# Improving the Rank Consistency of One-Shot NAS: Collaborative Solution to Multi-Model Forgetting and Unfair Gradient Descent

Zhaokai Zhang, He Cai, Chunnan Sheng, Lamei Chen, Tianpeng Feng*, Yandong Guo
OPPO Research, China
zhangzhaokai1, caihe, shengchunnan, chenlamei, fengtianpeng, guoyandong@oppo.com

## Abstract

*One-shot Neural Architecture Search (NAS) is a weight-sharing method that can significantly reduce the training cost compared with traditional NAS methods. However, the consistence issue still remains to be solved. The performance of the sub-network sampled from the supernet is inconsistent with the performance of the same network trained independently. Progressive Shrinking works effectively in improving the consistency between candidates and the supernet but it has severe side-effects because of the unfair training. To address the drawback above, we propose Adapted Progressive Shrinking (APS) and Sub-supernet Enhancing (SSE). APS is to progressively train the supernet with distinct shrinking process of depth adapted for different network stages. SSE separates the one-shot supernet into several sub-supernets to efficiently enhance underestimated sub-networks with distillation. In addition, we come up with Candidate Plasticity Training (CPT) that optimizes sampled sub-networks and a subset of the most diverse sub-networks in every step to regularize the supernet training. Our approach wins 2nd place in the supernet track of the CVPR 2022 lightweight NAS competition. Comprehensive experiments are conducted to demonstrate the efficacy of our method.*

## 1. Introduction

Neural Architecture Search (NAS) is an automatic architecture engineering method that has made huge progress on many computer vision tasks such as image classification [7] and object detection [10]. Various search strategies are adopted to explore the optimal neural architectures, including evolutionary methods [6], reinforcement learning (RL) [10], and gradient-based methods [5], etc. Most of these methods have outperformed hand-designed models but still suffer from unaffordable computation costs. One-shot NAS greatly alleviates the computational burden since it shares weights between architectures that have edges of the supernet in common and no extra training is required other than the weights of a single supernet [3].

However, the performance of models with inherited weights from the supernet is often biased. Such bias inevitably results in an improper ranking of candidate performance. NSAS [8] designs a greedy novelty search method to find the most representative constraints and reduce the multi-model forgetting in supernet training. FairNAS [2] believes that the bias is generated by inherent unfairness and proposes Strict Fairness principle to fairly select sub-networks. Once-for-all (OFA) [1] proposes a progressive shrinking algorithm to avoid interference among sub-networks of different sizes. Inspired by the methods above, we design our framework based on resnet-48.

First of all, we conduct APS on the supernet with distinct shrinking process according to the network dimensions (depths and widths) of different stages. Specially, we start to train sub-networks that have a shallow 4th unit at the beginning of training because they tend to be more sensitive to the progressive depth shrinking process. In addition, we also propose SSE that separates the one-shot supernet into multiple sub-superents and helps to enhance underestimated subnets with Knowledge Distillation [4].

To overcome multi-model forgetting, we come up with CPT, optimizing sampled sub-networks and a subset of the most diverse sub-networks in every step to regularize the supernet training. The most diverse sub-networks are referred to as a constraint candidate group, whose population keeps evolving during each training step. We define the diversity of sub-networks according to their widths. CPT further improves the rank consistency of the one-shot supernet. The complete framework of our approach is illustrated in Fig. 1.

## 2. Method

### 2.1. Progressive shrinking

Progressive shrinking(PS) sequentially optimizes parameters of the largest sub-network to the smallest. Different from the uniform sampling strategy that tries to promise an equal number of iterations over each sub-network, pro-

