Prior-Guided One-shot Neural Architecture Search

Peijie Dong[1], Xin Niu[1], Lujun Li[2], Linzhen Xie[1], Wenbin Zou[3], Tian Ye[4], Zimian Wei[1], Hengyue Pan[1]
[1] School of Computer, National University of Defense Technology, Hunan, China
[2] Chinese Academy of Sciences, Beijing, China
[3] Fujian Provincial Key Laboratory of Photonics Technology, Fujian Normal University, Fuzhou, China
[4] School of Ocean Information Engineering, Jimei University, Xiamen, China
{dongpeijienudt, niuxin, weizimian16, hengyuepan} @nudt.edu.cn
lilujunai@gmail.com, 18810698745@163.com, alexzou14@foxmail.com, 201921114031@jmu.edu.cn

Abstract

Neural architecture search methods seek optimal candidates with efficiency weight-sharing supernet training. However, recent studies indicate poor ranking consistency about the performance between stand-alone architectures and shared-weight networks. In this paper, we present Prior-Guided One-shot NAS (PGONAS) to strengthen the ranking correlation of supernets. Specifically, we first explore effect of activation functions and propose a balanced sampling strategy based on the Sandwich Rule to alleviate weight coupling in the supernet. Then, FLOPs and Zen-Score are adopted to guide the training of supernet with ranking correlation loss. Our PGONAS ranks the 3rd place in the supernet Track Track of CVPR2022 Second lightweight NAS challenge. Code is available in https://github.com/pprp/CVPR2022-NAS-competition-Track1-3th-solution.

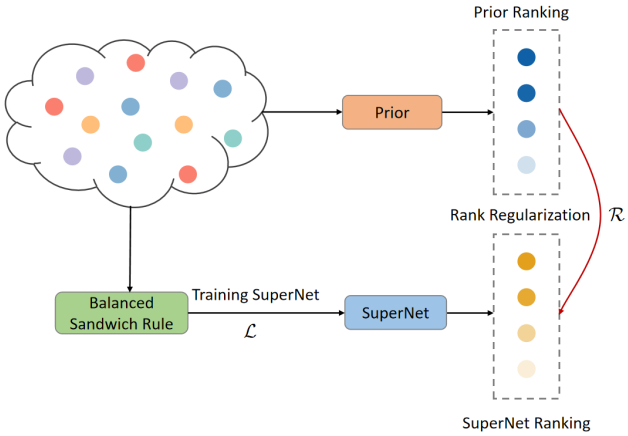

Fig. 1. Overview of the Prior-Guided One-shot NAS (PGONAS). The Balanced Sandwich Rule means that the proposed balanced sampling strategy is based on the Sandwich Rule [27]. During the training phase, priors are employed to guide the supernet training with rank loss.

1. Introduction

There are tremendous advances in deep learning, including automated machine learning. Neural Architecture Search (NAS) is a branch of automated machine learning (AutoML) that has sparked increased interest due to its remarkable progress in a variety of computer vision tasks [5,7,14,16,18,29]. By balancing performance and resource constraints, it aims to reduce the cost of human efforts in manually designing network architectures and discover promising models automatically. Many early NAS works are based on Reinforcement Learning (RL) and Evolutionary algorithm (EA). However, these techniques [25, 31] require sampling and evaluating a large number of network architectures from the search space, which can take hundreds of days with thousands of GPUs.

To alleviate this challenge, ENAS [21] is the first to present a weight-sharing mechanism for efficient NAS to optimize computation. It defines the supernet in which all models share a single copy of weights, rather than training models from scratch. DARTS [17] and DARTS-like methods [3, 22, 23] are another popular type of weight-sharing approach. They optimize the continuous architecture parameters and network weights during supernet training. In contrast, One-shot NAS [9, 12, 28] is a new paradigm that separates architecture search from supernet training. One-shot NAS uses evolutionary algorithms to sample numerous architectures in the training stage of supernet. Then, these candidates are then evaluated by inheriting weights from the well-trained supernet in seache stage. and finally, we retrain the best-performing ones. It requires less memory and is more efficient because only a portion of the candidates are activated and optimized.

