Pattern Recognition 132 (2022) 108909

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Joint operation and attention block search for lightweight image restoration

Hao Shen[a,b], Zhong-Qiu Zhao[a,b,c,d,*], Wenrui Liao[a,b], Weidong Tian[a,b], De-Shuang Huang[e]

[a] School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China
[b] Intelligent Interconnected Systems Laboratory of Anhui Province (Hefei University of Technology), China
[c] Guangxi Academy of Sciences, China
[d] Intelligent Manufacturing Institute of Hefei University of Technology, China
[e] EIT Institute for Advanced Study, Tongxin Road No.568, Ningbo 315201, Zhejiang, China

A R T I C L E   I N F O

Article history:
Received 29 September 2021
Revised 7 May 2022
Accepted 16 July 2022
Available online 19 July 2022

Keywords:
Image restoration
Neural architecture search
Attention mechanism

A B S T R A C T

Recently, block-based design methods have shown effectiveness in image restoration tasks, which are usually designed in a handcrafted manner and have computation and memory consumption challenges in practice. In this paper, we propose a joint operation and attention block search algorithm for image restoration, which focuses on searching for optimal combinations of operation blocks and attention blocks. Specifically, we first construct two search spaces: operation block search space and attention block search space. The former is used to explore the suitable operation of each layer and aims to construct a lightweight and effective operation search module (OSM). The latter is applied to discover the optimal connection of various attention mechanisms and aims to enhance the feature expression. The searched structure is called the attention search module (ASM). Then we combine OSM and ASM to construct a joint search module (JSM), which serves as the basic module to build the final network. Moreover, we propose a cross-scale fusion module (CSFM) to effectively integrate multiple hierarchical features from JSMs, which helps to mine feature corrections of intermediate layers. Extensive experiments on image super-resolution, gray image denoising, and JPEG image deblocking tasks demonstrate that our proposed network can achieve competitive performance. The source code is available on https://github.com/it-hao/JSNet.

1. Introduction

Image restoration (IR) aims to recover high-quality (HQ) images from low-quality (LQ) images corrupted by various kinds of degradations, which is a classical low-level task and has drawn much attention. However, due to the irreversible nature of the degradation process, the IR task is an ill-posed problem and very challenging.

Recently, deep learning-based methods [1–3] adopt a data-driven manner to remove the possible corruptions by mapping the degraded images to the latent clean versions, which have been widely investigated and achieve promising performance in image super-resolution (SR) [4,5], image denoising (DN) [6], and JPEG image deblocking [7], etc. Among them, the tremendous advances are mainly benefited from the developments of various handcrafted neural network architectures including residual connection [8], dense connection [9], attention mechanism [10], and multi-scale design [5], etc. However, most above methods focus on improving the quality of the restored images, and largely neglect the model complexity and inference speed. For example, MPRNet [3] contains 20M parameters and 760G Multi-Adds when the input image 256×256. This severely restricts the practical use of the CNN-based image restoration methods in the real world.

To overcome the drawback, most methods that concentrate on architecture designs are proposed to reduce model complexity. In these handcrafted designs, some methods [11,12] reduce the number of network parameters by utilizing them recursively, they further improve the reconstruction performance using residual units, memory, or feedback modules but at the cost of running time. Some methods [4,13] utilize cascaded or multi-branch architectures or exploit different types of convolutions to decrease computational burdens and memory cost. Other methods [14,15] combine various attention mechanisms such as channel attention, spatial attention, and non-local attention to better guide feature extraction, thus improving the quality of restored images. Although these methods offer a good compromise in terms of PSNR and model complexity or speed, they also rely on overweight manual

* Corresponding author.
E-mail address: z.zhao@hfut.edu.cn (Z.-Q. Zhao).

https://doi.org/10.1016/j.patcog.2022.108909
0031-3203/© 2022 Elsevier Ltd. All rights reserved.

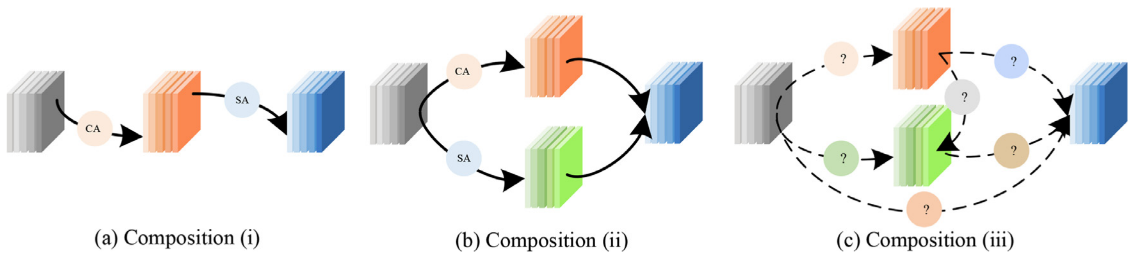

Fig. 1. Comparisons of various connection patterns of attention mechanisms, where CA and SA denote channel attention and spatial attention, respectively. Composition (i) is based on sequential manner, composition (ii) is based on parallel manner, and composition (iii) is our proposed attention search space that contains potential connection patterns of various attentions. It is worth noting that composition (i) and (ii) are special cases of composition (iii).

designs and expert experience, which leads to spending considerable consumption on unnecessarily repetitive designs. For instance, attention-based architectures often employ an attention module to a cell or a block. As shown in Fig. 1(a) and (b), channel attention and spatial attention can be organized in a sequential or parallel manner, which results in the representational capability of the model is subject to previous artificial arrangements. However, a common characteristic is that most of these methods are block-based design mode. This also fully demonstrates the feasibility of this design mode.

Recently, Neural Architecture Search (NAS) algorithm [16,17] has been proposed, which refers to automatically find a desirable neural architecture by using one of the following search strategies, namely, evolutionary algorithm (EA), reinforcement learning (RL), gradient-based methods, etc. Compared with the manually designed architectures, the networks found by NAS algorithms achieve better performance and have fewer parameters. However, EA-based and RL-based search methods [16,17] often face the explosion problem of architecture combination, which is computationally inefficient and time-consuming. In contrast, gradient-based methods [18,19] can effectively reduce the training time and attract wide attention.

Motivated by the search efficiency of the gradient-based NAS [18], effectiveness of attention mechanism, and block-based design mode, we propose a joint operation and attention block search algorithm to hunt for efficient lightweight image restoration networks, namely **J**oint **S**earch **Net**work (**JSNet**). This may be the first attempt towards an automatic search of desirable attention-based neural architectures in the image restoration field. Specifically, the method consists of two search spaces: operation block search space and attention block search space. The operation block search space aims to discover the optimal building block at the appropriate location, thus obtaining the best combination of various types of operations with as few parameters as possible. We called the searched structure operation search module. There are two manners to utilize attention mechanisms: one is to embed various independent attention blocks in the operation block search space to form a multi-branch structure, and the other is to build an attention block search space that can progressively emphasize meaningful features. It is obvious that the former just add more branches to the connection as shown in Fig. 1(b), and the latter helps to discover potential forms of connection and inner correlation of various attention mechanisms. Therefore, constructing an attention block search space structure, depicted in Fig. 1(c), should be a better choice. We call the searched structure attention search module. Finally, taking two modules together, we design a joint search module (JSM) to build the final network, as shown in Fig. 2(a).

