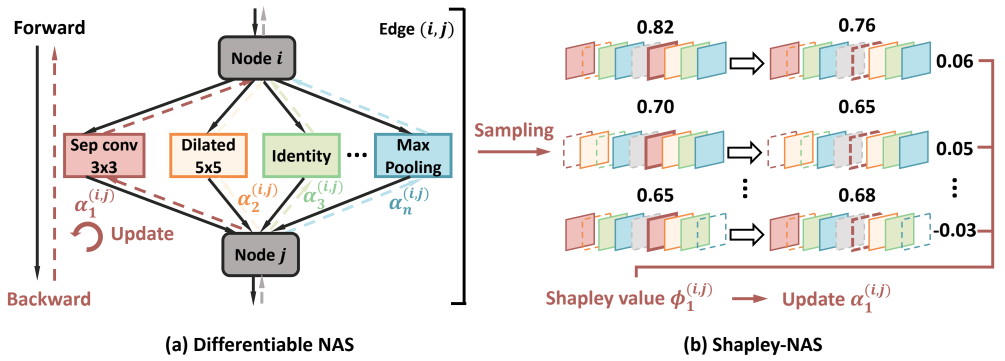


Figure 1: The comparison between DARTS and our Shapley-NAS. (a) DARTS constructs a weight-sharing supernet which consists of all candidate operations. The architecture parameters are optimized by gradient descent, which can not reflect the actual importance of operations. (b) The proposed Shapley-NAS method directly evaluates operation contribution to the task performance, and updates architecture parameters via the actual influence on accuracy.

on task performance according to the Shapley value of corresponding operations. The operation aggregation in the supernet based on performance contribution enables effective optimization of supernet weights, so that architectures with more promising performance are acquired. Figure 1 shows the difference between our Shapley-NAS and existing DARTS methods. More specifically, we iteratively evaluate the Shapley value for architecture parameter assignments and update the supernet weights. We employ the Monte-Carlo sampling with early truncation for operation set permutations to efficiently approximate the Shapley value of individual operations, and the architecture parameters are determined by the Shapley value that reveals actual component contribution. Moreover, we update the architecture parameters with momentum rather than direct assignment of the Shapley value, so that the fluctuation of operation set permutation sampling in Shapley value approximation is alleviated. We conducted extensive experiments on image classification and optimal architecture search across various search space, where our Shapley-NAS outperforms the state-of-the-art differentiable architecture search methods. We achieve an error rate of 2.43% on CIFAR-10 (Krizhevsky et al., 2009) according to the search space of DARTS and obtain the top-1 accuracy of 23.9% on ImageNet (Deng et al., 2009) under the mobile setting. Furthermore, our Shapley-NAS acquire the optimal architectures on two datasets and the near-optimal solution on the NAS-Bench-201 benchmark (Dong & Yang, 2020).

2 Related Work

Differentiable NAS: Differentiable architecture search (DARTS) was first proposed by Liu et al. (2018b) with the goal of significant search cost reduction in NAS. They formulated a bi-level objective that simultaneously optimizes the architecture parameters and supernet weights according to the overall objective, so that the efficient gradient descent was leveraged to search the graphical representation of the optimal network architectures. Since DARTS optimizes the single point on the simplex of continuous search space and discretizes the final architecture after search, the generalizability (Chen et al., 2019; Li et al., 2020; Xie et al., 2018; Yu et al., 2019) and the stability (Chen et al., 2020; Chen & Hsieh, 2020; Zhang et al., 2021; Zela et al., 2019; Wang et al., 2021b) are challenged. In order to mitigate the performance gap between the training set and the validation data, SNAS (Xie et al., 2018) and GDAS (Dong & Yang, 2019) adopted the differentiable Gumbel-Softmax (Jang et al., 2016) to imitate the one-hot encoding during architecture discretization. SGAS (Li et al., 2020) chose and pruned the candidate operations based on edge importance, selection certainty and selection stability to alleviate the degenerate of search-evaluation correlation, which reflects the true rankings of operation importance. RobustDARTS (Zela et al., 2019) found that the solutions generalize poorly when they coincide with high validation loss curvature during optimization, which results in significant performance drop after the architecture parameter discretization. Aiming at eliminating the instability in architecture discretization, they performed early stop regularization based on the largest eigenvalue. SmoothDARTS (Chen & Hsieh, 2020) further smoothed the loss landscape via perturbation based regularization including random smoothing and adversarial attack. Moreover, the large memory and computing overheads obstruct the potential efficiency enhancement of the DARTS framework (Yang et al., 2021). To address these, PC-DARTS (Xu et al., 2019) only searched the partially-connected operations to reduce the redundancy in network space exploration, where edge normalization degraded the search uncertainty to prevent edge selecting inconsistency. However, empirical studies (Wang et al., 2021b; Zhou et al., 2021) have demonstrated the learnable architecture parameter in DARTS framework fails to reveal the operation importance in the supernet, which requires effective metrics that fairly evaluate the operation contribution during architecture search.

Shapley value: Shapley value has been widely studied in game theory as it fairly evaluates the player contribution in the cooperative system (Roth, 1988; Winter, 2002; Shapley, 2016). Recently, Shapley value was adopted in explainable machine learning to discover the importance of model components, which can be divided into three groups: explaining feature importance (Mase et al., 2019; Lundberg & Lee, 2017; Lundberg et al., 2020; Ancona et al., 2019; Strumbelj & Kononenko, 2010), model component importance (Ancona et al., 2020; Wang et al., 2021a; Ghorbani & Zou, 2020) and data importance (Jia et al., 2019; Yona et al., 2021). For the first regard, Ancona et al. (2019) conducted an axiomatic comparison to show the advantage of the Shapley value over the attribution methods for feature map explanation in deep networks. SHAP (Lundberg & Lee, 2017) presented the additive feature attribution based on the Shapley value of features to acquire higher consistency with human intuition. For model component importance explanation, ShapNets (Wang et al., 2021a) leveraged the Shapley transform that transforms the input into Shapley representations so that the network prediction can be explained during the forward pass. Neuron Shapley (Ghorbani & Zou, 2020) pruned the neurons with the lowest Shapley value for deep networks, so that the model efficiency is significantly strengthened without sizable performance degradation. For the last aspect, Ghorbani & Zou (2019) quantified the contribution of individual data points which identified the outliers and corrupted data. Since computing the exact Shapley value is NP-hard, Monte-Carlo sampling (Ghorbani & Zou, 2019; 2020), perturbation-based approximation (Ancona et al., 2019), influence function and many others were presented for efficient acquisition of Shapley value. In this paper, we extend the Shapley value to operation importance evaluation in DARTS framework, so that the optimal architectures are derived by selecting the operations that contribute significantly to the tasks.