Despite the fact that One-shot NAS significantly improves search efficiency, it is prone to poor performance estimation during the evaluation process. It typically searches a large and complex network space with billions of options to find the best ones, in which the supernet parameters are tightly coupled. When child models are sampled and trained in different iterations, they interfere with each other, and their accuracy in the supernet is unavoidably averaged, blurring the lines between strong and weak architectures. As a result, a well-trained supernet finds it difficult to obtain a stable and accurate performance ranking of candidate models.

In this paper, we propose Prior-Guided One-shot NAS (PGONAS), an effective strategy to address the weak correlation problem of weight sharing supernet. The PGONAS improvement for One-shot NAS can be highlighted in three aspects: architecture enhancement, sampling strategies, and prior-based regularization. First, we utilize PreLU to replace the second activation of the block to effectively enhance the predictability of the supernet in the channel dimension. Then, unlike the naive sampling strategy, we use the Sandwich Rule (maximum, middle, and minimum) sampling strategy and in place distillation. These two strategies achieve 81.56 absolute correlation coefficient, while baseline is 72.49. Encouraged by the recent train-free NAS, we also evaluate some train-free metrics and are surprised with its relatively advanced consistency (78.43 for FLOPs and 81.29 for Zen-Score). Therefore, we introduce these train-free metrics to guide the training of the supernet as a priori. Specifically, we add a consistent regularization loss for arbitrary pairs of candidates about its train-free metrics and the training losses on the supernet. To further improve, we carefully tune the weights of the loss and architectural distances. Finally, with the above advanced supernet training techniques, we obtain a consistency of 83.20 and ranked 3rd in the supernet track of the CVPR 2022 NAS Challenge.

2. Related Work

In this section, we briefly summarize the investigative techniques behind our approach, including One-shot NAS and prior metrics.

One-shot NAS. In a wide variety of computer vision tasks [10, 13, 30], manually designed neural networks had a great deal of success. Artificial architectures, on the other hand, are commonly believed to be sub-optimal. Both academia and industry have recently become more interested in neural architecture search (NAS). The early NAS work trains the child networks individually by RNN and reinforcement learning but consumes a large computer resource. Recent One-shot NAS utilize weight-sharing super-network for efficient NAS to reduce computation costs. Different child models in these algorithms share the same weights. They initially train an over-parameterized super-network utilizing different sample strategies and after that search a discrete search space with numerous candidates. Sampling strategies are important in the training stage because they affect how to train an accurate and stable super-network for performance estimation. [1], for example, trains the over-parameterized network while dropping out operators with increasing probability, allowing their weights to co-adapt. SPOS [12] proposes the uniform sampling method for supernt training based on this. Only one path is activated for each optimization step, and regular gradient-based optimizers are used to optimize it. FairNAS [6] strengthens the One-shot method by adhering to strict fairness for both super-network sampling and training. AutoSlim [26] improves correlation by optimizing the maximum minimum and intermediate paths by in-place distillation. OFA proposes a one-stage supernet training strategy through progressive shrinkage. As an alternative technology paradigm, the gradient-based methodologies [17, 22] initiate architecture parameters with each operator, including using back-propagation to jointly optimize them and network weights. Finally, magnitudes of architecture parameters are used to select the best model. In contrast, we propose a new One-shot NAS method based on Autoslim [26] & priori-guided regularization, which is not present in previous work.

Training-free Metric. In recent years, many researchers began to realize that some training-free metrics can be used to measure the capacity of neural networks. These methods are generally based on Gaussian initialization of the model and the ability to characterize random features. Specifically, TE-NAS estimates expressiveness by directly computing the number of active regions RN on randomly sampled images. The number of active regions is calculated directly on the images. NASWOT calculates the architecture score based on the kernel matrix of binary activation patterns between small batches of samples. Zen-score takes into account not only the distribution of linear regions, but also the complexity, resulting in a more accurate estimate of the expressiveness of the network. In addition, the number of parameters and the computational volume of the model are also indicators that can initially represent the capacity of the model. In many scenarios, the larger model has better performance. In this work, we select FLOPs and Zen-score to guide the training of the supernet using ranking losses.

3. Prior-guided One-shot NAS

In this section, we present our architecture enhancement & sampling strategies for weight-sharing supernet, and ranking regularization based on a priori metrics, respectively.

