Supplementary of "ROME: Robustifying Memory-Efficient NAS via Topology Disentanglement and Gradient Accumulation"

Xiaoxing Wang*[1], Xiangxiang Chu*[2], Yuda Fan[1], Zhexi Zhang[1], Bo Zhang[2], Xiaokang Yang[1], and Junchi Yan†[1]

[1]Dep. of Computer Science and Engineering & MoE Key Lab of AI, Shanghai Jiao Tong University
[2]Meituan

A. Proof of Gumbel-Top2 Process

This section will prove that the sampling scheme proposed in ROME-v2, Gumbel-Top2 technique,is equivalent to sampling two different edges without replacement, with the probability simplex $p_i = \frac{\exp \beta_i}{\sum_{i'} \exp(\beta_{i'})}$.

To complete this, we need to prove that each edge has the same probability of being selected in these two schemes. Let p_i be the probability of choosing i-th edge among n edges at one time. Without loss of generality, we suppose that e_1 is chosen.

i). We first discuss sampling two edges in order without replacement. The cases that e_1 is chosen can be divided into two disjoint parts:

A) It is selected by the first choice, whose probability is p_1.

B) It is selected by the second choice, and the probability is $\sum_{i=2}^{n} p_i \frac{p_1}{1-p_i}$, where $\frac{p_1}{1-p_i}$ is the scaled probability when taking i-th edge away without putting it back.

In total, the probability of e_1 being chosen is

$$p_1 + \sum_{i=2}^{n} p_i \frac{p_1}{1-p_i}. \tag{1}$$

ii). Further, we discuss the Gumbel Top-2 scheme, in which we sample n real numbers ϵ_k from $U[0,1]$ at first, and the probability of choosing each edge e_k is $\tilde{q}_k$:

$$q_k = \log p_k - \log(-\log \epsilon_k), \qquad \tilde{q}_k = \frac{\exp(q_k)}{\sum_{k'=1}^{n} \exp(q_{k'})}. \tag{2}$$

There are also two cases where e_1 will be chosen:

A) $\tilde{q}_1$ is the largest one among all edges, that is:

$$q_1 > q_j, \forall j \notin \{1\}. \tag{3}$$

By reformatting these inequalities, we have

$$\epsilon_j < \epsilon_1^{p_j/p_1}, \forall j \notin \{1\} \tag{4}$$

Since each ϵ_i is sampled from $U[0,1]$ independently, we can obtain the joint probability of all these events.

$$P = \prod_{j=2}^{n} P(\epsilon_j < \epsilon_1^{p_j/p_1}) = \prod_{j=2}^{n} \left[\int_0^1 \int_0^{\epsilon_1^{p_j/p_1}} 1 \, \mathrm{d}\epsilon_j \mathrm{d}\epsilon_1 \right]$$
$$= \int_0^1 \prod_{j=2}^{n} \epsilon_1^{p_j/p_1} \mathrm{d}\epsilon_1 = \int_0^1 \epsilon_1^{\frac{1}{p_1}-1} \mathrm{d}\epsilon_1 = p_1 \tag{5}$$

So the probability of case A) is p_1.

B) $\tilde{q}_1$ is the second largest one only next to $\tilde{q}_i$. That is

$$q_1 < q_i; \quad q_1 > q_j, \forall j \notin \{1, i\}. \tag{6}$$

By reformatting these inequalities, we have

$$\epsilon_i > \epsilon_1^{p_i/p_1}; \quad \epsilon_j < \epsilon_1^{p_j/p_1}, \forall j \notin \{1, i\}. \tag{7}$$

Similar to case A), since each ϵ_i is sampled from $U[0,1]$ independently, we can get the joint probability of all these events.

$$P = \int_0^1 (1 - \epsilon_1^{p_i/p_1}) \prod_{j \notin \{1,i\}}^{n} \epsilon_1^{p_j/p_1} \mathrm{d}\epsilon_1 = \int_0^1 (1 - \epsilon_1^{\frac{p_i}{p_1}}) \epsilon_1^{\frac{1-p_i}{p_1}-1} \mathrm{d}\epsilon_1$$
$$= \int_0^1 \epsilon_1^{\frac{1-p_i}{p_1}-1} - \epsilon_1^{\frac{1}{p_1}-1} d\epsilon_1 = \frac{p_1}{1-p_i} - p_1 = p_i \frac{p_1}{1-p_i}. \tag{8}$$

Enumerating i from 2 to n, the probability of case B) is $\sum_{i=2}^{n} p_i \frac{p_1}{1-p_i}$.

