DDSAS: Dynamic and Differentiable Space-Architecture Search

Longxing Yang[1,2,3] yanglongxing20b@ict.ac.cn
Yu Hu[1,2,3] * huyu@ict.ac.cn
Shun Lu[1,2,3] lushun19s@ict.ac.cn
Zihao Sun[1,2,3] sunzihao18z@ict.ac.cn
Jilin Mei[1,2,3] meijilin@ict.ac.cn
Yiming Zeng[4] zyms5244@gmail.com
Zhiping Shi[5] shizp@cnu.edu.cn
Yinhe Han[1,2,3] yinhes@ict.ac.cn
Xiaowei Li[2,3] lxw@ict.ac.cn
[1]*Research Center for Intelligent Computing Systems, Institute of Computing Technology, CAS.*
[2]*State Key Laboratory of Computer Architecture, Institute of Computing Technology, CAS.*
[3]*School of Computer Science and Technology, University of Chinese Academy of Sciences.*
[4]*Tecent ADlab.*
[5]*Capital Normal University.*

Editors: Vineeth N Balasubramanian and Ivor Tsang

Appendix A. Method

A.1. Approximation in Differentiale Search

Here we further explain why we use Eq. (7) as an approximation. As a comparison, directly using weight without approximation, i.e., normalized Eq. (9), represents the *w/o approximation* case. We conduct experiments on CIFAR10 and CIFAR100 to show the impact of the approximation on test error. The experimental settings are the same as that in Section 4.2 of the main text. Table 5 shows the results. Compared with the *w/ approximation* case, the *w/o approximation* case has higher test errors. As shown in Eq. (13), the gradient of space probability p_k is very small due to the multiplication of multiple probabilities in the *w/o approximation* case. For example, with 8 operations, the maximum of $\prod_{l \neq k}(p_l)^{Z_l^{i,j}}(\bar{p}_l)^{\bar{Z}_l^{i,j}}$ is $(0.5^7)^2 \approx 6 \times 10^{-5}$. In fact, the approximation helps to alleviate the vanishing gradients problem, and helps the differentiable search to converge better.

$$\frac{\partial P(Z^{i,j}, A^{i,j})}{\partial p_k} = q_k \cdot (\prod_{l \neq k}(p_l)^{Z_l^{i,j}}(\bar{p}_l)^{\bar{Z}_l^{i,j}}) \tag{13}$$

Method	CIFAR10 (%)	CIFAR100 (%)
w/o approximation	2.74±0.16	18.00±0.62
w/ approximation	2.59±0.17	17.15±0.33

Table 5: Comparison of test error between without and with approximation on CIFAR10 and CIFAR100.

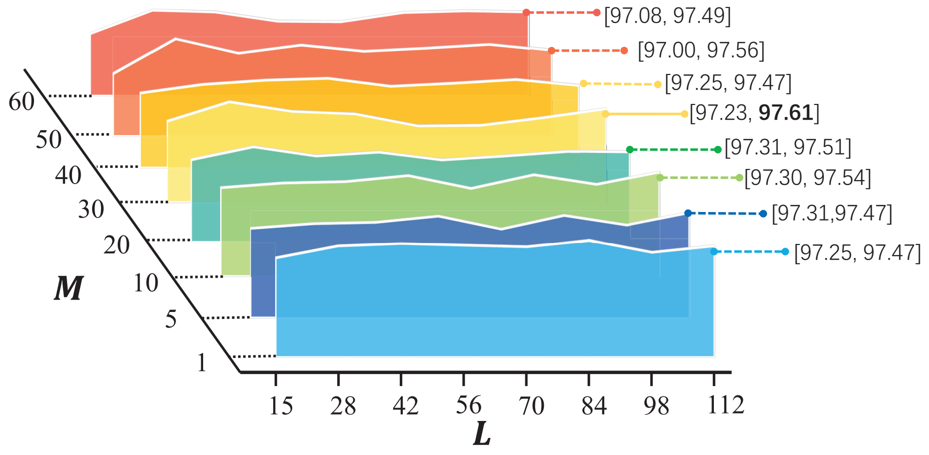


Figure 7: The effect of hyper-parameters L and M. The figure shows the best accuracy of DDSAS with different L and M.

