A Fast Evolutionary Knowledge Transfer Search for Multiscale Deep Neural Architecture

Ruohan Zhang, *Student Member, IEEE*, Licheng Jiao, *Fellow, IEEE*, Dan Wang, *Student Member, IEEE*, Fang Liu, *Senior Member, IEEE*, Xu Liu, *Member, IEEE*, and Shuyuan Yang, *Senior Member, IEEE*

***Abstract*— The emergence of neural architecture search (NAS) algorithms has removed the constraints on manually designed neural network architectures, so that neural network development no longer requires extensive professional knowledge, trial and error. However, the extremely high computational cost limits the development of NAS algorithms. In this article, in order to reduce computational costs and to improve the efficiency and effectiveness of evolutionary NAS (ENAS) is investigated. In this article, we present a fast ENAS framework for multiscale convolutional networks based on evolutionary knowledge transfer search (EKTS). This framework is novel, in that it combines global optimization methods with local optimization methods for search, and searches a multiscale network architecture. In this article, evolutionary computation is used as a global optimization algorithm with high robustness and wide applicability for searching neural architectures. At the same time, for fast search, we combine knowledge transfer and local fast learning to improve the search speed. In addition, we explore a multiscale gray-box structure. This gray box structure combines the Bandelet transform with convolution to improve network approximation, learning, and generalization. Finally, we compare the architectures with more than 40 different neural architectures, and the results confirmed its effectiveness.**

***Index Terms*— Bandelet, evolutionary optimization, knowledge transfer, neural architecture search (NAS).**

I. Introduction

NEURAL networks have been widely used in recent years. The powerful feature representation capabilities of neural networks have enabled major breakthroughs and progress in many fields. The emergence of neural networks has removed the constraints imposed by the use of manually designed features in traditional methods. Neural networks can not only automatically learn useful features through efficient network architectures but also greatly improve the performance on tasks such as image and speech recognition.

Various efficient and functional neural networks [5], including residual networks (ResNets) [6], inception networks [7], and dense networks (DenseNets) [8], have been proposed. The performance of the networks has gradually improved. Experiments have shown that the network architecture is crucial to the representation of data and the performance of the network. A more effective network framework leads to better network performance. The architecture of a neural network largely determines its performance. A good neural network architecture can not only improve performance but also reduce computational costs. However, these excellent network architectures are usually artificially designed based on amassed professional knowledge and trial. The development of an excellent network architecture requires intense and time-consuming efforts, limiting the development of neural networks. To reduce the amount of work required to design the architecture, the researchers proposed the neural architecture search (NAS) [4] technology. Recently, NAS technology [4], [9] has been developed and is used in a wide range of applications [10], [11]. The NAS algorithm can avoid manual repetitive exploration and design, which enables a high-performance network architecture to be designed in an automated manner with less manual intervention. The performance of an NAS-designed network is comparable to the performance of networks with the most advanced artificial neural network architectures. Since the NAS algorithm can specifically design methods that generate network architectures according to the needs of tasks, it effectively reduces the number of calculations and the implementation cost of a neural network. The advantages of the neural network architecture search algorithm have motivated widespread research interest in this topic.

The NAS algorithm can be regarded as a target optimization problem. The performance of the network architecture is evaluated by the accuracy of the network architecture in the dataset. Generally, NAS uses a search strategy to gradually search the neural network architecture in a given search space. By evaluating the performance of the network architecture on the test dataset, a certain selection strategy is used to search for

Manuscript received 5 February 2021; revised 6 June 2022, 31 January 2023, and 22 May 2023; accepted 4 August 2023. Date of publication 23 August 2023; date of current version 3 December 2024. This work was supported in part by the Key Scientific Technological Innovation Research Project by the Ministry of Education; in part by the State Key Program and the Foundation for Innovative Research Groups of the National Natural Science Foundation of China under Grant 61836009; in part by the National Natural Science Foundation of China under Grant U22B2054, Grant 62076192, Grant 62006177, Grant 61902298, Grant 61573267, Grant 61906150, and Grant 62276199; in part by the 111 Project; in part by the Program for Cheung Kong Scholars and Innovative Research Team in University under Grant IRT 15R53; in part by the Science and Technology (ST) Innovation Project from the Chinese Ministry of Education; in part by the Key Research and Development Program in Shaanxi Province of China under Grant 2019ZDLGY03-06; in part by the National Science Basic Research Plan in Shaanxi Province of China under Grant 2022JQ-607; in part by the China Postdoctoral Fund under Grant 2022T150506; and in part by the Scientific Research Project of the Education Department in Shaanxi Province of China under Grant 20JY023. *(Corresponding author: Licheng Jiao.)*

The authors are with the Key Laboratory of Intelligent Perception and Image Understanding, Ministry of Education, School of Artificial Intelligence, Xidian University, Xi'an, Shaanxi 710071, China (e-mail: ruohan950427@gmail.com; lchjiao@mail.xidian.edu.cn).

Digital Object Identifier 10.1109/TNNLS.2023.3304291

the optimal network architecture [4]. NAS can autonomously search for network architectures. However, it often incurs a high computational cost. For example, NAS with reinforcement learning (RL) [12] requires 800 GPUs to run for 22 400 GPU days on CIFAR10; 450 GPUs for three to four days are required for learning transferable NAS (NASNet-A) [13]. Obviously, these search methods have high computational costs, which greatly limits the development of NAS.

Therefore, researchers have also proposed optimized algorithms for solving the problem caused by the massive number of calculations required for NAS. By optimizing the NAS algorithm, the number of calculations during the search process is reduced, and the search time is shortened. For example, the evolutionary NAS (ENAS) method via parameter sharing [14] uses the weight sharing method to accelerate the verification process; the differentiable architecture search (DARTS) method [15] uses a differentiable method to search, and by rethinking the value of network pruning, structured pruning and fine-tuning are used to reduce the computational cost, while echo state networks [16] use learning rule adaptation to solve the neural network training problem.

To improve the efficiency and effectiveness of the NAS approach to search, this article presents a novel and fast general approach to neural network architecture search. The motivation for the approach proposed in this article is twofold. First, neural network architecture search faces high computational costs, which we expect to alleviate for evolutionary search networks. Using the method in this article reduces the amount of computation and time required for search. Low-cost search will make the task of neural network architecture search more popular and advance the development of architecture search. Second, this article explores multiscale network architectures. This is a new type of architecture search that connects the Bandelet transform with convolution. In recent years, the combination of convolution and multiscale geometric analysis has received a lot of attention from researchers [2], [3], [17]. Such combinations can enhance model interpretability. It allows NAS to change from exploring black-box network structures to exploring gray-box structures. Our search method is based on ENAS optimization while organically integrating local knowledge transfer, equivalent to global Darwinian and local Lamarckian approaches. Specifically, the global Darwinian approach is the evolutionary search computation, and the local Lamarckian approach is the knowledge transfer optimization network. By organically combining global and local optimization, we provide an efficient framework for the fast search of multiscale network structures.

The innovations of this article can be described as follows. Our algorithm introduces knowledge transfer strategies and multiscale geometric analysis. First, in the search strategy section, the computational cost of search is effectively reduced. In the process of reducing the search cost, we use knowledge transfer to reduce the amount of training of the model. The whole process is a combination of local and global optimization. The multiobjective evolutionary optimization is used for global optimization of network architecture, and the knowledge transfer is used for local optimization of the network under global evolution. Specifically, the optimization architecture is searched in the full search space, and the architecture is optimized locally. Using such an approach reduces the search time for the entire process. A more similar approach to ours is the weight-sharing method. In contrast to the weight-sharing approach, our knowledge transfer approach takes into account the interactions between parameters in a neural network architecture. Prior knowledge from the same task and similar architectures is transferred to the new architecture. It also uses local training to fuse the old and new knowledge, so that the parameters are better matched to the architecture. This reduces the number of parameters to be trained for the new architecture, saving training time and ensuring the accuracy of the ranking. Second, we propose to search for a new multiscale network that introduces the Bandelet transform into the convolutional network. So, we search for a new kind of gray-box structured network and design the corresponding search space. Multiscale geometric filter banks can prevent the network from falling into local optima prematurely and enhance the interpretability and controllability of the network. It also enhances the robustness of the network from multiscale and anisotropic perspectives and improves the network performance [2], [17]. The Bandelet transform integrates multiscale features into the convolutional neural network (CNN) architecture and enhances the network feature learning capability. Multiscale features improve the approximation ability of the network and facilitate saddle point escape in network optimization. The searched network combines Bandelet transform with CNN to explore how multiscale and multidirectional wavelet textures are connected to convolution. In addition, the results of our search architectures contain Bandelet transform connections, demonstrating that the combination of convolution and multiscale transformation is indeed effective.

The main contributions of this article are summarized as follows.

1) The global evolution and local optimization are combined to propose a novel search framework, which is based on evolutionary learning and knowledge transfer.
2) A new end-to-end multiscale architecture that introduces the Bandelet transform into the convolutional network was searched. The novel search space is designed, which is conducive to getting multiscale networks.
3) A fast local learning based on knowledge transfer strategy is proposed, reducing redundant computations and greatly speeding up the global search.
4) Using a multiobjective evolutionary algorithm to increase the diversity of the search process, a new multiscale network architecture incorporating the Bandelet transform was searched for, which performed well on the classified dataset.

The rest of this article is organized as follows. Section II introduces the preliminaries of this work, including ENAS and the Bandelet transform. Section III describes the details of the proposed algorithm. The experimental settings and results are presented in Section IV, which compares our results with 42 popular human-designed architectures and state-of-the-art NAS methods. Section V concludes the article.