In addition, we design a cross-scale fusion module (CSFM), which can effectively integrate all features extracted by JSMs. The benefits of this module are twofold. First, the module adopts butterfly structure [20], which can generate various linear combinations of multi-scale features, thus enhancing the information communication between different types of features. Second, in the specific IR task (*i.e.*, image super-resolution), the number of feature maps in sub-pixel convolution affects both the computational complexity and performance of the network. However, the proposed CSFM has reduced the number of feature maps before the features are fed into a high-dimensional space, thus achieving a great trade-off between performance and the number of parameters. The detailed structure of CSFM is shown in Fig. 4(b).

Part of our previous work has been reported in [21]. Compared to the preliminary version, in this manuscript, we have made improvements in the following aspects: (1) The initial version merely solves the image super-resolution task. This paper, however, deals with multiple image restoration tasks using the NAS-based algorithm. Besides, we construct a larger attention search space in the search phase and stack more searched blocks to formulate the final network for the gray image denoising and JPEG image deblocking tasks. (2) We perform a more comprehensive survey of existing related works, *e.g.*,adding the review of image restoration and neural architecture search works in Section 2. (3) We conduct more empirical evaluations and more experimental analysis in Section 4. In addition, more experimental details and more comparison experiments are provided. In brief, although there are some literal overlaps, the new content in this manuscript makes the proposed search method much more general, comprehensive, and convincing. In summary, the main contributions of this paper are listed as follows:

(1) Based on the differentiable architecture search, we propose a joint operation and attention block search algorithm that enables operation type search and attention mechanism search simultaneously. It provides more room for searching for better networks of image restoration.
(2) We propose the operation block search space and attention block search space to find the optimal combination of operation block and attention block, respectively. Taking these two searched modules together, we construct a joint search module to formulate the final network.
(3) We propose the cross-scale fusion module (CSFM) based on butterfly structures and multi-scale features, which can be embedded in the searched network, thus helping to expand representation space for achieving more powerful networks.

2. Related work

2.1. CNN-based image restoration

To date, many image restoration methods [5,22] have been proposed, which achieve remarkable performance in various applications. For instance, SRCNN [23] firstly apply three convolutional layers to image super-resolution, which achieves superior performance compared with conventional methods. ARCNN [7] uses a

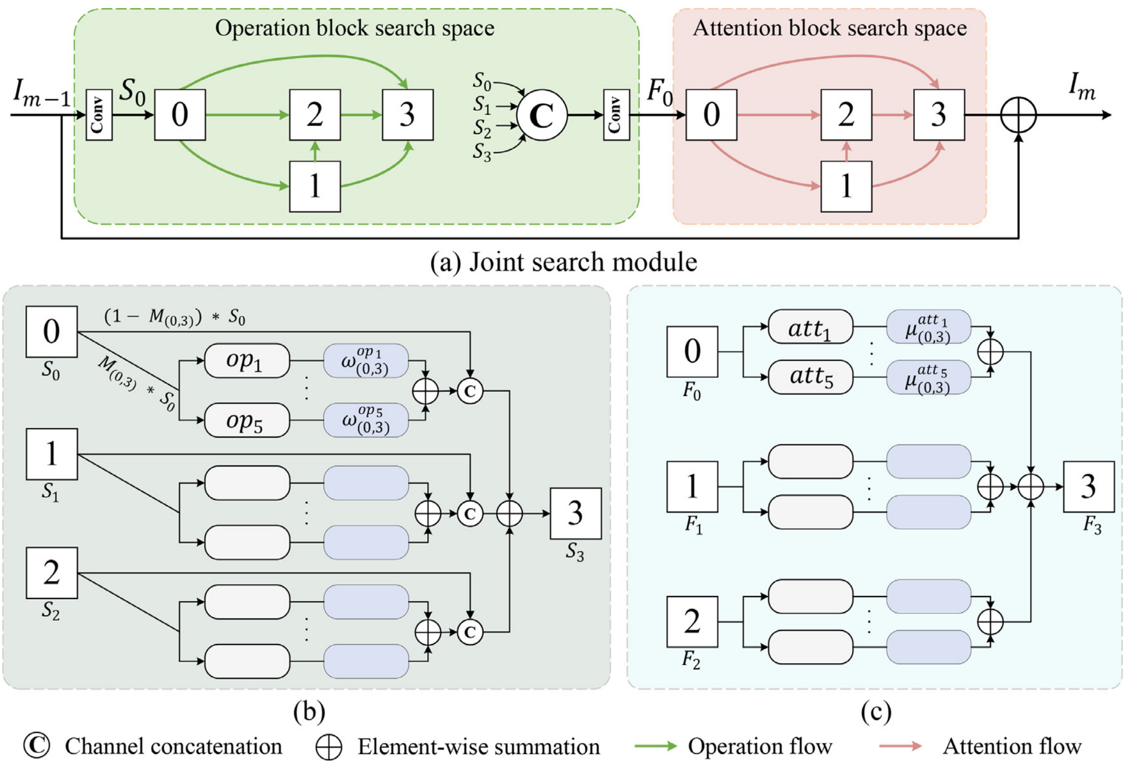

Fig. 2. (a) The architecture of joint search module. As an example, (b) and (c) indicate how the features are propagated to the third node in the operation block search space and the attention block search space, respectively. Each node with an index outputs a latent representation (*e.g.*, feature maps). In operation block search space, an operation flow from the i-th node to j-th node is formulated by weighting candidate operations (denoted as op_k) with a set of hyper-parameters, namely, $\{\omega^{op_k}_{(i,j)}\}$. We only sample part of input features with a mask $M_{i,j}$ in the channel dimension. In attention block search space, the mixing process of our candidate attentions (denoted as att_k) with $\mu^{att_k}_{(i,j)}$ is denoted as an attention flow from the i-th node to j-th node.

four-layer convolutional network for JPEG image deblocking. To stack more convolutional layers and ease the difficulty of training a deep network, VDSR [8] and DRCN [11] employ gradient clipping, residual learning, or recursive supervision to tackle image super-resolution tasks. FFDNet [6] constructs a flexible network to deal with noise on different levels, as well as spatially variant noise. CDNet [24] investigates the potentials of complex-valued CNNs for image denoising. YOLY [25] proposes an unsupervised and untrained neural network for image dehazing for the first time. However, these methods are for specific image restoration tasks. Unlike these methods, [14,15] design a unified model framework that can be generalized to different image restoration tasks. NLRN [14] incorporates non-local operations into a recurrent neural network for image restoration. A-CubeNet [15] combine multiple attentions to enhance feature representations. AirNet [26] designs a unified framework to recover images from multiple corruptions in an all-in-one fashion. Although the overall performance of image restoration has dramatically boosted, with it come the increases of the number of parameters and the amount of computation. Besides, these hand-crafted networks are labor-intensive to seek an optimal architecture, while the image restoration performance is sensitive to neural architecture according to the advances in recent years.

2.2. Neural architecture search

Search strategy. NAS aims to automatically discover satisfactory network architecture [16,17] by using various search strategies such as evolutionary algorithm (EA), reinforcement learning (RL), gradient-based methods, etc. Some early works [16,17] mainly adopt EA for optimizing neural architecture and parameters, which obtain the best architecture via the iterative crossovers and mutations of population. RL-based algorithms, as an alternative, adopt policy gradients [27] and Q-learning techniques to train a recurrent neural network that acts as a meta-controller to generate potential architectures by exploring a predefined search space. However, both EA-based and RL-based methods are inefficient in search and often require a large number of computations and training time. To solve this issue, recent works have been focused on gradient-based methods, such as DARTS [18] and PC-DARTS [19]. The core idea of this method is to relax the discrete and non-differentiable architecture to a continuous and differentiable surrogate, thus allowing the efficient search of the architecture using gradient descent. Many recent works including ours and [28,29] are inspired by this differentiable NAS.