Appendix B. Experiments

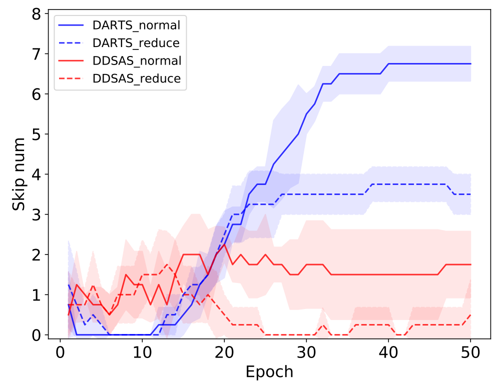
B.1. Hyper-parameter Settings

CIFAR10 and CIFAR100. In the searching phase, we leverage bi-level optimization to update the parameters. Half of the images are used to update the space and architecture parameters by an Adam optimizer with a learning rate 0.0003, weight decay 0.0001, and momentum (0.5, 0.999). The other half is used to update the weight parameters by an SGD optimizer with an initial learning rate 0.025, momentum 0.9, and weight decay 0.0003. In the training phase, the network consists of 20 layers with 36 initial channels. The SGD optimizer is used to train 600 epochs for the searched network with a batch size 96. The initial learning rate is 0.025 and decayed to 0 based on cosine scheduler. The other settings are also the same as DARTS: the length of cutout is 16, the weight of auxiliary towers is 0.4, and the probability of path dropout is 0.3.

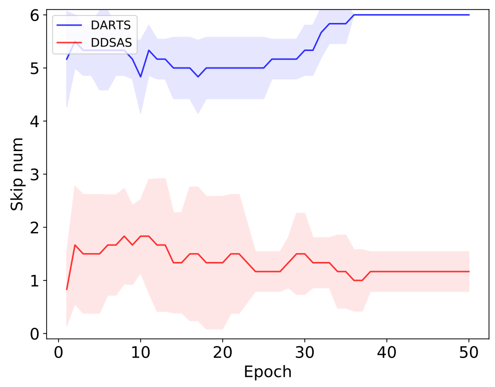
ImageNet. The network is built with 14 cells and the input size is 224×224. An SGD optimizer is used to train the network for 250 epochs with an initial learning rate of 0.5, weight decay of 0.00003, and a momentum of 0.9. The batch size is 1024. We train the network on eight V100 for around three days.

B.2. The Effect of L and M on Performance

We evaluate L and M on the CIFAR10 dataset in the DARTS search space, with a total of 64 configurations of $L \in \{15, 28, 42, 56, 70, 84, 96, 112\}$ and $M \in \{1, 5, 10, 20, 30, 40, 50, 60\}$. The trend of accuracy obtained by the best cells is shown in Figure 7. We can see that DDSAS has stable performance under different L and M configurations, showing its robustness and effectiveness.

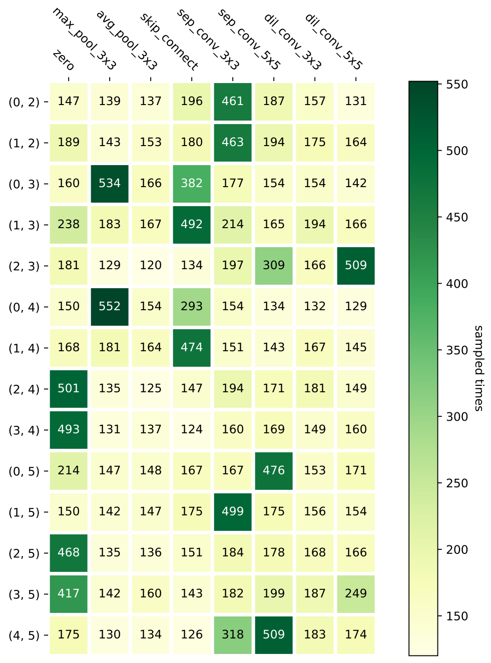


(a) Skip number in CIFAR100

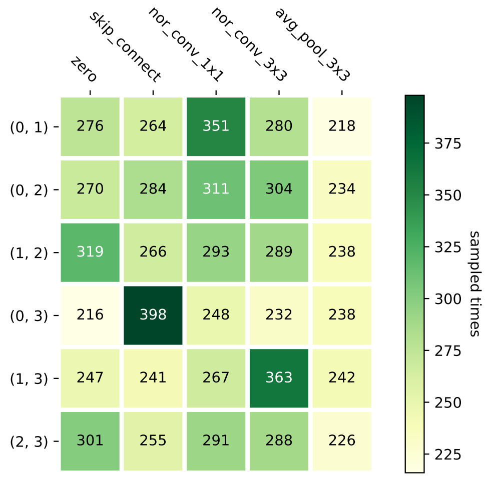


(b) Skip number in NAS-Bench-201

Figure 8: The number of skip operation along epochs in CIFAR100 and NAS-Bench-201.