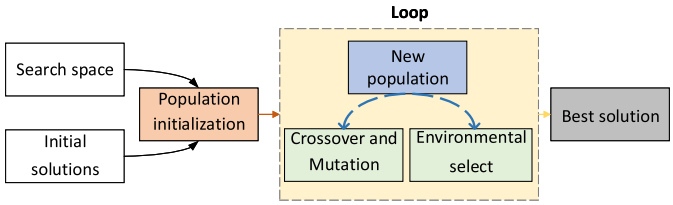

Fig. 1. Generic evolutionary search framework.

II. Preliminaries

In this section, the basic background of ENAS and Bandelet transform are reviewed, and we briefly introduce their calculation processes.

A. Evolutionary NAS

Essentially, NAS can be regarded as a process of finding the optimal architecture in a certain search space. NAS generally implements the following steps. First, define the search space and predefine some network operation sets (convolution, pooling, etc.) and basic network connection structure. Second, a search strategy is set to obtain a certain number of candidate network structures. In the third step, the accuracy of these candidate networks is fed back to the search strategy, so as to adjust the search strategy to obtain a new round of candidate networks. Repeat this process. The fourth step is to set the termination conditions (number of searches, accuracy, etc.). When the termination condition is reached, the search is stopped and the network architecture corresponding to the highest accuracy is found. Finally, the optimal architecture found by the final generation is the final result of the entire search.

ENAS is based on evolutionary algorithm to search the neural network structure. Essentially, ENAS learns and selects the wiring relationships between cells, and through different wiring relationships. A large number of neural network model structures are created, from which the optimal wiring is selected. This is equivalent to "designing" a new neural network model. This process is illustrated in Fig. 1. The algorithm searches for the best individuals by simulating natural evolution and selection. In ENAS, each network architecture is regarded as an evolving individual, and the architecture is encoded, crossed, and mutated to produce a new architecture. The search for the best individual (neural architecture) is carried out in the continuous evolution process and ultimately obtains the optimal neural network structure [4].

In recent years, many researchers have investigated ENAS algorithms [18], [19]. The superiority of evolutionary algorithms in NAS was also demonstrated [20]. However, due to the massive search space of NAS, it is often necessary to train and evaluate the model to determine the performance of the model, which leads to high computational and time costs. Some ENAS algorithms require thousands of GPU days to obtain a good model structure for a small dataset, such as CIFAR [21]. Thus, further optimization of the ENAS algorithm is necessary.

In this article, we combine global evolution with local optimization. Inspired by Lamarck's idea of "acquired inheritance," we design a local optimization strategy based on knowledge transfer to improve the efficiency of global evolutionary search.

Algorithm 1 Bandelet Transform

Input: One image
Output: Bandelet transform features
1 Perform a 2-D discrete orthogonal wavelet multiscale transform;
2 Combine quadratic tree segmentation and the classification and regression tree (CART) (bottom-up fusion) algorithms to establish the optimal quadratic tree decomposition method for each subband of the multiscale image to obtain Bandelet blocks;
3 For each Bandelet block, find the optimal direction according to the Lagrange penalty function method;
4 According to the geometric flow direction of most parts of each Bandelet block, obtain a 1-D discrete signal by orthogonal projection and wavelet coefficient rearrangement and apply a 1-D discrete wavelet transform to it.

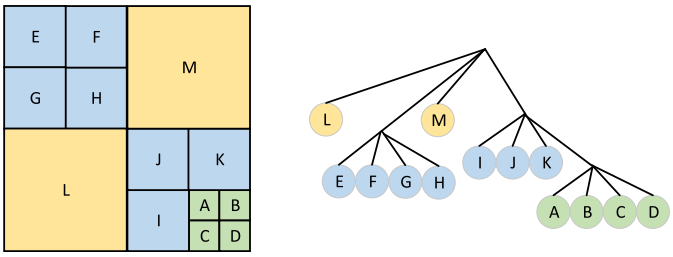

Fig. 2. Quadtree optimization framework.

B. Bandelet Transform

Multiscale geometric analysis is a signal analysis method developed in the mathematical analysis, computer vision, pattern recognition, statistical analysis, and physiology fields. It has been widely used in image processing, computer vision, and other fields. Multiscale geometric analysis methods can analyze the geometric characteristics of image edges, textures, and structures from four perspectives: multiscale, locality, directionality, and anisotropy [22]. Multiscale geometric analysis methods include the ridgelet, curvelet, Bandelet, and contourlet transformations. In this article, the Bandelet transform is used.

In multiscale geometric analysis, the Bandelet transform [23] is based on the edge-based image representation method, which can adaptively track the geometric canonical direction of an image. The computational complexity is $O(N^{3/2})$, which is almost linear.

The overall steps of the second-generation Bandelet transform algorithm are in Algorithm 1. And the details of the quadtree and selection of the best geometry are as follows.

1) Construction of the Quadtree: As shown in Fig. 2, a quadtree is a binary partition method. Each subband is equally divided into four subbands, and each subband is divided into four subbands in the segmentation of the next layer until the

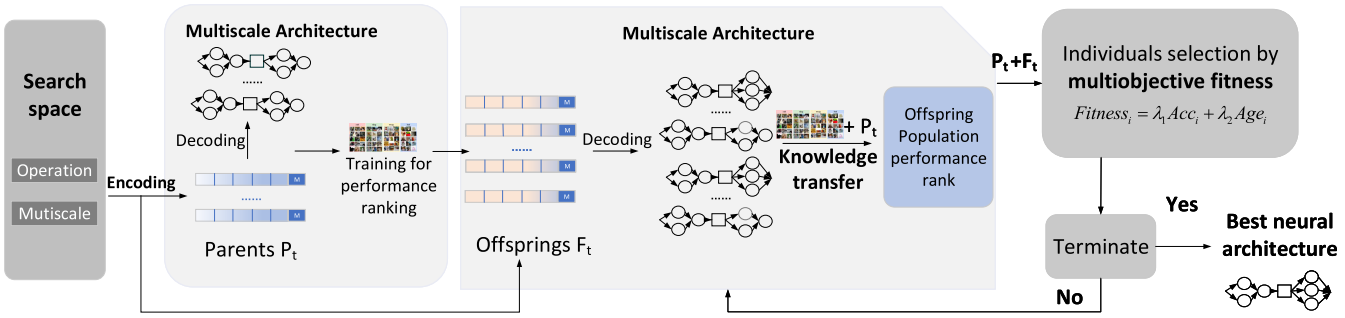

Fig. 3. Overall framework of EKTS. First, individuals are encoded, and populations are initialized according to the search space. And the initial populations are trained to obtain accuracy ranking. Second, crossover or mutation of the parent generates offspring, and offspring uses knowledge transfer strategy to obtain the accuracy ranking. Third, a new population is selected for all individuals based on the mutiobjective fitness function. Finally, if the iteration is complete, the best individual is selected, otherwise the cycle continues.

bottom subband reaches the preset minimum scale. Due to the additivity of the Lagrangian and quadtree structures, a fast bottom-up algorithm is adopted. Specifically, for each block of size $L \times L$, the corresponding optimal geometry d and the smallest Lagrangian function value $L_0(S)$ are recorded. Let $L = 2L$, we calculate the optimal geometric flow direction of the $L \times L$ square, and the minimum Lagrange function value $L(S)$ and its subsquares (S_1, S_2, S_3, S_4) combined with the Lagrange function value (the additional λT^2 is due to the split cost RS = 1 bit)

$$\tilde{L}(S) = L_0(S_1) + L_0(S_2) + L_0(S_3) + L_0(S_4) + \lambda T^2. \quad (1)$$

Then, $L_0(S) = \min(L(S), \tilde{L}(S))$ is updated. If L is less than the largest segmentation scale, let $L = 2L$ and recalculate $L_0(S)$.

2) Selection of the Best Geometry: For a subsquare of size $L \times L$, we perform sampling of angles. Discrete the angles such as the circumference angle $[0, \pi]$ into $L^2 - 1$ angles. Then, the possible values of θ are

$$\theta = \frac{k\pi}{L^2 - 1}, \quad k = 0, 1, 2, \ldots, L^2 - 1. \quad (2)$$

In the case of no geometric flow, the mark is $\theta = \inf$, which means that Bandeletization is not implemented.

For a threshold T, the direction θ, which generates a smaller approximation error, is calculated through the following steps.

First, construct a large $L \times L$ grid for each point on the square and calculate the orthogonal projection error of the sampling angle at each grid point

$$t = -\sin(\theta) \cdot x(i) + \cos(\theta) \cdot y(i). \quad (3)$$

Second, sort by error value from small to large and perform a 2-D discrete wavelet transform. Then, resort according to the transform coefficients to obtain the signal f_d. Perform a 1-D wavelet transform on f_d to obtain f_θ, and its quantized value $\tilde{f}_\theta$ is

$$Q_T(x) = \begin{cases} 0, & |x| \le T \\ \text{sign}(x) \cdot \left(q + \frac{1}{2}\right) \cdot T, & qT \le |x| \le (q+1). \end{cases} \quad (4)$$

T is the quantization threshold, $q \in Z$. Considering both the coding bit rate and the distortion degree, the optimal geometric flow direction in the area S should be such that the following Lagrange function is minimized:

$$L(f_\theta, R) = \|f_\theta - \tilde{f}_\theta\|^2 + \lambda T^2 (R_G + R_B) \quad (5)$$

where R_G is the number of bits needed to code the geometric parameter θ with an entropy coder, R_B is the number of bits needed to code the quantized coefficients $\tilde{f}_\theta$, and λ is the Lagrange multiplier.

Efficient feature extraction is a very important way to improve the performance of neural networks. In recent years, methods that combine multiscale geometric analysis with neural networks [1] have emerged to justify the combination of the two and demonstrated that the sparsity and interpretability of multiscale geometric analysis can enhance the ability of geometric transformations in CNNs. In addition, multiscale filter banks are interpretable and do not require training, significantly reducing the computational effort of search methods and increasing the approximation, learning, and generalizability of network architectures, making them a promising neural architecture model.