3 METHODOLOGY

In this section, we first briefly introduce differentiable architecture search (DARTS), which suffers from degenerate architectures due to the mismatch between the architecture parameters and operation importance. Then we introduce a fair attribution metric called Shapley value to quantify the relative contribution of operations, and also present the Monte-Carlo sampling algorithm with early truncation for efficient approximation of Shapley value. Finally, we propose Shapley-based architecture search (Shapley-NAS) which can effectively identify the optimal architectures with the most important operations in the large search space.

3.1 PRELIMINARIES

The differentiable architecture search (DARTS) is one of the most popular solutions to identify effective architectures, as it largely reduces the search cost by continuously relaxing the architecture search space. The search space is constructed by repetitions of normal and reduction cells. Each cell is represented by a directed acyclic graph (DAG) with $\mathcal{N}$ nodes and $\mathcal{E}$ edges, where each node $x^{(i)}$ defines a latent representation and each edge (i, j) is associated with an operation $o^{(i,j)}$. The core idea of DARTS is to apply continuous relaxation to the search space to perform gradient-based search. Concretely, the intermediate node is computed as a softmax mixture of candidate operations:

$$\bar{o}^{(i,j)}(x^{(i)}) = \sum_{o\in\mathcal{O}} \frac{\exp(\alpha_o^{(i,j)})}{\sum_{o'\in\mathcal{O}} \exp(\alpha_{o'}^{(i,j)})} o(x^{(i)}), \tag{1}$$

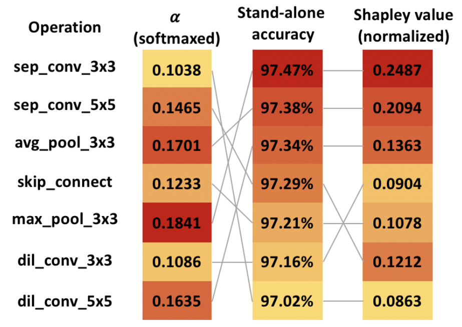
where $\mathcal{O}$ is the set of all candidate operations and $\alpha_o^{(i,j)}$ denotes the mixing weight for operation $o^{(i,j)}$ to construct the architecture. With such relaxation, the architecture search can be performed by jointly optimizing the network weight w and architecture parameters α in a differentiable manner with the following bi-level objective:

$$\min_{\alpha} \ \mathcal{L}_{val}(w^*, \alpha) \qquad \text{s.t.} \qquad w^* = \arg\min_{w} \ \mathcal{L}_{train}(w, \alpha). \tag{2}$$

During the search stage, a weight-sharing supernet containing all these candidate operations is optimized by gradient descent. At the end of the search stage, the final architecture is derived by selecting the operation with the largest architecture parameter α on every edge across all operation choices. This magnitude-based architecture selection process relies on an important assumption that the magnitude of architecture parameters represents the operation importance. However, this assumption has been proved to be untrue in most cases (Wang et al., 2021b), where the value of architecture parameters does not reflect the operation contribution to the performance of the supernet. To alleviate this issue, Wang et al. (2021b) proposes a perturbation-based architecture selection method which measures the operation importance by its discretization accuracy. However, their method greedily selects the best operation and performs discretization on each edge on top of DARTS, which only includes first-order approximation of the supernet and neglects the interactions between different edges.

3.2 Operation Importance Evaluation

The architecture parameters optimized by gradient descent can not reflect the actual operation importance. To further validate our assumption, we make a comparison between α and their corresponding performance. We get the stand-alone test accuracy by discretizing the edge to every candidate operation and training the derived architecture from scratch. Figure 2 shows the comparison between α and stand-alone accuracy, where we use different colors to represent their relative rankings and connect the units with the same ranking. As shown, the operation with the largest α does not result in the highest final accuracy and there is no obvious correlation between their rankings.



Operation	α (softmaxed)	Stand-alone accuracy	Shapley value (normalized)
sep_conv_3x3	0.1038	97.47%	0.2487
sep_conv_5x5	0.1465	97.38%	0.2094
avg_pool_3x3	0.1701	97.34%	0.1363
skip_connect	0.1233	97.29%	0.0904
max_pool_3x3	0.1841	97.21%	0.1078
dil_conv_3x3	0.1086	97.16%	0.1212
dil_conv_5x5	0.1635	97.02%	0.0863

Figure 2: The comparison between architecture parameters α in DARTS, their corresponding Shapley value and stand-alone accuracy, where we use different colors to show their relative rankings.

It is crucial to propose a fair attribution metric to evaluate operation contribution instead of relying on values of the architecture parameter α. Due to the complex interactions between operations on different edges, the task performance will change a lot when composed of different subsets of operations. To address this, we model the differentiable architecture search process as a cooperative game. In a cooperative game with a set of N players, a value function V maps each subset of players $S \subseteq N$ to a real value $V(S)$, which represents the expected payoff a set of the players can obtain by cooperation. In differentiable NAS, the supernet is composed of several layers with identical cell structure, and each cell has $|\mathcal{E}|$ edges each with $|\mathcal{O}|$ operations. Therefore, a set of individual operations, $N = \mathcal{O} \times \mathcal{E} = \{o^{(i,j)}\}_{o \in \mathcal{O}, (i,j) \in \mathcal{E}}$, can be modeled as players in the cooperative game, where all players work together towards the supernet's performance $V(N)$. It has been proved that, Shapley value (Roth, 1988; Winter, 2002; Shapley, 2016), denoted as $\phi_o^{(i,j)}$ in our problem, is the only method that uniquely distributes the total gains of all players $V(N)$ to each player in N with the following properties:

Efficiency The performance of the entire supernet is the sum of contributions of individual operations, i.e. $\sum_{o^{(i,j)} \in N} \phi_o^{(i,j)} = V(N)$.