In all, the probability of e_1 being chosen is $p_1 + \sum_{i=2}^{n} p_i \frac{p_1}{1-p_i}$ as well, which meets Eq. 1. Therefore, these two schemes i) and ii) are equivalent.

*Equal contribution, † Correspondence author.

Table 1: Comparison in RobustDARTS [8] reduced search spaces and 3 datasets. We report the **lowest error rate** of 3 found architectures. †: using the settings of [8] where CIFAR-100 and SVHN models have 8 layers and 16 initial channels, CIFAR-10 models have 20 layers and 36 initial channels except that S2 and S4 have 16 initial channels. ⋆: using the settings in S-DARTS [1], where all models have 20 layers and 36 initial channels. Others utilize the settings in RobustDARTS. The best is underlined and in bold, the second-best is in bold.

Benchmark		DARTS†	R-DARTS†		DARTS†		SDARTS-RS†	ROME (ours)†		PC-DARTS⋆	SDARTS-RS⋆	ROME (ours)⋆	
			DP	L2	ES	ADA		v1	v2			v1	v2
C10	S1	3.84	3.11	2.78	3.01	3.10	2.78	**2.68**	**2.62**	3.11	2.78	**2.68**	**2.62**
	S2	4.85	3.48	3.31	3.26	3.35	3.33	**3.24**	**2.95**	3.02	**2.75**	2.79	**2.62**
	S3	3.34	2.93	**2.51**	2.74	2.59	**2.53**	2.65	2.58	**2.51**	**2.53**	2.65	2.58
	S4	7.20	3.58	3.56	3.71	4.84	4.84	**3.21**	**3.31**	3.02	**2.93**	3.61	**2.68**
C100	S1	29.46	25.93	24.25	28.37	24.03	23.51	22.34	**22.04**	18.87	**17.02**	17.27	**17.24**
	S2	26.05	22.30	22.24	23.25	23.52	22.28	**21.95**	**22.12**	18.23	17.56	**17.09**	**17.06**
	S3	28.90	22.36	23.99	23.73	23.37	**21.09**	22.56	**22.11**	18.05	17.73	**16.95**	**16.94**
	S4	22.85	22.18	21.94	**21.26**	23.20	21.46	21.33	**20.44**	17.16	17.17	**15.99**	**16.18**
SVHN	S1	4.58	2.55	4.79	2.72	2.53	2.35	**2.33**	**2.27**	2.28	2.26	**2.07**	**2.14**
	S2	3.53	2.52	2.51	2.60	2.54	2.39	**2.39**	**2.30**	2.39	2.37	**2.14**	**2.07**
	S3	3.51	2.49	**2.48**	2.50	2.50	**2.36**	2.58	2.51	2.27	2.21	**2.14**	**2.07**
	S4	3.05	2.61	2.50	2.51	2.46	2.46	**2.43**	**2.34**	2.37	2.35	**2.00**	**1.99**

B. Detailed Discussion with Prior Works

B.1. Topology Disentanglement

DOTS [4] is related to our work that proposes to decouple the operation search and topology search into two separate stages. However, It is methodologically different from ROME. We highlight some key features of ROME. **1) No-Prior:** To alleviate the collapse, DOTS uses a strong human grouping prior as StacNAS, which classifies the operations into two groups: parametric and non-parametric. ROME uses no prior at all. **2) Single-phase searching with no extra hyperparameters:** DOTS contains two phases: search operations first, and then search topologies with the fixed operations. It uses three carefully designed and tuned hyperparameters ($T_0, T_\beta, T_{\alpha_{On}}$) to control the percentage of two phases for different datasets (through our communication with the authors of DOTS). In contrast, ROME is single-phase as DARTS and it requires no specific hyperparameters tailored for different datasets. **3) Memory efficiency:** ROME (2.3G) costs 1/4 of DOTS's memory (9.5G), since DOTS trains the whole supernet during the operation search.

