# EcoNAS: Finding Proxies for Economical Neural Architecture Search

Dongzhan Zhou[1]* Xinchi Zhou[1]* Wenwei Zhang[2] Chen Change Loy[2]
Shuai Yi[3] Xuesen Zhang[3] Wanli Ouyang[1]
[1]The University of Sydney, SenseTime Computer Vision Research Group, Australia
[2]Nanyang Technological University [3]SenseTime Research
{d.zhou, xinchi.zhou1, wanli.ouyang}@sydney.edu.au {wenwei001, ccloy}@ntu.edu.sg
{yishuai, zhangxuesen}@sensetime.com

## A. Two Model Examples in Introduction

In the introduction, we mentioned that some architectures applied with certain reduction factors win in the reduced setting but perform worse in the original setup on CIFAR-10 [5]. The normal and reduction cells for the aforementioned models A and B are shown in Fig. A1 and Fig. A2, respectively. The results on CIFAR-10 of the two models in the original setting and reduced setting are shown in Table A1. The original setting is $c_0r_0s_0e_{600}$ while the reduced setting is $c_0r_0s_0e_{30}$. The training details about the two settings are provided in Section E.2. The results show that the rank of performance evaluated in the reduced setting is not guaranteed to be the same as that in the original setting.

## B. Reliability of Spearman Coefficient

The final accuracy of each network might have minor variation due to the randomness in training. To make the Spearman Coefficient reliable to the accuracy variation, we adjust $\rho_{sp}$ to make it tolerant to the small variations of accuracy and re-analyze the results based on existing records. In the new metric, if the absolute accuracy difference of two models within an interval $b$ are in both original and reduced setting, they will be considered as having no ranking difference. $b$ (0.15% in our implementation) is used to ignore the minor accuracy variations. For instance, if the accuracy differences of two models are 0.1 in both original and reduced settings, then the new metric will regard these models of having no ranking difference despite the ranking change between these two models. We find that this new $\rho_{sp}$ is highly consistent with previous metric (the normalized correlation is 0.96). And the good settings in Fig. 8 are consistent with the new metric in Fig. A3. We further test some settings with 100 models and observe consistent results, which also confirms the reliability.

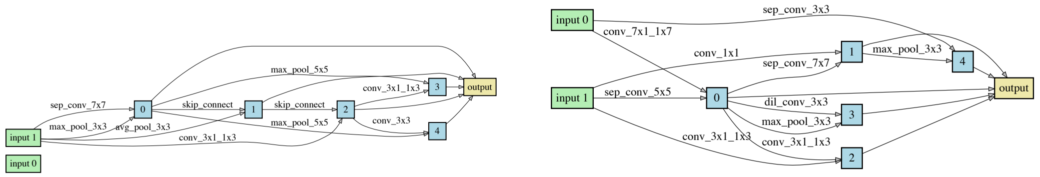
## C. Construction of Model Zoo

This section provides the details on constructing the model zoo (Section 3). Each network architecture in the model zoo is a stack of normal cells alternating with reduction cells. In each network, these two cells are all generated separately according to the common selection steps in [7, 9, 11] and we just replace the search algorithm in these approaches by random sampling. The number of nodes inside the cells is 5 and every cell receives two initial inputs. For cell $k$, the two initial inputs are denoted as $h_{k-2}$ and $h_{k-1}$, which are outputs of previous cells $k-2$ and $k-1$ or the input of images. The output of each cell is the depthwise concatenation of all the intermediate nodes (two initial inputs excluded). The generation steps of each intermediate node are as follows:

- **Step 1.** Randomly select an input from the input set, which contains two initial inputs of the cell and the set of outputs from previous nodes within the cell.
- **Step 2.** Randomly select another input from the same input set as in Step 1.
- **Step 3.** Randomly select an operation from the operation set and apply this operation to the first input selected in Step 1.
- **Step 4.** Randomly select another operation to apply to the second input selected in Step 2.
- **Step 5.** Add the outputs of Step 3 and Step 4 to create the output of the current node.

The original 'Step 5' in [11] provides two combination methods: element-wise addition and depth-wise concatenation. However, previous work [7] mentions that the concatenation method are never chosen during search. Therefore, we only use addition as the combination operation. We selected 13 operations to build our operation set considering their prevalence in the NAS literature [1, 8, 10, 11], which are listed as below:

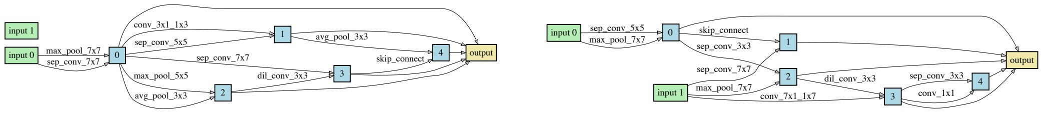
*Equal contribution.



