Supplementary of "FairNAS: Rethinking Evaluation Fairness of Weight Sharing Neural Architecture Search"

Xiangxiang Chu* Bo Zhang Ruijun Xu
{chuxiangxiang,zhangbo11,xuruijun}@xiaomi.com

1. More Discussion of Algorithms

1.1. Supernet Training

Figure 4 details the supernet training stage of our approach. In fact, it's inherently efficient regarding GPU utilization. Even on powerful machines such as Tesla V100, it can make full use of GPU without special optimization. As most of the existing deep learning frameworks allow paralleled execution between data generation and gradient calculation, our algorithm can exploit this parallelism to the extreme since a mini-batch of data is reused by m times of backpropagation. The GPUs are always busy because the data is ready whenever required, which shortens the training time. Moreover, our method works in a single-path way, which is memory friendly.

Irregular Search Spaces. Note that SF in the paper can be easily extended by a preprocessing function in case of irregular search spaces (i.e. the number of operations are not the same for each layer). We only need to make a minor modification of Algorithm 1. Say the l-th layer has m_l choices. Suppose $M = \max(m_l)$, we randomly choose $M - m_l$ extra operations from m_l choices and regard these extra options as different ones from the original search space. Therefore, the input condition of Algorithm 1 still hold and we can use it directly. This procedure can be regarded as an approximated SF. However, perfect SF for irregular cases remains as our future work.

1.2. Evolutionary Searching Pipeline

With our supernet fairly trained as a model evaluator, we adopt an evolutionary-based algorithm for searching, detailed in Algorithm 2 (main text) and Figure 1. Generally, it is built on the ground of MoreMNAS [5] by replacing its incomplete-training evaluator with our fairly trained supernet. FairNAS supernet exhibits tremendous speed-up in terms of GPU days by two orders of magnitudes. We also use Proximal Policy Optimization as the default reinforcement algorithm [11].

*This work was done when all the authors were at Xiaomi AI Lab.

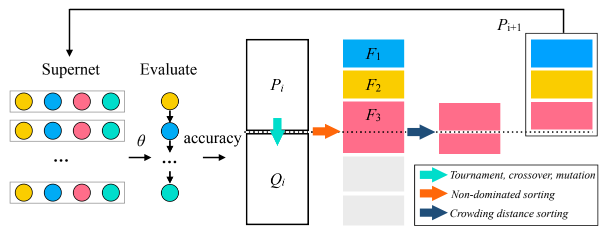

Figure 1. Evolutionary searching with the supernet trained with *strict fairness*. In each generation, candidate models in the current population inherit weights from the supernet for evaluation. Their estimated accuracies are fed into the searching pipeline as one of the objectives. The evolution loops till Pareto optimality.

NAS Methods	Type	C_t	C_s	EF	SF
SMASH [2]	Hypernet	-	-	✗	✗
One-Shot [1]	Supernet	$4^{\ddagger}$	3.3	✗	✗
DARTS [9]	Gradient-based	$0.5^{\dagger}$	0	✗	✗
FBNet [13]	Gradient-based	9	0	✗	✗
ProxylessNAS [3]	Gradient-based/RL	8.3	0	✗	✗
SPOS [6]	Supernet+EA	12	<1	✗	✗
Single-Path NAS [12]	Gradient-based	$1.25^{\ddagger}$	0	✓	✗
FairNAS (Ours)	Fair Supernet+EA	10	2	✓	✓

Table 1. Comparison of state-of-the-art weight-sharing NAS methods as per cost and fairness basis. C_t, C_s: train and search cost measured in GPU days. EF: Expectation Fairness, SF: Strict Fairness. †: searched on CIFAR-10, ‡: TPU

2. A Fairness Taxonomy

We compare current weight-sharing NAS methods based on the defined fairness in Table 1. SPOS [6] satisfies Expectation Fairness, while FairNAS meets Strict Fairness.

3. Experiment Details

Dataset. The supernet experiments are performed on ImageNet [10] and we randomly select 50,000 images from the training set as our validation set (50 samples from each class). The remaining training set is used as our training set, while the original validation set is taken as the test set to measure the final performance of each model.

3.1. Architectures of Searched Models

The searched FairNAS-A, B and C models are illustrated in Figure 2.

3.2. Hyperparameters for MoreMNAS variant

We list the hyperparameters for the adopted MoreMNAS [5] variant in Table 2. It has a population N of 64 models. It also takes a hierarchical mutation strategy. Respectively, p_{rm}, p_{re}, p_{pr} indicate probabilities for random mutation, reinforce mutation and prior regulator, where p_{re} again is divided into p_{K-M} for *roulette wheel selection*, and p_M for reinforced controller.