III. Proposed Algorithm

The specific process of the fast ENAS framework for evolutionary search of multiscale convolutional networks in this article is described in Algorithm 2 and Fig. 3.

As shown in Algorithm 2, the coding rules are set first to initialize the population. After entering the iterative process, crossing or mutating individuals in the population is carried out to generate new offspring, and the individual fitness values of the offspring are calculated based on the knowledge transfer strategy (described later in Section IV). Then, mutiobjective selection operations are performed on the populations containing parents and offspring. Finally, a new generation of the population is generated, and the first two steps are repeated until the termination condition is met.

A. Architecture Encoding

As illustrated in Fig. 4, the network has to be divided into the normal block and reduction block [13], [24]. The search space is improved based on NAS [13]. In this article, we explore a new type of multiscale network that fuses the Bandelet transform with convolution. And the search space

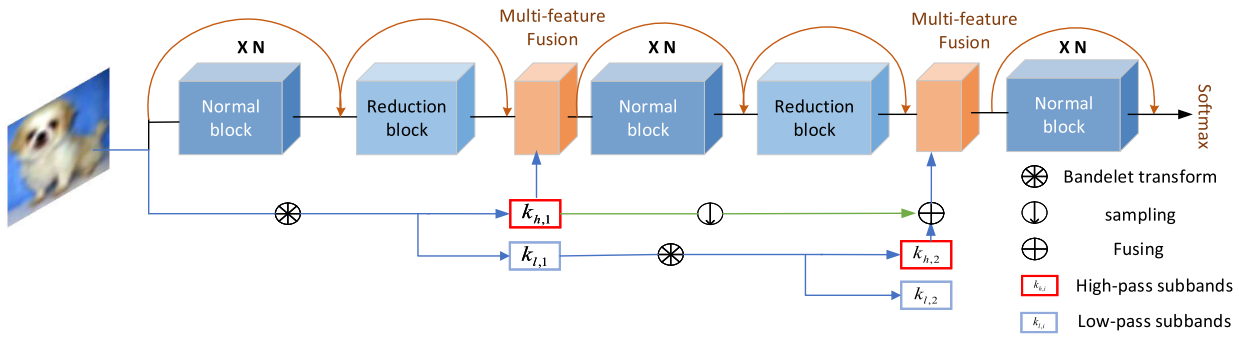

Fig. 4. Neural architecture consists of normal blocks (dark blue), reduction blocks (light blue), and multiscale geometric modules (orange). The various parts of the architecture are interconnected to form the entire network. Here, the inputs of the blocks are the outputs of the previous two blocks.

Algorithm 2 Framework of EKTS

Input: population size K, maximum number of generations G, genetic operator P_c, P_m, training data D_{train}, validation data D_{valid}

Output: the best neural network architecture

population initialization: initialize a population P_0 with a size of K;
t ← 0;
decode population P_0 into the corresponding neural network and training networks according to the performance ranking hypothesis on D_{train};
record the accuracy and age of each individual on D_{valid};
while $t \leq G$ **do**
  randomly sample two individuals from P_t;
  generate new offspring F_t by crossover or mutation;
  for *each individual in* F_t **do**
    obtain the fitness of the individual with the knowledge transfer strategy;
  $candidate_t = P_t \cup F_t$;
  new population P_{t+1} is selected and K individuals are selected from $candidate_t$ by environment selection;
  t ← t+1;
Return the best individual from P_t and decode it into the corresponding neural network ;

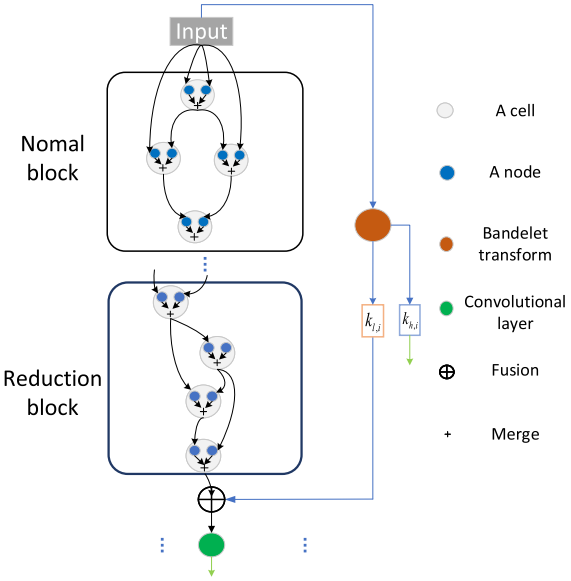

Fig. 5. Example of two blocks with four cells and multiscale connections in the neural architecture. The first block is the input, which is the beginning of the neural architecture. The first block contains four cells, where each cell is composed of two computing nodes, and the block output is a combination of the four cell outputs. The second block also contains four cells, but it differs from the normal block in that the stride of the node convolution is set to 2, and the output of the reduction block can be combined with multiscale features. Here, the solid arrows indicate the flow of information in the architecture.

in this article is defined and determined according to the target network architecture. The size of the search space is the number of all possible combinations of the architecture encoding. The design of the target network structure for the search and the description of the search space are described below.

As illustrated in Fig. 4, each block is a fully convolutional network; it inputs an $H_{\text{input}} \times W_{\text{input}} \times C_{\text{input}}$ tensor and outputs another $H_{\text{output}} \times W_{\text{output}} \times C_{\text{output}}$ tensor. Block uses stride 1 convolution in the normal block, and $H_{\text{input}} = H_{\text{output}}$ and $W_{\text{input}} = W_{\text{output}}$. In the reduction block, block uses stride 2, and $H_{\text{output}} = H_{\text{input}}/2$ and $W_{\text{output}} = W_{\text{input}}/2$. After the reduction block, the convolution features and the Bandelet transform features are combined according to the architecture chromosome.

In particular, the specific structures of the block are shown in Fig. 5, and each block is a topological structure composed of four cells. Each cell contains two operators (nodes). They are nested into each other and are connected by the relationships between the input and output. The chromosome of each block can be expressed as $([O_{11}, O_{12}, l_{11}, l_{12}], \ldots [O_{i1}, O_{i2}, l_{i1}, l_{i2}])$ and $[M]$.

There are four parameters $[O_{i1}, O_{i2}, I_{i1}, I_{i2}]$ denoted by the ith cell of the block, where I is the input of the node in the current cell and O is the type of operation operator for each node. As shown in Fig. 5, there are two corresponding nodes in each cell. Hence, there are two pairs of I and O characters. The elementwise addition of the output tensors of the two nodes is the feature map output by the current cell.

Furthermore, in addition to the four characters $[O_1, O_2, I_1, I_2]$, each reduction block corresponds to multiscale characters $[M]$, which represent whether the block is concatenated with multiscale features. This is the architecture encoding rule for evolutionary knowledge transfer search (EKTS). Using the above search encoding for search, the number of possible combinations of network structures, which is the size of the search element space, is

$$O = (\text{ops})^{2i}((i+1)!)^2 \cdot (3^M - 1)$$

where ops is the number of considered operations, i is the number of cells, and M is a possible location for multiscale connections. If a block has four cells, the overall size of the encoded search space is approximately 10^{33}. Once the encoding is determined, we can decode the encoding into the corresponding network structure according to the predefined structure. Finally, the input-output relationships are used to connect each node with multiscale features to form a complete neural network.

B. Multiscale Connection

The Bandelet transform in multiscale geometric analysis methods is combined with convolutional networks to generate multiscale networks to improve the approximation, learning, and generalization of convolutional networks.

In the uniformly regular function $f(x_1, x_2) \in C^\alpha$, α indicates that the function f is α-order differentiable. The Bandelet can achieve the optimal attenuation rate-like wavelet, that is, $CM^{-\alpha}$, and can make full use of the geometric regularity in the image while using fewer basis functions to perform a sparser approximation. For 2-D smooth functions with piecewise discontinuities on the boundary, the Bandelet cannot reduce the attenuation order due to the discontinuities but still achieves the best approximation performance [25]. Below, the approximation characteristics of the Bandelet are briefly described.

An image $f \in L^2[0,1]^d$ can be expanded on a set of orthogonal bases $\beta = \{g_m\}_{m \in N}$ as follows:

$$f = \sum_{m=0}^{+\infty} \langle f, g_m \rangle g_m. \tag{6}$$

The sparse representation of an image can be represented by the partial sum of the above formula. Therefore, in the Bandelet transform basis function, the sparse representation of an image is as follows:

$$f_M = \sum_{m \in I_m} \langle f, g_m \rangle g_m \tag{7}$$

where I_M represents the set of intrinsic product indexes whose magnitude is greater than the threshold T_M and whose number is M

$$I_M = \{m \in N : |\langle f, g_M \rangle > T_M\}. \tag{8}$$

The resulting approximation error is

$$\|f - f_M\|^2 = \sum_{m \notin I_m} |\langle f, g_M \rangle|^2. \tag{9}$$

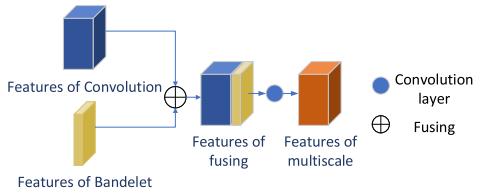

Fig. 6. Structure of the multiscale connection. The blue cube represents the convolution features, and the yellow cube represents the Bandelet transform features. After fusing the two features and passing through a convolution layer, the number of channels is the same as the number of input convolution feature channels.

When f_M with M parameters adaptively approximates f, the optimal attenuation rate is satisfied

$$\|f - f_M\|^2 \le CM^{-\alpha} \tag{10}$$

where α indicates only the regularity of the geometric image, and C is a constant.