3.1. Weight-sharing supernet Enhancement

Weight sharing in the supernet can lead to gradient conflicts, making the supernet difficult to converge and affect-

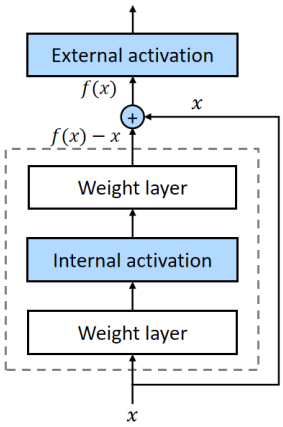

Fig. 2. Illustration of the location of activations in residual blocks.

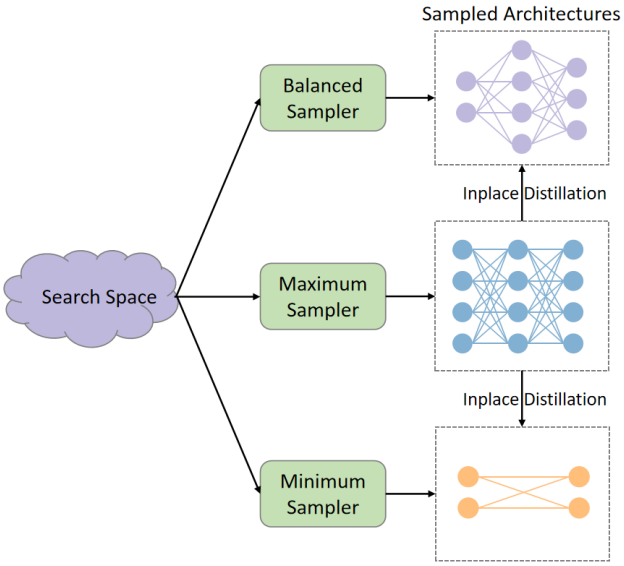

Fig. 3. Illustration of the Sandwich Rule with Balanced Sampler.

ing the ranking consistency of the supernet. Therefore, in this paper, we explore the effects of activation functions, sampling strategies, and regularization on the ranking consistency of the supernet.

Activation Functions. Inspired by [24], we explore the effects of activation functions in detailed, including the location and type of activation functions. ReLU is a nonlinear activation function that allows complex patterns in the data to be learned. However, ReLU cannot learn examples for which their activation is zero. If the input is less than 0, then it outputs zero, and the neural network cannot continue the back propagation algorithm, which is known as the dying ReLU problem. We observe that replacing ReLU with smoother activation functions helps alleviate the ranking disorder problem. Take Mish as example, it provides a strong regularization effect and helps make gradients smoother, which make it eaiser to optimize function contour. On the other hand, smoother activation functions have wider minima, which improve generalization compared to ReLU. In addition, the location of the activation functions also matters. As shown in Figure 1, there are two activation functions in the residual block. Define the activation function between the two weight layer as internal activation, and the activation function after shortcut as external activation. We found that internal activation plays an important role in the training of supernets. If we replace the internal ReLU activation with smoother activation functions such as PReLU, the supernet would suffer from gradient explosion and is difficult to converge. Also, the supernet is not sensitive to converting the external activation functions.

Sampling Strategies. Two-stage NAS requires sampling from the search space during training. Thus, sampling strategies would directly impact the optimization of the supernet. A suitable sampling strategy can alleviate the interference between sub-networks in the supernet. There are many sampling strategies in One-shot NAS, such as the uniform sampling strategy in SPOS [12], the progressive shrinking strategy in Once for all [2], and the Sandwich Rule in Autoslim [26] and BigNAS [27]. Among them, we found that the Sandwich Rule achieves a better rank correlation in our search space. we proposed a noval sampling strategy based on the Sandwich Rule. Specifically, we replace the original random sampler with a balanced sampler, which is motivated by BNAO [19]. Training of architectures of different sizes is imbalanced, which causes the evaluated performances of the architecture to be less predictable of their stand-alone accuracy. BNAO proposed sampling based on the model size of sub-networks. As shown in Figure 3, there are three samplers in PGONAS:

1. Maximum Sampler: sample the subnet with largest width and largest depth.
2. Minimum Sampler: sample the subnet with smallest width and smallest depth.
3. Balanced Sampler: sample N subnets with random width and random depth and sample one of them based on $p_i = \frac{FLOPs_i}{\sum_j FLOPs_j}$.

In-place distillation is also adopted in PGONAS. The largest model is training with true label, the smallest model and the middle size model are trained with in-place distillation.