Null Player If the operation has no impact on the performance when added to or removed from any subsets of the supernet, then its contribution is zero. That is, if $V(S) = V(S \cup \{o^{(i,j)}\})$ for any operation subset $S \subseteq N \setminus \{o^{(i,j)}\}$, we can derive $\phi_o^{(i,j)} = 0$. For example, the zero operation in DARTS search space has no impact on the final performance and thus has zero attribution.

Symmetry If two different operations could be exchanged without affecting the performance, they should be assigned with equal contributions. For any operation subset $S \subseteq N \setminus \{o^{(i,j)}, o'^{(k,l)}\}$, $V(S \cup \{o^{(i,j)}\}) = V(S \cup \{o'^{(k,l)}\})$, then we have $\phi_o^{(i,j)} = \phi_{o'}^{(k,l)}$.

Linearity If the performance metric V is a linear combination of other metrics (i.e. $V = a \times V_1 + b \times V_2$), the total contribution of each operation also satisfies $\phi_o^{(i,j)}(V) = \phi_o^{(i,j)}(V_1) + \phi_o^{(i,j)}(V_2)$.

The Shapley value of operations provides a fair scheme to quantify the operation contribution as it considers all possible combinations as a weighted mean and accounts for high correlations between individual elements. Therefore, it can help us discover important operations which contribute the most to the task performance during the search process.

For operation $o^{(i,j)}$ in our problem, its Shapley value can be computed as:

$$\phi_o^{(i,j)}(V) = \frac{1}{|N|} \sum_{S \subseteq N \setminus \{o^{(i,j)}\}} \frac{V(S \cup \{o^{(i,j)}\}) - V(S)}{\binom{|N|-1}{|S|}} \tag{3}$$

The operation importance represents the marginal contribution to the accuracy, which is obtained by evaluating the performance difference between all operation permutation and the counterparts without the given operation. Based on (3), we compute the Shapley value of different operations and compare them with stand-alone accuracy in Figure 2. The ranking with Shapley value matches the accuracy ranking very well, which demonstrates that Shapley value is an effective metric for evaluating operation importance, especially for identifying the most important operations.

3.3 Shapley value Approximation

Although Shapley value is an desirable attribution metric for quantifying the contribution of operations, directly computing Shapley value from (3) requires $2^{|\mathcal{O}| \times |\mathcal{E}|}$ network evaluations caused by enumerating all possible subsets. Therefore, exact computation of Shapley value becomes expensive since $|\mathcal{O}| \times |\mathcal{E}|$ in the common search space is usually large. To efficiently estimate the Shapley value, we present an approximate method based on Monte-Carlo sampling (Castro et al., 2009). Specifically, the Shapley value of operation $o^{(i,j)}$ is equivalent to estimating the mean of a random variable, which can be written as:

$$\phi_o^{(i,j)}(V) = \sum_{R \in \pi(N)} \frac{1}{N!} [V(R_{Pre(o^{(i,j)})} \cup \{o^{(i,j)}\}) - V(R_{Pre(o^{(i,j)})})] \tag{4}$$

where $\pi(N)$ denotes the set of permutations of all elements in N, and $R_{Pre(o^{(i,j)})}$ is the set of predecessors of $o^{(i,j)}$ in a given permutation $R \in \pi(N)$. Based on (4), we can get an unbiased approximation of every operation's Shapley value by sampling permutations of operation set N. Notably, the Monte-Carlo estimation reduces the exponential calculation complexity to polynomial time $M \times (|\mathcal{O}| \times |\mathcal{E}|)$, where M is the number of samples. Although this sampling-based estimation of Shaley value requires repetitions of accuracy evaluation on the validation set, it only includes the forward process through the supernet and no back-propagation is needed, thus enabling efficient approximation of operation Shapley value.

Moreover, when the number of operations in $R_{Pre(o^{(i,j)})}$ becomes too small, we find the task performance degrades dramatically and yields unstable sampling results. Therefore, to reduce the fluctuation of Shapley value estimation, we utilize the early truncation technique during the Monte-Carlo sampling procedure. Specifically, when the masked out operations lead to an extreme performance drop exceeding a pre-defined threshold, we break off the current sampling. This early truncation technique also reduces nearly half of computation cost, which makes the overall computational overheads comparable with gradient-based architecture parameter optimization in DARTS. The full algorithm of Shapley value estimation is illustrated in Appendix A.1.

3.4 Shapley-based Architecture Search

In order to reveal the actual operation importance to performance, we utilize Shapley value of operations to guide the architecture search to find the best solutions. Figure 1 shows the difference between our Shapley-NAS and conventional differential NAS. Rather than updating the architecture parameters by gradient descent in DARTS, we leverage Shapley value to represent the relative strength of operations. Specifically, we use the performance on validation set $\mathcal{L}_{val}$ as the metric V and thus reformulate the bi-level optimization problem in DARTS given in (2) as follows:

$$\alpha = \phi(\mathcal{L}_{val}(w^*, \alpha))) \qquad \text{s.t.} \quad w^* = \arg\min_{w} \ \mathcal{L}_{train}(w, \alpha). \tag{5}$$

The overall search process can be divided into two stages. At the first stage, we pre-train a supernet by only fine-tuning its network weight w while keeping architecture parameters α frozen. This warm-up process is essential for the initialized Shapley estimation and we keep α frozen to ensure fair comparison. At the second stage, we iteratively optimize the network weight w and the mixing operation weight α according to its Shapley value estimated by the algorithm in Section 3.3:

$$\alpha_t = \alpha_{t-1} + \epsilon \cdot \frac{s_t}{||s_t||_2} \tag{6}$$

where α_t means the architecture parameter in the t_{th} step during the optimization, s_t represents the accumulated Shapley value in the t_{th} step, $||\cdot||_2$ is the L_2 norm and ϵ is defined as the step size. To reduce undesired fluctuation in updating caused by random sampling, we introduce the momentum into the iteration to stabilize the optimization:

$$s_t = \mu \cdot s_{t-1} + (1 - \mu) \cdot \frac{\phi(\mathcal{L}_{val}(w_{t-1}, \alpha_{t-1}))}{||\phi(\mathcal{L}_{val}(w_{t-1}, \alpha_{t-1}))||_2} \tag{7}$$

where μ is the momentum coefficient that balances the accumulated Shapley value and the current sampling result. After the search stage is finished, we derive the final architecture by selecting the operation with the largest contribution on each edge. The detailed algorithm of our Shapley-NAS can be found in Appendix A.2.