NAS for image restoration. To date, some works apply the NAS strategy to image restoration. E-CAE [30] employs EA to search for architecture autoencoders for image impainting and denoising. FALSR [17] treats the super-resolution task as a constrained multi-objective optimization problem and utilizes RL and EA to search lightweight models. ESRN [16] constructs multiple handcrafted efficient residual dense blocks and then resorts to EA to search for the optimal network architecture in these given building blocks for image super-resolution. However, all these methods mentioned above require enormous computational resources and take a large amount of GPU time for searching. Two more related works are HiNAS [28] and CLEARER [29] that employ gradient-based search strategy. HiNAS employs operations with adaptive receptive field to build a flexible search space then applies differentiable architecture search to image denoising and deraining with less search time. CLEARER designs a multi-scale search space that contains different task-flexible modules and then leverages the differentiable search strategy to search for a super-network.

Motivated by the superior performance and search efficiency, we also employ the gradient-based approach as our search strat-

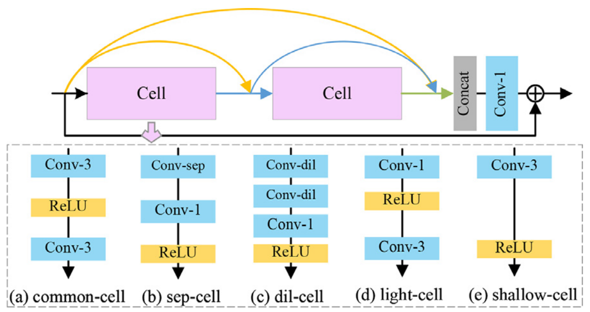

Fig. 3. The architecture of the candidate operation block and its components. The block contains two basic cells and is constructed based on residual dense block [9]. The 'Conv-3' and 'Conv-1' denote the 3 × 3 convolution and 1 × 1 convolution, respectively, 'Conv-sep' denotes the separated convolution, and 'Conv-dil' denoted the dilated convolution.

egy but with slight differences. Firstly, both HiNAS and CLEARER are differentiable. However, our work is closely related to PC-DARTS [19], which samples a subset of channels into the operation selection block while bypassing the rest directly in a shortcut, therefore performing a more efficient search. Secondly, we construct two search spaces to find optimal operation blocks and optimal attention blocks, respectively. HiNAS can search for the operation of each layer. However, CLEARER only explores when to fuse low-level and high-level features based on three task-flexible modules, and thus neglects to search the concrete module design. This allows CLEARER to search for larger network within the same search time, but also leads to a sub-optimal solution.

3. Proposed method

3.1. Constructing the joint search module

Many previous works [9,10] focus on designing efficient building blocks to extract features. These blocks typically consist of a series of convolution layers and specific attention mechanisms, thus neglecting the connections between different types of convolutions or various attention mechanisms, resulting in sub-optimal results. In this section, we utilize neural architecture search algorithms to obtain the optimal operation search module (OSM) and attention search module (ASM) from two search spaces. As shown in Fig. 2(a), a joint search module (JSM) is proposed based on these two modules and residual learning strategy. We denote I_{m-1} and I_m as the input and output of the JSM at the m-th layer. The process of feature propagation in the module can be formulated as:

$$F_0 = H_{OSM}(I_{m-1}), \tag{1}$$

$$I_m = H_{ASM}(F_0) + I_{m-1}, \tag{2}$$

where H_{OSM} and H_{ASM} denote the function of OSM and ASM respectively, F_0 denotes the output and input of the m-th OSM and the m-th ASM.

3.1.1. Operation block search

For the image restoration task, we redesign a series of candidate operation blocks. Inspired by the effectiveness of the residual dense block (RDB) [9], we construct several novel operation blocks based on the following two aspects: (1) the convolution number in RDB is set to 2, and the growth rate [9] is set to 16; (2) the common convolution layer is replaced with various cells such as common-cell, sep-cell, dil-cell, light-cell, and shallow-cell to ensure effective and lightweight network. Therefore, apart from the skip connection and none [18], there are five types of candidate operation blocks, as depicted in Fig. 3.

Then, we first adopt a 1 × 1 convolution layer to adaptively control the dimensions of feature maps. Given the input feature I_{m-1}, we have:

$$S_0 = W_{conv}(I_{m-1}), \tag{3}$$

where W_{conv} is the weight of 1 × 1 convolution layer. Then the S_0 will be inputted into the operation block search space. Here, we adopt the cell-based [27] manner to construct search space, where the cell is defined by a directed acyclic graph with several nodes. In this search space, we denote N_1 operation nodes with the index from 0 to $N_1 - 1$, each node takes the outputs of all previous nodes as input and then produces new feature maps. Note that the output of the node with index 0 equals S_0 (so-called feature maps). Taking the j-th node as an example, the output of this node is calculated as follows:

$$S_j = \sum_{i=0}^{j-1} O_{(i,j)}(S_i), 0 < j < N_1, \tag{4}$$

where S_j is the output of the j-th node. $O_{(i,j)}(\emptyset)$ represents the operation flow that transforms S_i from the i-th node to j-th node, where $i < j$. Let $\mathcal{O}$ be the set of candidate operations, every operation $o \in \mathcal{O}$ from i-th node to j-th node has been allocated an architecture parameter $\alpha^o_{i,j}$. We compute the architecture weight adopting $softmax$ function for every operation from i-th node to j-th node:

$$\omega^o_{(i,j)} = \frac{\exp\left\{\alpha^o_{i,j}\right\}}{\sum_{o' \in \mathcal{O}} \exp\left\{\alpha^{o'}_{i,j}\right\}}. \tag{5}$$

Here, to obtain the output of each node, we adopt the channel sampling strategy [19] to sample a subset of channels into the operation flow:

$$O_{(i,j)}(S_i) = [(1 - M_{(i,j)}) * S_i, \sum_{k=1}^{R} \omega^{op_k}_{(i,j)} \cdot op_k(M_{(i,j)} * S_i)], \tag{6}$$

where $[\cdots]$ denotes the channel concatenation operation, $\{op_1, op_2, \cdots, op_R\}$ denotes R possible operation blocks, and $\omega^{op_k}_{(i,j)}$ corresponds to the weight of operation op_k from i-th node to j-th node. $M_{(i,j)}$ denotes the mask which assigns 1 to the selected channels and 0 to the remaining ones. Therefore, $M_{(i,j)} * S_i$ and $(1 - M_{(i,j)}) * S_i$ represent the selected and remaining channels, respectively. Finally, the outputs of all nodes are fused by the concatenation operation followed by a 1 × 1 convolution layer as follows:

$$F_0 = W_{conv}[S_0, S_1, \cdots, S_{N_1 - 1}], \tag{7}$$

where F_0 denotes the output of operation block, W_{conv} is the weight of 1 × 1 convolution layer.