Furthermore, the Bandelet transform can obtain the anisotropic geometric features of an image, while the CNN can extract the features of an image. The convolutional layer in the CNN can be expressed as

$$Y = W_k * x \tag{11}$$

where x and Y are the input and output of the convolution layer, W represents the weights including the bias, k represents the step size of the convolution kernel, and $*$ represents the convolution operator. Considering the convolution expression, the Bandelet transform can be expressed as follows:

$$x_l = W_2^l * x \tag{12}$$

$$x_h = W_2^h * x. \tag{13}$$

W_2^l and W_2^h are two different wavelet kernels. Their step size is 2, which indicates that the size of the generated output becomes half that of the input. The multiscale analysis performs a hierarchical decomposition by repeatedly applying (12) and (13)

$$x_{l,t+1} = W_2^{l,t} * x_{l,t} \tag{14}$$

$$x_{h,t+1} = W_2^{h,t} * x_{l,t}. \tag{15}$$

The features after the Bandelet transform are half the size of the input features. Their size is exactly the same as the size of the neural network after the downsampling pooling operation, so two features of the same size can be merged.

Therefore, a Bandelet transform connection as a multiscale connection block for NAS is proposed. First, the input image is decomposed in multiple stages using the Bandelet transform, judge whether to merge with the CNN features according to the network architecture coding, and then cascade the Bandelet transform features with the CNN features of matching size. In addition, as shown in Fig. 6, if the architecture chromosome chooses to merge features, a layer of convolution integration feature layers is added after the merge layer so that the input blue convolution layer and the orange output convolution layer are the same size.

In chromosomes, M represents whether or not multiscale connection is performed, and it has three possible values

$\{0, 1, 2\}$: A value of 0 indicates that M is not connected by multiscale connections. A value of 1 indicates that the Bandelet transform features of the same size are connected. A value of 2 means that all Bandelet transform features are fed into the network and merged, where the Bandelet transform features that do not match in size are downsampled and changed to the same size. The addition of the Bandelet transform features compensates for the single feature extracted in the end-to-end classification of the CNN and makes the entire network more robust.

From an information-theoretic point of view, the network can achieve better results by introducing additional information. From a feature perspective, the combination of spatial and spectral features makes the whole network more robust. And multiscale connection allows the network to improve its ability to learn sparse effective feature representations with fewer trainable parameters.

C. Knowledge Transfer

In global search, the network often needs to be trained to evaluate the network structure. To obtain individual performance metrics, each individual generated is trained for convergence, a process that incurs significant computational costs. To reduce the computational cost and speed up the search, the local optimization approach is used to evaluate individual performance. EKTS aims to propose a method that not only reduces redundant computations through local optimization but also expresses the performance ranking of candidate individuals in order to select the best one. Inspired by shared methods [14], [26], [27], where children in global search have similar structure and parameters with their parents, Lamarckian "acquired inheritance" is used for knowledge transfer from parents. In contrast to the weight sharing approach, our knowledge transfer approach takes into account the interactions between parameters in a neural network architecture.

A knowledge transfer strategy is proposed: new knowledge is combined with existing knowledge. While the old knowledge is inherited, the new knowledge is trained. In addition, the knowledge transfer strategy accelerates the convergence of the network, allowing us to further reduce the amount of computation during training.

Mathematically, a multilayer neural network is essentially a composite function

$$\varphi(f(x)) = \sigma(w_i * \sigma(w_{i-1} * \sigma(w_{i-2} * \cdots \sigma(w_0 * x)))) \quad (16)$$

where $*$ represents the convolution operator, σ is a nonlinear function, and w represents the convolution weight. It can be seen that the computational results of a composite function are derived step by step from layers of progressive parameters. There is a one-to-one correspondence between the parameters and the structure of the function. Inappropriate parameters can affect the performance of the architecture. The structure and parameters interact with each other to complete the computation of the neural network. The structure and parameters in a neural network do not operate independently. Therefore, if the structure or parameters of one of the nodes are changed, it will affect the computation of the whole network.

To avoid the mismatch problem, the network is usually trained from scratch to obtain the parameter values of the most suitable architecture. In general, the traditional weight update formula can be simply expressed as

$$W_{\text{new}} = W_{\text{old}} - \eta \nabla_w J(w) \quad (17)$$

where W represents the weights, $\nabla_w J(\bullet)$ represents the gradient of the loss function, and η is the learning rate. However, obtaining matching parameters requires a lot of computation. Our knowledge transfer strategy is designed with a parameter γ to control the updating of parameters in training

$$W_i = W_{i(\text{old})} - \gamma \eta \nabla_w J(w) \quad (18)$$

$$\gamma = \begin{cases} 0, & \text{structural inheritance} \\ 1, & \text{structure changed} \end{cases} \quad (19)$$

where γ is always 1 in the usual parameter update methods. Here, only the parameter of the changed part of the network structure is trained. And γ is set to 0 in other parts and the weights are kept unchanged. In this way, only a small part of the network is needed to train for network training, which is equivalent to the pretrained network. After such a knowledge transfer strategy, the new parameters of the changed structural part can match to the overall network parameters, maintaining the existence of joint dependences and adaptations between the convolutional layers. And suitable parameters can avoid inaccurate performance ranking due to the mismatch between the network structure and parameters.

For the standard evolutionary network architecture search, the population size is N. Assume that the computational volume of the parameters to be calculated for each individual is M. The optimal neural network architecture is obtained after K iterations. Then, the amount of computation required for this search is

$$C = N \cdot M \cdot K. \quad (20)$$

Among them, the number of parameters tends to be around the 20M–2M level. The process of computing $\nabla_W J(\bullet)$ largely affects the computational cost of the search. In the search proposed in this article, improvements are made for $\nabla_W J(\bullet)$. In the algorithm of this article, the weights are filtered using γ to obtain new weights. The weights to be transferred are $W_i = W_{i(\text{old})} - \gamma \eta \nabla_w J(w)$. The weight that does not need to be transferred is $W_i = W_{i(\text{old})}$.

When the weights do not need to be transferred, the new weights are equal to the on weights. At this time, there is no need to calculate the value of $\nabla_W J(\bullet)$. Then, the computation required for search by the proposed algorithm in this article is

$$C = P(\gamma = 1) \cdot N \cdot M \cdot K \quad (21)$$

where $P(\gamma = 1)$ is less than 1. Therefore, it can be said that the knowledge transfer achieves local optimization for fast learning, reducing search computational costs and improving search efficiency in the search process.

Taking the mutation operation as an example, knowledge transfer methods are presented in Fig. 7. There are neural network structures. In Fig. 7, the small box is the network structure before the mutation, and the large box below is the

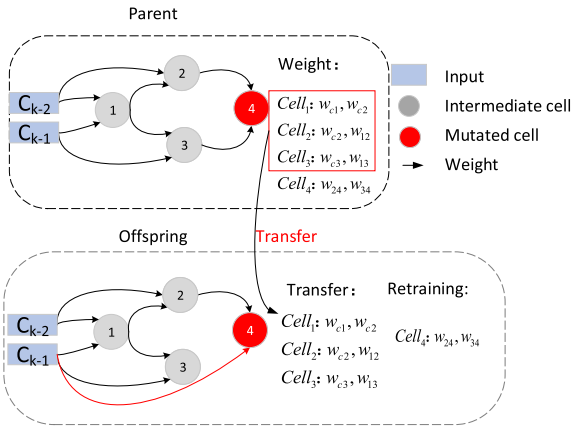

Fig. 7. Knowledge transfer in the mutations.

structure after the mutation. Cell 4 in the red circle is the cell with a mutation, and its operation and connection change. As shown in the figure, the new structure migrated the weights of Cell 1, Cell 2, and Cell 3 in the old structure, and the weights of the mutant Cell 4 were obtained by retraining. This forms a new weighted structure.

Since a large number of individuals are generated during the evolutionary search process, reducing the computational cost per individual can significantly reduce the search time. With the above knowledge transfer-based local optimization strategy, a ranking of population performance is obtained while reducing redundant computations, allowing us to easily select structures with outstanding performance for evolution.

D. Multiobjective Fitness

The multiobjective fitness is the driving force behind evolutionary algorithms. The multiobjective fitness is selected according to the individual degree of fitness. Among other things, the design of the degree of fitness and the change in the multiobjective fitness at each stage have an impact on evolution. The selection directs the evolutionary algorithm search to promising regions of the search space. Multiobjective fitness optimization is used to find the Pareto optimal (or nondominated) solutions [28], [29], [30], making the set of solutions found more diverse and preventing premature convergence.

Based on the tournament selection method, in addition to individual precision, the ability to control the direction of evolution by the degree of aging of the individuals [19] is added. A number of iterations in individual survival are used to eliminate older individuals, making the algorithm more biased toward younger individuals

$$\text{Fitness}_i = \lambda_1 \text{Acc}_i + \lambda_2 \text{Age}_i. \tag{22}$$

Here, Fitness_i is the multiobjective fitness of the ith individual; λ_1 and λ_2 are constant and represent the weighting of each item. Acc_i is the precision of the ith individual in the population. Age_i of the ith individual is the number of epochs that the ith individual has survived the population renewal. In terms of the optimization problem, a multiobjective optimization problem is used. With this setup, a precision term is used to maintain reasonable differences between individuals when their relative age differences are small, accelerating the competition. Moreover, when individual precision is small, applying the age term to eliminate the most senior individuals increases diversity and prevents premature convergence.