4 Experiments

In this paper, we conducted extensive experiments to evaluate our method on the DARTS search space with CIFAR-10 (Krizhevsky et al., 2009) and ImageNet (Deng et al., 2009) for image classification, as well as on a widely used NAS benchmark dataset, NAS-Bench-201 (Dong & Yang, 2020). In the following ablation study, we analyzed the effectiveness of the proposed Shapley value evaluation, as well as the influence of hyperparameters on task performance and search cost.

4.1 Comparison with the State-of-the-art NAS Methods

4.1.1 Results on CIFAR-10

For the CNN search space in DARTS, we first performed experiments on CIFAR-10 for the image classification task. We employed the same operation space $\mathcal{O}$ as DARTS, constructed the supernet by stacking 8 cells (6 normal cells and 2 reduction cells) and set the initial channel number as 16. We utilized the partial connection strategy in PC-DARTS (Xu et al., 2019) to reduce memory overhead and increase batch size. We set the partial channel parameter $K = 4$ and trained the supernet for 50 epochs with a batch size of 256 on a single GTX 1080Ti GPU (we first finetuned the network weights for 15 epochs to warm up). The training set of CIFAR-10 containing 50K images was divided into two parts with equal size, one for optimizing the network weights and the other for evaluating Shapley value. We set the number of samples M to be 10 in the Monte-Carlo sampling and the early truncation threshold η to be 0.5 in each iteration. In momentum-based updating of architecture parameters, the momentum coefficient μ and step size ϵ were assigned to 0.8 and 0.1 respectively. At the evaluation phase, We simply followed the DARTS experimental settings for fair comparison and retrained the network from scratch for 600 epochs on the entire 50K training set.

Table 1 shows the performance of Shapley-NAS on CIFAR-10 compared with the state-of-the-art NAS methods, and the architecture of searched normal and reduction cells is visualized in Appendix A.4. Our Shapley-NAS achieves an average test error of 2.47% while only using 0.3 GPU days, significantly surpassing the DARTS baseline in both search cost and accuracy. The test error of the best single run in our experiments is 2.43%, ranking top amongst popular NAS methods. Although ProxylessNAS (Cai et al., 2018b) achieves a lower test error of 2.08%, it performs architecture search on a different space with heavy search cost. The low variance of the experimental results also demonstrates the stability of the proposed search method.

Table 1: Comparison with state-of-the-art image classifiers on CIFAR-10.

Architecture	**Test Error (%)**	**Params (M)**	**Search Cost (GPU days)**	**Search Method**
DenseNet-BC (Huang et al., 2017)	3.46	25.6	-	manual
NASNet-A (Zoph et al., 2018)	2.65	3.3	2000	RL
AmoebaNet-A (Real et al., 2019)	3.34 ± 0.06	3.2	3150	evolution
AmoebaNet-B (Real et al., 2019)	2.55 ± 0.05	2.8	3150	evolution
PNAS (Liu et al., 2018a)	3.41 ± 0.09	3.2	225	SMBO
ENAS (Pham et al., 2018)	2.89	4.6	0.5	RL
NAONet (Luo et al., 2018)	3.53	3.1	0.4	NAO
RandomNAS (Li & Talwalkar, 2020)	2.85 ± 0.08	4.3	2.7	Random
DARTS (1st order) (Liu et al., 2018b)	3.00 ± 0.14	3.3	0.4	gradient
DARTS (2nd order) (Liu et al., 2018b)	2.76 ± 0.09	3.3	1.0	gradient
SNAS(moderate) (Xie et al., 2018)	2.85 ± 0.02	2.8	1.5	gradient
GDAS (Dong & Yang, 2019)	2.93	3.4	0.3	gradient
BayesNAS (Zhou et al., 2019)	2.81 ± 0.04	3.4	0.2	gradient
ProxylessNAS (Cai et al., 2018b)	2.08	5.7	4.0	gradient
P-DARTS (Chen et al., 2019)	2.50	3.4	0.3	gradient
PC-DARTS (Xu et al., 2019)	2.57 ± 0.07	3.6	0.1	gradient
SGAS (Cri 1. avg) (Li et al., 2020)	2.66 ± 0.24	3.7	0.25	gradient
SDARTS-RS (Chen & Hsieh, 2020)	2.61 ± 0.02	3.4	0.4	gradient
DrNAS (Chen et al., 2020)	2.54 ± 0.03	4.0	0.4	gradient
DARTS+PT (Wang et al., 2021b)	2.61 ± 0.08	3.0	0.8	gradient
Shapley-NAS(avg.)‡	2.47 ± 0.04	3.4	0.3	sampling
Shapley-NAS(best)	2.43	3.6	0.3	sampling

‡ Means and standard deviations are obtained by repeated experiments with 4 random seeds.

4.1.2 RESULTS ON IMAGENET

ImageNet contains about 1.2 million training and 50K validation images from 1000 categories, which is much more challenging than CIFAR-10. We randomly sampled 10% and 2.5% images from the entire 1.3M training set of ImageNet for training network weights and estimating Shapley value respectively. The supernet was trained for 50 epochs with batch size 1024 and the architecture parameters remained frozen in the first 25 epochs. The other hyper-parameters were the same with section 4.1.1. At the evaluation stage, we trained the network from scratch for 250 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}.

The comparison results on ImageNet with other methods is demonstrated in Table 2. We trained the best-found architecture on CIFAR-10 on ImageNet to evaluate its transferability. The searched cells on CIFAR-10 achieve a competitive result with $24.3\%/7.3\%$ top-1/5 test error, which verifies the generalization ability of our Shapley-NAS. We also evaluated the optimal architecture directly searched on ImageNet and obtained a top-1/5 test error of $23.9\%/7.2\%$, which outperforms all other NAS methods with light search cost.