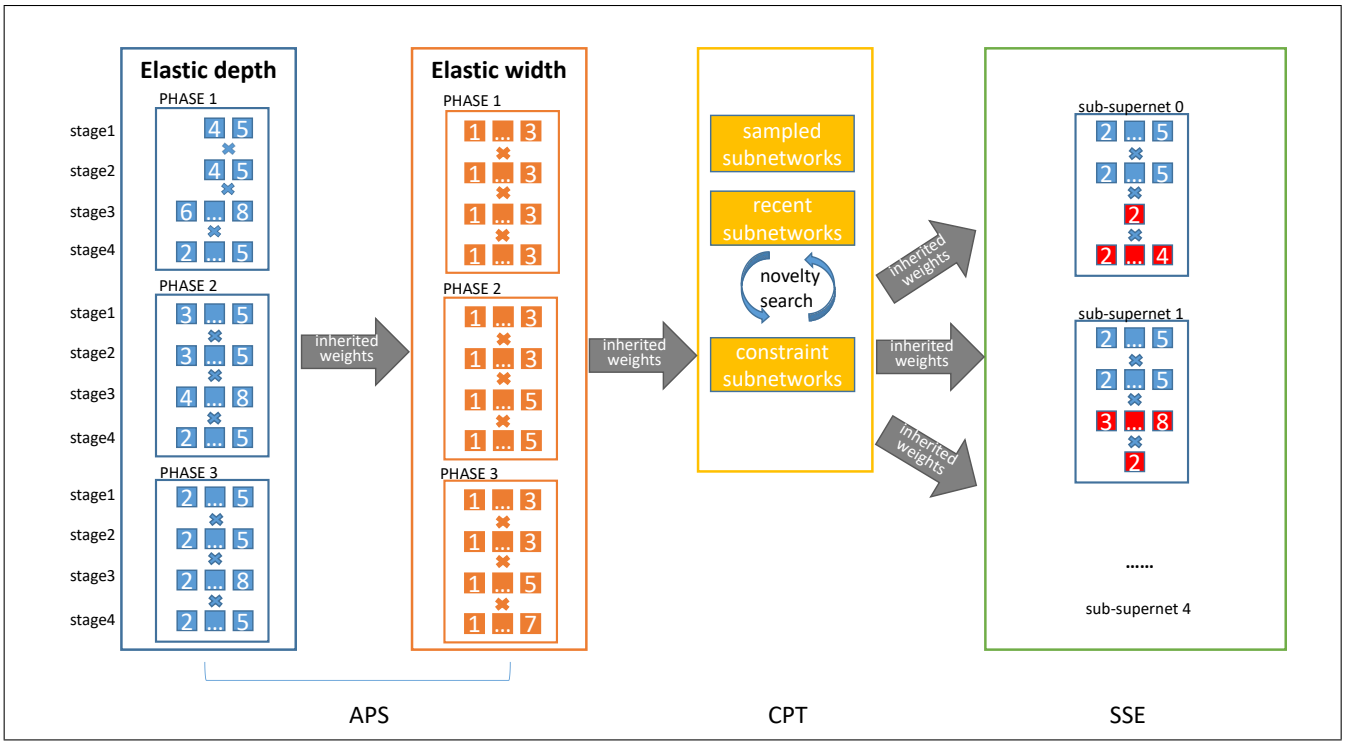


Figure 1. The framework of our approach. We first conduct APS to progressively optimize the supernet. Then, we adopt CPT to regulate supernet training. Finally, we come up with SSE to separate the supernet into 5 sub-supernets and further promise fair and sufficient training among subnets of different depths and widths.

gressive shrinking unfairly puts more training emphasis on larger subnets. The j-th phase of progressive shrinking is denoted by $S_j$. Then, the optimization of the supernet can be formalized as:

$$\min_{W} \sum_{j} \sum_{i} \mathcal{L}_{train}(S_j(W, arch_i)) \tag{1}$$

where $W$ denotes weights of the supernet, and $arch_i$ denotes a architecture in current sampling space. Such unfair training dramatically reduces the interference between sub-networks of different sizes and further improves the ranking consistency of most sub-networks selected from the supernet. However, the unfairness inevitably leads to severe degradation of some sub-networks. We propose our solution to this issue in Sec. 2.2 and Sec. 2.3.

## 2.2. Adapted Progressive Shrinking

All the units share the same shrinking process of depth in PS, but it does not work well on the resnet-48 network. Besides, we find that the performance of units closer to the output layer tends to be more sensitive to depth shrinking. If we allocate the same shrinking process for all units, sub-networks that have fewer layers in stage 4 are more likely to be underestimated in the supernet.

To get more sub-networks fairly and sufficiently trained, we propose Adapted Progressive Shrinking, namely APS. APS aims to customize the shrinking process for each unit according to its sensitivity. When it comes to the search space of resnet-48, we set the same shrinking process for the first two stages. The PS process of the third and fourth stages are differently configurated according to their network properties respectively (details explained in Sec. 3.2).

## 2.3. Sub-supernet Enhancing

Although APS largely reduces the side effects caused by unfair training, the underestimation over performance of some sub-networks in the supernet still remains to be settled. We suspect that the reason for the underestimation is insufficient and unfair training. It is difficult to optimize only the underestimated sub-network in a one-shot supernet, so we introduce the idea of few-shot NAS [9] and separate the supernet into multiple sub-supernets, each covering different regions of the search space.

