中国应用数学通报
Bulletin of Chinese Applied Mathematics
2023 年第 01 卷第 01 期
Vol. 01, No. 01, 2023

A Review of Evolutionary Deep Neural Architecture Search with Performance Predictor

QIAO Zenglin[1,2], ZHAO Xinchao[1,2,*], WU Lingyu [1,2]

(1. School of Science, Beijing University of Posts and Telecommunications, Beijing 100876, China;
2. Key Laboratory of Mathematics and Information Networks(Beijing University of Posts and Telecommunications), Ministry of Education, Beijing 100876, China)

Abstract: Automated design of deep neural networks using performance predictor has become a hot topic in current research. Neural architecture search(NAS) methods can be used to enable automatic design of neural network structures by defining different search spaces, search strategies, or optimization strategies. Evolutionary computation by many researchers as the search strategy for NAS, which is called evolutionary NAS(ENAS). However, ENAS is time-consuming in evaluating the performance of network structures, which hinders the development of ENAS. Therefore, predicting network architecture performance using performance predictor can improve ENAS search speed and save computational resources. This paper summarizes several ENAS methods that utilize performance predictor, and discusses an outlook on search space, search strategies, and the future directions of ENAS assisted by performance predictor.

Keywords: Neural architecture search; evolutionary algorithm; deep learning; convolutional neural network

DOI: 10.48014/bcam.20230822002

Citation: QIAO Zenglin, ZHAO Xinchao, WU Lingyu. A review of evolutionary deep neural architecture search with performance predictor[J]. Bulletin of Chinese Applied Mathematics, 2023, 1(1): 1-9.

1 Introduction

In recent years, with the continuous development of the field of artificial intelligence, deep neural networks, as the main technology of deep learning, have become a popular research area. Deep neural networks have been widely used in fields such as natural language processing[1], image processing[2], and data mining[3]. Especially in the field of image processing, some famous deep neural networks have achieved good classification and segmentation results in applications such as image classification[4] and image segmentation[5]. For example, AlexNet[4], GoogLeNet[6], and ResNet[2].

However, these famous neural network architectures are all designed manually by experts who have rich experience and knowledge in relevant field. As deep neural networks and other disciplines

* 通讯作者 **Corresponding author**: ZHAO Xinchao, zhaoxc@bupt.edu.cn

收稿日期: 2023-08-22; **录用日期**: 2023-09-15; **发表日期**: 2023-12-28

continue to cross-develop, some users from other interdisciplinary areas hope to design efficient deep neural networks for their research areas, but they lack knowledge and experience in the field of deep learning and cannot design networks manually. This has led to a surge of interest in automatic design of deep neural network architectures. On the other hand, automatic design of deep neural network architectures can also promote the development of deep learning in other disciplines. ExistingNeural architecture search (NAS) methods can be divided into two categories depending on whether domain knowledge is needed when using them. The first category is "automatic + manual tuning" neural network architecture design, which still requires manual tuning based on professional knowledge of designing neural network architectures. This category includes genetic CNN[7], hierarchical evolution[8], efficient architecture search (EAS)[9], and neural architecture search (NASNet)[10]. The second category is "automatic" CNN architecture design, which does not require any manual tuning by users when using it. This category includes large-scale evolution[11], cartesian genetic programming (CGP-CNN)[12] and meta-modelling method (MetaQNN)[13]. Clearly, automatic neural network architecture search methods are more in line with user needs.

According to the adopted techniques, NAS can also be divided into evolutionary neural architecture search (ENAS), reinforcement learning NAS (RNAS), gradient-based NAS, and so on. However, experimental evidence has shown that RNAS usually require more computational resources than ENAS[14]. ENAS conforms to the basic process of evolutionary algorithms and is more suitable for architecture search of deep neural networks. Evolutionary algorithms[15] are a class of population-based metaheuristic optimization methods inspired by biological evolution. Typical evolutionary algorithms include genetic programming[16], evolutionary strategies[17], among which genetic algorithms are the most popular due to their theoretical foundation[18] and good performance in solving different optimization problems[19-22]. Genetic algorithms generate high-quality optimal solutions by using bio-inspired operators, namely mutation, crossover, and selection[32]. The basic idea of evolutionary algorithms is to represent the solution to a problem as an individual and iteratively optimize the solution by performing selection, crossover, and mutation operations on individuals. Specifically, the basic steps of evolutionary algorithms include initializing the population, selection operation, crossover operation, mutation operation, and environmental selection. Evolutionary algorithms have the advantages of strong global search ability, low requirements for problem specifications, and good parallelism.