B.2. Gumbel Reparameterization in NAS

GDAS and SNAS are contemporary works based on Gumbel-softmax reparameterization technique. Nevertheless, GDAS is memory-efficient since it sample and activate a sub-set of candidate operations, while SNAS still belongs to one-shot NAS since all operations participate the forward and backward at each iteration in the search stage. This work research on GDAS and point out that the performance collapse issue also exists. We attribute it to two aspects, that differs from the reason in DARTS: 1) the topology inconsistency between searching and evaluation and 2) the stochastic nature of sampling for candidate operations. Topology disentanglement and gradient accumulation techniques are proposed to stabilize the search process for GDAS. Our method, ROME inherits the property of GDAS, i.e., low GPU memory requirement and high speed for searching. In comparison, SNAS has little relation to ROME. It requires vast GPU memory like DARTS and still suffers performance collapse issue.

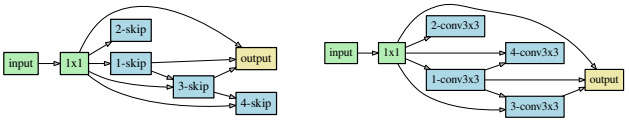
B.3. Dynamic Network

DDW [7] is a kind of dynamic network whose topology dynamically changes based on the input. In contrast, ROME is a NAS method, whose topology is fixed after searching. Unlike DDW that limited to some handcrafted architectures e.g. ResNet, MobileNetV2, ROME supports more complex topologies as DARTS's search space. Moreover, DDW is not memory efficient as it keeps the whole supernet in the memory, while ROME requires much less GPU memory.

C. Further Experiments

C.1. Robustness evaluation on 12 hard benchmarks

We follow RobustDARTS [8] and evaluate the performance and generalization of our method across three datasets on S1-S4 search spaces, where DARTS severely suffers from performance collapse. We independently search four times under different random seeds for each



(a) GDAS [3] (b) ROME

Figure 1: GDAS fails on NAS-Bench-1Shot1 [9] on CIFAR-10 when adding skip connection to the second search space. Notice that nodes with no out-degrees have no contribution to the output.

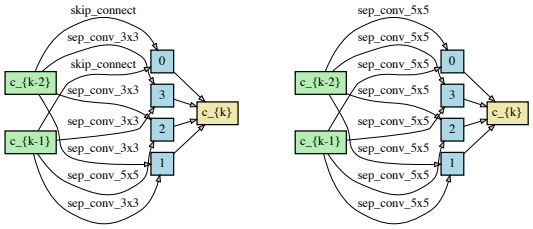


Figure 2: Best normal and reduction cells discovered by ROME-v1 on CIFAR-10.

benchmark and train the discovered models to report their mean and standard variance performance. This process is recommended by [5, 1, 8, 2] to fairly compare different NAS methods. Table 1 reports the best performance, showing that our methods robustly outperform RobustDARTS with a clear margin across the 12 benchmarks. The best cells found by ROME are shown in the next section.

C.2. Discussion on collapse behavior across popular NAS benchmarks.

We argue that excluding an important operation for search space can cause illusive conclusions. Specifically, NAS-Bench-1Shot1 [9] suggests that Gumbel-based NAS is quite robust. However, this observation is laying on the basis that popular skip connections are not included in the search space [6]. After adding skip connection into the choices, we perform the GDAS search using their released code [1]. The best model found is full of skip connections, which again supports our discovery of collapse issue in single-path based NAS, see Fig. 1 and more in Fig. 14 in the supplemental material. Instead, we do not suffer the same issue while performing ROME in these search spaces (see Fig. 15).

D. Figures of Genotypes

Genotypes of the discovered architectures by ROME are illustrated in Fig. 2 - Fig. 15.

