Journal of Sensor Science and Technology
Vol. 30, No. 5 (2021) pp. 290-294
http://dx.doi.org/10.46670/JSST.2021.30.5.290
pISSN 1225-5475/eISSN 2093-7563

Robust architecture search using network adaptation

Amrita Rana[1] and Kyung Ki Kim[1,+]

Abstract

Experts have designed popular and successful model architectures, which, however, were not the optimal option for different scenarios. Despite the remarkable performances achieved by deep neural networks, manually designed networks for classification tasks are the backbone of object detection. One major challenge is the ImageNet pre-training of the search space representation; moreover, the searched network incurs huge computational cost. Therefore, to overcome the obstacle of the pre-training process, we introduce a network adaptation technique using a pre-trained backbone model tested on ImageNet. The adaptation method can efficiently adapt the manually designed network on ImageNet to the new object-detection task. Neural architecture search (NAS) is adopted to adapt the architecture of the network. The adaptation is conducted on the MobileNetV2 network. The proposed NAS is tested using SSDLite detector. The results demonstrate increased performance compared to existing network architecture in terms of search cost, total number of adder arithmetics (Madds), and mean Average Precision(mAP). The total computational cost of the proposed NAS is much less than that of the State Of The Art (SOTA) NAS method.

Keywords : Neural architecture search, Deep neural networks, Search space, Parameter remapping, Detection.

1. INTRODUCTION

Modern deep neural networks have numerous types of different layers with varied connections between layers. For example, skip-connection and its sub-modules are used to promote convergence of models. Currently, most deep neural network architectures are developed based on human experience, through a long and tedious process of trial and error. With the growing interest in neural architecture search (NAS), attempts have been made to automate the process of identifying effective architectures for a given deep learning problem. NAS can be defined as a technique for automating the design of artificial neural networks (ANN) [1]. The early concept of NAS included defining a search space, search strategy, and performance estimation strategy. Fig. 1 shows the concept of NAS, which is a gradient-based method for identifying good architectures [2]. The concept is based on the observation that the structure and connectivity of a neural network can be typically specified by a variable-length string.

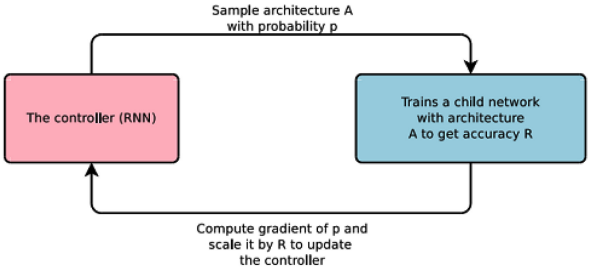

Fig. 1. Concept of NAS [2].

NAS is computationally expensive and time-consuming, and uses 450 GPUs for up to 3–4 days. Meanwhile, efficient neural architecture search (ENAS) has been proposed, which is a fast and inexpensive approach for automatic model design [3]. The main contribution of ENAS was to improve the efficiency of NAS by sharing parameters among child models, using much fewer GPU-hours than existing automatic model design approaches. The best existing architecture search algorithms are computationally demanding despite their outstanding performance.

For instance, obtaining a state-of-the-art architecture for CIFAR-10 and ImageNet required 2000 GPU days of reinforcement learning (RL), or 3150 GPU days of evolution [4,5]. Moreover, in all the popular existing approaches, architecture search is treated as a black-box optimization problem over a discrete domain, which requires a large number of architecture evaluations. Therefore, differentiable architecture search (DARTS) was a method proposed

[1] Department of Electronic Engineering, Daegu Universtiy
Daegudaero 201, Gyeongsan, Gyeongbuk 38543, Korea
[+] Corresponding author: kkkim@daegu.ac.kr
(Received: Sep. 13, 2021, Revised: Sep. 26, 2021, Accepted: Sep. 29, 2021)

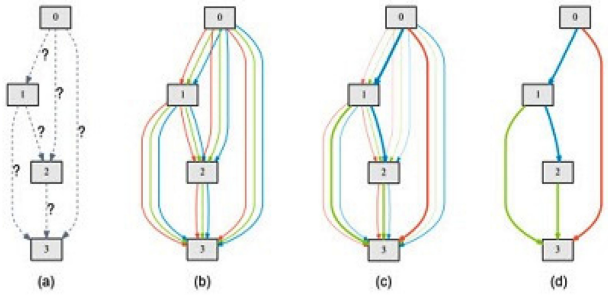

Fig. 2. Overview of DARTS [6] ; (a) Operations on the edges are initially unknown, (b) Continuous relaxation of the search by placing a mixture of candidate operations on each edge, (c) Joint optimization of mixing probabilities, and network weights by solving a bilevel optimization problem, and (d) Inducing the final architecture from the learned mixing probabilities.

for efficiently searching architecture, which tackled the problem differently. Its main contribution was to relax the search space to be continuous, instead of searching over a discrete set of candidates, such that the architecture could be optimized with respect to its validation set performance by gradient descent [6]. Fig. 2 shows the concept of DARTS.