3.1.2. Attention block search

Previous works [10,31] employ different attention mechanisms to a basic building block to enhance the representational capability of models. These methods require subtle handcrafted designs. In this subsection, we construct an attention block search space, as illustrated in Fig. 2(a). The search space is a directed acyclic graph containing a sequence of N_2 nodes. Each node is a potential representation of features, and each directed edge is regarded as an attention flow. Similar to how we compute the architecture weight of each operation block above, we obtain the probability of each attention block using a $softmax$ function overall candidate attention blocks:

$$\mu^o_{(i,j)} = \frac{\exp\left\{\beta^o_{i,j}\right\}}{\sum_{o' \in \mathcal{O}} \exp\left\{\beta^{o'}_{i,j}\right\}}. \tag{8}$$

Taking the flow from the i-th node to the j-th node for example, where $i < j$, the core idea of an attention flow is to formulate

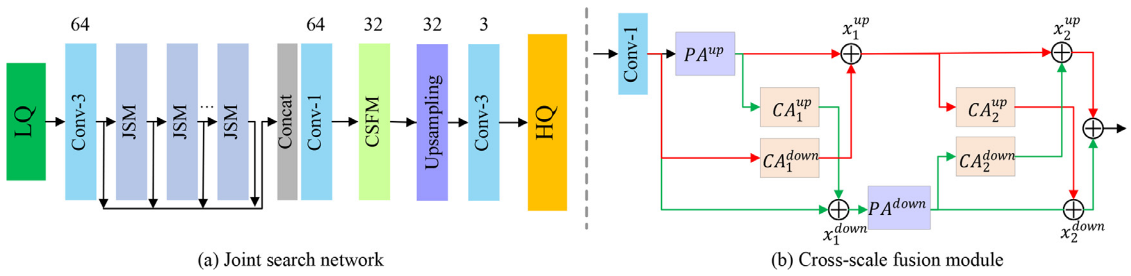

Fig. 4. (a) The overall architecture of the proposed joint search network (JSNet). The 'Upsampling' consists of one 3 × 3 convolution followed by one sub-pixel [36] convolution layer and is specific for the image super-resolution task. (b) The proposed cross-scale fusion module (CSFM), the '*PA*' denotes the pixel attention proposed in [34], the '*CA*' denotes channel attention proposed in [10].

Table 1
The list of candidate attention blocks to be searched.

Type	Channel-wise	Pixel-wise	Spatial
1	Channel Attention (CA) [10]	Pixel-wise Attention (PA) [34]	Spatial Attention (SA) [32]
2	Contrast-aware Channel Attention (CCA) [13]	Cost-effective Attention (CEA) [35]	Enhanced Spatial Attention (ESA) [33]

the features propagated from i-th node to j-th node as a weighted summation of T candidate attentions:

$$A_{(i,j)}(F_i) = \sum_{k=1}^{T} \mu_{(i,j)}^{att_k} \cdot att_k(F_i), \tag{9}$$

where $\{att_1, att_2, \cdots, att_T\}$ denotes T possible attention types. F_i denotes the output of the i-th attention node. We mix candidate attentions in a continuous relaxation way by weighting $att_k(\emptyset)$ with $\mu_{(i,j)}^{att_k}$. The output of each node in attention block is the summation of all associated attention flows, which can be denoted as:

$$F_j = \sum_{i=0}^{j-1} A_{(i,j)}(F_i), 0 < j < N_2. \tag{10}$$

Note that the output of the first node equals to F_0, therefore, the output of overall attention block is denoted as F_{N_2-1}. Due to the use of residual structure strategy in JSM, we compute the output by summing F_{N_2-1} with I_{m-1}.

Consider that lean blocks are essential to design fast and lightweight image restoration networks. Therefore, we introduce a variety of lean and effective attention mechanisms to build attention block search space. All these attention mechanisms can be divided into three categories, as shown in Table 1. The first type is based on the channel level, such as channel-wise attention [10,13]. The second type is spatial attention, including SA [32] and ESA [33]. The third type is based on pixel-wise, such as PA [34] and CEA [35]. Besides, skip connection is added to the attention search space. These attention mechanisms are all convenient to be embedded in our attention search block, as shown in Fig. 2(a). It is noted that CCA and SA are only used in gray image denoising and JPEG image deblocking tasks. Therefore, apart from skip connection, there are four and six candidate attention operations in attention block search space for image SR and other image restoration tasks, respectively.

3.2. From search to evaluate

3.2.1. Overall search procedure

Benefiting from the continuously relaxed representation of the search space, we can search for the super-network by updating the architecture parameters α, β and weight parameters θ of the network using gradient descent algorithms such as ADAM. We train the network with the following L_1 loss:

$$(\alpha, \beta, \theta) = \underset{\alpha,\beta,\theta}{\operatorname{argmin}} \sum_{i=1}^{N} \|\mathcal{F}(\boldsymbol{y}_i; \alpha, \beta, \theta) - \boldsymbol{x}_i\|_1, \tag{11}$$

where y_i and x_i are the i-th pair of low quality and high quality image patches respectively, and $\mathcal{F}(\boldsymbol{y}_i; \alpha, \beta, \theta)$ denotes the reconstructed image patch. To ensure better search results, we split the search procedure into two stages. In the first stage, we only optimize the weights (kernels in convolution layers) for enough epochs to avoid the performance being too bad, we call it a warm-up. In the second stage, we activate the architecture search. We alternatively optimize the weights parameters by descending $\nabla_\theta \mathcal{L}_{\text{train}}(\theta, \alpha, \beta)$ on the training set, and optimize the architecture parameters by descending $\nabla_{\alpha,\beta} \mathcal{L}_{\text{val}}(\theta, \alpha, \beta)$ on the validation set. After the search stage, for each operation flow $O_{(i,j)}(\emptyset)$ and attention flow $A_{(i,j)}(\emptyset)$, we select the operation block and attention block with the maximum value which is determined by ω and μ, respectively. Thereby, we obtain the exact architecture of the joint search module.

3.2.2. Deriving the final networks

Based on searched modules above, we build a lightweight joint search network (JSNet) for image restoration tasks, as illustrated in Fig. 4(a). To effectively integrate all features extracted by JSMs, we propose a cross-scale fusion module (CSFM) which reduces the channel dimension without losing contextual information of deep features. As shown in Fig. 4(b), our CSFM is a multi-branch structure with pixel-attention modules [34] in each branch and contains several channel attention modules between two branches. We employ a 1 × 1 convolution layer at the beginning to reduce feature dimensions by half. Let x denote the features after the reduction operation. We can denote the linear combination of feature propagation as the following equations:

$$\begin{aligned}
x_1^{up} &= PA^{up}(x) + CA_1^{down}(x),\\
x_1^{down} &= CA_1^{up}(PA^{up}(x)) + x,\\
x_2^{up} &= CA_2^{down}(PA^{down}(x_1^{down})) + x_1^{up},\\
x_2^{down} &= CA_2^{up}(x_1^{up}) + PA^{down}(x_1^{down}),
\end{aligned} \tag{12}$$

where the superscripts and subscripts denote the position and the order that appeared on the module. Notably, we replace the 3 × 3 convolution with 5 × 5 convolution in the PA^{up} module to extract multi-scale spatial information. Finally, x_2^{up} and x_2^{down} are summed to obtain the outputs of the module. The elaborate settings of networks will be introduced in Section 4.1.