ENAS is a neural architecture search method based on evolutionary algorithms. It iteratively evolves the structure and parameters of neural networks to find the optimal neural network architecture. Some ENAS have achieved good results, such as genetic CNN[7] and CNN-GA[23]. However, the difficulty with ENAS lies in the fact that evaluating neural architectures is very time-consuming. For example, A fully machine designed neural network[24] requires parallel operation on Tesla K40 GPUs for nearly a month. This high computational cost sacrifices the savings achieved by manual architecture design. Therefore, reducing the computational cost is one of the key goals of ENAS. Some researchers use the method of reducing the number of epochs to save running costs, but this method can lead to underfitting and inaccurate results. Recently, using performance predictor to predict the performance of neural architectures for ENAS has become a popular research direction. In the process of ENAS, using performance predictors to predict the performance of architectures can reduce the time required to evaluate architectures.

Therefore,this paper summarizes the ENAS with performance predictor of recent years,summarizes them, and looks forward to future development directions.

2 Neural Architecture Search

Neural architecture search(NAS)is generally defined as a global optimization problem. In the search space S ,NAS can be defined as Eq. (1).

$$S = arg\min_{s \in S} f(s) \tag{1}$$

where f is the performance measure of the neural architecture s , NAS always requires some search strategy, as it is impossible to evaluate the performance of all architectures through the training-validation process under limited search cost.

NAS can be divided into three parts, namely search space, search strategy, and performance evaluation, as shown in Fig. 1. The search space defines the possible structures and components of the neural network. The design of the search space affects the efficiency and results of the search. Generally, the search space should include common neural network structures and components, while also allowing a certain degree of freedom and innovation, so that the search algorithm can find better solutions. The search strategy defines how to conduct the search in the search space. Search strategies typically include heuristic search, genetic algorithms, reinforcement learning, and other methods. Different search strategies have their own advantages and disadvantages, and choosing the appropriate search strategy can improve the efficiency and quality of the search results.