[1] https://github.com/automl/nasbench-1shot1

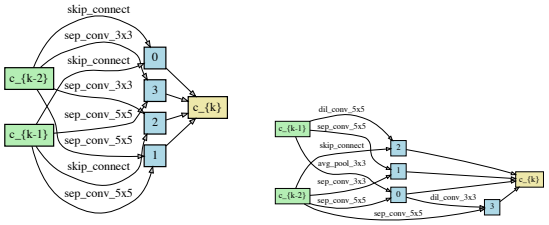
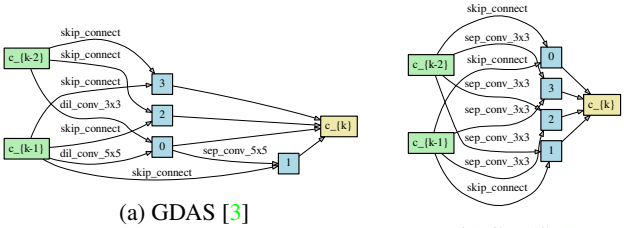


Figure 3: Best normal and reduction cells discovered by ROME-v2 on CIFAR-10.



(a) GDAS [3] (b) GDAS [3]

Figure 4: The architecture of normal cells searched by GDAS and ROME on ImageNet under S0 search space. Network searched by GDAS is dominated by skip connection and only obains 72.5% accuracy on ImageNet, while our method is much more stable and achieves 75.5% accuracy.

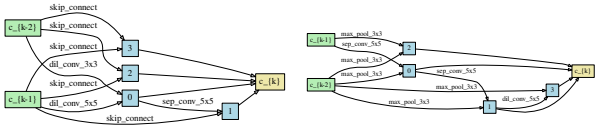


Figure 5: The best architecture found by GDAS on ImageNet in S0. Skip connection dominate the searched architecture. Top-1 accuracy on the validation set is 72.5%.

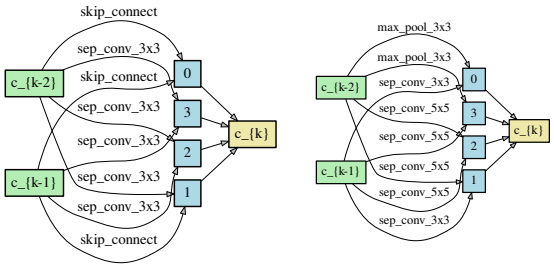
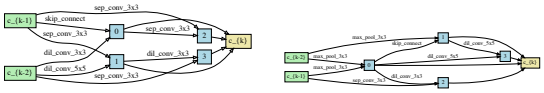


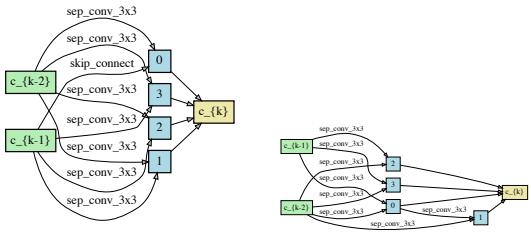
Figure 6: The best architecture found by ROME on ImageNet in S0. No performance collapse occurs. Top-1 accuracy on the validation set is 75.5%

References

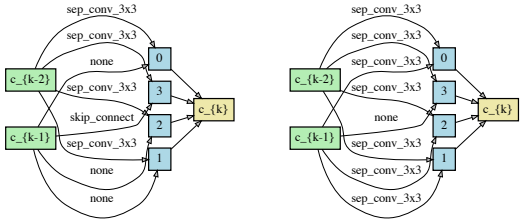
[1] Xiangning Chen and Cho-Jui Hsieh. Stabilizing differentiable architecture search via perturbation-based regularization. In



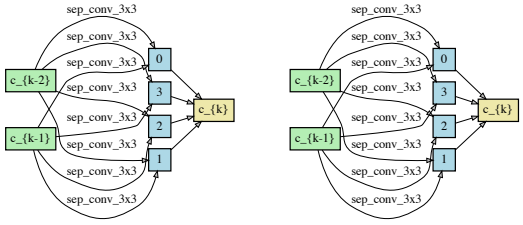
(a) S1



(b) S2

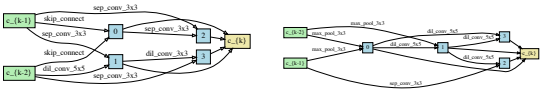


(c) S3

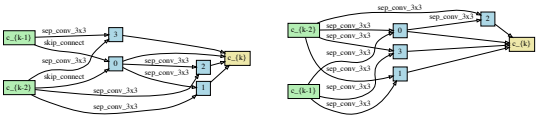


(d) S4

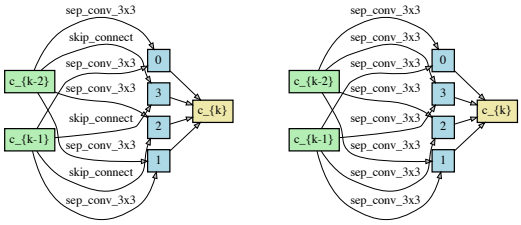
Figure 7: ROME-V1 best cells (paired in normal and reduction) on CIFAR10 in reduced search spaces of Robust-DARTS.