4.1.3 RESULTS ON NAS-BENCH-201

We also performed experiments on the NAS-Bench-201 space to further evaluate the performance of our Shapley NAS. NAS-Bench-201 is a popular benchmark to analyze NAS algorithms, as it provides performance of all candidate architectures which can be directly obtained by querying. In the search space of NAS-Bench-201, the operation set $\mathcal{O}$ has 5 elements 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 more details about the datasets can be found in their paper (Dong & Yang, 2020). Following previous works (Dong & Yang, 2020; Yan et al., 2020), we used the results obtained by training 12 epochs on CIFAR-10, and 200 epochs on CIFAR-100 and ImageNet-16-120. Specifically, we acquired the task-specific performance by directly searching on the evaluation dataset, and report the mean and standard deviation for the best architecture from 4 independent runs with different random seeds.

As shown in Table 3, our Shapley-NAS achieves outstanding performance with 94.37%, 73.51% and 46.85% test accuracy on CIFAR-10, CIFAR-100 and ImageNet-16-120 respectively. Notably,

Table 2: Comparison with state-of-the-art image classifiers on ImageNet under the mobile setting.

Architecture	**Test Error(%)**		**Params**	**Search Cost**	**Search**
	top-1	**top-5**	**(M)**	**(GPU days)**	**Method**
Inception-v1 (Szegedy et al., 2015)	30.1	10.1	6.6	-	manual
MobileNet (Howard et al., 2017)	29.4	10.5	4.2	-	manual
ShuffleNet $2\times$ (v1) (Zhang et al., 2018)	26.4	10.2	~ 5	-	manual
ShuffleNet $2\times$ (v2) (Ma et al., 2018)	25.1	-	~ 5	-	manual
NASNet-A (Zoph et al., 2018)	26.0	8.4	5.3	2000	RL
AmoebaNet-C (Real et al., 2019)	24.3	7.6	6.4	3150	evolution
PNAS (Liu et al., 2018a)	25.8	8.1	5.1	225	SMBO
MnasNet-92 (Tan et al., 2019)	25.2	8.0	4.4	-	RL
DARTS (2nd) (Liu et al., 2018b)	26.7	8.7	4.7	1.0	gradient
SNAS (mild) (Xie et al., 2018)	27.3	9.2	4.3	1.5	gradient
GDAS (Dong & Yang, 2019)	26.0	8.5	5.3	0.3	gradient
BayesNAS (Zhou et al., 2019)	26.5	8.9	3.9	0.2	gradient
ProxylessNAS (GPU) (Cai et al., 2018b)†	24.9	7.5	7.1	8.3	gradient
P-DARTS (CIFAR-10) (Chen et al., 2019)	24.4	7.4	4.9	0.3	gradient
P-DARTS (CIFAR-100) (Chen et al., 2019)	24.7	7.5	5.1	0.3	gradient
PC-DARTS (CIFAR-10) (Xu et al., 2019)	25.1	7.8	5.3	0.1	gradient
PC-DARTS (ImageNet) (Xu et al., 2019)†	24.2	7.3	5.3	3.8	gradient
SGAS (Cri 1. best) (Li et al., 2020)	24.2	7.2	5.3	0.25	gradient
SDARTS-ADV (Chen & Hsieh, 2020)	25.6	8.2	6.1	0.4	gradient
DrNAS (ImageNet) (Chen et al., 2020)†	24.2	7.3	5.2	3.9	gradient
Shapley-NAS (CIFAR-10)	24.3	7.3	5.1	0.3	sampling
Shapley-NAS (ImageNet)†	23.9	7.2	5.4	4.2	sampling

† indicates the results obtained by searching on ImageNet, otherwise on CIFAR-10 or CIFAR-100.

Table 3: Comparison results with state-of-the-art NAS methods on NAS-Bench-201.

Method	**CIFAR-10**		**CIFAR-100**		**ImageNet-16-120**	
	validation	**test**	**validation**	**test**	**validation**	**test**
ResNet (He et al., 2016)	90.83	93.97	70.42	70.86	44.53	43.63
Random (baseline)	90.93 ± 0.36	93.70 ± 0.36	70.60 ± 1.37	70.65 ± 1.38	42.92 ± 2.00	42.96 ± 2.15
RSPS (Li & Talwalkar, 2020)	84.16 ± 1.69	87.66 ± 1.69	45.78 ± 6.33	46.60 ± 6.57	31.09 ± 5.65	30.78 ± 6.12
REINFORCE (Zoph et al., 2018)†	91.09 ± 0.37	93.85 ± 0.37	71.61 ± 1.12	71.71 ± 1.09	45.05 ± 1.02	45.24 ± 1.18
ENAS (Pham et al., 2018)	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 (Liu et al., 2018b)†	39.77 ± 0.00	54.30 ± 0.00	15.03 ± 0.00	15.61 ± 0.00	16.43 ± 0.00	16.32 ± 0.00
DARTS (Liu et al., 2018b)	39.77 ± 0.00	54.30 ± 0.00	38.57 ± 0.00	38.97 ± 0.00	18.87 ± 0.00	18.41 ± 0.00
SNAS (Xie et al., 2018)	90.10 ± 1.04	92.77 ± 0.83	69.69 ± 2.39	69.34 ± 1.98	42.84 ± 1.79	43.16 ± 2.64
GDAS (Dong & Yang, 2019)	90.01 ± 0.46	93.23 ± 0.23	24.05 ± 8.12	24.20 ± 8.08	40.66 ± 0.00	41.02 ± 0.00
PC-DARTS (Xu et al., 2019)	89.96 ± 0.15	93.41 ± 0.30	67.12 ± 0.39	67.48 ± 0.89	40.83 ± 0.08	41.31 ± 0.22
iDARTS (Zhang et al., 2021)†	89.96 ± 0.60	93.58 ± 0.32	70.57 ± 0.24	70.83 ± 0.48	40.38 ± 0.59	40.89 ± 0.68
DrNAS (Chen et al., 2020)	91.55 ± 0.00	94.36 ± 0.00	73.49 ± 0.00	73.51 ± 0.00	46.37 ± 0.00	46.34 ± 0.00
Shapley-NAS	$\mathbf{91.61 \pm 0.00}$	$\mathbf{94.37 \pm 0.00}$	$\mathbf{73.49 \pm 0.00}$	$\mathbf{73.51 \pm 0.00}$	$\mathbf{46.57 \pm 0.08}$	$\mathbf{46.85 \pm 0.12}$
optimal	91.61	94.37	73.49	73.51	46.77	47.31

† Results are obtained by searching on CIFAR-10, otherwise by directly searching on the evaluation dataset.