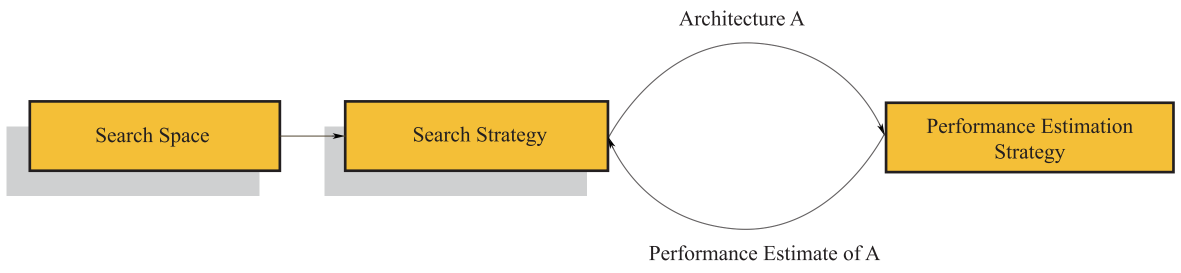

Fig. 1 Neural Architecture Search

3 Evolutionary neural architecture search (ENAS)

Evolutionary algorithms are a class of optimization algorithms that draw inspiration from the natural process of selection and evolution. These algorithms are utilized in a variety of fields, including engineering, finance, and biology, to solve complex optimization problems. At the core of evolutionary algorithms is the maintenance of a population of candidate solutions, with genetic operators such as mutation and crossover applied iteratively to generate new solutions. The fitness of these solutions is then evaluated according to a fitness function, with the most promising individuals selected to become parents of the next generation. This process continues for several generations until a satisfactory solution is achieved. One of the primary advantages of evolutionary algorithms is their ability to effectively explore large search spaces and avoid being trapped in local optima. This makes them particularly suitable for complex optimization problems with non-linear and non-convex objective functions. There are several types of evolutionary algorithms, including genetic algorithms, evolutionary strategies, and genetic programming, each with unique strengths and weaknesses. The selection of the appropriate algorithm depends on the specific problem being addressed. In recent years, hybrid algorithms combining evolutionary algorithms with other optimization techniques such as swarm intelligence and machine learning have emerged, resulting in more powerful and efficient algorithms. These hybrid algorithms have been applied to diverse applications such as image and signal processing, data mining, and robotics. Overall, evolutionary algorithms are versatile and powerful optimization algorithms that have demonstrated their effectiveness in solving a broad range of complex problems. Fig. 2 shows the standard optimization process of genetic algorithm.

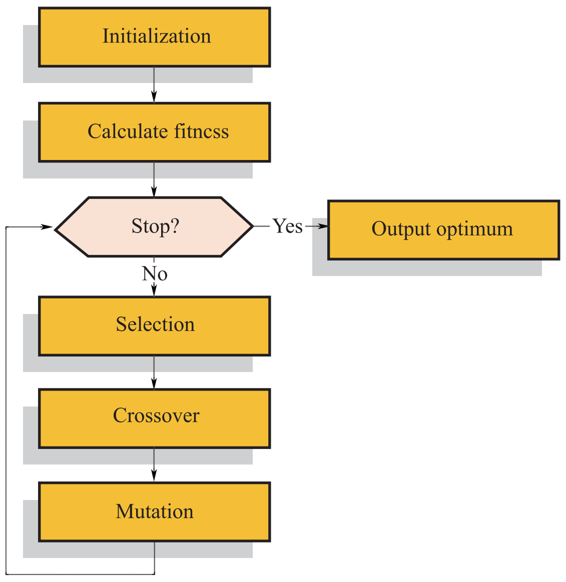

Fig. 2 Genetic algorithm

Evolutionary algorithms have been shown to possess global search capabilities that make them an efficient search strategy for NAS. Several methods have been proposed, including EvoNAS-Rep[25], Evolutionary NAS[26], and IPPSO[27]. EvoNAS-Rep is a recent ENAS-based method that introduces a new RepVGG node. It proposes a new encoding strategy that maps fixed-length coded individuals to variable-depth DL structures using RepVGG nodes. GA is used to search for the optimal individual and its corresponding DL model. An iterative training process is designed to train the deep learning model and evolutionary genetic algorithm. EvoNAS-Rep is validated on CIFAR 10 and CIFAR 100 datasets, achieving high accuracy of 96.35% and 79.82%, respectively, with less than 0.2 GPU days. Evolutionary NAS proposes an effective evolutionary method that uses a customized crossover operator to inherit parent structures'information, improving convergence speed. IPPSO utilizes particle swarm optimization (PSO) to automatically search for CNN's optimal architecture without human intervention. Three improvements are made on the traditional particle swarm algorithm, including proposing a new encoding strategy inspired by computer networks, designing a disabled layer to enable the method to learn variable-length CNN architecture, and randomly selecting a portion of the dataset for evaluation to increase evaluation speed. IPPSO is the first work that uses particle swarm algorithms to automatically evolve CNN architectures. Table 1 summarizes the evolutionary algorithms used by the three algorithms and the types of basic units in the search space.