In our experiment, We use five sub-supernets, grouped by the depth of stages 3 and 4 in resnet-48 and their weights are inherited from the same one-shot supernet. Specifically, sub-supernet 0 is built of the following architectural configurations: '22'(the first number is the depth of stage 3, and the second number is the depth of stage 4), '23', '24'.

Similarly, sub-supernet 1 is built of '32', '42', '52', '62', '72', '82', sub-supernet 2 is built of '83', '84', '85', sub-supernet 3 is built of '25', '35', '45', '55', '65', '75' and sub-supernet 4 is built of all the combinations left. We observe that most of the networks in sub-supernets 0, 1, 2, 3 are under-trained compared with the networks in sub-supernets 4. So sub-supernets 0,1,2,3 take more training steps than sub-supernets 4 to get enhanced performence.

### 2.4. Candidate Plasticity Training

It's generally accepted that optimizing some sub-networks in a supernet often inevitably leads to the degradation of others because the weights they partially share are inherited from the same supernet. Inspired by the Novelty Search based Architecture Selection (NSAS) [8], we keep a set of sub-networks as constraint candidates and optimize the sampled sub-networks as well as constraint candidates in each training step. Constraint candidates are the most different sub-networks based on their widths. We redesign the distance between two sub-networks according to the number of channels of each layer.

$$D_{i,j} = \sum_{k=0}^{n} |C_i^k - C_j^k| \tag{2}$$

Where $C_i^k$ is the width encoding bit of k-th layer in sub-network i. We replace '0' in width encoding bits with '9' because the actual number of channels that respond to '0' is closer to '7' than '1'.

## 3. Experiments And Results

In this section, we show comprehensive experiments on our proposed framework. First, we train the supernet following our three-phase strategy of APS and compare three versions of APS. Next, we conduct candidate plasticity training and sub-supernet enhancing experiments on the best APS version with inherited weights. All proposed methods have shown positive effects on improving the ranking consistency of candidate performance.

### 3.1. Calibrate BN

The statistics data of BN is not available after supernet training because of varying architectures. To precisely evaluate sub-networks, we re-calibrate the batch norm statistics for 50 steps with training data and a batch size of 256. We also try more configurations of calibration steps (Eg. 100, 200, and 300 steps) and figure out that 50 steps are quite enough.

### 3.2. Adapted Progressive Shrinking

First, we apply our 3-phase strategy of APS to support elastic depth (D). In each training step, 4 sub-networks are sampled and their gradients are accumulated before updating parameters. The first phase (D1: [5,4], D2:[5,4], D3:[8,6], D4:[5,4]) takes 5 epochs with an initial learning rate of 0.001. The second phase (D1: [5,3], D2:[5,3], D3:[8,4], D4:[5,3]) takes 10 epochs with an initial learning rate of 0.002. The third phase (D1: [5,2], D2:[5,2], D3:[8,2], D4:[5,2]) takes 15 epochs with an initial learning rate of 0.004.

Next, we continue to apply our 3-phase strategy of APS to support elastic width (W). Four sub-networks are sampled in each training step. The first phase (W1: [1,3], W2:[1,3], W3:[1,3], W4:[1,3]) takes 5 epochs with an initial learning rate of 0.001. The second phase (W1: [1,3], W2: [1,3], W3:[1,5], W4:[1,5]) takes 10 epochs with an initial learning rate of 0.002. The third phase (W1: [1,3], W2: [1,3], W3:[1,5], W4:[1,7]) takes 25 epochs with an initial learning rate of 0.004.

Furthermore, we have adjusted the depth shrinking process to fully support elastic depth on our supernet. Apart from searching for the best learning rate and the number of epochs, we focus on the sampling space in each phase. We believe that the fourth stage makes up the deepest part of the whole network and thus plays the most important role in performance evaluation. So we carry out more experiments in different configurations on stage 4. We replace the sampling space in the second phase (D4:[5,3]) with (D4:[5,2]), named APS_v2. Based on APS_v2, we change the first phase (D4:[5,4]) to (D4:[5,2]), named APS_v3.

| Setting | Pearson Coeff. |
|---|---|
| baseline | 0.79884 |
| APS_v2 | 0.80141 |
| APS_v3 | 0.80263 |

Table 1. The ranking correlation of sub-networks with different APS version. "APS" denotes Adapted Progressive Shrinking.

The results of different versions of APS are displayed in Tab. 1. The Pearson Coeff. of APS_v3 is 0.379% higher than the baseline without extra computational cost. We find it more beneficial to train subnets that are shallow in stage 4 earlier.