All the aforementioned architectures are highly successful in computer vision tasks, and image classification has always served as a fundamental task for NAS. Hence, networks designed and pre-trained on classification tasks are commonly used. However, the object-detection task requires both localization and classification information for each instance, and here, the backbone architecture influences the performance. Ultimately, the architecture designed for image classification may not perform well for detection. To overcome this problem, some existing works proposed ways to improve the backbone architecture [7-10].

Pre-training is an unavoidable and costly procedure. However, training from the beginning on the target task takes numerous iterations when compared with fine-tuning from a pre-trained one. As ImageNet pre-training has been a standard rule for many computer vision tasks, there are plenty of trained models available [11]. To take advantage of these models, an NAS model was proposed based on the parameter adaptation paradigm. The proposed NAS could adapt the parameters of the manually designed base network, which is pre-trained on ImageNet or MobileNetV2 [12]. The base network is expanded to the super network, which is essentially the search space in the proposed NAS. Thus, this architecture search could obtain the optimal target architecture for detection. With this method, there is no need for pre-training on a large-scale dataset. The effectiveness of the proposed NAS is tested via experiments on detection tasks.

2. PROPOSED NAS

MobileNetV2, the most commonly used network for designing search spaces in NAS methods, is adapted as the base model. To adapt the network for the detection task, the two architecture elements (that is, kernel size and depth) of the network model are fine-tuned. Fig. 3 shows the overall framework of the proposed NAS, wherein the base network is expanded to its super-network for searching. Further, the parameters of the base network are added to the final architecture. The framework is composed of two important aspects: (1) remapping the parameters of the base model, and (2) network adaptation.

2.1 Remapping the parameter of base network

The paradigm focuses on mapping the parameters of the base model to another one. The number of inverted residual blocks *(MBConvs)* is adjusted in each step of the network.

Then, the parameters of the base network Ns can be denoted as $\{W^{(1)}\ s, W^{(2)}\ s, \ldots, W^{(l)}\ s\}$, and similarly, the parameters for the corresponding stage with m layers in the new network, Nn, can be denoted as $\{W^{(1)}\ n, W^{(2)}\ n, \ldots, W^{(m)}\ n\}$. This implies that the parameters of layers in Nn, which also exist in Ns, are simply copied from Ns. All the parameters of new layers are copied from the last layer in Ns, as expressed in the following equation:

$$f(i) = min(i, l),\ Wn^{(i)} = Ws^{(f(i))},\ \forall 1 \le i \le m \quad (1)$$

The kernel size of 3×3 is commonly used in manually designed networks. A larger kernel size is introduced in the adaptation process to expand the receptive field and capture abundant features in the detection task. To expand the 3×3 kernel to a larger size, its values are assigned to the parameters of the central 3×3 region in the large kernel. The other regions

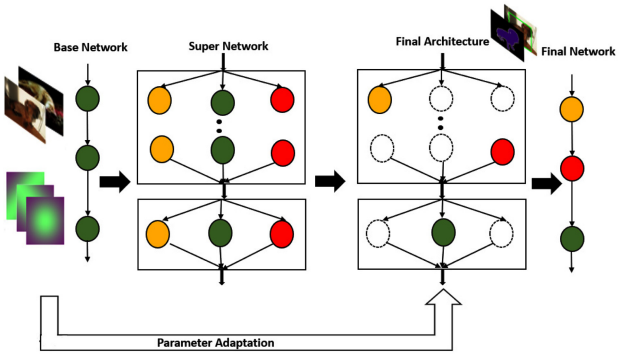

Fig. 3. Framework of proposed NAS.

surrounding the central part are assigned as 0. W^{3x3} denotes the parameters of the original 3×3 kernel and W^{kxk} denotes the parameters of the larger kernel $k \times k$. The following equation defines the remapping process of the kernel:

$$W^{kxk}_{:,:,h,w} = \begin{cases} W^{3x3}_{:,:,h,w} \text{ if } (k-3)/2 < h, w \leq (k+3)k \\ 0 \quad otherwise \end{cases}, \tag{2}$$

where h,w denote the indices of the spatial dimension.