Table 1 Summary of three different ENAS

Algorithms	Evolutionary algorithm	Basic Unit
EvoNAS-Rep	Genetic algorithm	Block
Evolutionary NAS	Genetic algorithm	Cell
IPPSO	Particles warm optimization	Layer

4 Evolutionary neural architecture search with performance predictor

ENAS generally uses gradient back propagation to evaluate the performance of the searched architectures when searching for architecture evaluation, which makes ENAS very computer resource-intensive and time-consuming. Therefore, the researchers proposed using a performance predictor to evaluate deep network architectures. Instead of using gradient backpropagation to obtain the evaluation value for each searched network architecture by the evolutionary algorithm, the performance predictor performs a fast evaluation. Some researchers have proposed efficient ENAS using different performance predictor. Sun et al.[28] proposed using an end-to-end random forest-based performance predictor (E2EPP) as a performance predictor for the evaluation of network structures. Fig. 3 shows the framework of the proposed performance predictor and the associated evolutionary deep learning algorithm(EDL). The random forest-based performance predictor was learned from a few different CNN architectures that have been trained for accuracy in the classification task. Compared to the CNN training process, random forest construction and prediction are less computationally expensive and can be repeated for the entire evolutionary optimization, thus alleviating the higher computational cost in EDL. Fig. 3 consists of three modules: data acquisition, E2EPP, and EDL. The proposed E2EPP performance predictor is part of the EDL. First, EDL is used to collect data for training the random forest agent, a process that does not use E2EPP. Samples are composed of CNN architectures and classification accuracies obtained by training using gradient backpropagation. Secondly, these CNN architectures are encoded into discrete codes for training the random forest model. At each cycle of the EDL, the CNN architectures are used as input to the random forest, which is then used to predict the performance of the CNN architectures without using the time-consuming gradient backpropagation. When the EDL reaches the termination condition, the CNN architecture with the best prediction performance is output. Gradient backpropagation training of the CNN is not required for the optimization process.