TABLE I
NETWORK OPERATION CODE SPACE

Operation type	Kernel size	Code
None	-	0
skip_connect	-	1
Convolution	7x1, 1x7	2
Separable convolution	3x3	3
Separable convolution	5x5	4
Separable convolution	7x7	5
Dilated separable convolution	3x3	6
Dilated separable convolution	5x5	7
Average pooling	3x3	8
Max pooling	3x3	9

TABLE II
SUMMARY OF THE PARAMETER SETTINGS IN THE SEARCH

Parameter name	Parameter value
Crossover probability	0.95
Mutation probability	0.05
Batch size	128
Weight decay	3×10^{-4}
Momentum	0.9
Learning rate	1×10^{-3}
Optimizer	SGD
Nesterov	True
Drop Path	0.2
Population size	10
Epoch	100

IV. EXPERIENCE AND VALIDATION

The EKTS model proposed in this article aims to reduce the computational cost and can quickly search for efficient multiscale neural networks. In this section, our experiments in detail are introduced. The possible operations in the architecture search are shown in Table I. We search on the CIFAR-10 dataset [21] and evaluate on the CIFAR-10 and ILSVRC 2012 ImageNet datasets [31]. The results are compared with popular human-designed architectures and state-of-the-art NAS methods; it verify the effectiveness of the proposed method.

A. Parameter Setting

Regarding the details of the evolution process, the parameter values during the architecture search is shown in Table II. As shown in the table, crossover and mutation operations are performed on the corresponding node. The crossover and mutation rates are set to 0.95 and 0.05, respectively. For the crossover, a one-cut point crossover and swap mutation are used to modify a certain segment of the neural network. In particular, the one-cut point crossover refers to randomly

TABLE III
CHROMOSOMES OF THE ARCHITECTURES

Block	Arch1	Arch2	Arch3	Arch4
Normal	[7,7,0,1]	[5,9,0,1]	[7,2,1,0]	[7,2,1,0]
	[5,5,0,1]	[1,2,0,1]	[4,4,1,2]	[4,4,1,2]
	[4,4,1,2]	[7,0,0,1]	[6,0,0,1]	[6,0,0,1]
	[7,1,0,2]	[5,1,0,4]	[3,8,2,3]	[5,8,2,3]
Reduction	[3,9,0,1]	[9,1,0,1]	[3,4,0,1]	[3,4,0,1]
	[9,4,0,2]	[9,9,0,1]	[2,3,0,2]	[2,3,0,2]
	[3,0,2,3]	[8,1,0,3]	[1,3,0,1]	[1,3,0,1]
	[9,5,0,2]	[9,2,0,3]	[1,6,1,2]	[1,6,1,2]
M	[0,1]	[1,1]	[0,1]	[0,1]

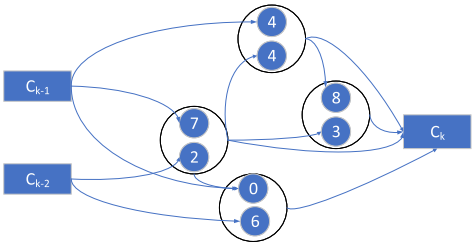

Fig. 8. Normal block of Arch 3.

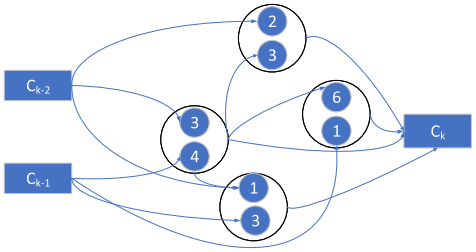

Fig. 9. Reduction block of Arch 3.

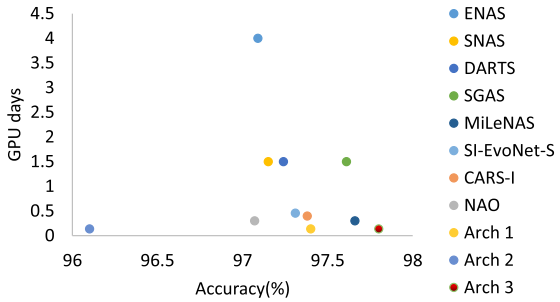

Fig. 10. Comparison of search time and performance with EKTS and other excellent algorithms on CIFAR-10.

selecting two network structures and randomly selecting a section of the same length in the two network structures for exchange. The swap mutation refers to randomly selecting two parts of a network structure with the same length for exchange. All experiments are performed on a TITAN Xp GPU.

B. Search on CIFAR-10

The CIFAR-10 [21] is a commonly used classification dataset. It has ten categories and is composed of 60 000 32 × 32 RGB images, where each category has 6000 images. A total of 10 000 of the 60 000 images are used as test images, and the rest are used as training images.

We use the CIFAR-10 training set to train the neural architecture on the search and the test sets to verify the accuracy of the architecture. The design population is 10. After 100 generations of environmental selection, the best individual is selected as our target structure. To better demonstrate the performance of the search strategy, we run the search algorithm three times. The three searches finally obtain three network architectures, Arch 1, Arch 2, and Arch 3. The above-mentioned chromosomes of Arch 1, Arch 2, Arch 3, and a mutated Arch 4 are shown in Table III. Specifically, the architecture of Arch 3 is shown in Figs. 8 and 9.

The searched architectures are retrained to obtain the accuracy on the CIFAR-10 dataset after convergence. Then, the searched architectures are compared with state-of-the-art models. Table IV shows the CIFAR-10 results of the convolutional architectures. RL, the evolution-based algorithm (EA), gradient descent (GD), multinomial distribution learning (MDL), and surrogate model-based optimization (SMBO) are the models for comparison.

By comparing the data in Table IV, we can see the advantages of this proposed algorithm. First, the algorithm proposed in this article significantly improves the speed of the evolutionary neural network architecture search algorithm. In the evolutionary search process, the iterative 100-generation search only took 0.14 GPU days [50] and achieved good results. Although evolutionary computation has good global optimality, the slow search speed has been a challenge to overcome. As seen from the table, the search using evolutionary algorithms generally require more search time than other search algorithms. By optimizing the evolutionary algorithm, the redundant computation can be effectively reduced to improve the search speed. The SI-EvoNet-S algorithm further reduces the computation in the search process by optimizing the evolutionary algorithm and reducing the number of parameters of the network structure to be searched. However, the SI-EvoNet-S algorithm still requires high hardware support. The search algorithm proposed in this article can search medium-sized network structures and also has advantages in speed compared to other search algorithms. By optimizing the evolutionary search, the algorithm has less search time and GPU hardware requirements. The algorithm has low implementation cost, which can promote the landing and development of search algorithms for neural network architectures. Second, the table also shows that we search for small-scale neural network architectures. In general, larger model parameters are more likely to fit the data distribution and achieve higher accuracy. However, the searched neural network architectures can achieve higher accuracy. This is due to the fact that in the search network framework of this article, we introduce multiscale structures. The multiscale structure is used to improve the representational ability of the network structure to improve the accuracy of the network results. It also reduces the number of parameters in the network

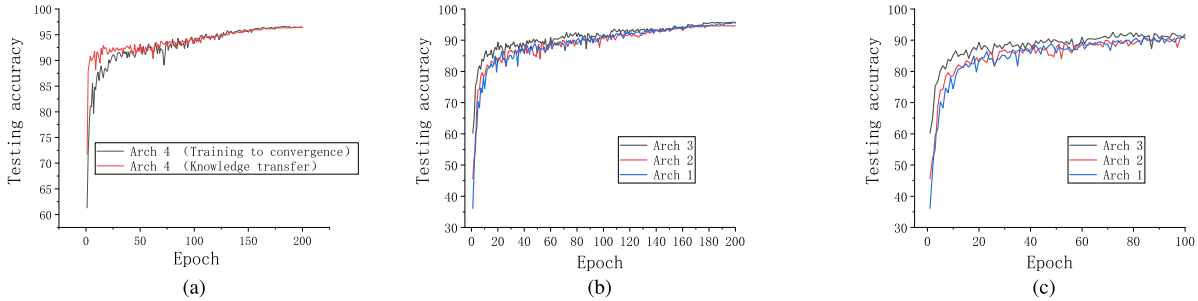

Fig. 11. Different neural architectures are selected for training and testing on CIFAR-10. In (a), Arch 4 changes a node of the neural architecture of Arch 3. Knowledge transfer and retraining are performed to obtain the accuracy curve of architecture during the training process. (b) and (c) Accuracy curves of the three neural architectures during the training process. The optimal architecture maintains the optimal accuracy throughout the training process. (a) Comparison of the accuracy in two methods. (b) 200 epoch. (c) 100 epoch.

architecture by reducing repeated convolutional computations. This can reduce the computation in the network training and further speed up the whole search process. It can be seen that the introduction of multiscale possesses higher accuracy than simple convolution for the same number of parameters. As shown in Fig. 10, the searched structure is excellent not only in terms of speed but also in terms of performance. Obviously, KT-ENAS can quickly explore the search space and generate a good architecture.

1) Effectiveness of Knowledge Transfer: To verify the effectiveness of the knowledge transfer-based local optimization strategy, we randomly selected Arch 3 for the following experiments.

As shown in Table III, the mutation operation in global evolution is simulated and mutated Arch 3 to produce a new network architecture (Arch 4). Knowledge transfer training and scratch training until convergence are separately performed on the new Arch 4 network architecture and recorded and compared the changes in network architecture performance between the two training methods. Regarding the parameters in both training methods, we used a stochastic GD (SGD) optimizer to train the network. The initial learning rate was set to 0.025 and annealed to zero according to a cosine schedule, where the momentum was 0.9 and the weights decayed to 3×10^4. The learning rate of the polynomial parameter was set to 0.01. The batch of training data was set to 96.