(a) Normal Cell

(b) Reduction Cell

Figure A1. Normal and reduction cell structures of model A



(a) Normal Cell

(b) Reduction Cell

Figure A2. Normal and reduction cell structures of model B

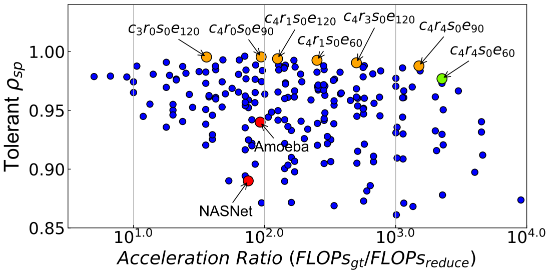


Figure A3. New $\rho_{sp}$ (Y-axis) and acceleration ratio (X-axis) of reduced settings. Blue points show all settings, the orange ones are good settings and the green one is adopted in EcoNAS.

Table A1. The top-1 accuracy on CIFAR-10 for two models in the original setting ($c_0r_0s_0e_{600}$) and the reduced setting ($c_0r_0s_0e_{30}$).

| Model | $c_0r_0s_0e_{600}$ | $c_0r_0s_0e_{30}$ |
|---|---|---|
| $A$ | 95.27% | 82.42% |
| $B$ | 94.58% | 86.21% |

- 3x3 average pooling
- 3x3 max pooling
- 5x5 max pooling
- 7x7 max pooling
- Identity
- 1x1 Convolutions
- 3x3 Convolutions
- 3x3 Separable Convolutions
- 5x5 Separable Convolutions
- 7x7 Separable Convolutions
- 3x3 Dilated Convolutions
- 1x3 then 3x1 Convolutions
- 1x7 then 7x1 Convolutions

## D. Detailed Information About *Entropy*

In Fig. 5 and Fig. 6 of Section 3.3, a new measurement called *entropy* is used. This section provides the details on how *entropy* is calculated.

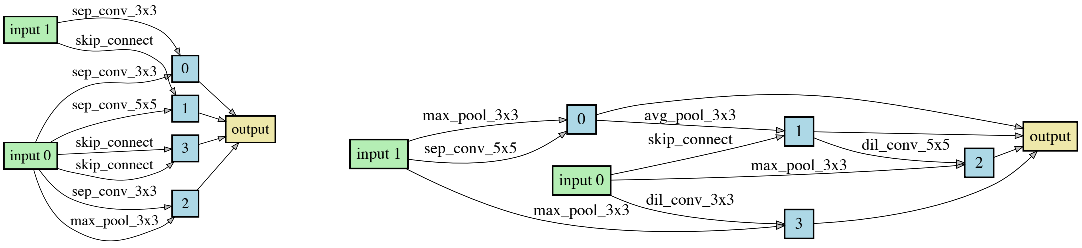
We use *entropy*, denoted by $\rho_e$, to measure the monotonicity of a given objective set. The *entropy* $\rho_e$ is the Spearman Coefficient measuring the rank difference between the objective set and an arbitrary increasing collection (called base set, such as $\{1, 2, 3, 4, 5\}$). The objective set is the collection of $\rho_{sp}$ along a certain reduction factor dimension, such as $\rho_{sp}$ of reduced settings $\{c_0r_0s_0e_{30}, c_1r_0s_0e_{30}, c_2r_0s_0e_{30}, c_3r_0s_0e_{30}, c_4r_0s_0e_{30}\}$ along the dimension of reduction factor $c$. If the absolute value of $\rho_e$ is closer to 1, it indicates that the objective set has a more apparent increasing ($\rho_e$ approximates 1) or decreasing ($\rho_e$ approximates $-1$) trend. Otherwise (e.g., $\rho_e$ approximates 0) the monotonicity of the objective set is less apparent. Since the true values of the inputs will be transferred to the ranks, the choice of base set will not affect the final results if it is a set of increasing numbers.