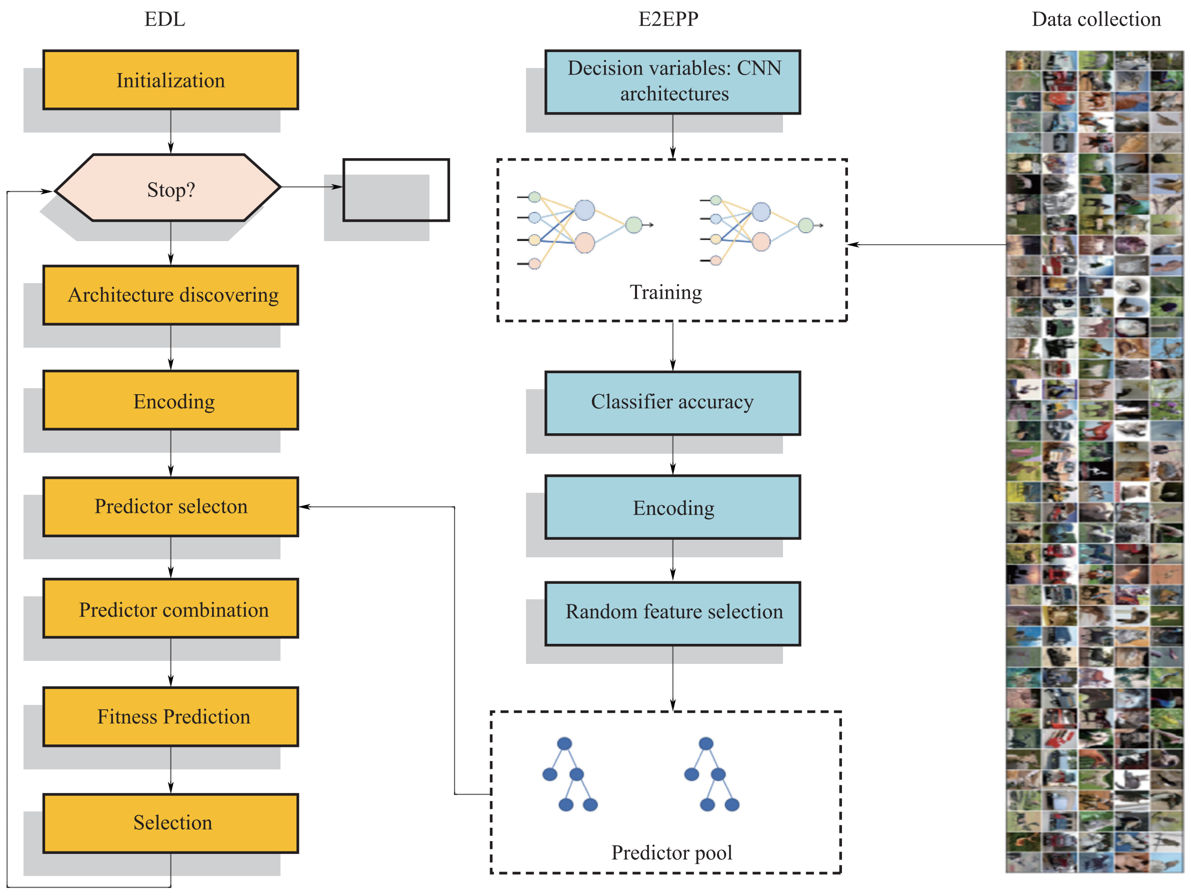

Fig. 3 Main frame of the E2EPP associated EDL

The NSGANetV2 algorithm proposed by Lu et al.[29] utilizes multiple performance predictor to assist in a multi-objective evolutionary neural architecture search. Four agent models are used in the approach, including Classification and Regression Trees (CART)[28], Radial Basis Function

(RBF)[30], Multi-Layer Perceptron (MLP)[31], and Gaussian Process (GP)[32]. The authors note that no single agent model outperforms the others for multiple targets, hence the use of a selection mechanism called adaptive switching (AS). In each iteration of the algorithm, AS constructs four different types of agent models, and then selects the model with the best predictive performance using cross-validation. The goal of NSGANetV2 is to optimize both prediction accuracy and the number of parameters within the network structure, with a focus on finding efficient neural network architectures. In testing on the CIFAR-10 dataset, NSGANetV2 can achieve a 98.4% accuracy while keeping the number of parameters at 468 M. On the popular ImageNet dataset, the algorithm achieves a 75.9% accuracy while maintaining the number of parameters at 225 M. Overall, the use of multiple agent models and AS selection mechanism appears to be an effective method for seeking optimal neural network architectures.

EMONAS-Net[33] is a state-of-the-art neural structure search framework that uses agent assistance to efficiently search for optimal neural network architectures. It achieves this using an inexpensive random forest agent function that ranks the performance of candidate architectures during the evolutionary algorithm search. This helps to significantly reduce the overall search time compared to other search frameworks. In addition, EMONAS-Net makes use of a proxy model based on a random forest, which has a very small number of hyperparameters and does not require a large amount of training data during training. This makes it particularly well-suited for use with various datasets. EMONAS-Net is unique in that it considers both the micro-and macro-structures of the architecture when designing the search space. This reduces the need for manual intervention and minimizes the size of the architecture, resulting in more efficient and effective searches for optimal neural network architectures.

Table 2 Summary of five different ENAS with performance predictors

Algorithms	Agents	Basic Unit	Fields of application
E2EPP	Random forest	CNN layer	Image classification
NSGANetV2	CART, RBF, MLP and GP	Block	Image classification
PRE-NAS	Random forest	Cell	Image classification
NPENAS	Bayesian optimization	Cell	Image classification
EMONAS-Net	Random Forest	Block	3D Image segmentation

NPENAS[34] is a recently proposed evolutionary neural architecture search algorithms that use performance predictors to rank candidate architectures. NPENAS contains two types of performance predictors, the NPEANS-BO and NPEANS-NP, which are used to rank candidate architectures. The NPEANS-BO is an acquisition function, which is defined from a graph-based uncertainty estimation network, while the NPEANS-NP is a graph-based performance predictor. These predictors are used to rank candidate architectures.