2.2 Network Adaptation

The adaptation process is divided into three steps. First, the base network is expanded to a super network, which is an actual representation of the search space in the network adaptation process. Next, the DARTS method is used to implement network adaptation on the architecture level to obtain the final architecture. Finally, the parameters of the final architecture are adapted, and the final network is obtained. The parameter remapping discussed in Section 2.1 is implemented before the final architecture. For the *MBConv* layer, kernel settings of {3,5,7} and {3,6} expansion ratios are implemented. Similar to most differentiable NAS methods, the search space relaxes every

layer as a weighted sum of all candidate operations, as derived by the following equation:

$$\bar{o}(i)(x) = \sum_{o \epsilon O} \frac{\exp(\alpha_o^{(i)})}{\Sigma_{o' \epsilon} \exp(\alpha_{o'}^{(i)})} o^{(x)} \tag{3}$$

The proposed NAS adapts the MobileNetV2 base network to a common use detection model, SSDLite, also known as a lightweight detector [13].

3. RESULTS AND DISCUSSION

An ImageNet pre-trained MobileNetV2 model is used for the proposed NAS, and an object detection task is performed. The experiment is conducted on the MS-COCO dataset [14]. In the search process of architecture adaptation, 50% of the data is randomly sampled from the original set for validation. The input images are resized to 320×320. A standard RMSProp optimizer with a weight decay of 4×10^{-5} is used for operation weights in the search process. For the first 500 iterations, the learning rate is set from 0 to 0.03. After that, the learning rate is decayed by 0.1 at 22 epochs. The architecture optimization begins at 20 epochs. λ is set to 0.2, and τ is set to 10 for the loss function, wherein λ denotes the cross-entropy loss and τ term controls the multiplication and addition arithmetics. The search process takes 24 epochs in total, that is, 76 h on a single TITAN Xp. Table 1 lists the results on the COCO dataset, with a comparison to the literature DetNas [10] in terms of a number of parameters, λ, MAdds, and detection accuracy. The proposed NAS proves to be lightweight, using only 4.6M parameters, which is far less than those of DetNas. However, the mAP is lower than the existing one. The MAdds also exhibits comparable results, reducing the operations to 1B only.

Table 1. Object detection results on MS-COCO.

Methods	Params	MAdds	mAP(%)
DetNAS	13.41M	133.26B	33.3
Proposed NAS	4.6	1B	25.3

Table 2. Comparison of computational cost on the object detection tasks.

Methods	Total cost	Super Network (search cost)	Final network
DetNAS	68GDs	44GDs	24GDs
Proposed NAS	9GDs	3GDs	6GDs

Table 2 shows the comparison of computational cost for the object detection task. The experiments using the proposed NAS are conducted on a single TITAN Xp GPU. Here GDs denotes the number of GPU days. The MMDetection framework is used which is an open source object detection toolbox based on PyTorch.

Fig. 4 shows the architecture found after the search process, where Kx_Ey denotes the kernel size of the depthwise convolution is 'x' and expansion ratio is 'y'.

Fig. 5 shows the visualization of the searched architecture on

```
[[32, 16], ['k3_e1'], 1]|
[[16, 24], ['k7_e6', 'k5_e6', 'k3_e6', 'k3_e3'], 2]|
[[24, 32], ['k7_e6', 'k3_e6', 'k5_e6', 'k7_e3'], 2]|
[[32, 64], ['k5_e6', 'k3_e6', 'k3_e3', 'k3_e6'], 2]|
[[64, 96], ['k3_e6', 'k7_e6', 'k3_e6', 'k7_e3'], 1]|
[[96, 160], ['k5_e6', 'k7_e6', 'k7_e3', 'k5_e3'], 2]|
[[160, 320], ['k3_e6'], 1]
```

Fig. 4. Obtained architecture.

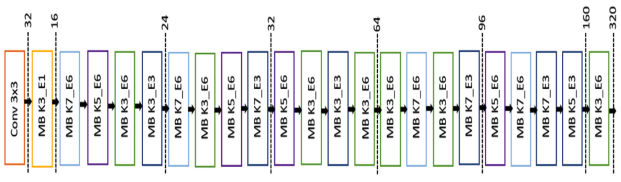

Fig. 5. Proposed NAS on SSDLite.