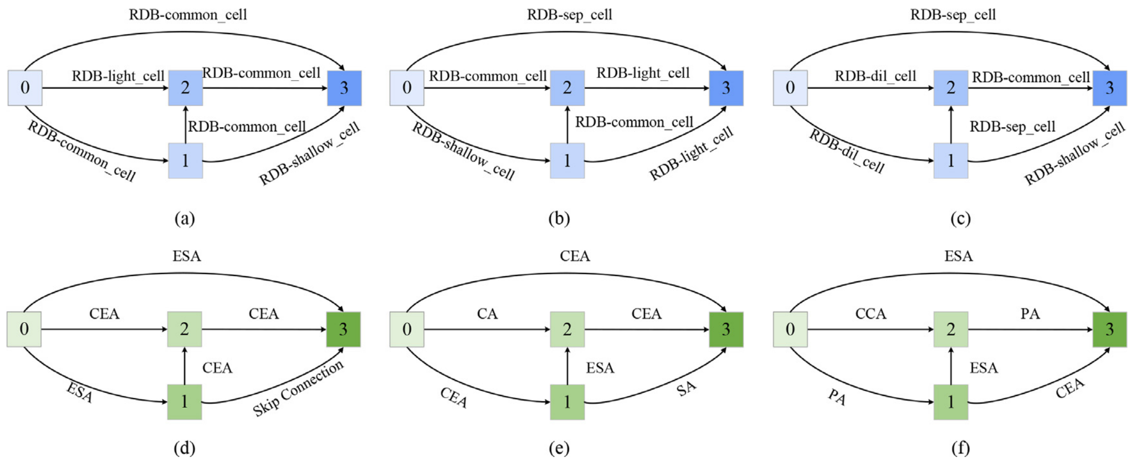

Fig. 5. The derived operation search module for (a) image super-resolution, (b) gray image denoising, and (c) JPEG image deblocking. The derived attention search module for (d) image super-resolution, (e) gray image denoising, and (f) JPEG image deblocking.

4. Experiments

4.1. Experimental settings

4.1.1. Datasets

We apply our JSNet to three classical image restoration tasks: image super-resolution, gray image denoising, and JPEG image deblocking. The DIV2K [37] dataset is used to train all of our models. The noisy images are generated by adding white Gaussian noise to the corresponding clean image with $\sigma = 10, 30, 50, 70$. The compressed images are generated by using Matlab JPEG encoder with JPEG quality setting $q = 10, 20, 30, 40$. As for image super-resolution, we follow the same setting as CARN [4]. Set5 [38], Set14 [39], BSD100 [40], and Urban100 [41] are adopted as the test datasets. For the gray image denoising, we follow the same setting as IRCNN [42]. BSD68 [40] and Kodak24 are used as the test datasets. For JPEG image deblocking, we follow the same setting as ARCNN [7]. LIVE1 [43] and Classic5 [44] are applied as the test datasets. We adopt the mean PSNR and/or SSIM to evaluate the results.

4.1.2. Search settings

The search network stacks four JSMs to construct the overall network and each search space has four nodes. The number of channels in candidate operation blocks and attention blocks is set to 16 and 64, respectively. The input size of the LQ image is set to 64×64 and the minibatch size is set as 16. During the search stage, 800 training images $\mathbb{D}_{train}$ from DIV2K are used to optimize the weights, and 100 validation images $\mathbb{D}_{val}$ from DIV2K are used to optimize the architecture parameters. All datasets are augmented by flipping horizontally or vertically and rotating 90°. We optimize the θ, α, β parameters with two ADAM optimizers. For weight parameter θ, the learning rate is set to 10^{-4}, the momentum parameter and exponential moving average parameter are set as (0.9,0.999) and the weight decay is set to 0. For architecture parameters α and β, the learning rate is set to 10^{-3}, the momentum parameter and exponential moving average parameter are set as (0.9,0.999) and the weight decay is set to 10^{-3}. The learning rates of the warm-up process and searching process are both set to 10^{-4}. The warm-up and overall search processes take about 2×10^4 iterations and 4×10^5 iterations, respectively.

4.1.3. Training settings

The final network architecture for image super-resolution consists of five JSMs, one upsampling module, and one CSFM. However, for other image restoration tasks, we replace the upsampling module with one JSM, which means there are six JSMs across the final network. For retraining the final network, we use dataset $\mathbb{D}_{train}$ with the same data augmentation as the searching stage. We train the model in 10^6 iterations and randomly select 16 LQ images sized by 64×64 as the inputs. The ADAM algorithm with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$ is adopted to optimize the network. The learning rate is set as 2×10^{-4} and then decreases to half for every 2×10^5 iterations. Our network is implemented by PyTorch framework with NVIDIA 2080Ti GPU.

4.2. Searched results

The derived operation search module and attention search module for various image restoration tasks are shown in Fig. 5. We can observe that:

(1) The structures of our derived modules for three different image restoration tasks are quite different from each other. However, many previous works focus on utilizing unified frameworks to solve various image restoration tasks, thus resulting in worse performance.
(2) We provide enough candidate blocks of different sizes to choose from, including five operation blocks and six attention blocks in the operation block search space and attention block search space, as mentioned in Fig. 3 and Table 1. Therefore, the searched structures can achieve a better trade-off between performance and model complexity.
(3) The fact that searched modules select proper operation blocks and attention blocks at suitable positions instead of simply integrating complex blocks indicates the proposed search algorithm is effective.

4.3. Ablation study

4.3.1. Benefits of searching for operation blocks

In this section, to evaluate the benefits of searching for operation blocks, we apply the proposed method to the image super-resolution task. In detail, we first obtain the operation search module (OSM) by operation block search space. The searched structure is shown in Fig. 5(a). Based on the searched results, we replace the searched operation blocks between paired nodes in the OSM with one based on common_cells, light_cells, and shallow_cells, respectively. We denote these operation blocks as RDB-common_cell, RDB-light_cell, and RDB-shallow_cell. The detailed structures of several cells are shown in Fig. 3. Here, we do not add ASM and CSFM in the network in order to evaluate the performance of this

Table 2
Comparisons of the number of parameters and mean values of PSNR evaluated on various models. We record the best results for ×4 image SR in 500 epochs.

Operation block	Parameters	Set5	Set14	BSD100	Urban100
RDB-common_cell	631K	31.89	28.42	27.44	25.64
RDB-light_cell	484K	31.74	28.33	27.38	25.48
RDB-shallow_cell	520K	31.80	28.37	27.41	25.57
RDB-searched	588K	**31.91**	**28.44**	**27.47**	**25.69**

Table 3
Comparison of the number of parameters and mean values of PSNR obtained by using various attention mechanism. We record the best results for ×4 image SR in 500 epochs.

Attention block	Parameters	Set5	Set14	BSD100	Urban100
RB	610K	31.60	28.26	27.34	25.37
RB-CA	615K	31.67	28.27	27.36	25.44
RB-SA	617K	31.74	28.40	27.40	25.53
RB-PA	627K	31.89	28.42	27.44	25.64
RB-CCA	617K	31.69	28.30	27.41	25.47
RB-CEA	620K	31.72	28.31	27.46	25.57
RB-ESA	623K	31.78	28.45	27.46	25.60
RB-ASM	651K	**31.95**	**28.48**	**27.48**	**25.80**

module separately. The comparison results for ×4 image super-resolution on several datasets are lists in Table 2.