Peng et al.[35] proposed PRE-NAS, an evolutionary neural architecture search algorithm that uses an performance predictor. The performance predictor can be trained with a limited number of samples through a selection strategy. PRE-NAS achieved a test error rate of 2.40% on CIFAR-10 and a top-1 test error rate of 24% on ImageNet. Fig. 4. shows that, in the search space, PRE-NAS builds a neural network by stacking several repeating cell modules on top of each other. The networks in the dashed line on the right are called cell networks. Each cell network consists of several predefined operations. Each complete neural network consists of a stack of cell networks. The number of cell networks in the network, and the depth of the network are obtained by a search algorithm.

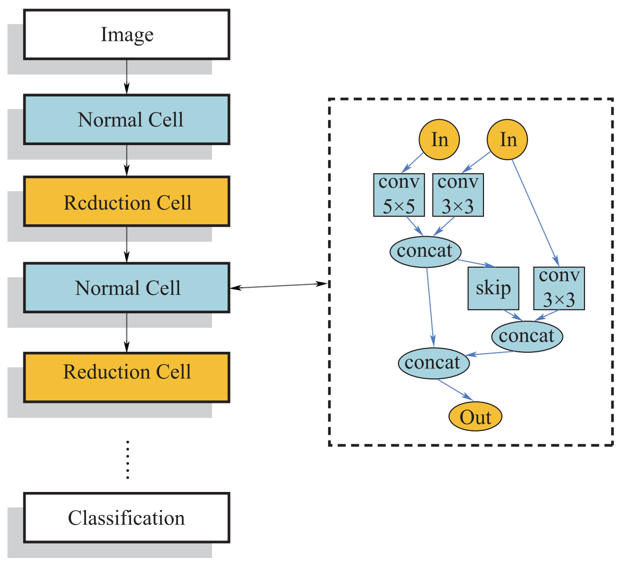

Fig. 4 Neural network based on cell construction in PRE-NAS

The expectation in designing a proxy predictor is that the predictor can be trained on a limited amount of data and contain a small number of parameters, as training a larger network would slow down the search. PRE-NAS experimentally evaluated several common agent networks, including multi-layer perceptron (MLP), linear regressor, bayesian ridge regressor, support vector regressor, graph convolutional network (GCN), kernel ridge, and random forest regressor. The results showed that random forest was the best-performing predictor. The experimental results are shown in Table 3.

Table 3 The results of training 7 regression variables with 100 samples for multiple proxy predictors in PRE-NAS

Predictors	Spearman coefficient	Training time(ms)	Inference time(ms)
Random Forest	0.65±0.07	690±100	50±7
Support Vector	0.38±0.11	0.7±0.1	0.3±0.05
GCN	0.45±0.13	41000±728	13±5
MLP	0.07±0.11	480±50	3±0.5
Linear Regressor	0.05±0.11	0.6±0.2	0.1±0.07
Kernel Ridge	0.07±0.10	0.8±0.9	0.3±0.2
Bayesian Ridge	0.05±0.11	1.5±0.9	0.1±0.04

5 Conclusions

This paper provides a summary of evolutionary neural architecture search approach that utilizes performance predictors, and describes the basic forms and components of neural architecture search. Automatic search of network architectures enables users without deep learning knowledge to design networks that meet their needs, and NAS with performance predictors can accelerate this process and reduce the cost of use. However, designing the performance predictor requires meeting several requirements. Firstly, the performance predictor should not contain too many parameters and weights and should not be too large or deep. A performance predictor that is too large can slow down the training process, which is contrary to the original design intention. Secondly, the performance predictor should be trained on a small number of samples to achieve good performance. Due to the high cost of evaluating neural architectures, there are very few data samples available for training the predictor. Therefore, the focus of the next step is to design efficient performance predictors that are lightweight and do not require many samples for training.

Conflict of interest: The authors declare no conflict of interest.

References

[1] D. W. Otter, J. R. Medina, and J. K. Kalita. A survey of the usages of deep learning for natural language processing[J]. IEEE transactions on neural networks and learning systems, 2020, 32(2): 604-624. doi: 10.1109/TNNLS.2020.2979670

[2] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition[J]. Proceedings of the IEEE conference on computer vision and pattern recognition, 2016: 770-778.