### 3.3. Candidate Plasticity Training

For candidate plasticity training, we keep 100 sub-networks as recent sub-networks and 10 sub-networks as constraint sub-networks. When updating constraint sub-networks, we take the mean distance of the top 10 nearest neighbors as a surrogate. The loss ratios of constraint sub-networks and sampled sub-networks are both 0.5. It takes 30 epochs for candidate plasticity training with an initial learning rate of 0.001. And we sample one sub-network in each step to improve training efficiency.

| Setting | Pearson Coeff. |
|---|---|
| APS_v3 | 0.80263 |
| APS_v3-CPT | 0.81415 |

Table 2. The ranking correlation of sub-networks with candidate plasticity training. "CPT" denotes candidate plasticity training.

The results of candidate plasticity training are shown in Tab. 2. After candidate plasticity training, the Pearson Coeff. has increased by 1.152% compared with the previous supernet trained with APS_v3. It is believed that candidate plasticity training has even greater potential if properly tuned with suitable parameters.

### 3.4. Sub-supernet Enhancing

First, we implement sub-supernet enhancing on stage 4. The search space is divided into 3 groups. The first group is identical with sub-supernet 0, which includes all sub-networks that the depth of stage 4 is 2. The second group contains all sub-networks whose depth in stage 4 is 5. The third group, sub-supernet 2, covers all sub-networks left. The result, denoted as APS_v3-CPT-SSE4, is shown in Tab. 3.

We further conduct sub-supernet enhancing on both stages 3 and 4 following the configurations in Sec. 2.3. Sub-supernets 0,1,2,3 are trained with an initial learning rate of 0.003. The ratios of cross-entropy losses of sampled sub-networks, total cross-entropy losses of constraint sub-networks, and distillation losses of sampled sub-networks are 0.4, 0.2, and 0.4, respectively. To hold the training balance among sub-networks of different depths, we train sub-supernet 0 and 2 for 6 epochs and 9 epochs for sub-supernet 1 and sub-supernet 3.

| Setting | Pearson Coeff. |
|---|---|
| APS_v3-CPT | 0.81415 |
| APS_v3-CPT-SSE4 | 0.82166 |
| APS_v3-CPT-SSE34 | 0.83768 |
| APS_v3-CPT-SSE34-Dis | 0.84652 |

Table 3. The ranking correlation of sub-networks with few-shot training. "SSE4" denotes the sub-supernet enhanced on stage 4. "SSE34" denotes the sub-supernet enhanced on stage 3 and 4. "Dis" means using distillation.

Tab. 3 exhibits the experimental results of sub-supernet enhancing. The result of sub-supernet enhancing based on stage 4 is 0.751% higher than APS_v3-CPT and applying sub-supernet enhancing on stages 3 and 4 raises the Pearson Coeff to 0.83768. Besides, the final result reaches 0.84652 after distillation.

## 4. Conclusion

In this paper, we improve the ranking consistency of supernet with Adapted Progressive Shrinking(APS), Candidate Plasticity Training(CPT), and Sub-supernet Enhancing(SSE). APS and SSE can improve the performance of underestimated sub-networks in the supernet, which alleviates the side effects of unfair training. CPT selects a representative subset of constraint sub-networks to regularize the supernet training to overcome multi-model forgetting problems. Comprehensive experiments show that the key points we propose are effective and can be effectively integrated to achieve better results.

## References


[1] Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once-for-all: Train one network and specialize it for efficient deployment. In *International Conference on Learning Representations*, 2019. 1

[2] Xiangxiang Chu, Bo Zhang, and Ruijun Xu. Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12239–12248, 2021. 1

[3] 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. In *European Conference on Computer Vision*, pages 544–560. Springer, 2020. 1

[4] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv e-prints*, pages arXiv–1503, 2015. 1

[5] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *International Conference on Learning Representations*, 2018. 1

[6] Yuqiao Liu, Yanan Sun, Bing Xue, Mengjie Zhang, Gary G Yen, and Kay Chen Tan. A survey on evolutionary neural architecture search. *IEEE transactions on neural networks and learning systems*, 2021. 1

[7] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pages 4780–4789, 2019. 1

[8] Miao Zhang, Huiqi Li, Shirui Pan, Xiaojun Chang, and Steven Su. Overcoming multi-model forgetting in one-shot nas with diversity maximization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7809–7818, 2020. 1, 3

[9] Yiyang Zhao, Linnan Wang, Yuandong Tian, Rodrigo Fonseca, and Tian Guo. Few-shot neural architecture search. In *International Conference on Machine Learning*, pages 12707–12718. PMLR, 2021. 2

[10] 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*, pages 8697–8710, 2018. 1