3.2. Prior-guided Ranking Regularization

Zero-shot Prior We adopt FLOPs and Zen-Score as prior, respectively, to guide the training procedure. With the efficient zero-shot NAS [4, 15, 20], the search cost has

been greatly reduced. Among them, we finally choose the Zen-Score [15] as prior because of its stable and scale-insensitive estimation ability. Zen-Score is a zero-shot predictor for ranking architectures, which can stably correlate with the model accuracy without training the network parameters. Zen-Score can measure the network expressivity by averaging the Gaussian complexity of the linear function in each linear region. In practice, the feature map before the global average pool layer (pre-GAP) is adopted to avoid information loss.

Computation Prior Motivated by NATSBench [8], we found that the FLOPs of each subnet have a good correlation with the accuracy of the standalone model. Using FLOPs as prior to measure rank consistency is intuitive. Models with high FLOPs usually have larger capacity and are likely to achieve better performance.

Pairwise Rank Loss We first revisit the ranking consistency problem in One-shot NAS. Let Ω be the search space, defined as N candidate architectures $a_i, i \in [1, N]$. Pairwise rank loss is used to constrain the optimization process of supernet as a regular term. Assuming that the architecture a_i is better than the performance of the architecture a_j, then with the above ranking constraints, the following relationship is theoretically satisfied.

$$\mathcal{L}_{\mathcal{A}}\left(x, \theta_{a_i}^*\right) \leq \mathcal{L}_{\mathcal{A}}\left(x, \theta_{a_j}^*\right) \Rightarrow \mathcal{L}\left(x, \theta_{a_i}^s\right) \leq \mathcal{L}\left(x, \theta_{a_j}^s\right) \tag{1}$$

where $\mathcal{L}_{\mathcal{A}}$ denotes the real accuracy corresponding to stand alone training, and $\mathcal{L}$ denotes the corresponding loss function ranking in the supernet. Then the loss functions computed by architecture a_i and architecture a_j are as follows:

$$\mathcal{R}_{(i,j)}\left(\theta^s\right) = \max\left(0, \mathcal{L}\left(x, \theta_{a_i}^s\right) - \mathcal{L}\left(x, \theta_{a_j}^s\right)\right) \tag{2}$$

We introduce the balanced sampling strategy with Sandwich Rule and pairwise rank loss described by Algorithm 1.

Algorithm 1 Training supernet S with pairwise rank loss.

Require: Define $depth$ range and $width$ range.
Require: Define n as number of sampled subnets in each iteration, m as number of sampling pairs in computing rank loss, k as prior in guiding rank loss.
initialize supernet parameters θ^s with pretrained weights.
while $step$ t < T **do**
  Get next mini-batch of data x and label y.
  Execute max-network, $\hat{y} = S_{max}(x)$.
  Compute loss, $loss = criterion(\hat{y}, y)$.
  Accumulate graidents, $loss.backward()$.
  Stop graidents of $\hat{y}$ as label, $\hat{y} = \hat{y}.detach()$.
  Execute min-network, $\bar{y} = S_{min}(x)$.
  Compute loss, $loss = criterion(\hat{y}, \bar{y})$.
  Accumulate gradients, $loss.backward()$.
  while $step\ i < (n-2)$ **do**
    Get n networks $a_i (i = 1, .., N)$ and their FLOPs $FLOPs_i (i = 1, ..., N)$.
    Sample network based on probability $p_i = \frac{FLOPs_i}{\sum_j FLOPs_j}$
    Execute the sampled network, $\tilde{y} = S_{rand}(x)$.
    Compute loss, $loss = criterion(\hat{y}, \tilde{y})$.
    Accumulate gradients, $loss.backward()$.
  end while
  while $step\ i < m$ **do**
    Sample random network a and b.
    Execute random network, $y_a = S_a(x), y_b = S_b(x)$.
    Compute prior, $k_a = (S_a), k_b = K(S_b)$.
    Compute loss, $loss_a = criterion(y_a, y), loss_b = criterion(y_b, y)$.
    Compute rank loss, $loss_r = \lambda \mathcal{R}_{(a,b)} = \lambda \mathcal{L}_r(k_a, k_b, loss_a, loss_b)$.
    Accumulate gradients, $loss_r.backward()$.
  end while
  Update weights, $optimizer.step()$.
  Clear graidents, $optimizer.zero\_grad()$.
end while