From Table 2, it is obviously observed that the results of RDB-searched outperform the results using other three types of operation blocks. Compared with RDB-common_cell, our RDB-searched could improve the PSNR by 0.02dB, 0.02dB, 0.03dB and 0.05dB on Set5, Set14, BSD100 and Urban100 with fewer parameters. It indicates that the operation search module has found the optimal architecture from candidate operation blocks.

4.3.2. Benefits of searching for attention blocks

We obtain the attention search module (ASM) by attention block search space. To evaluate the effect of searched ASM, we use the search network as the basic network and compare the performance in several classical attention mechanisms. As done in PAN [34], we also replace the JSM with the same number of residual blocks (RB), residual blocks with channel attention (RB-CA), residual blocks with spatial attention (RB-SA), residual blocks with pixel attention (RB-PA), residual blocks with contrast-aware channel attention (RB-CCA), residual blocks with cost-effective attention (RB-CEA), residual blocks with enhanced spatial attention (RB-ESA), and residual blocks with our attention module (RB-ASM), respectively. Note that all mentioned attention layers are embedded in the tail of RB. The quantitative comparison results are reported in Table 3. From these results, it is obvious that our RB-ASM outperforms the other attention methods and significantly improves the PSNR value compared with the baseline. This fully demonstrates that we have found optimal attention combinations based on the attention block search space.

4.3.3. Benefits of JSM and CSFM

To make a fair comparison, we replace JSM in the search network with the same number of residual blocks (RB) to construct baseline model. In Table 4, the baseline achieves the lowest PSNR value on Set14 (×4). When JSM or CSFM is adopted, the PSNR values are increased by + 0.25 dB and + 0.08 dB compared with the baseline on Set14 (×4), respectively. Note that the model only with CSFM has 1/3 fewer parameters than the baseline. Moreover, the results of the last column demonstrate that the combination of our proposed JSM and CSFM achieve a comprehensive balance of the number of parameters and performance.

In Table 5, we further study the setting of multi-scale in the CSFM. When the size of convolutional kernel in PA^{up} and PA^{down} is set 3 × 3 and 5 × 5, the PSNR value is inferior to the result of cross-scale setting. This fully demonstrates the proposed CSFM integrates the multi-scale spatial information, boosting the representational capability of the network.

Table 4
Investigations of JSM and CSFM. We record the best PSNR for ×4 image super-resolution in 500 epochs.

Name	1	2	3	4
JSM	✗	√	✗	√
CSFM	✗	✗	√	√
Parameters	610K	629K	411K	430K
PSNR on Set5	31.60	31.98	31.79	**32.04**
PSNR on Set14	28.27	28.52	28.35	**28.54**

Table 5
Investigation of scale setting in cross-scale fusion module.

Cross-scale fusion module			
Scale	all 3 × 3	all 5 × 5	cross-scale
PSNR on Set5	31.99	31.92	32.04
PSNR on Set14	28.50	28.47	28.54

Table 6
Comparisons of running time (seconds) of various image SR methods on HR images of sizes 256 × 256, 512 × 512, and 1024 × 1024 for ×2 scaling factor.

Size	256 × 256	512 × 512	1024 × 1024
CARN [4]	0.018	0.032	0.126
SRFBN-S [45]	0.034	0.054	0.211
IMDN [13]	0.026	0.039	0.137
FALSR [17]	0.074	0.090	0.421
LAPAR-A [46]	0.045	0.062	0.287
JSNet (Ours)	0.031	0.056	0.244

4.4. Comparisons with the state-of-the-art methods

We first make a brief complexity analysis of some lightweight image super-resolution methods and then compare the proposed JSNet with some state-of-the-art methods in image super-resolution, gray image denoising, and JPEG image deblocking.

4.4.1. Computational complexity

To obtain a more comprehensive understanding of the model complexity, we compare our JSNet against various benchmark methods in terms of the Multi-Adds and the number of the parameters on the Set5 dataset ×2 dataset. We assume the high-resolution image size to be 720p (1280 × 720) to calculate Multi-Adds. As shown in Fig. 6, our JSNet outperforms all state-of-the-art models that have less than 1M parameters. Especially, JSNet has fewer network parameters and Multi-Adds than IMDN, LAPAR-A, FALSR-A, and ESRN-F, but our method outperform all these models. As shown in Table 6, we also use official codes of the compared methods to obtain the average running time on various HR sizes. Each result is an average value obtained by repeating five experiments to ensure a fair comparison. Although our proposed JSNet cannot achieve the best result in each metric, our method makes a better trade-off between performance and model complexity.

4.4.2. Image super-resolution

We compare our JSNet with ten representative image super-resolution methods: FSRCNN [47], VDSR [8], DRRN [48], MemNet [12], IDN [49], CARN [4], IMDN [13], LAPAR [46], FALSR [17], and ESRN [16]. Note that, the first nine methods are hand-crafted architectures and the last two are NAS-based methods. Table 7 shows quantitative comparisons for scaling factors ×2, ×3, and ×4 on two commonly-used metrics: peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM). FALSR

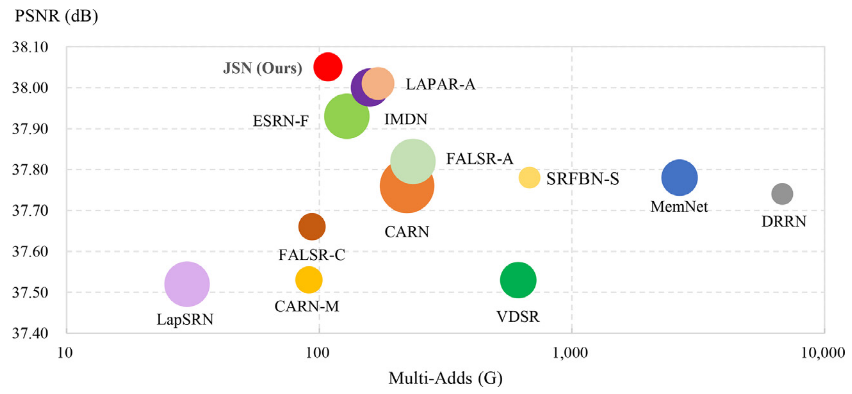

Fig. 6. Comparison between our proposed JSNet and other lightweight methods on Set5 for ×2 setting. Circle sizes are set proportional to the number of parameters.

Table 7
Comparisons on multiple benchmark datasets for lightweight image super-resolution networks. The Multi-Adds is calculated corresponding to the 1280 × 720 high-resolution image. The best results are emphasized with **bold**.