In Fig. 11(a), it shows the change in accuracy on the test set during the training iterations for both training methods. The red curve represents the change in accuracy of the Arch 4 architecture trained with the knowledge transfer strategy. The red curve inherits some parameters of the Arch 3 architecture before training and freezes these parameters during training, only training the weight values of the nodes where the changes occur. The black curve represents the training of the Arch 4 architecture from scratch until convergence with all weight parameters. It is clear that the red curve converges faster than the black curve and eventually arrives at a comparable convergence accuracy. In contrast, the local optimization algorithm of knowledge transfer can achieve a convergence degree similar to that of the traditional 70th generation in the 10th generation.

From the above, it can be concluded that the locally optimized knowledge transfer strategy is able to represent the performance of the network architecture and accelerate the convergence of the network training. This strategy allows the network architecture to be trained with only a small number of trainable parameters, and it converges quickly and is feasible for reducing the computational cost of global evolution.

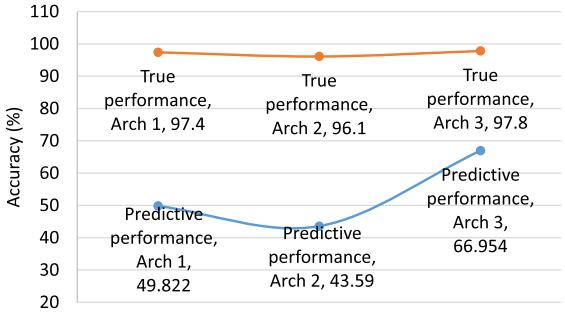

Fig. 12. Predictive performance (blue line) and true performance (red line) of individuals on CIFAR-10.

2) Precision Ranking Hypothesis in Search: To further reduce the computational effort, the performance ranking hypothesis [53] is used. The premise of the performance ranking hypothesis is that in obtaining the performance of individuals, they are not trained to convergence. The strategy only expects to obtain the accuracy ranking of the population. Three networks which searched from the CIFAR-10 dataset (Arch 1, Arch 2, and Arch 3) are chosen and plotted their accuracy with increasing chronology, as shown in Fig. 11(b) and (c). The figures illustrate that Arch 3 showed better performance than the other architectures (Arch 1 and Arch 2) in the initial phase of training, which continued until final convergence. Although the accuracy rankings for Arch 2 and Arch 3 are confusing, this error is within our acceptance range due to the relatively similar performance of the two architectures. This means that we can obtain a ranking of the accuracy for network architectures, at the beginning of the iteration without training each network architecture until convergence. Moreover, we apply this approach to all training of individuals, including individuals that have undergone a knowledge transfer strategy. As shown in Fig. 12, the predicted values of the performance ranking hypothesis are compared with the true value, which represents the performance ranking of the candidate individuals. Therefore, the performance ranking hypothesis applies to our NAS for identifying optimal performance architectures.

TABLE IV
COMPARISON WITH STATE-OF-THE-ART IMAGE CLASSIFIERS ON THE CIFAR-10 DATASET

Architecture	Test Error (%)	Params (M)	Search Cost (GPU days)	GPUs	Search Method
VGG [32]	6.66	28.05	-	-	manual
ResNet(depth=101) [6]	6.43	1.7	-	-	manual
Pre-ResNet [33]	4.46	10.3	-	-	manual
Wide-ResNet [34]	4.17	36.5	-	-	manual
DenseNet(L = 40; k = 12) [8]	2.24	25.6	-	-	manual
IGCV3-D [35]	5.04	2.2	-	-	manual
MobileNetV2 [36]	5.44	2.1	-	-	manual
ShuffleNet [37]	9.13	1.06	-	-	manual
PNAS [38]	3.4	3.2	225	-	SMBO
ENAS + cutout [14]	2.91	4.2	4	1	RL
NASNet-A + cutout [13]	2.65	3.3	2,000	500 P100	RL
NASv3 [12]	4.47	7.1	22,400	800 K40	RL
Faster BlockQNN [39]	3.15	34.4	0.83	1 1080Ti	RL
MetaQNN [40]	6.92	11.2	90	10	RL
DPP-Net-M [41]	5.84	0.45	8	4 GTX 1080	RL
PPP-Net-A [42]	5.28	0.45	-	Titan X Pascal	RL
Proxyless NAS [43]	2.08	5.7	1,500	-	RL
SNAS (moderate) + cutout [44]	2.85	2.8	1.5	1	GD
DARTS (first) + cutout [15]	3.0	3.3	1.5	4 1080Ti	GD
DARTS (second) + cutout [15]	2.76	3.3	4	4 1080Ti	GD
SGAS [45]	2.39	3.8	0.25	1 1080Ti	GD
MiLeNAS [46]	2.34	3.87	0.3	-	GD
AmoebaNet-A + cutout [19]	3.12	3.1	3,150	450 K40	EA
Hierarchical [47]	3.75	15.7	300	200	EA
Large-scale [18]	5.4	5.4	2,750	250	EA
CGP-CNN [48]	5.98	1.7	27	2	EA
AE-CNN [49]	4.7	2.0	27	-	EA
AE-CNN+E2EPP [50]	5.3	4.3	8.5	-	EA
LEMONADE [51]	2.58	13.1	90	16 Titan X	EA
NSGA-Net [52]	2.75	3.3	4	-	EA
SI-EvoNet-S [26]	2.69	0.51	0.458	2 RTX 2080Ti	EA
MdeNAS [53]	2.55	3.61	0.16	1 GTX 1080Ti	MDL
CARS-I [54]	2.62	3.6	0.4	-	EA+GD
NAO [55]	2.93	2.5	0.3	1 V100	GD+SMBO
Arch 1	**2.63**	**4.0**	**0.14**	**1 Titan Xp**	**EA**
Arch 2	**3.87**	**5.2**	**0.14**	**1 Titan Xp**	**EA**
Arch 3	**2.20**	**5.8**	**0.14**	**1 Titan Xp**	**EA**

3) Effectiveness of the Multiscale Connection Module: Our framework seeks to improve the accuracy of the neural network without a large increase in its computational cost. Therefore, a multiscale connection module is designed in a neural network. To test the effectiveness of the multiscale connection modules on neural network performance, multiscale ablation experiments are conducted. The multiscale geometric connection is removed in architectures to compare the impact of the multiscale geometric connection on the neural architecture.

To test the accuracy in this training experiment, the parameter settings of the two training situations are kept exactly the same, and the results are shown in Table V. The removal of the multiscale geometric connection in the Arch 1 architecture reduces the accuracy by 1.62% points. It can be seen that adding multiscale connection modules to the network structure improves the performance of the network. As shown in Table V, the multiscale connection module can achieve high classification accuracy in image classification tasks.

TABLE V
COMPARISON OF THE IMPACT OF MULTISCALE CONNECTION ON ARCHITECTURE ACCURACY

Architecture	Multiscale geometric connection	CIFAR-10 (%)
Arch 1	✓	**97.37**
	×	95.75
Arch 2	✓	**96.13**
	×	94.97
Arch 3	✓	**97.79**
	×	96.53

4) Effectiveness of the Multiobjective Fitness Module: Multiobjective evolution is beneficial to the evolutionary process, as shown in Fig. 13. Age and precision as a fitness judging function is combined in evolution. Unlike [19], we combine the age and precision of each individual rather than simply discarding older individuals to avoid losing good individuals.

The difference in the distribution of the two results can be seen in Fig. 13. First, the distribution of the yellow points, which uses multiobjective fitness module, is more clustered and the blue points are more spread out. This indicates that the

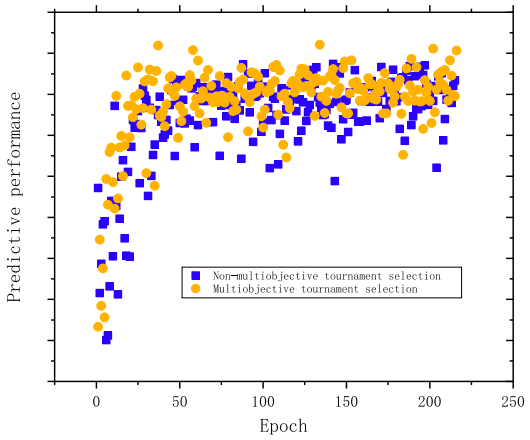

Fig. 13. Comparison between our multiobjective tournament selection and the non-multiobjective tournament selection in the same population sizes.

TABLE VI
COMPARISON WITH THE SCATTERING NETWORK ON THE CIFAR-10 DATASET

Method	Architecture	CIFAR-10 (%)
Scattering	Trans.,Order 1	72.6
	Trans.,Order 2	80.3
	Trans.,Order 2 + OLS	81.6
EKTS	**Arch 1**	**97.4**
	Arch 2	**96.1**
	Arch 3	**97.8**

gap between the yellow points is smaller, and the distribution is more stable than the blue points. Furthermore, it can be seen in the figure that the yellow dots are more prominent in terms of performance, with higher performing individuals appearing 50 generations ago. From this, we can conclude that the multiobjective fitness has higher performance than the non-multiobjective, and the performance of individuals in the population is more stable.

5) Comparison With the Scattering Network: Bruna and Mallat [56] have proposed scattering networks, which are based on a network structure composed of predefined complex wavelet filters. It cascades wavelet transform convolutions with a nonlinear modulus and averaging operators for the image classification. In contrast to deep learning models, scattering networks do not require training, and this architecture performs well in applications [57]. Unlike scattering networks, our proposed network replaces the scattering transform with the Bandelet transform. Taking into account the geometric invariants and training-free nature of the multiscale geometric filter, the multiscale geometric transform is fused with a convolutional network in the hopes of generating more robust and powerful features for image classification. The performances are compared of both on the CIFAR-10 dataset and prove that our structures can produce better features.

In Table VI, Trans.,Order 1 and Trans.,Order 2 represent the scattering network with a first-order coefficient matrix and a second-order coefficient matrix, respectively. The orthogonal least square (OLS) represents the scattering network with OLS feature reduction.