[3] G. Nguyen et al. Machine learning and deep learning

frameworks and libraries for large-scale data mining: a survey [J]. Artificial Intelligence Review, 2019, 52: 77-124.
doi: https://doi.org/10.1007/s10462-018-09679-z

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks [J]. Communications of the ACM, 2017, 60(6): 84-90.
doi: 10.1145/3065386

[5] C. Liang-Chieh, G. Papandreou, I. Kokkinos, et al. Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs [C]. International Conference on Learning Representations, 2015.
doi: https://doi.org/10.48550/arXiv.1412.7062

[6] W. L. Christian Szegedy, Yangqing Jia, Pierre Sermanet, et al. Going Deeper With Convolutions [J]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015: 1-9.
doi: 10.1109/CVPR.2015.7298594

[7] L. Xie and A. Yuille. Genetic cnn [J]. Proceedings of the IEEE international conference on computer vision, 2017: 1379-1388.
doi: 10.1109/ICCV.2017.154

[8] H. Liu, K. Simonyan, O. Vinyals, et al. Hierarchical Representations for Efficient Architecture Search [C]. International Conference on Learning Representations, 2018.
doi: https://doi.org/10.48550/arXiv.1711.00436

[9] H. Cai, T. Chen, W. Zhang, et al. Efficient architecture search by network transformation [J]. Proceedings of the AAAI Conference on Artificial Intelligence, 2018, vol. 32, no. 1.
doi: https://doi.org/10.1609/aaai.v32i1.11709

[10] B. Zoph, V. Vasudevan, J. Shlens, et al. Learning transferable architectures for scalable image recognition [J]. Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 8697-8710.
doi: https://doi.org/10.48550/arXiv.1707.07012

[11] E. Real et al. Large-scale evolution of image classifiers [C]. International Conference on Machine Learning, 2017: PMLR, pp. 2902-2911.

[12] M. Suganuma, S. Shirakawa, and T. Nagao. A genetic programming approach to designing convolutional neural network architectures [C]. Proceedings of the genetic and evolutionary computation conference, 2017, pp. 497-504.
doi: 10.1145/3071178.3071229

[13] B. Baker, O. Gupta, N. Naik, et al. Designing Neural Network Architectures using Reinforcement Learning [C]. International Conference on Learning Representations, 2016.
doi: 10.48550/arXiv.1611.02167

[14] Y. Sun, B. Xue, M. Zhang, et al. Evolving deep convolutional neural networks for image classification [J]. IEEE Transactions on Evolutionary Computation, 2019, 24(2): 394-407.
doi: 10.1109/TEVC.2019.2916183

[15] T. Back. Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms [M]. Oxford university press, 1996.
doi: https://doi.org/10.1108/k.1998.27.8.979.4

[16] W. Banzhaf, P. Nordin, R. E. Keller, et al. Francone, Genetic programming: an introduction: on the automatic evolution of computer programs and its applications [M]. Morgan Kaufmann Publishers Inc., 1998.
doi: dl.acm.org/doi/abs/10.5555/323917

[17] H.-G. Beyer and H.-P. Schwefel. Evolution strategies-a comprehensive introduction [J]. Natural computing, 2002, 1: 3-52.
doi: https://doi.org/10.1023/A:1015059928466

[18] L. M. Schmitt. Theory of genetic algorithms [J]. Theoretical Computer Science, 2001, 259(1-2): 1-61.
doi: https://doi.org/10.1016/S0304-3975(00)00406-0

[19] K. Deb, A. Pratap, S. Agarwal, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II [J]. IEEE transactions on evolutionary computation, 2002, 6(2): 182-197.
doi: 10.1109/4235.996017

[20] M. Jiang, Z. Huang, L. Qiu, et al. Transfer learning-based dynamic multiobjective optimization algorithms [J]. IEEE Transactions on Evolutionary Computation, 2017, 22(4): 501-514.
doi: 10.1109/TEVC.2017.2771451