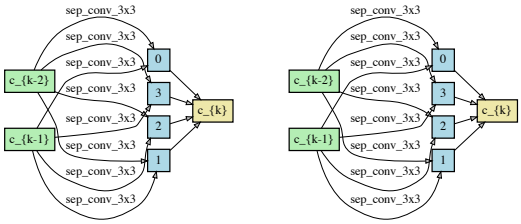
(a) S1



(b) S2



(c) S3



(d) S4

Figure 8: ROME-V1 best cells (paired in normal and reduction) on CIFAR100 in reduced search spaces of Robust-DARTS.

ICML, 2020. 2, 3

[2] Xiangxiang Chu, Xiaoxing Wang, Bo Zhang, Shun Lu, Xiaolin Wei, and Junchi Yan. {DARTS}-: Robustly stepping out of performance collapse without indicators. In *ICLR*, 2021. 3

[3] Xuanyi Dong and Yi Yang. Searching for a Robust Neural Architecture in Four GPU Hours. In *CVPR*, pages 1761–1770, 2019. 3

[4] Yuchao Gu, Lijuan Wang, Yun Liu, Yi Yang, Yu-Huan Wu, Shao-Ping Lu, and Ming-Ming Cheng. DOTS: decoupling operation and topology in differentiable architecture search. In *CVPR*, 2021. 2

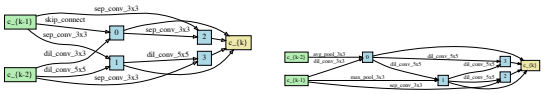
[5] Antoine Yang, Pedro M. Esperança, and Fabio M. Carlucci. Nas evaluation is frustratingly hard. In *ICLR*, 2020. 3

[6] Chris Ying, Aaron Klein, Eric Christiansen, Esteban Real, Kevin Murphy, and Frank Hutter. NAS-bench-101: Towards reproducible neural architecture search. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 7105–7114, Long Beach, California, USA, 09–15 Jun 2019. PMLR. 3

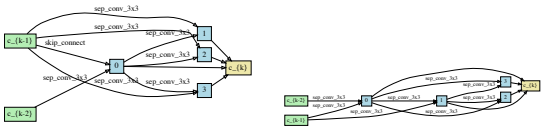
[7] Kun Yuan, Quanquan Li, Shaopeng Guo, Dapeng Chen, Aojun Zhou, Fengwei Yu, and Ziwei Liu. Differentiable dynamic wirings for neural networks. In *ICCV*, pages 327–336, 2021. 2

[8] Arber Zela, Thomas Elsken, Tonmoy Saikia, Yassine Marrakchi, Thomas Brox, and Frank Hutter. Understanding and robustifying differentiable architecture search. In *ICLR*, 2020. 2, 3

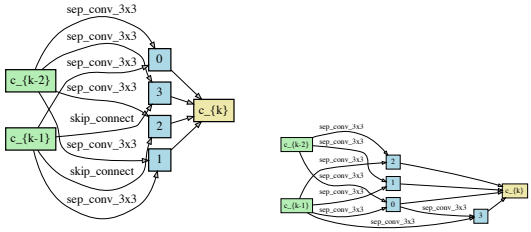
[9] Arber Zela, Julien Siems, and Frank Hutter. Nas-bench-1shot1: Benchmarking and dissecting one-shot neural architecture search. In *ICLR*. OpenReview.net, 2020. 3, 7, 8