By comparing the accuracy results of the scattering and EKTS methods in the table, it is clear that the EKTS method performs better on the dataset. Our proposed EKTS method combining Bandelet and convolution can improve the accuracy by at least 15% points over the scattering method. This suggests that our proposed method is effective for image classification task.

C. Results on ILSVRC2012

Since searching for the architecture on the ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012) [31] is more time consuming than on CIFAR-10, researchers generally consider experimenting with migration operations on ILSVRC2012. Moreover, the transferability of the searched architectures for the network structure is evaluated [13], [15]. The models are searched on CIFAR-10, transferred to ILSVRC2012, and trained on the ILSVRC2012 dataset [31] to evaluate the searched architectures.

To adapt our experiment to the ILSVRC2012 dataset, the number of normal block repetitions is changed from $N = 6$ to $N = 4$ [15], and the rest of the structure remained unchanged. In the experiment, we use the SGD optimizer for training, the initial learning rate is set to 0.1, and the learning rate is annealed to zero according to a cosine schedule, where the momentum is 0.9 and the weight decays to 3×10^5. The batch size during training is set to 128, and the training involves 200 epochs. The results are shown in Table VII. It can be seen that the EKTS models, which were found on CIFAR-10, also performed well on the ILSVRC2012 dataset. EKTS Arch 3 achieves top 1 and top 5 accuracies of 73.8% and 91.6%, respectively. The EKTS achieves better performance than most popular human-designed architectures and state-of-the-art NAS frameworks. The ILSVRC2012 dataset is one of the more challenging color image databases due to its high class variabilities. It is further evidence of the generalizability and superiority of our architecture. These results also demonstrate that the models found on CIFAR-10 can be directly used to effectively learn larger datasets, such as the ILSVRC2012 dataset.

V. APPLICATION

There are many application scenarios for neural network architecture search algorithms. Neural network architecture search algorithms are also one of the current hot topics of research in the field of visual computers. The neural network architecture search algorithm aims to design suitable neural networks in an automated way by using limited computational resources.

In this article, we propose a fast search method for multiscale deep neural architecture. In reality, the proposed neural network architecture search algorithm has many application scenarios. First, it saves the cost of designing neural network architectures because expert experience and iterative trial and error are not used. It can also automatically design the appropriate neural network architecture according to the actual needs. The algorithm searches out a network with high generalization ability and friendly hardware requirements.

TABLE VII
COMPARISON WITH STATE-OF-THE-ART IMAGE CLASSIFIERS ON THE ILSVRC2012 DATASET. THE EKTS MODELS USE THE ARCHITECTURES FOUND ON THE CIFAR-10 DATASET

Architecture	Top1 Acc (%)	Top5 Acc (%)	Params (M)	Search Cost (GPU days)	GPUs	Search Method
ResNet50 [6]	75.3	92.2	25.6	-		manual
MorphNet [58]	75.2	-	15.5	-	-	manual
InceptionV1 [7]	69.8	90.1	6.6	-	-	manual
MobileNetV2 (1x) [36]	72.0	90.4	3.4	-	-	manual
ShuffleNetV2 (2x) [59]	74.9	90.1	7.4	-	-	manual
PNAS [38]	74.2	91.9	5.1	224	-	SMBO
SemiNAS [60]	76.5	93.2	6.32	4	-	SMBO
Block-wise QNN [61]	81.0	95.42	91	96	32 1080Ti	RL
NASNet-A [13]	74.0	91.6	5.3	2,000	500 P100	RL
MNasNet-A1 [62]	75.2	92.5	3.9	1.666	-	RL
SNAS (mild) [44]	72.7	90.8	4.3	1.5	-	GD
GDAS [63]	72.5	90.9	4.4	0.17	1	GD
DARTS [15]	73.3	91.3	4.7	4	-	GD
RCNet [64]	72.2	91.0	3.4	8	-	GD
FBNetV2-L1 [65]	77.2	-	-	25	8 V100	GD
Arch 1	**72.8**	**90.8**	**5.5**	**0.14**	**1 Titan Xp**	**EA**
Arch 2	**73.1**	**90.9**	**7.1**	**0.14**	**1 Titan Xp**	**EA**
Arch 3	**73.8**	**91.6**	**8.1**	**0.14**	**1 Titan Xp**	**EA**

In the e-commerce industry, it can be used to automate image processing. Search for neural architectures adapted to different products to classify and label product images. In the education industry, it can help students with learning disabilities and handicaps. Searching out neural architectures to recognize images and convert images into speech. In manufacturing, different neural architectures can be designed for image recognition at different stages of the manufacturing cycle. Reduce manufacturing defects by identifying anomalies in the manufacturing process. Due to the convenience of automatic search in neural network architectures, it can be applied in many industries. Meanwhile, this article introduces the combination of multiscale and convolution and proves the effectiveness of the combination of both. Network architectures combining multiscale and depth can be further investigated.

VI. CONCLUSION

In this article, knowledge transfer strategies and multiscale geometric analysis are combined to provide a fast and efficient general method for searching neural network architectures. The local optimization based on knowledge transfer is combined with a global evolutionary search strategy, which considers the existence of joint dependences and adaptations among convolutional layers. And knowledge transfer achieves local optimization for fast learning, reducing search computational costs and improving search efficiency. At the same time, the integration of convolution and multiscale geometry in designing the network architecture can effectively improve the approximation, learning, and generalization of the neural architecture. The results of our search are compared with popular artificial design architectures and state-of-the-art NAS methods. Experimental results demonstrate that the method proposed in this article effectively alleviates the computational cost of neural network architecture search and explores a novel multiscale network structure combining Bandelet transform and convolution.

The proposed EKTS shows potential for development. In the future, we will further explore neural network architectures for more complex scenarios as well as new paradigms for more effective neural network architectures. In addition, we will extend the neural network architecture search algorithm to more application scenarios.

REFERENCES

[1] T. Williams and R. Li, "Wavelet pooling for convolutional neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2018.
[2] M. Liu, L. Jiao, X. Liu, L. Li, F. Liu, and S. Yang, "C-CNN: Contourlet convolutional neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 6, pp. 2636–2649, Jun. 2021.
[3] A. A. M. Muzahid, W. Wan, F. Sohel, L. Wu, and L. Hou, "CurveNet: Curvature-based multitask learning deep networks for 3D object recognition," *IEEE/CAA J. Autom. Sinica*, vol. 8, no. 6, pp. 1177–1187, Jun. 2021.
[4] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *J. Mach. Learn. Res.*, vol. 20, pp. 55:1-55:21, Mar. 2019.
[5] L. Xu, "An overview and perspectives on bidirectional intelligence: Lmser duality, double IA harmony, and causal computation," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 4, pp. 865–893, Jul. 2019.
[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[7] C. Szegedy et al., "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.
[8] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2261–2269.
[9] M. Loni et al., "FastStereoNet: A fast neural architecture search for improving the inference of disparity estimation on resource-limited platforms," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 52, no. 8, pp. 5222–5234, Aug. 2022.
[10] F. E. Fernandes and G. G. Yen, "Automatic searching and pruning of deep neural networks for medical imaging diagnostic," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 12, pp. 5664–5674, Dec. 2021.
[11] H. Zhu and Y. Jin, "Multi-objective evolutionary federated learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 4, pp. 1310–1322, Apr. 2020.
[12] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," in *Proc. 5th Int. Conf. Learn. Represent.*, Toulon, France, Apr. 2017.

[13] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8697–8710.

[14] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean, "Efficient neural architecture search via parameter sharing," in *Proc. ICML*, 2018, pp. 4092–4101.

[15] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable architecture search," in *Proc. Int. Conf. Learn. Represent.*, 2019.

[16] X. Wang, Y. Jin, and K. Hao, "Evolving local plasticity rules for synergistic learning in echo state networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 4, pp. 1363–1374, Apr. 2020.

[17] S. Fujieda, K. Takayama, and T. Hachisuka, "Wavelet convolutional neural networks," 2018, *arXiv:1805.08620*.

[18] E. Real et al., "Large-scale evolution of image classifiers," in *Proc. ICML*, 2017, pp. 2902–2911.

[19] E. Real, A. Aggarwal, Y. Huang, and Q. Le, "Regularized evolution for image classifier architecture search," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Feb. 2018, pp. 1–12.

[20] M. Cui, L. Li, M. Zhou, J. Li, A. Abusorrah, and K. Sedraoui, "A bi-population cooperative optimization algorithm assisted by an autoencoder for medium-scale expensive problems," *IEEE/CAA J. Autom. Sinica*, vol. 9, no. 11, pp. 1952–1966, Nov. 2022.

[21] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Comput. Sci. Dept., Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 1, Jan. 2009.

[22] J. Licheng and T. Shan, "Multi-scale geometric analysis of images: Review and outlook," *Electron. J.*, vol. 31, no. b12, pp. 1975–1981, 2004.

[23] E. Le Pennec and S. Mallat, "Bandelet image approximation and compression," *Multiscale Model. Simul.*, vol. 4, no. 3, pp. 992–1039, Jan. 2005.

[24] Y. Xue, Y. Wang, J. Liang, and A. Slowik, "A self-adaptive mutation neural architecture search algorithm based on blocks," *IEEE Comput. Intell. Mag.*, vol. 16, no. 3, pp. 67–78, Aug. 2021.

[25] E. L. Pennec and S. Mallat, "Sparse geometric image representations with bandelets," *IEEE Trans. Image Process.*, vol. 14, no. 4, pp. 423–438, Apr. 2005.

[26] H. Zhang, Y. Jin, R. Cheng, and K. Hao, "Efficient evolutionary search of attention convolutional networks via sampled training and node inheritance," *IEEE Trans. Evol. Comput.*, vol. 25, no. 2, pp. 371–385, Apr. 2021.