(a) Sampled times in CIFAR100



(b) Sampled times in NAS-Bench-201

Figure 9: The heat maps of operations sampled times in CIFAR100 and NAS-Bench-201.

B.3. About Skip Aggregation

Although DARTS suffers from skip aggregation, DDSAS has no this problem in all experiments. On CIFAR100 and NAS-Bench-201, where skip aggregation is easy to appear as reported by P-DARTS and NAS-Bench-201, Figure 8 shows that skip numbers ($mean \pm std$) of DDSAS is much less than that of DARTS, specifically 1.75 ± 0.83 vs. 6.75 ± 0.43 on CIFAR100 and 1.17 ± 0.37 vs. 6.00 ± 0.00 on NAS-Bench-201. Furthermore, Figure 9 shows the sampled times of each operation. We observe that the number of sampling for skip, nor_conv_1×1 and nor_conv_3×3 is 264/284/241, 351/311/267 and 280/304/363 in $edge(0,1)(0,2)(1,3)$ on NAS-Bench-201. The root cause is that excessive sampling skip operations reduces the UCB value thus other operations with higher UCB value will be sampled into subspaces.

B.4. The influence of epochs

The influence of epochs is shown in Figure 5(d) where the accuracy gradually increases and converges along with more epochs, indicating that the sampled subspace contains better architectures. Meanwhile, the cosine distance between architectures decreases, showing that the sampling prefers more exploitation in later epochs.

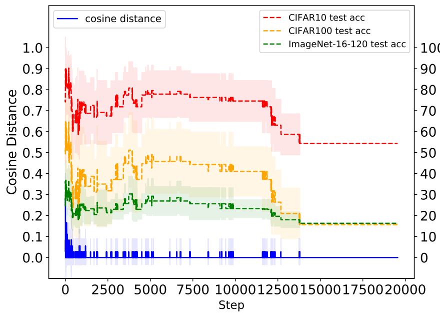
B.5. Calculation of Cosine Distance

The cosine distance is calculated by the one-hot encoding of architectures between adjacent epochs. Here we give an example for illustration. Suppose that there are two edges and three operations in a supernet. The one-hot encoding of $arch1$ and $arch2$ can be expressed as:

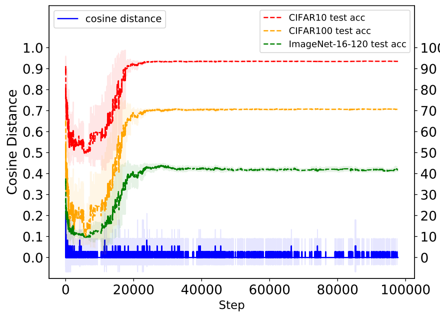
$$arch1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad arch2 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

We convert the matrix to a single row so that $arch1$ and $arch2$ can be written as $(0,1,0,1,0,0)$ and $(0,1,0,0,0,1)$. And then we can calculate the cosine distance between $arch1$ and $arch2$:

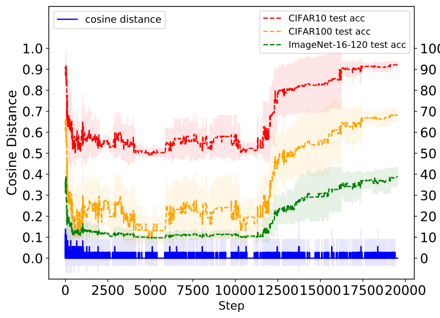
$$Dis(arch1, arch2) = 1 - \frac{\langle arch1, arch2 \rangle}{||arch1|| \; ||arch2||} = 0.5$$