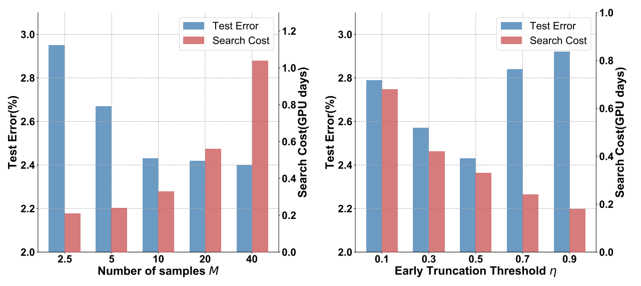
we obtain the global optimal architectures on CIFAR-10 and CIFAR-100, which indicates that the proposed method can identify important operations and derive the best architecture from the large search space. On the ImageNet-16-120 dataset, we also acquire a near-optimal solution, which outperforms the state-of-the-art algorithms, again verifying the effectiveness of our Shapley-NAS.

4.2 Ablation study

Effectiveness of Shapley value evaluation To verify the effectiveness of Shapley-NAS, we conducted experiments on 4 simplified search spaces S1-S4 proposed by Zela et al. (2019) across 3 datasets (CIFAR-10, CIFAR100 and SVHN). For comparison, we built a baseline, DARTS+Shapley, by combining the proposed Shapley value evaluation method with DARTS. We took a pretrained supernet from DARTS and applied Shapley value evaluation at the final discretization step, i.e. selecting operations based on their Shapley values instead of α. Moreover, we also tested the performance under the same settings but keeping α frozen, denoted as DARTS+Shapley*. As can be seen from the results in Table 4, DARTS achieves competitive results with the proposed Shapley evaluation method, even when α is not optimized in the training. Notably, our Shapley-NAS still outperforms DARTS+Shapley and DARTS+Shapley*, since taking Shapley value into the supernet optimization can further alleviate the problem caused by gradient-based NAS methods.

Method	C10			
	S1	S2	S3	S4
DARTS	3.84	3.11	2.95	2.82
DARTS+Shapley	4.85	2.92	2.84	2.55
DARTS+Shapley*	3.34	2.58	2.67	2.42
Shapley-NAS	7.20	3.45	2.94	2.63
Method	**C100**			
	S1	**S2**	**S3**	**S4**
DARTS	29.46	28.21	25.24	23.60
DARTS+Shapley	26.05	24.51	24.66	22.77
DARTS+Shapley*	28.90	23.67	22.39	21.92
Shapley-NAS	22.85	22.78	22.15	21.53
Method	**SVHN**			
	S1	**S2**	**S3**	**S4**
DARTS	4.58	2.59	2.88	2.36
DARTS+Shapley	3.53	2.72	2.64	2.49
DARTS+Shapley*	3.41	2.83	2.49	2.34
Shapley-NAS	3.05	2.65	2.58	2.41

Table 4: The test error(%) of different search algorithms on S1-S4. DARTS+Shapley denotes the combination of DARTS and Shapley value evaluation, and $*$ means freezing α during the search.



(a) Varying number of samples M (b) Varying early truncation threshold η

Figure 3: The test error (%) and search cost (GPU days) of the proposed method on CIFAR-10 with (a) different sampling times and (b) various thresholds of early truncation in the Shapley value estimation.

Table 5: The test error (%) and parameter storage cost (M) of the final architectures w.r.t. different values of momentum coefficient μ and different assignments of step size ϵ.

step size ϵ	$\mu = 0.2$		$\mu = 0.5$		$\mu = 0.8$		$\mu = 0.9$	
	Test Error(%)	Params(M)	Test Error(%)	Params(M)	Test Error(%)	Params(M)	Test Error(%)	Params(M)
0.01	2.89 ± 0.21	4.0	2.87 ± 0.16	3.7	2.67 ± 0.06	3.5	2.74 ± 0.11	3.8
0.05	2.85 ± 0.18	3.6	2.79 ± 0.12	3.4	2.55 ± 0.07	3.2	2.68 ± 0.07	3.5
0.1	2.82 ± 0.11	3.7	2.66 ± 0.10	3.3	2.47 ± 0.04	3.4	2.61 ± 0.06	4.1
0.5	2.92 ± 0.19	3.5	2.84 ± 0.13	4.2	2.71 ± 0.12	3.8	2.83 ± 0.15	3.9

Influence of sampling times M and early truncation threshold η We also explored the influence of sampling times M and early truncation threshold η in the Monte-Carlo sampling algorithm. The values of sampling times M and early truncation threshold η are significant for accurate Shapley value estimation, which also affect the overall search cost. Figure 3 shows the test error (%) and search cost (GPU days) on CIFAR-10 with various M and η. Reducing number of samples results in lower search cost while degrades the performance since the sampling is not enough to make accurate estimation. However, the estimation accuracy with samples larger than 10 is not sensitive to number of samples, and we choose $M = 10$ for search efficiency. Meanwhile, medium η also achieves the best accuracy-complexity trade-off as it mitigates the fluctuation in sampling as well as reducing the search cost.

Impact of momentum coefficient μ and step size ϵ To investigate the influence of momentum coefficient μ and step size ϵ on search accuracy, we implemented the architecture parameter assignment with different μ and ϵ. The test error range and model parameter cost is demonstrated in Table 5, where medium ϵ outperforms other values. Small step sizes fail to achieve the optimal distribution when reaching the maximum update iterations, and large step sizes make the supernet optimization hard to converge. With the increase of μ, the training stabilization becomes enforced, where μ with 0.8 achieves the best accuracy.