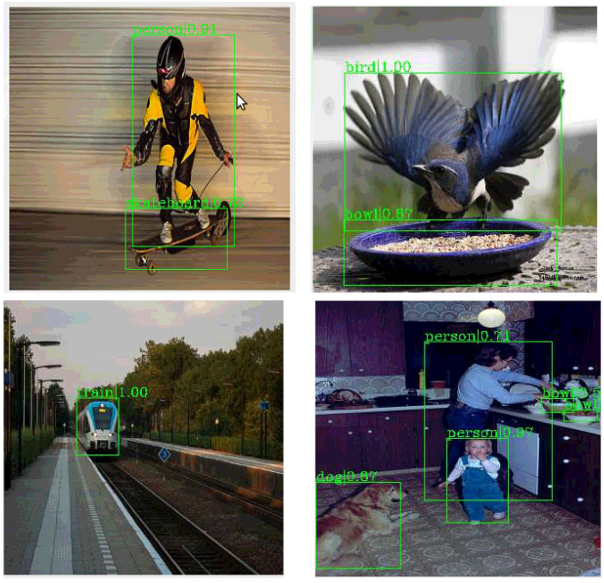

Fig. 6. Detection on MS-COCO validation dataset.

the SSDLite framework, where MB denotes the inverted residual block proposed in MobileNetV2.

Fig. 6 shows the visualization of object detection results on the MS-COCO validation dataset.

4. CONCLUSIONS

In this study, the existing popular NAS methods are reviewed, and an efficient NAS method for object detection tasks is proposed. However, the method is limited, and is not explored in different frameworks. Our main contribution is taking complete advantage of the models pre-trained on ImageNet. The manually designed network is adopted as the base model. The strategy of remapping the parameters fully utilizes the base model, which, in turn, accelerates the overall network efficiently. With this method, researchers can quickly adapt other manually designed networks to various other tasks. In future, it could be more beneficial to perform other computer vision tasks by introducing various adaptation techniques.

ACKNOWLEDGMENT

This research was supported by the Ministry of Science and ICT (MSIT), Korea, under the Information Technology Research Center (ITRC) support program (IITP-2021-0-02052) supervised by the Institute for Information & Communications Technology Planning & Evaluation (IITP).

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (2020-0-01080, Variable-precision deep learning processor technology for high-speed multiple object tracking).

REFERENCES

[1] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey", *J. Mach. Learn. Res.,* Vol. 20, No.1, pp. 1-21, 2019.

[2] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning", *Proc. of Int. Conf. on Learn. Representations*, pp. 1611.01578(1)-1611.01578(16), Toulon, France, 2017.

[3] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean, "Efficient neural architecture search via parameter sharing", *Proc. of Int. Conf. on Mach. Learn*., pp. 4095-4104, Stockholm, Sweden, 2018.

[4] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le. "Learning transferable architectures for scalable image recognition", *Proc. of IEEE Conf. on Comput. Vis. Pattern Recognit.*, pp. 8697-8710, Salt Lake City, Utah, 2018.

[5] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le. "Regularized evolution for image classifier architecture search", *Proc. of AAAI Conf. on artif. intel.*, Hawaii, United States, pp. 4780-4789, 2019.

[6] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable architecture search", *Proc. of Int. Conf. on Learn. Representations,* pp. 1806.09055(1)- 1806.09055(13), New Orleans, United States, 2019.

[7] R. J. Wang, X. Li, and C. X. Ling, "Pelee: A real-time object detection system on mobile devices", *Proc. of Adv. Neural. Inf. Process. Syst.,* pp. 184.06882(1)- 184.06882(10), Montreal, Canada, 2018.

[8] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks", *Proc. of IEEE Conf. on Comput. Vis. Pattern Recognit.*, pp.4700-4708, Honolulu, Hawaii, 2017.

[9] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun, "Detnet: Design backbone for object detection", *Proc. of Comput. Vis. ECCV,* pp. 334-350, Munich, Germany, 2018.

[10] Y. Chen, T. Yang, X. Zhang, G. Meng, X. Xiao, and J. Sun, "DetNAS: Backbone search for object detection", *Proc. of Adv. Neural. Inf. Process. Syst.,* pp. 6642-6652, Montreal, Canada, 2019.

[11] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and F. F. Li, "Imagenet: A large-scale hierarchical image database", *Proc. IEEE Conf. on Comput. Vis. Pattern Recognit.*, pp. 248-255, Miami, Florida, 2009.

[12] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobilenetV2: Inverted residuals and linear bottlenecks", *Proc. of IEEE Conf. on Comput. Vis. Pattern Recognit.,* pp. 4510-4520, Salt Lake City, Utah, 2018.

[13] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector", *Proc. of Comput. Vis. ECCV*, pp. 21-37, Amsterdam, Netherlands, 2016.
[14] T. Lin, M. Maire, S. J. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollar, "Microsoft coco: Common objects in context", *Proc. of Comput. Vis. ECCV*, pp. 740-755, Zürich, Switzerland, 2014.