Item	value	Item	value
Population N	64	Mutation Ratio	0.8
p_{rm}	0.2	p_{re}	0.65
p_{pr}	0.15	p_M	0.7
p_{K-M}	0.3		

Table 2. Hyperparameters for the second-stage EA search.

3.3. Training of stand-alone models

We picked 13 models to train from scratch whose one-shot accuracies are approximately evenly spaced, ranging in $[0.641, 0.7]$. We keep the exactly same hyperparameters as the supernet training. Their corresponding stand-alone accuracies are within $[0.692, 0.715]$. Figure 3 plots the training process, from which we observe the ranking of one-shot models are generally maintained. The model-meta (indices of operations) of these 13 models are listed in Table 3. Besides, the mapping from an index in model-meta to a searchable operation is given in Table 4.

Index	Model Meta
0	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
1	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]
2	[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0]
3	[0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0]
4	[0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0]
5	[0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0]
6	[0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 2, 0]
7	[0, 1, 0, 1, 1, 4, 1, 1, 1, 1, 1, 0, 0, 1, 2, 0]
8	[0, 1, 4, 1, 0, 3, 1, 1, 1, 1, 1, 1, 0, 2, 0, 0]
9	[0, 1, 0, 0, 1, 5, 1, 1, 0, 5, 1, 1, 0, 1, 2, 3]
10	[3, 1, 4, 1, 3, 4, 1, 4, 1, 3, 1, 1, 3, 1, 2, 0]
11	[0, 1, 4, 3, 1, 3, 1, 1, 1, 3, 4, 1, 3, 1, 2, 3]
12	[1, 5, 3, 2, 1, 4, 3, 4, 1, 5, 1, 1, 3, 5, 5, 3]

Table 3. Model-meta of 13 sampled stand-alone models for ranking analysis.

Model Meta Index	kernel	Expansion Rate
0	3	3
1	5	3
2	7	3
3	3	6
4	5	6
5	7	6

Table 4. Mapping between model-meta index and operations

3.4. Evolutionary Searching

The evolutionary search of FairNAS based on MoreMNAS variant [5] is shown in Figure 4. At each generation, 64 models are evaluated by our fair supernet, after 200 generations, the evolution converges, the Pareto-front is shown in bright yellow, each dot represents a candidate network.

3.5. Object Detection

For object detection, we treat FairNAS models as drop-in replacements for RetinaNet's backbone [7]. We follow the same setting as [7] and exploit MMDetection toolbox [4] for training. All the models are trained and evaluated on MS COCO dataset (train2017 and val2017 respectively) [8] for 12 epochs with a batch size of 16. The initial learning rate is 0.01 and decayed by $0.1\times$ at epochs 8 and 11.

All baselines in the paper are mobile networks. The input features from these backbones to the FPN module are from the last depthwise layers of stage 2 to 5[1]. The number of output channels of FPN is kept 256 as [7]. We also use $\alpha = 0.25$ and $\gamma = 2.0$ for the focal loss. Given longer training epochs and other tricks, the detection performance can be improved further. However, it's sufficient to compare the transferability of various methods.

[1] We follow the typical nomination for the definition of stages and the orders start from 1.

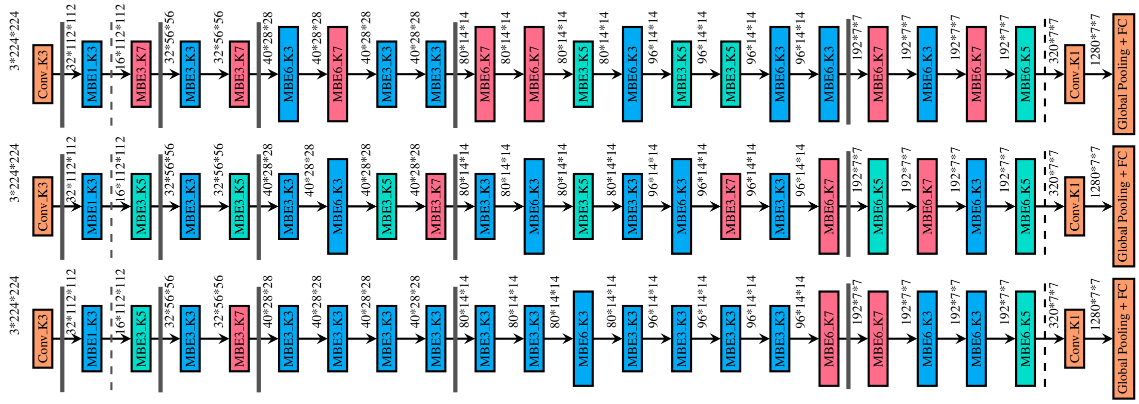

Figure 2. Architectures of FairNAS-A,B,C (from top to bottom). MBEx_Ky means an expansion rate of x and a kernel size of y for its depthwise convolution

