Towards to automatic machine learning

LE Van Linh

August, 2018

Abstract

In recent years, convolutional neural networks (CNNs) have been employed to many tasks of computer sciences, (i.e. images classification, object recognition, language modeling, ...) with many proposed networks: AlexNet, VGG, ... Almost the networks are manually proposed and evaluated, which is time-consuming. Because of this, we are very interested in the methods of *automating neural architecture search (NAS)*. In this study, we will provide an overview and the elements of neural architecture search. On that basis, we would like to apply the NAS to automatically propose the CNNs to predict the landmarks. By that, we can compare the results that obtained from automatic-model and manual-model (ICPRS-2018).

1 Neural Architecture Search

The successes of deep learning are remarkable by a lot of CNNs have been proposed in recent years. The networks, which have been designed from the simple to the complex, have helped the researchers solve the problems that we cannot finish before or we need a lot of time to provide the solution. However, most of CNNs are designed manually. In this period, NAS [1] is known as a subfield of automated machine learning and it is considered as a solution to provide the neural architecture automatically. The NAS includes three components:

- **Search space**: defines which architectures can be represented in principle.
- **Search strategy**: details how to explore the search space.
- **Performance estimation strategy**: refers to the process of estimating the performance of the architectures.

Fig.1 shows the relation of the components in NAS.

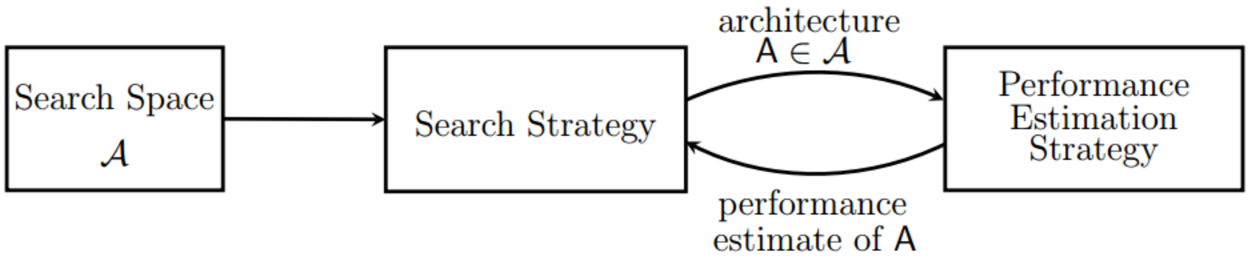

Figure 1: Illustration of Neural Architecture Search method

The **Search space** contains almost the parameters of neural network. The **Search strategy** selects a network architecture A from the search space. The selected network is passed to a **performance estimation strategy** to estimated the performances of network A and return it to the search strategy. This process will be continued until reaching the stop conditions (i.e. how many trials?). For each performance time, the search strategy tries to modify the network A to obtain the best performance on the dataset.

Search space

The search space defines which architectures can be represented in principle. For example to build a CNN, the search space is parameterized by:

- The number of layers
- The type of layers (i.e. convolutional layer, pooling layer, full-connected layer, ...)
- The hyperparameters associated with the operations (i.e. CONV kernel, CONV filters, POOL kernel, ...)

The network hyperparameters are selected from the search space. They can be one-branch [2, 3] or multi-branch [4, 5], repeated motifs architectures [6, 7, 5]. Note that the difficulty of the optimization problem is depended on the size of the search space.

Search strategy

Search strategy details the methods to explore the search space. It includes the algorithm to select a good parameters for an architecture after each step. Many different search strategies have been applied:

- Random search
- Bayesian optimization
- Evolutionary methods
- Reinforcement learning (RL) [8] has been used in recent years
- Gradient-based methods

Performance estimation strategy

Performance estimation strategy refers to the process of finding the architecture A that maximizes the performance measure, such as accuracy, and reduce the cost of these performance estimation. The simplest way is to train A on training data and to evaluate its performance on validation data. But training and evaluating A from scratch are time-consuming. So, to reduce the computation, some solutions have been proposed to get the better performance on this problem such as:

- Shorter training times:
- Training on a subset of the data
- Training with less filters per layer
- Initialize the weight of next architecture based on the weights of other architectures that have been trained before (a kind of fine-tuning)

2 Propose the solutions for landmarking networks

Currently, *most of the works on NAS are performed on images classification problem.* So, a strategy for a regression problem can be considered. Based on the knowledge about NAS, I would like to propose two strategies for detecting the landmarks on beetles.

In the first strategy, purpose is comparing the predicted landmarks which have obtained from a manual model (ICPRS-18) with the result of an automatic model. Firstly, a generic model has been designed with the same number of layers and the same type of the layers as the manual model. Then, the NAS will be applied to predict the hyperparameters for each layer in each trial during performance process. The model after each prediction will be trained and evaluated on the dataset. At the end of the process, we will compare the coordinates of the landmarks which have predicted by the manual and the automatic model.

In the second strategy, we will provide to NAS the number of layers and try to ask it the output of the network models. The models then are trained and evaluated on the dataset. Of course, at the end of the process, we also compare the results of the automatic models with the manual model.

In both of strategies, we use Reinforcement learning as a solution for the search strategy. Besides, all the proposed models will be trained from the scratch (before try with other solutions).

References

[1] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *arXiv preprint arXiv:1808.05377*, 2018.

[2] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

[3] Han Cai, Jiacheng Yang, Weinan Zhang, Song Han, and Yong Yu. Path-level network transformation for efficient architecture search. *arXiv preprint arXiv:1806.02639*, 2018.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[5] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, volume 1, page 3, 2017.

[6] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. *arXiv preprint arXiv:1707.07012*, 2(6), 2017.

[7] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[8] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction.* MIT press, 1998.