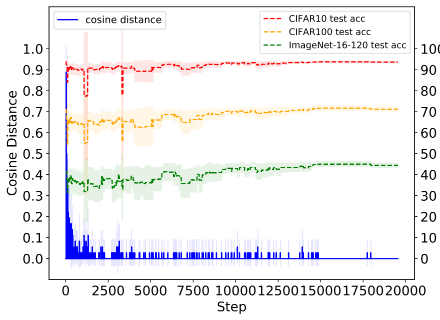
(*a*) DARTS, search with 50 epochs



(*b*) GDAS, search with 250 epochs



(*c*) GDAS, search with 50 epochs



(*d*) DDSAS, search with 50 epochs

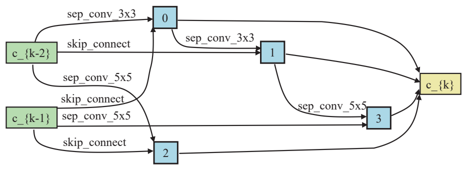
Figure 10: The exploration-exploitation capability of DARTS, GDAS, and DDSAS on NAS-Bench-201. The above figures show the cosine distance between adjacent **steps** and the corresponding accuracy, which are the fine-grained diagrams of Figure 4 in the main text.

B.6. Stepwise Cosine Distance and Accuracy

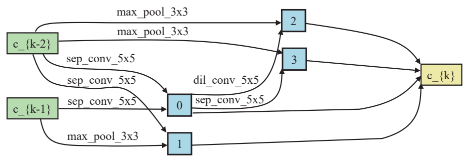
In Section 4.5.1 of the main text, we illustrated the exploration and exploitation of the search space with cosine distance between *adjacent epochs* and accuracy. For the sake of clarity, we show the cosine distance between *adjacent steps* and the corresponding accuracy in Figure 10. We can see the trends of cosine distance between adjacent steps are consistent with that between adjacent epochs, which verifies the conclusions of Section 4.5.1 in the main text.

B.7. Visualization of the Best Cells

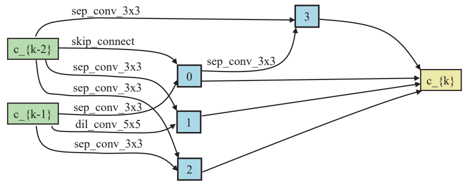
We visualized the best cells of DDSAS on CIFAR10, CIFAR100, and ImageNet. Figures 11 show the best cells of CIFAR10, CIFAR100, and ImageNet. The cell of ImageNet is transferred from CIFAR10, where the size of the sampled subspace L is 28, the number of gradient descent steps M is 50, and the accuracy is 97.56%.



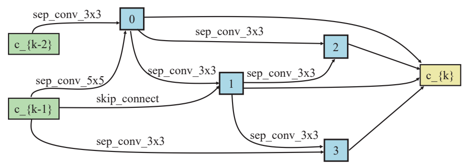
(a) CIFAR10 normal cell



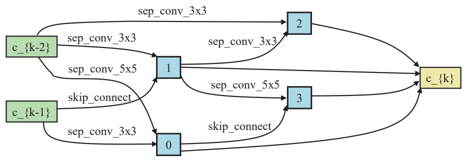
(b) CIFAR10 reduction cell



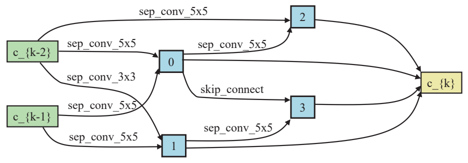
(c) CIFAR100 normal cell



(d) CIFAR100 reduction cell



(e) ImageNet normal cell



(f) ImageNet reduction cell

Figure 11: Best cells on CIFAR10, CIFAR100, and ImageNet searched by DDSAS.