5 Conclusion

In this paper, we have presented Shapley-NAS, a Shapley value based operation contribution evaluation method for neural architecture search. Since the learnable architecture parameters in DARTS can not reveal the actual importance of operations on the task performance, we propose to evaluate the marginal contribution of operations on accuracy via Shapley value. Specifically, the Shapley value of operations can be efficiently approximated by Monte-Carlo sampling based algorithm with early truncation, and thus enabling the optimization of the supernet whose architecture parameters are directly updated with the operation contribution. Shapley-NAS achieves state-of-the-art performance on CIFAR-10, ImageNet and NAS-Bench-201 benchmarks, which proves its effectiveness to identify the optimal architectures with the most important operations in neural architecture search.

REFERENCES

Marco Ancona, Cengiz Oztireli, and Markus Gross. Explaining deep neural networks with a polynomial time algorithm for shapley value approximation. In *ICML*, pp. 272–281, 2019.

Marco Ancona, Cengiz Öztireli, and Markus Gross. Shapley value as principled metric for structured network pruning. *arXiv preprint arXiv:2006.01795*, 2020.

Han Cai, Tianyao Chen, Weinan Zhang, Yong Yu, and Jun Wang. Efficient architecture search by network transformation. In *AAAI*, volume 32, 2018a.

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

Javier Castro, Daniel Gómez, and Juan Tejada. Polynomial calculation of the shapley value based on sampling. *Computers & Operations Research*, 36(5):1726–1730, 2009.

Xiangning Chen and Cho-Jui Hsieh. Stabilizing differentiable architecture search via perturbation-based regularization. In *ICML*, pp. 1554–1565, 2020.

Xiangning Chen, Ruochen Wang, Minhao Cheng, Xiaocheng Tang, and Cho-Jui Hsieh. Drnas: Dirichlet neural architecture search. *arXiv preprint arXiv:2006.10355*, 2020.

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

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

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

Xuanyi Dong and Yi Yang. Searching for a robust neural architecture in four gpu hours. In *CVPR*, pp. 1761–1770, 2019.

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

Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In *ICML*, pp. 2242–2251, 2019.

Amirata Ghorbani and James Zou. Neuron shapley: Discovering the responsible neurons. *arXiv preprint arXiv:2002.09815*, 2020.

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*, pp. 770–778, 2016.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Song, and Costas J Spanos. Towards efficient data valuation based on the shapley value. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 1167–1176, 2019.

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

Guohao Li, Guocheng Qian, Itzel C Delgadillo, Matthias Muller, Ali Thabet, and Bernard Ghanem. Sgas: Sequential greedy architecture search. In *CVPR*, pp. 1620–1630, 2020.

Liam Li and Ameet Talwalkar. Random search and reproducibility for neural architecture search. In *Uncertainty in artificial intelligence*, pp. 367–377. PMLR, 2020.

Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In *ECCV*, pp. 19–34, 2018a.

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

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *NeurIPS*, pp. 4768–4777, 2017.

Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. From local explanations to global understanding with explainable ai for trees. *Nature machine intelligence*, 2(1):56–67, 2020.

Renqian Luo, Fei Tian, Tao Qin, Enhong Chen, and Tie-Yan Liu. Neural architecture optimization. *arXiv preprint arXiv:1808.07233*, 2018.

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)*, pp. 116–131, 2018.

Masayoshi Mase, Art B Owen, and Benjamin Seiler. Explaining black box decisions by shapley cohort refinement. *arXiv preprint arXiv:1911.00467*, 2019.

Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In *ICML*, pp. 4095–4104, 2018.

Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *AAAI*, volume 33, pp. 4780–4789, 2019.

Alvin E Roth. *The Shapley value: essays in honor of Lloyd S. Shapley*. Cambridge University Press, 1988.

Lloyd S Shapley. *17. A value for n-person games*. Princeton University Press, 2016.

Erik Strumbelj and Igor Kononenko. An efficient explanation of individual classifications using game theory. *The Journal of Machine Learning Research*, 11:1–18, 2010.

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*, pp. 1–9, 2015.

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*, pp. 2820–2828, 2019.

Rui Wang, Xiaoqian Wang, and David I Inouye. Shapley explanation networks. *arXiv preprint arXiv:2104.02297*, 2021a.

Ruochen Wang, Minhao Cheng, Xiangning Chen, Xiaocheng Tang, and Cho-Jui Hsieh. Rethinking architecture selection in differentiable nas. *arXiv preprint arXiv:2108.04392*, 2021b.

Tianzhe Wang, Kuan Wang, Han Cai, Ji Lin, Zhijian Liu, Hanrui Wang, Yujun Lin, and Song Han. Apq: Joint search for network architecture, pruning and quantization policy. In *CVPR*, pp. 2078–2087, 2020.

Eyal Winter. The shapley value. *Handbook of game theory with economic applications*, 3:2025–2054, 2002.

Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. Snas: stochastic neural architecture search. *arXiv preprint arXiv:1812.09926*, 2018.

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.

Shen Yan, Yu Zheng, Wei Ao, Xiao Zeng, and Mi Zhang. Does unsupervised architecture representation learning help neural architecture search? *Advances in Neural Information Processing Systems*, 33, 2020.

Yibo Yang, Shan You, Hongyang Li, Fei Wang, Chen Qian, and Zhouchen Lin. Towards improving the consistency, efficiency, and flexibility of differentiable neural architecture search. In *CVPR*, pp. 6667–6676, 2021.

Gal Yona, Amirata Ghorbani, and James Zou. Who's responsible? jointly quantifying the contribution of the learning algorithm and data. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 1034–1041, 2021.

Kaicheng Yu, Christian Sciuto, Martin Jaggi, Claudiu Musat, and Mathieu Salzmann. Evaluating the search phase of neural architecture search. *arXiv preprint arXiv:1902.08142*, 2019.

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.

Miao Zhang, Steven Su, Shirui Pan, Xiaojun Chang, Ehsan Abbasnejad, and Reza Haffari. idarts: Differentiable architecture search with stochastic implicit gradients. *arXiv preprint arXiv:2106.10784*, 2021.

Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6848–6856, 2018.

Hongpeng Zhou, Minghao Yang, Jun Wang, and Wei Pan. Bayesnas: A bayesian approach for neural architecture search. In *International Conference on Machine Learning*, pp. 7603–7613. PMLR, 2019.