[27] G. Wang, J. Qiao, J. Bi, W. Li, and M. Zhou, "TL-GDBN: Growing deep belief network with transfer learning," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 2, pp. 874–885, Apr. 2019.

[28] S. Gao, M. Zhou, Y. Wang, J. Cheng, H. Yachi, and J. Wang, "Dendritic neuron model with effective learning algorithms for classification, approximation, and prediction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 2, pp. 601–614, Feb. 2019.

[29] C. Chen, N. Lu, B. Jiang, and C. Wang, "A risk-averse remaining useful life estimation for predictive maintenance," *IEEE/CAA J. Autom. Sinica*, vol. 8, no. 2, pp. 412–422, Feb. 2021.

[30] P. Zhang, S. Shu, and M. Zhou, "An online fault detection model and strategies based on SVM-grid in clouds," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 2, pp. 445–456, Mar. 2018.

[31] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[32] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent.*, Y. Bengio and Y. LeCun, Eds. San Diego, CA, USA, 2015.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Computer Vision—ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham, Switzerland: Springer, 2016, pp. 630–645.

[34] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proc. Brit. Mach. Vis. Conf.*, R. C. Wilson, E. R. Hancock, and W. A. P. Smith, Eds., 2016.

[35] K. Sun, M. Li, D. Liu, and J. Wang, "IGCV3: Interleaved low-rank group convolutions for efficient deep neural networks," in *Proc. Brit. Mach. Vis. Conf.*, 2018, p. 101.

[36] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[37] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6848–6856.

[38] C. Liu et al., "Progressive neural architecture search," in *Proc. ECCV*, 2018, pp. 19–35.

[39] Z. Zhong et al., "BlockQNN: Efficient block-wise neural network architecture generation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 7, pp. 2314–2328, Jul. 2021.

[40] B. Baker, O. Gupta, N. Naik, and R. Raskar, "Designing neural network architectures using reinforcement learning," in *Proc. 5th Int. Conf. Learn. Represent.*, Toulon, France, Apr. 2017.

[41] J.-D. Dong, A.-C. Cheng, D.-C. Juan, W. Wei, and M. Sun, "DPP-Net: Device-aware progressive search for Pareto-optimal neural architectures," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 1–11.

[42] J. Dong, A. Cheng, D. Juan, W. Wei, and M. Sun, "PPP-Net: Platform-aware progressive search for Pareto-optimal neural architectures," in *Proc. 6th Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, Apr. 2018.

[43] H. Cai, L. Zhu, and S. Han, "Proxylessnas: Direct neural architecture search on target task and hardware," in *Proc. 7th Int. Conf. Learn. Represent.*, New Orleans, LA, USA, May 2019, pp. 1–15.

[44] S. Xie, H. Zheng, C. Liu, and L. Lin, "SNAS: Stochastic neural architecture search," in *Proc. 7th Int. Conf. Learn. Represent.*, New Orleans, LA, USA, May 2019.

[45] G. Li, G. Qian, I. C. Delgadillo, M. Müller, A. Thabet, and B. Ghanem, "SGAS: Sequential greedy architecture search," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Seattle, WA, USA, Jun. 2020, pp. 1617–1627.

[46] C. He, H. Ye, L. Shen, and T. Zhang, "MiLeNAS: Efficient neural architecture search via mixed-level reformulation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11990–11999.

[47] H. Liu, K. Simonyan, O. Vinyals, C. Fernando, and K. Kavukcuoglu, "Hierarchical representations for efficient architecture search," in *Proc. 6th Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, Apr. 2018.

[48] M. Suganuma, S. Shirakawa, and T. Nagao, "A genetic programming approach to designing convolutional neural network architectures," in *Proc. Genetic Evol. Comput. Conf.*, Stockholm, Sweden, Jul. 2017, pp. 5369–5373.

[49] Y. Sun, B. Xue, M. Zhang, and G. G. Yen, "Completely automated CNN architecture design based on blocks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 4, pp. 1242–1254, Apr. 2020.

[50] Y. Sun, H. Wang, B. Xue, Y. Jin, G. G. Yen, and M. Zhang, "Surrogate-assisted evolutionary deep learning using an end-to-end random forest-based performance predictor," *IEEE Trans. Evol. Comput.*, vol. 24, no. 2, pp. 350–364, Apr. 2020.

[51] T. Elsken, J. H. Metzen, and F. Hutter, "Efficient multi-objective neural architecture search via Lamarckian evolution," in *Proc. 7th Int. Conf. Learn. Represent.*, New Orleans, LA, USA, May 2019.

[52] Z. Lu et al., "NSGA-Net: A multi-objective genetic algorithm for neural architecture search," in *Proc. Genetic Evol. Comput. Conf.*, 2019, pp. 419–427.

[53] X. Zheng, R. Ji, L. Tang, B. Zhang, J. Liu, and Q. Tian, "Multinomial distribution learning for effective neural architecture search," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1304–1313.

[54] Z. Yang et al., "CARS: Continuous evolution for efficient neural architecture search," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1826–1835.

[55] R. Luo, F. Tian, T. Qin, E. Chen, and T.-Y. Liu, "Neural architecture optimization," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran Associates, 2018, pp. 7827–7838.

[56] J. Bruna and S. Mallat, "Invariant scattering convolution networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1872–1886, Aug. 2013.

[57] E. Oyallon and S. Mallat, "Deep roto-translation scattering for object classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 2865–2873.

[58] A. Gordon et al., "MorphNet: Fast & simple resource-constrained structure learning of deep networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1586–1595.

[59] N. Ma, X. Zhang, H. Zheng, and J. Sun, "ShuffleNet V2: Practical guidelines for efficient CNN architecture design," in *Proc. ECCV*, 2018, pp. 122–138.

[60] R. Luo, X. Tan, R. Wang, T. Qin, E. Chen, and T. Liu, "Semi-supervised neural architecture search," in *Proc. Adv. Neural Inf. Process. Syst.*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., Dec. 2020.
[61] Z. Zhong, J. Yan, W. Wu, J. Shao, and C. Liu, "Practical block-wise neural network architecture generation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2423–2432.
[62] M. Tan et al., "MnasNet: Platform-aware neural architecture search for mobile," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Long Beach, CA, USA, Jun. 2019, pp. 2815–2823.
[63] X. Dong and Y. Yang, "Searching for a robust neural architecture in four GPU hours," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1761–1770.
[64] Y. Xiong, R. Mehta, and V. Singh, "Resource constrained neural network architecture search: Will a submodularity assumption help?" in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1901–1910.
[65] A. Wan et al., "FBNetV2: Differentiable neural architecture search for spatial and channel dimensions," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 12962–12971.

Ruohan Zhang (Student Member, IEEE) received the B.S. degree in applied physics from the Xi'an University of Posts and Telecommunications, Xi'an, Shaanxi, China, in 2017. She is currently pursuing the Ph.D. degree with the Key Laboratory of Intelligence Perception and Image Understanding, Ministry of Education of China, Xidian University, Xi'an.

Her current research interests include deep learning, object detection, and image understanding.

Licheng Jiao (Fellow, IEEE) received the B.S. degree from Shanghai Jiaotong University, Shanghai, China, in 1982, and the M.S. and Ph.D. degrees from Xi'an Jiaotong University, Xi'an, China, in 1984 and 1990, respectively.

Since 1992, he has been a Professor with Xidian University, Xi'an, where he is currently the Director of the Key Laboratory of Intelligent Perception and Image Understanding, Ministry of Education of China. His research interests include image processing, natural computation, machine learning, and intelligent information processing.

Dr. Jiao is a fellow of The Institution of Engineering and Technology (IET)/Chinese Association for Artificial Intelligence (CAAI)/Chinese Institute of Electronics (CIE)/China Computer Federation (CCF)/Chinese Association of Automation (CAA). He is also the Chairperson of the Awards and Recognition Committee, the Vice Board Chairperson of the Chinese Association of Artificial Intelligence, a Councilor of the Chinese Institute of Electronics, a Committee Member of the Chinese Committee of Neural Networks, and an Expert of the Academic Degrees Committee of the State Council.

Dan Wang (Student Member, IEEE) received the B.S. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2009. She is currently pursuing the Ph.D. degree with the Key Laboratory of Intelligent Perception and Image Understanding, Ministry of Education, School of Artificial Intelligence, Xidian University, Xi'an, China.

Her research interests include object detection and tracking, and image processing.

Fang Liu (Senior Member, IEEE) received the B.S. degree in computer science and technology from Xi'an Jiaotong University, Xi'an, China, in 1984, and the M.S. degree in computer science and technology from Xidian University, Xi'an, in 1995.

She is currently a Professor with the School of Computer Science, Xidian University. Her research interests include signal and image processing, synthetic aperture radar image processing, multiscale geometry analysis, learning theory and algorithms, optimization problems, and data mining.

Xu Liu (Member, IEEE) received the B.Sc. degree in mathematics and applied mathematics from the North University of China, Taiyuan, China, in 2013, and the Ph.D. degree from Xidian University, Xi'an, China, in 2019.

He is currently a Post-Doctoral Researcher with the Key Laboratory of Intelligent Perception and Image Understanding, Ministry of Education, School of Artificial Intelligence, Xidian University. His current research interests include machine learning and image processing.

Dr. Liu was the Chair of the IEEE Xidian University Student Branch from 2015 to 2019.

Shuyuan Yang (Senior Member, IEEE) received the B.A. degree in electrical engineering and the M.S. and Ph.D. degrees in circuit and system from Xidian University, Xi'an, China, in 2000, 2003, and 2005, respectively.

She is currently a Professor of electrical engineering with Xidian University. Her research interests include machine learning and multiscale geometric analysis.