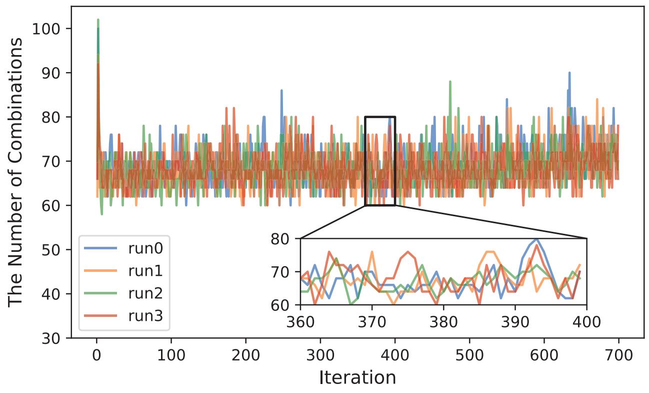
B.8. The Number of Combinations

In Section 4.5.2 of the main text, we provided the total number of combinations and the counts of combinations of $(A1_{k1}^{i,j} = 1, A2_{k2}^{i,j} = 1)$ within different ranges of the normal cell. Here we explain the calculation for the number of combinations. We independently ran the experiment on CIFAR10 for 4 times, with 700 sampled subspaces per run. To compare $A1^{i,j}$ and $A2^{i,j}$, two existing operations are selected with replacement in $Z^{i,j}$. Every two selected operations is considered as a combination. The total number of combinations is:

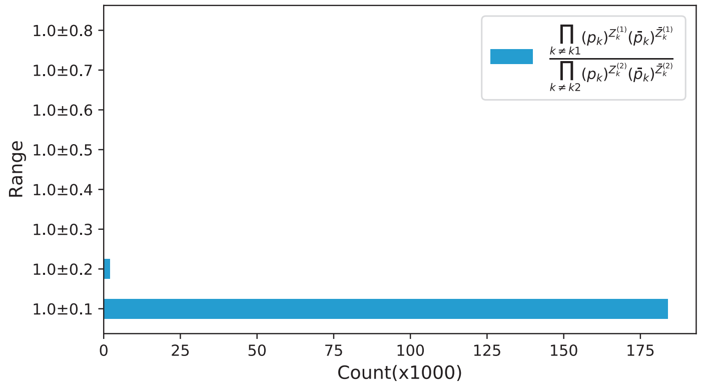
$$\begin{gathered}\sum_{a=1}^{4}\sum_{b=1}^{700}\sum_{c=1}^{14} y_{a,b,c}^2 \\ s.t. \sum_{c=1}^{14} y_{a,b,c} = L, \quad 1 \leq y_{a,b,c} \leq 8\end{gathered} \tag{14}$$

where $y_{a,b,c}$ is the number of operations existing in the a-th run, the b-th iteration and the c-th edge. The size of the sampled subspace L is 28. Figure 12(a) shows how the number of combinations of $(A1_{k1}^{i,j} = 1, A2_{k2}^{i,j} = 1)$ in the normal cell changes along iterations, where each curve represents a run of the experiments. For the four runs, the number of combinations is 48200, 47740, 47696, and 47640, respectively. Therefore, the total number of combinations is 191276.

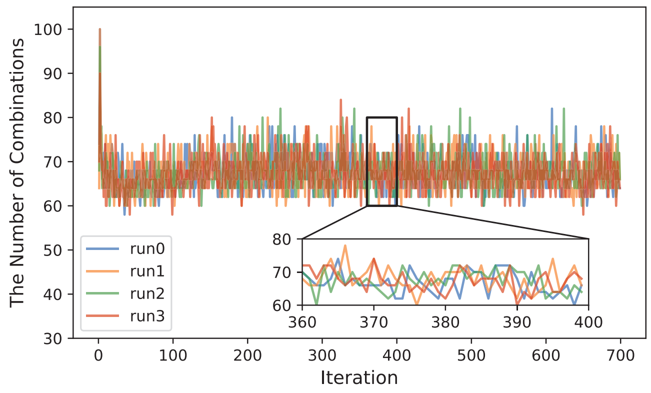
Besides, we report the corresponding data of the reduction cell in Figure 12(b) and Figure 12(c). In Figure 12(b), the total number of combinations is 188688 and 97.9% of combinations in which the second term of Eq. (12) is within 1.0 ± 0.1, confirming again the validity of our assumption for the ratio of the second term of Eq. (12).



(a) **Normal** cell: the number of combinations along iterations



(b) **Reduction** cell: the counts of combinations within different ranges



(c) **Reduction** cell: the number of combinations along iterations

Figure 12: **(a)** The number of combinations as iteration increases. Each point represents the number of combinations of all edges of the **normal** cell. **(b)** The counts of combinations in which the second term of Eq. (12) is within different ranges in the **reduction** cell. **(c)** This Figure has the same meaning as Figure (a) but it shows the number of combinations of the **reduction** cell.