[21] Z. Qiao et al. Gaussian bare-bones gradient-based optimization: Towards mitigating the performance concerns [J]. International Journal of Intelligent Systems, 2022, 37(6): 3193-3254.
doi: https://doi.org/10.1002/int.22658

[22] C. Yinnan, Y. Lingjuan, L. Rui, et al. A Multi-period Constrained Multi-objective Evolutionary Algorithm with Orthogonal Learning for Solving the Complex Carbon Neutral Stock Portfolio Optimization Model

[J]. Journal of Systems Science and Complexity, p. 0. doi: https://doi.org/10.1007/s11424-023-2406-3

[23] Y. Sun, B. Xue, M. Zhang, et al. Automatically designing CNN architectures using the genetic algorithm for image classification[J]. IEEE transactions on cybernetics, 2020, 50(9): 3840-3854. doi: 10.1109/TCYB.2020.2983860

[24] B. Zoph, and Q. Le. Neural Architecture Search with Reinforcement Learning[C]. International Conference on Learning Representations, 2016. doi: https://doi.org/10.48550/arXiv.1611.01578

[25] L. Wen, L. Gao, X. Li, et al. A new genetic algorithm based evolutionary neural architecture search for image classification[J]. Swarm and Evolutionary Computation, 2022, 75: 101191. doi: https://doi.org/10.1016/j.swevo.2022.101191

[26] C. He, H. Tan, S. Huang, et al. Efficient evolutionary neural architecture search by modular inheritable crossover[J]. Swarm and Evolutionary Computation, 2021, 64: 100894. doi: https://doi.org/10.1016/j.swevo.2021.100894

[27] B. Wang, Y. Sun, B. Xue, et al. Evolving deep convolutional neural networks by variable-length particle swarm optimization for image classification[C]. 2018 IEEE Congress on Evolutionary Computation (CEC), 2018: IEEE, pp. 1-8. doi: 10.1109/CEC.2018.8477735.

[28] Y. Sun, H. Wang, B. Xue, et al. Surrogate-assisted evolutionary deep learning using an end-to-end random forest-based performance predictor[J]. IEEE Transactions on Evolutionary Computation, 2019, 24(2): 350-364. doi: 10.1109/TEVC.2019.2924461

[29] Z. Lu, K. Deb, E. Goodman, et al. Nsganetv2: Evolutionary multi-objective surrogate-assisted neural architecture search[C]. Computer Vision-ECCV 2020: 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part I 16, 2020: Springer, pp. 35-51. doi: https://doi.org/10.1007/978-3-030-58452-8_3

[30] B. Baker, O. Gupta, R. Raskar, et al. Accelerating Neural Architecture Search using Performance Prediction, 2018. doi: https://doi.org/10.48550/arXiv.1705.10823

[31] C. Liu et al. Progressive neural architecture search[C]. Proceedings of the European conference on computer vision (ECCV), 2018, pp. 19-34. doi: https://doi.org/10.1007/978-3-030-01246-5_2

[32] X. Dai et al. Chamnet: Towards efficient network design through platform-aware model adaptation[C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 11398-11407. doi: 10.1109/CVPR.2019.01166

[33] M. B. Calisto, and S. K. Lai-Yuen. EMONAS-Net: Efficient multiobjective neural architecture search using surrogate-assisted evolutionary algorithm for 3D medical image segmentation[J]. Artificial Intelligence in Medicine, 2021, 119: 102154. doi: https://doi.org/10.1016/j.artmed.2021.102154

[34] C. Wei, C. Niu, Y. Tang, et al. Npenas: Neural predictor guided evolution for neural architecture search[J]. IEEE Transactions on Neural Networks and Learning Systems, 2022. doi: 10.1109/TNNLS.2022.3151160

[35] Y. Peng, A. Song, V. Ciesielski, et al. PRE-NAS: Evolutionary Neural Architecture Search with Predictor[J]. IEEE Transactions on Evolutionary Computation, 2022. doi: 10.1109/TEVC.2022.3227562