Yuan Zhou, Xukai Xie, and Sun-Yuan Kung. Exploiting operation importance for differentiable neural architecture search. *TNNLS*, 2021.

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

Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8697–8710, 2018.

A Appendix

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

```
Algorithm 1: Estimating Shapley value of operations
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 $\eta$
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 $v_{k-1} > \eta \cdot V(N)$ and $R[k] \neq zero$ then
            mask out operation $R[k]$ and re-evaluate the validation accuracy $V$;
            Update $v_k$: $v_k = V(R[k+1], R[k+2], ..., R[n])$;
        end
        else
            $v_k = v_{k-1}$
        end
        $\phi_{R[k]} = \phi_{R[k]} + (v_{k-1} - v_k)$
    end
end
Return $\phi_o^{(i,j)} = \phi_o^{(i,j)}/M$, for $o^{(i,j)} \in N$.
```

A.2 The full algorithm of Shapley-NAS

```
Algorithm 2: Shapley-NAS
Input: Initialized supernet weights $w_0$ and architecture parameters $\alpha_0$, warm-up epochs $\mathcal{T}_1$,
       search epochs $\mathcal{T}_2$, momentum efficient $\mu$, step size $\epsilon$.
Output: The final architecture with chosen operation on every edge $\{o_{(i,j)}\}$.
Initialization:accumulated Shapley $s_0 = 0$, $t = 0$.
Stage 1 (Warm-up)
while $t < \mathcal{T}_1$ do
    Update supernet weights $w_t$ by descending $\nabla_w \mathcal{L}_{train}(w_{t-1}, \alpha_{t-1})$;
    $t = t + 1$;
end
Stage 2 (Architecture Search)
while $t < \mathcal{T}_2$ do
    Update supernet weights $w_t$ by descending $\nabla_w \mathcal{L}_{train}(w_{t-1}, \alpha_{t-1})$;
    Estimate the Shapley value $\phi(\mathcal{L}_{val}(w_{t-1}, \alpha_{t-1}))$ by Monte-Carlo sampling according to
      Algorithm 1;
    Compute the accumulated Shapley value: $s_t = \mu \cdot s_{t-1} + (1 - \mu) \cdot \frac{\phi(\mathcal{L}_{val}(w_{t-1}, \alpha_{t-1}))}{||\phi(\mathcal{L}_{val}(w_{t-1}, \alpha_{t-1}))||_2}$ ;
    Update architecture parameters: $\alpha_t = \alpha_{t-1} + \epsilon \cdot \frac{s_t}{||s_t||_2}$;
    $t = t + 1$;
end
Derive the final architecture through argmax: $o^{(i,j)} = \arg\max_{o \in \mathcal{O}} \alpha_o^{(i,j)}$.
```

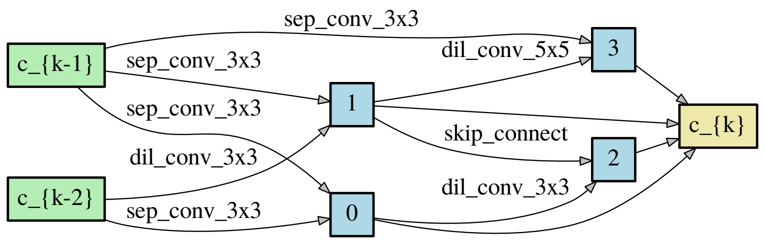
A.3 Training Details

CIFAR-10 The CIFAR-10 dataset 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 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 nodes). 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 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 (DeVries & Taylor, 2017) and drop-path (Zoph et al., 2018) with a rate of 0.3 for regularization.

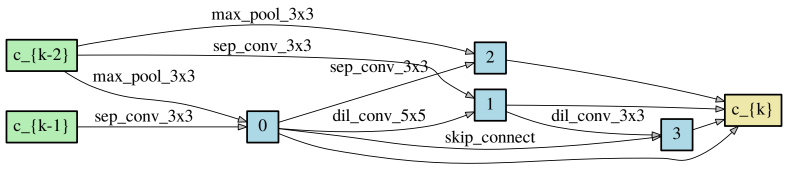
ImageNet Different from the architecture for CIFAR-10, the network for ImageNet starts with three convolution layers with stride of 2 which reduce the input resolution from 224×224 to 28×28 following previous works (Xu et al., 2019; Chen et al., 2019). At the evaluation stage, the network is composed 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 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 (Xu et al., 2019; Chen et al., 2019), label smoothing and an auxiliary loss tower are employed during the training.

NAS-Bench-201 In the search space of NAS-Bench-201, 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.

A.4 Searched Architectures on CIFAR-10



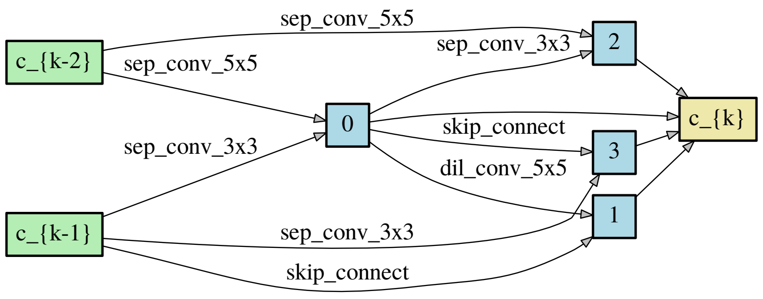
(a) Normal Cell



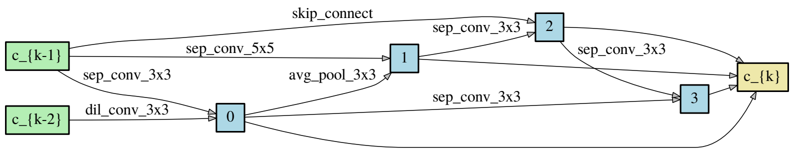
(b) Reduction Cell

Figure 4: Normal and Reduction cells discovered by Shapley-NAS on CIFAR-10

A.5 SEARCHED ARCHITECTURES ON IMAGENET



(a) Normal Cell



(b) Reduction Cell

Figure 5: Normal and Reduction cells discovered by Shapley-NAS on ImageNet