Methods	Scale	Train Data	Parameters↓	Multi-Adds↓	Set5 PSNR/SSIM↑	Set14 PSNR/SSIM↑	BSD100 PSNR/SSIM↑	Urban100 PSNR/SSIM↑
				Handcrafted methods				
FSRCNN [47]	×2	G100+Yang91	12K	6G	37.05/0.9560	32.66/0.9090	31.53/0.8920	29.88/0.9020
VDSR [8]		G100+Yang91	665K	613G	37.53/0.9590	33.05/0.9130	31.90/0.8960	30.77/0.9140
DRRN [48]		G100+Yang91	297K	6,797G	37.74/0.9591	33.23/0.9136	32.05/0.8973	31.23/0.9188
MemNet [12]		G100+Yang91	677K	2,662G	37.78/0.9597	33.28/0.9142	32.08/0.8978	31.31/0.9195
IDN [49]		G100+Yang91	579K	125G	37.85/0.9598	33.58/0.9178	32.11/0.8989	31.95/0.9266
CARN [4]		DIV2K	1,592K	223G	37.76/0.9590	33.52/0.9166	32.09/0.8978	31.92/0.9256
SRFBN-S[45]		DIV2K+Flickr2K	282K	680G	37.78/0.9597	33.35/0.9156	32.00/0.8970	31.41/0.9207
IMDN [13]		DIV2K	694K	159G	38.00/0.9605	**33.63**/0.9177	**32.19**/0.8996	**32.17**/0.9283
LAPAR-A [46]		DIV2K+Flickr2K	548K	171G	38.01/0.9605	33.62/**0.9183**	**32.19/0.8999**	32.10/0.9283
				NAS-based methods				
FALSR-A [17]	×2	DIV2K	1,021K	235G	37.82/0.9595	33.55/0.9168	32.12/0.8987	31.93/0.9256
FALSR-B [17]		DIV2K	326K	75G	37.61/0.9585	33.29/0.9143	31.97/0.8967	31.28/0.9191
FALSR-C [17]		DIV2K	408K	94G	37.66/0.9586	33.26/0.9140	31.96/0.8965	31.24/0.9187
ESRN-F [16]		DIV2K	1,019K	129G	37.93/0.9602	33.56/0.9171	32.16/0.8996	31.99/0.9276
ESRN-V [16]		DIV2K	324K	73G	37.85/0.9600	33.42/0.9161	32.10/0.8987	31.79/0.9248
JSNet (Ours)		DIV2K	476K	109G	**38.05/0.9608**	**33.63**/0.9180	**32.19**/0.8997	**32.17/0.9284**
				Handcrafted methods				
FSRCNN [47]	×3	G100+Yang91	12K	5G	33.18/0.9140	29.37/0.8240	28.53/0.7910	26.43/0.8080
VDSR [8]		G100+Yang91	665K	613G	33.67/0.9210	29.78/0.8320	28.83/0.7990	27.14/0.8290
DRRN [48]		G100+Yang91	297K	6,797G	34.03/0.9244	29.96/0.8349	28.95/0.8004	27.53/0.8378
MemNet [12]		G100+Yang91	677K	2,662G	34.09/0.9248	30.00/0.8350	28.96/0.8001	27.56/0.8376
IDN [49]		G100+Yang91	588K	56G	34.24/0.9260	30.27/0.8408	29.03/0.8038	27.99/0.8489
CARN [4]		DIV2K	1,592K	119G	34.29/0.9255	30.29/0.8407	29.06/0.8034	28.06/0.8493
IMDN [13]		DIV2K	703K	72G	34.36/0.9270	30.32/0.8417	29.09/0.8046	**28.17**/0.8519
SRFBN-S[45]		DIV2K+Flickr2K	376K	832G	34.20/0.9255	30.10/0.8372	28.96/0.8010	27.66/0.8415
LAPAR-A [46]		DIV2K+Flickr2K	594K	114G	34.36/0.9267	30.34/0.8421	**29.11/0.8054**	28.15/**0.8523**
				NAS-based methods				
ESRN-F [16]	×3	DIV2K	1,019K	72G	34.32/0.9268	30.35/0.8410	29.09/0.8046	28.11/0.8512
ESRN-V [16]		DIV2K	324K	36G	34.23/0.9262	30.27/0.8400	29.03/0.8039	27.95/0.8481
JSNet (Ours)		DIV2K	522K	53G	**34.37/0.9272**	**30.37/0.8424**	29.09/0.8047	28.16/0.8519
				Handcrafted methods				
FSRCNN [47]	×4	G100+Yang91	12K	5G	30.72/0.8660	27.61/0.7550	26.98/0.7150	24.62/0.7280
VDSR [8]		G100+Yang91	665K	613G	31.35/0.8830	28.02/0.7680	27.29/0.7260	25.18/0.7540
DRRN [48]		G100+Yang91	297K	6,797G	31.68/0.8888	28.21/0.7720	27.38/0.7284	25.44/0.7638
MemNet [12]		G100+Yang91	677K	2,662G	31.74/0.8893	28.26/0.7723	27.40/0.7281	25.50/0.7630
IDN [49]		G100+Yang91	600K	32G	31.99/0.8928	28.52/0.7794	27.52/0.7339	25.92/0.7801
CARN [4]		DIV2K	1,592K	91G	32.13/0.8937	28.60/0.7806	27.58/0.7349	26.07/0.7837
SRFBN-S[45]		DIV2K+Flickr2K	483K	1,037G	31.98/0.8923	28.45/0.7779	27.44/0.7313	25.71/0.7719
IMDN [13]		DIV2K	715K	41G	**32.21**/0.8948	28.58/0.7811	27.56/0.7353	26.04/0.7838
LAPAR-A [46]		DIV2K+Flickr2K	659K	94G	32.15/0.8944	**28.61/0.7818**	**27.61/0.7366**	**26.14/0.7871**
				NAS-based methods				
ESRN-F [16]	×4	DIV2K	1,019K	41G	32.15/0.8940	28.59/0.7804	27.59/0.7354	26.11/0.7851
ESRN-V [16]		DIV2K	324K	21G	31.99/0.8919	28.49/0.7779	27.50/0.7331	25.87/0.7782
JSNet (Ours)		DIV2K	513K	36G	**32.21/0.8949**	28.60/0.7812	27.59/0.7355	26.04/0.7839