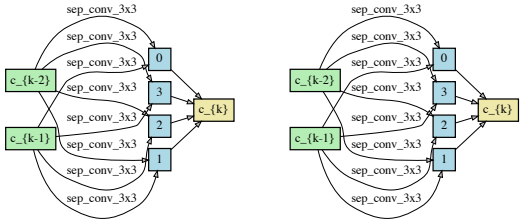
(a) S1



(b) S2

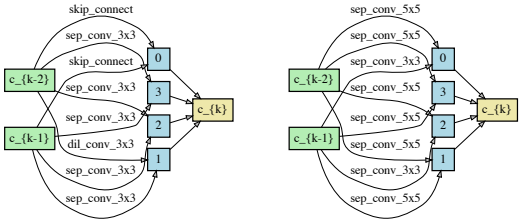


(c) S3

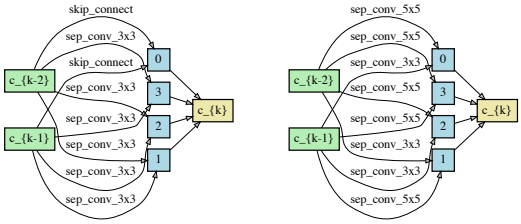


(d) S4

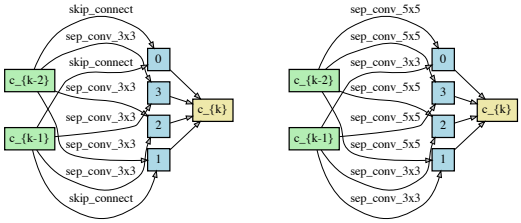
Figure 9: ROME-V1 best cells (paired in normal and reduction) on SVHN in reduced search spaces of RobustDARTS.



(a) Architecture 1

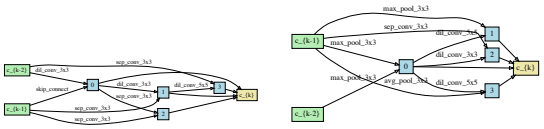


(b) Architecture 2

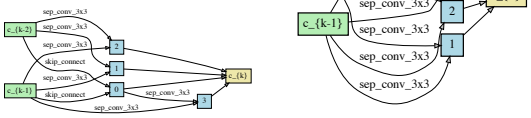


(c) Architecture 3

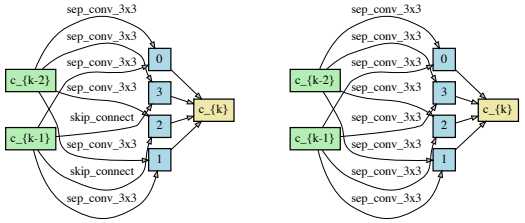
Figure 10: ROME-V1 cells (paired in normal and reduction) on CIFAR-100 in DARTS's search space.



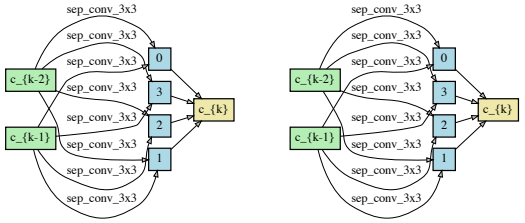
(a) S1



(b) S2

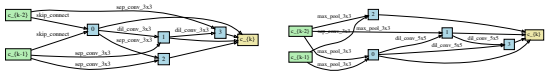


(c) S3

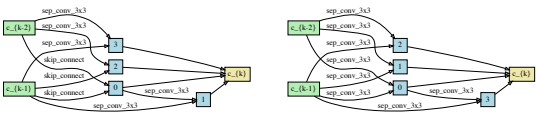


(d) S4

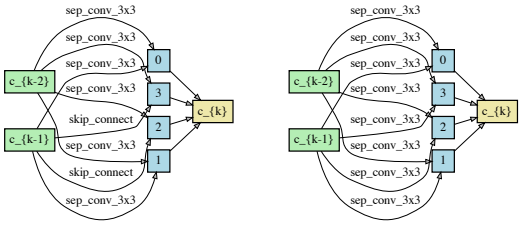
Figure 11: ROME-V2 best cells (paired in normal and reduction) on CIFAR10 in reduced search spaces of Robust-DARTS.