4. Experiments

Search Space. The Search Space is builded based on the ResNet48 backbone. Both the depth and the expansion ratios are searchable. There are 4 stages in the backbone. The basic channel configuration of 4 stages is 64, 128, 256 and 512. The candidate block number of the 1st, 2nd and 4th block is ranging from 2 to 5 and the candidate block number of the 3rd block is ranging from 2 to 8. The candidate expansion ratio is $[1.0, 0.95, 0.9, 0.85, 0.8, 0.75, 0.7]$ and the channel number should be divided by 8. There are around 5.06×10^{19} sub-networks in total.

Dataset. ImageNet-mini has 34,745 training images and 3,923 validation images in 1000 classes with resolution 224×224. We use the official training/validation split provided by Kaggle in our experiments.

Training Settings. For all experiments, we use the SGD optimizer with momentum 0.9; weight decay 0; initial learning rate 0.001 with batch size 256. We use cosing learning rate decay with warmup for 5 epoch. We first pretrain the supernet for 90 epochs in ImageNet-1k, and then training supernet with the Sandwich Rule for 70 epochs. Following the investigation in NASVIT [11], stronger regularization(e.g., large weight decay, dropout, dropPath) and stronger data augmentation schemes(e.g., CutMix, Mixup, Randaugment) are not utilized in our experiments.

Validation Metrics Following the setting of CVPR2022 challenge, the correlation of the rank consistency in One-shot NAS is quantified with the Pearson correlation coefficient. In these metrics, the rank correlation is bounded between -1 (disagreement) and 1 (agreement), where for 0 there is no significant correlation.

4.1. Results of Prior-Guided NAS

We evaluate the effectiveness of smoother activation functions, balanced sampling strategy and rank loss with different schedulers.

Results of Different Activations The experimental results are displayed in Table 1, where the original ReLU activation function is replaced with multiple smoother activation functions. From the table, we observe that all of the activation functions achieve better rank consistency, which prove that information loss caused by ReLU would reduce the ranking correlation of sub-networks in supernet.

Activation	Pearson Coeff.
ReLU	78.20
SELU	79.89
PReLU	80.65
Swish	80.60
Mish	**81.56**

Table 1. Influence of different types of activation functions.

Results of Different Sample Strategy The sampling strategy plays an important role in optimizing the supernet. The comparison results with the state-of-the-art sampling strategies are shown in Table 2. It is surprising that the progressive shrinking strategy in Once for all [2] obtain unsatisfactory results, which may indicate that although the One-stage NAS can converge well, the ranking consistency is not guaranteed. Instead, our balanced sampling strategy based on Sandwich Rule achieve good results. Specifically, we take the number of random sampled subnets as 2 and get two pairs for rank loss.

Sample Strategy	Pearson Coeff.
Uniform Sampling [12]	72.49
Progressive Shrinking [2]	69.03
Sandwich Rule [27]	78.20
Balanced Sampling	**81.63**

Table 2. Influence of different sample strategies.

Results of Loss coefficient λ and schedulers. In Table 3, we compare four different coefficient schedulers: constant regularization throughout training, warm-up scheduler that gradually increases regularization, cosine scheduler that increases and then decreases, and multistage scheduler which consists of zero stage, warm-up stage, constant stage and decreasing stage. For the warm-up scheduler, we gradually increase the loss from 0 to $\lambda_{max} = 2$ linearly in the first 20 epochs to avoid an abrupt change of the loss. In the initial phase of the training supernet, the coefficient should gradually increase to avoid an abrupt change of the loss. As shown in Table 3, the warmup scheduler gives the best results. We used this strategy in our experiments.

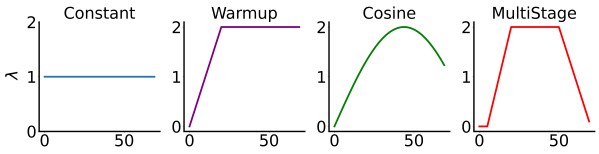

Fig. 4. Different schedules of the loss coefficient λ.

Type	Schedule	Pearson Coeff.
Rank Loss	Constant	81.97
	Warmup	83.20
	Cosine	81.31
	MultiStage	81.01

Table 3. Influence of the loss coefficient λ. Different schedules (c.f. top plots) to modify the regularization throughout training.