## E. Experiments

### E.1. Implementation Details of EcoNAS

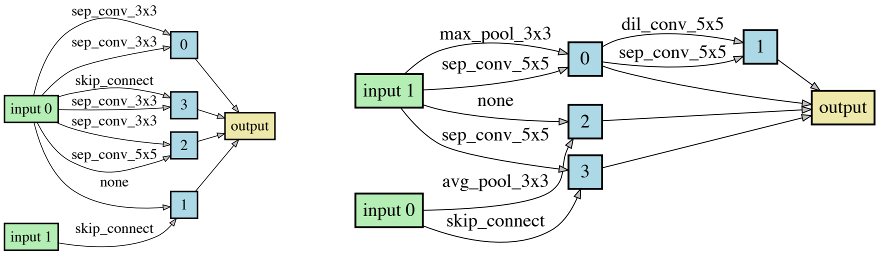
**Search space.** The search space of EcoNAS consists of 8 operations, which follow the previous work [8] and are listed as follows:

- Zeros
- 3x3 average pooling
- 3x3 max pooling
- 3x3 Separable Convolution
- Identity
- 5x5 Separable Convolution
- 3x3 Dilated Convolution
- 5x5 Dilated Convolution



(a) Normal Cell (b) Reduction Cell

Figure A4. Normal and reduction cell structures of first-place model, whose error rate is 2.62% on CIFAR-10.



(a) Normal Cell (b) Reduction Cell

Figure A5. Normal and reduction cell structures of second-place model, whose error rate is 2.67% on CIFAR-10.

Each cell in the network consists of 4 nodes (Fig. 2). The generation of each node follows the 5 steps described in Section C, except that the operation sets are different. In one cell, the node outputs that are not used will be concatenated together as the cell output [10, 11].

**Search strategy.** We use the setting of $c_4r_4s_0$ and the batch size is 384. Every network is trained on a single GPU. In every cycle, the chosen networks will be trained for 20 epochs and the maximum training length for each network is 60 epochs, *i.e.*, the complete reduce setting is $c_4r_4s_0e_{60}$, which has been found to be effective in the main text. The other hyper-parameters remain the same as stated in Section E.2. We use $P_{20}$, $P_{40}$ and $P_{60}$ to denote the networks trained for 20, 40, and 60 epochs, respectively. In each cycle, 16 networks will be chosen from the population and be mutated, and the top-8 and top-4 networks in $P_{20}$ and $P_{40}$ will continue to be trained for 20 epochs, which means that no more than half of the networks in the $P_{20}$ and $P_{40}$ set will get chance to be continually trained. When the process is finished, we only retrain and find the best model from top-5 models from $P_{60}$. The searched models that achieve the best and the second best results are shown in Figure A4 and Figure A5, respectively.

## E.2. Implementation Details on CIFAR-10

This section provides the details of training strategies for the original and reduced settings on CIFAR-10 (Section 3 and 5.1). In the original setting, we train each network from scratch for 600 epochs with batch size of 96. Cosine learning rate schedule is used with $lr_{max} = 0.025$ and $lr_{min} = 0.001$ and the weight decay is $3e{-}4$ [8]. Additional enhancements including cutout [2], path dropout [6], and common data augmentations follow the previous work [8].

The implementation for the reduced setting follows that for the original setting, except those as follows:

1. The number of training epochs is decided by the reduction factor $e$. But the cosine learning rate scheduler still finishes a completed cosine cycle within the reduced epochs.
2. Path dropout is excluded in the reduced setting because we empirically find that the evaluation ability of reduction settings will increase if path dropout is excluded. The possible reason is that we use very small number of epochs, which is not favored by path dropout.
3. The images are resized to reduced resolution after padding and random cropping, and the cutout length is adjusted according to the reduced resolution.

## E.3. Implementation Details on ImageNet

This section provides the the details on training strategies on ImageNet (Section 5.1). The networks are trained for 150 epochs with batch size 2048 on 32 GPUs. The learning rate also follows a cosine annealing schedule with $lr_{max} = 0.8$ and $lr_{min} = 0.0$. We use warmup [3] to start the learning rate from $0.2$ and then increase it linearly to $0.8$ in the first 2 epochs. The weight decay for all networks is $3e{-}5$. We also use common data augmentation methods following [4].

## References

[1] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. In *ICLR*, 2019.

[2] Terrance DeVries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. *CoRR*, abs/1708.04552, 2017.

[3] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *CoRR*, abs/1706.02677, 2017.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, Jun 2016.

[5] Alex Krizhevsky. Learning multiple layers of features from tiny images, 2009. Technical report,.

[6] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. In *ICLR*, 2017.

[7] 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*, 2018.

[8] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. In *ICLR*, 2019.

[9] Hieu Pham, Melody Y. Guan, Barret Zoph, Quoc V. Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. In *ICML*, 2018.

[10] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. Regularized evolution for image classifier architecture search. In *AAAI*, 2019.

[11] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning transferable architectures for scalable image recognition. In *CVPR*, 2018.