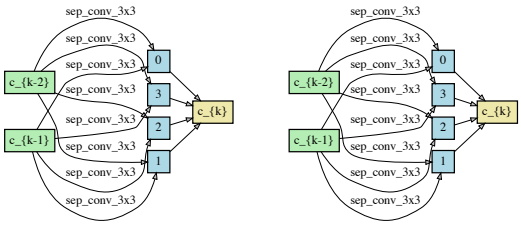
(a) S1



(b) S2

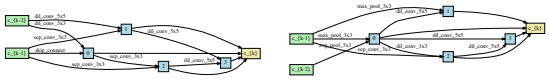


(c) S3

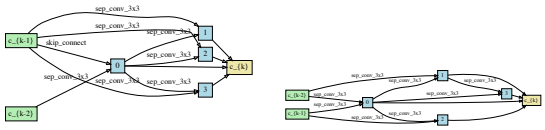


(d) S4

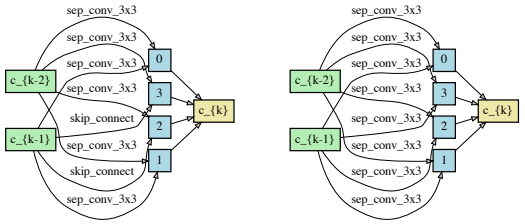
Figure 12: ROME-V2 best cells (paired in normal and reduction) on CIFAR100 in reduced search spaces of Robust-DARTS.



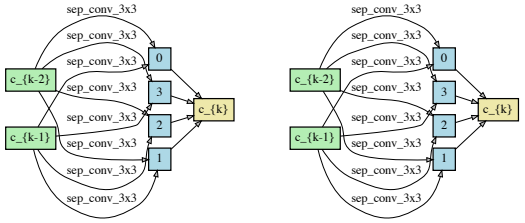
(a) S1



(b) S2

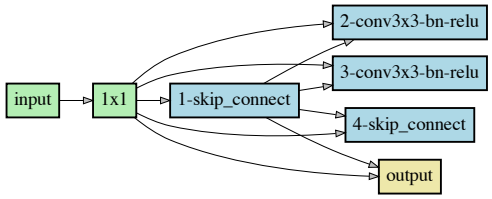


(c) S3

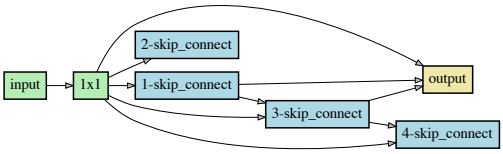


(d) S4

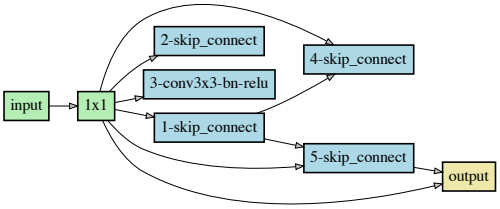
Figure 13: ROME-V2 best cells (paired in normal and reduction) on SVHN in reduced search spaces of Robust-DARTS.



(a) S1

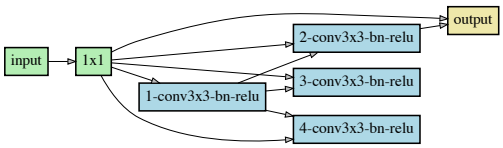


(b) S2

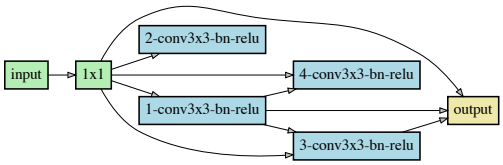


(c) S3

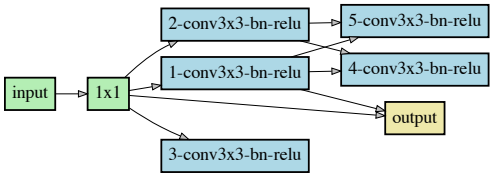
Figure 14: GDAS fails on NAS-Bench-1Shot1 [9] when searching on CIFAR-10 in all three search spaces when skip connection are added into choices. In each MixedOp, we have three choices: {maxpool3x3, conv3x3-bn-relu, skip-connect}.



(a) S1



(b) S2



(c) S3

Figure 15: ROME-V2 resolves the aggregation of skip connections on NAS-Bench-1Shot1 [9]. Notice intermediate nodes concatenate their outputs as the input for the output node, while some have loose ends and don't feed to the output node.