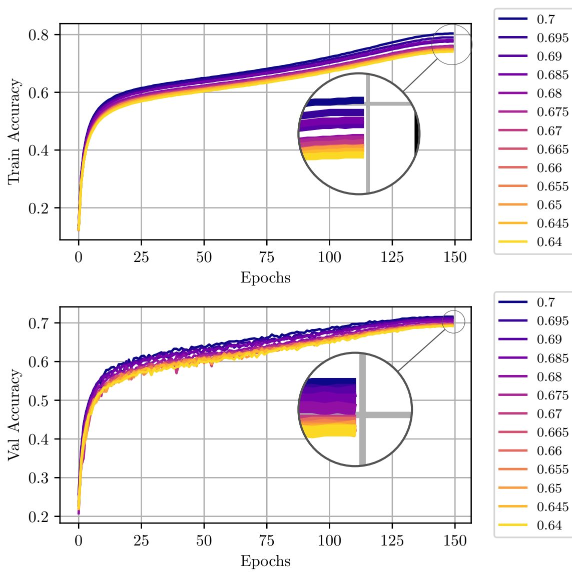

Figure 3. Train and validation accuracies (ground truth) of all 13 stand-alone models when being fully trained with the same hyperparameters. Lines are labelled with corresponding one-shot accuracies (predicted) sorted in descending order (as reflected by color gradient).

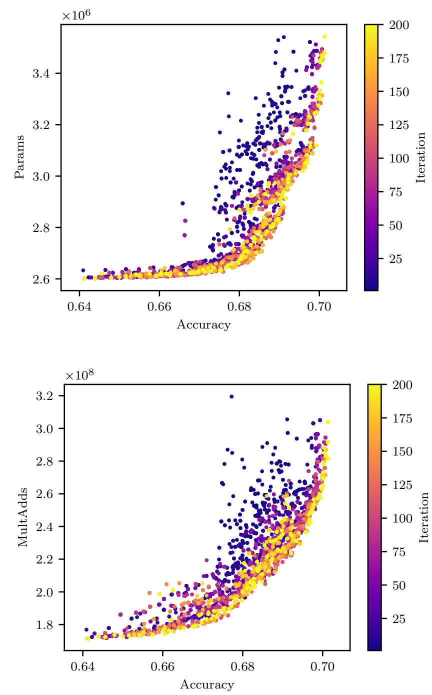

Figure 4. FairNAS evolution process of 200 generations, with 64 models sampled in each generation. Number of parameters, multiply-adds are charted with top-1 accuracies on the ImageNet validation set.

References

[1] Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc Le. Understanding and Simplifying One-Shot Architecture Search. In *International Conference on Machine Learning*, pages 549–558, 2018. 1

[2] Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. SMASH: One-Shot Model Architecture Search through HyperNetworks. In *International Conference on Learning Representations*, 2018. 1

[3] Han Cai, Ligeng Zhu, and Song Han. ProxylessNAS: Direct Neural Architecture Search on Target Task and Hardware. In *International Conference on Learning Representations*, 2019. 1

[4] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 2

[5] Xiangxiang Chu, Bo Zhang, and Ruijun Xu. Multi-Objective Reinforced Evolution in Mobile Neural Architecture Search.

In *European Conference on Computer Vision Workshop*, 2020. 1, 2

[6] Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, and Jian Sun. Single Path One-Shot Neural Architecture Search with Uniform Sampling. *European Conference on Computer Vision*, 2020. 1

[7] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection. In *International Conference on Computer Vision*, 2017. 2

[8] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision*, 2014. 2

[9] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable Architecture Search. In *International Conference on Learning Representations*, 2019. 1

[10] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 1

[11] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. *arXiv preprint. arXiv:1707.06347*, 2017. 1

[12] Dimitrios Stamoulis, Ruizhou Ding, Di Wang, Dimitrios Lymberopoulos, Bodhi Priyantha, Jie Liu, and Diana Marculescu. Single-Path NAS: Designing Hardware-Efficient ConvNets in less than 4 Hours. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2019. 1

[13] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. FBNet: Hardware-Aware Efficient ConvNet Design via Differentiable Neural Architecture Search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 1