Results of different prior for guiding We investigate the impact of different priors in Table 4. The Params, FLOPs and Zen-Score of sub-networks are adopted as zero-cost proxy and positively correlate with the model accuracy. Among them, FLOPs and Zen-Score achieve competitive results and are used as prior in rank loss. Performance consistently improves with the help of rank loss, which highlights the importance of prior.

Type	Prior	Pearson Coeff.
w/o Rank Loss	Params	67.23
	FLOPs	78.43
	Zen-Score	81.29
w/ Rank Loss	w/o Prior	80.65
	FLOPs Guided	**83.20**
	Zen-Score Guided	83.00

Table 4. Comparison of different types of prior.

Visualization of ranking correlation Table 4 illustrates that the ranking correlation of Zen-Score is better than FLOPs without the guide of rank loss. However, FLOPs guided rank loss achieves better ranking correlation than Zen-Score guided rank loss. Figure 5 shows the correlation between different types of priors. Although the correlation between Zen-Score and FLOPs is as high as 95.97, they have different tendencies. The FLOPs proxy is con-

tinuous while the Zen-Score proxy concentrate on certain values that results in a ribbon pattern.

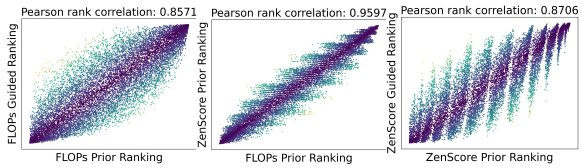

Fig. 5. Correlation between different types of priors.

5. Conclusion

In this paper, we propose Prior-Guided One-shot NAS (PGONAS) to improve the ranking correlation of supernet. First, we explore the effect of the location and type of smoother activation functions on the ranking correlation. Second, we propose a balanced sampling strategy based on the Sandwich Rule to alleviate weight coupling in the supernet. Finally, FLOPs and Zen-Score are adopted as before to guide the training of supernet with rank loss, which consistently improves the ranking correlation. Detailed experiments demonstrate that the above methods can improve the ranking correlation of sub-networks in weights inheriting and stand-alone training. In the future, we will further explore traing-free metrics and training strategies for supernets. We hope that our approach will attract the attention of the research community for NAS and new insight.

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. 2

[2] Han Cai, Chuang Gan, and Song Han. Once for all: Train one network and specialize it for efficient deployment. *arXiv preprint arXiv:1908.09791*, 2019. 3, 5

[3] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. *arXiv preprint arXiv:1812.00332*, 2018. 1

[4] Wuyang Chen, Xinyu Gong, and Zhangyang Wang. Neural architecture search on imagenet in four gpu hours: A theoretically inspired perspective. *ArXiv*, abs/2102.11535, 2021. 3

[5] Yifang Chen, Zheng Wang, Z Jane Wang, and Xiangui Kang. Automated design of neural network architectures with reinforcement learning for detection of global manipulations. *IEEE Journal of Selected Topics in Signal Processing*, 14(5):997–1011, 2020. 1

[6] Xiangxiang Chu, Bo Zhang, Ruijun Xu, and Jixiang Li. Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search. *arXiv preprint arXiv:1907.01845*, 2019. 2

[7] Zixiang Ding, Yaran Chen, Nannan Li, Dongbin Zhao, Zhiquan Sun, and CL Philip Chen. Bnas: Efficient neural architecture search using broad scalable architecture. *IEEE Transactions on Neural Networks and Learning Systems*, 2021. 1

[8] Xuanyi Dong, Lu Liu, Katarzyna Musial, and Bogdan Gabrys. Nats-bench: Benchmarking nas algorithms for architecture topology and size. *IEEE transactions on pattern analysis and machine intelligence*, PP, 2021. 4

[9] Xuanyi Dong and Yi Yang. One-shot neural architecture search via self-evaluated template network. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 3681–3690, 2019. 1

[10] Jun Fu, Jing Liu, Jie Jiang, Yong Li, Yongjun Bao, and Hanqing Lu. Scene segmentation with dual relation-aware attention network. *IEEE Transactions on Neural Networks*, pages 1–14, 2020. 2

[11] Chengyue Gong, Dilin Wang, Meng Li, Xinlei Chen, Zhicheng Yan, Yuandong Tian, Vikas Chandra, et al. Nasvit: Neural architecture search for efficient vision transformers with gradient conflict aware supernet training. In *International Conference on Learning Representations*, 2021. 4

[12] 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. *arXiv preprint arXiv:1904.00420*, 2019. 1, 2, 3, 5

[13] Hongwei Hu, Bo Ma, Jianbing Shen, and Ling Shao. Manifold regularized correlation object tracking. *IEEE Transactions on Neural Networks*, 29(5):1786–1795, 2018. 2

[14] Hanliang Jiang, Fuhao Shen, Fei Gao, and Weidong Han. Learning efficient, explainable and discriminative representations for pulmonary nodules classification. *Pattern Recognition*, 113:107825, 2021. 1

[15] Ming Lin, Pichao Wang, Zhenhong Sun, Hesen Chen, Xiuyu Sun, Qi Qian, Hao Li, and Rong Jin. Zen-nas: A zero-shot nas for high-performance image recognition. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 337–346, 2021. 3, 4

[16] Chenxi Liu, Liang-Chieh Chen, Florian Schroff, Hartwig Adam, Wei Hua, Alan L Yuille, and Li Fei-Fei. Autodeeplab: Hierarchical neural architecture search for semantic image segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 82–92, 2019. 1

[17] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018. 1, 2

[18] Jiaheng Liu, Shunfeng Zhou, Yichao Wu, Ken Chen, Wanli Ouyang, and Dong Xu. Block proposal neural architecture search. *IEEE Transactions on Image Processing*, 30:15–25, 2020. 1

[19] Renqian Luo, Tao Qin, and Enhong Chen. Balanced one-shot neural architecture optimization. *arXiv: Learning*, 2019. 3

[20] Joseph Charles Mellor, Jack Turner, Amos J. Storkey, and Elliot J. Crowley. Neural architecture search without training. *ArXiv*, abs/2006.04647, 2021. 3

[21] Hieu Pham, Melody Y Guan, Barret Zoph, Quoc V Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. *arXiv preprint arXiv:1802.03268*, 2018. 1

[22] 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*, pages 10734–10742, 2019. 1, 2

[23] Yuhui Xu, Lingxi Xie, Wenrui Dai, Xiaopeng Zhang, Xin Chen, Guo-Jun Qi, Hongkai Xiong, and Qi Tian. Partially-connected neural architecture search for reduced computational redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 1

[24] Ziwei Yang, Ruyi Zhang, Zhi Yang, Xubo Yang, Lei Wang, and Zheyang Li. Improving ranking correlation of supernet with candidates enhancement and progressive training. *ArXiv*, abs/2108.05866, 2021. 3

[25] Qing Ye, Yanan Sun, Jixin Zhang, and Jian Cheng Lv. A distributed framework for ea-based nas. *IEEE Transactions on Parallel and Distributed Systems*, 2020. 1

[26] Jiahui Yu and Thomas Huang. Autoslim: Towards one-shot architecture search for channel numbers. *arXiv: Computer Vision and Pattern Recognition*, 2019. 2, 3

[27] Jiahui Yu, Pengchong Jin, Hanxiao Liu, Gabriel Bender, Pieter-Jan Kindermans, Mingxing Tan, Thomas Huang, Xiaodan Song, and Quoc V. Le. Bignas: Scaling up neural architecture search with big single-stage models. In *ECCV*, 2020. 1, 3, 5

[28] Miao Zhang, Huiqi Li, Shirui Pan, Xiaojun Chang, Chuan Zhou, Zongyuan Ge, and Steven W Su. One-shot neural architecture search: Maximising diversity to overcome catastrophic forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 1

[29] Man Zhang, Yong Zhou, Jiaqi Zhao, Shixiong Xia, Jiaqi Wang, and Zizheng Huang. Semi-supervised blockwisely architecture search for efficient lightweight generative adversarial network. *Pattern Recognition*, 112:107794, 2021. 1

[30] Zhong-Qiu Zhao, Peng Zheng, Shou-Tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE Transactions on Neural Networks*, 30(11):3212–3232, 2019. 2

[31] Zhao Zhong, Zichen Yang, Boyang Deng, Junjie Yan, Wei Wu, Jing Shao, and Cheng-Lin Liu. Blockqnn: Efficient block-wise neural network architecture generation. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 1