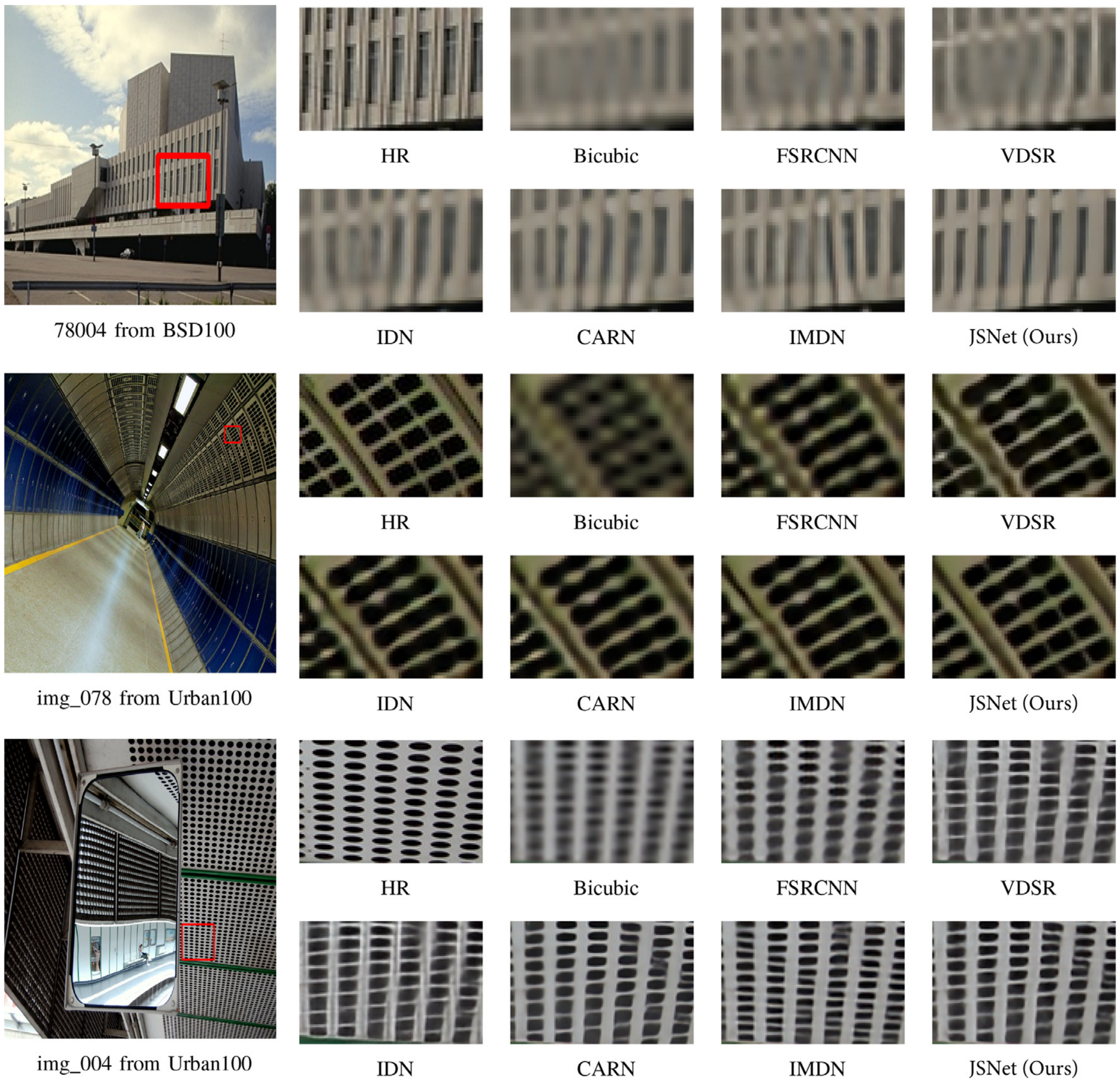

Fig. 7. Visual comparison of $\times 4$ image super-resolution of various methods on BSD100 and Urban100 datasets.

Table 8
Searching cost of NAS-based image super-resolution methods.

NAS-based SR method	GPU	GPU days
FALSR [17]	Tesla V 100	24
ESRN [16]	Tesla V 100	8
JSNet (Ours)	NVIDIA 2080Ti	2

only shows the results on scaling factor $\times 2$ because other results are unavailable. As shown in the table, our method outperforms most manually-designed methods with even fewer parameters and Multi-Adds. Compared with LAPAR-A, although relatively inferior results are obtained, our method uses fewer training images (DIV2K *vs.* DIV2K + Flickr2K). Compared with NAS-based methods, our JSNet outperforms the other methods on most benchmark datasets. Regarding the $\times 2$ and $\times 3$ setting, our JSNet stands out as the best. Besides, as shown in Table 8, the search cost of our method is significantly less than NAS-based methods. FALSR [17] takes around 3 days on 8 Tesla V100 GPUs to execute their model once. ESRN [16] takes around one day on 8 Tesla V100 GPUs to search. However, our JSNet, based on gradient algorithms, takes around 2 days on one NVIDIA 2080Ti GPU. It is worth noting that RDN [9], RCAN [10], and RFANet [33] have higher performance than ours. However, we do not compare these models because it is meaningless to compare two models with large differences in parameter and depth.

The visual comparison of $\times 4$ super-resolution on BSD100 and Urban100 is shown in Fig. 7. To facilitate subjective comparison visually, we enlarge selected regions in the super-resolved images. One can observe that most of compared methods cannot recover lost details in the low-resolution image. In contrast, our method can recover sharper and clear edges.

4.4.3. Gray image denoising

In this part, we make further exploration of gray image denoising. For this application, the upsampling module in Fig. 4 is removed. We compare the proposed JSNet with BM3D [50], TNRD [51], DnCNN [52], MemNet [12], IRCNN [42], FFDNet [6], and A-CubeNet [15]. As shown in Table 9, our method achieves the second-best results with most noise levels. Although A-CubeNet obtains slightly better results, it has more than twice the number of parameters than ours. And it utilizes the non-local operation to expand the receptive field, however, the operation is time-consuming, which defeats the purpose of designing an efficient network. It should be also noted that our JSNet achieves slight gains over other methods on BSD68 datasets. This is mainly because training images of these methods contain BSD68 datasets,

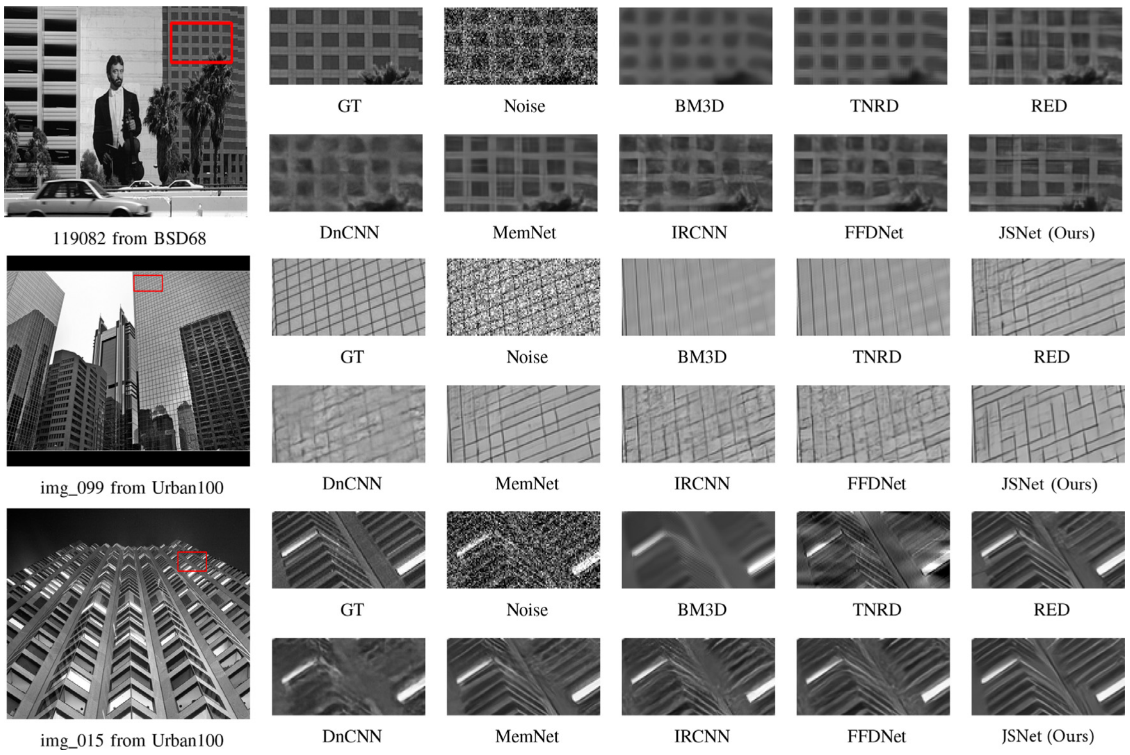

Fig. 8. Visual comparison of gray image denoising of various methods on BSD68 and Urban100 datasets with noise level 50.

Table 9
Quantitative results about gray image denoising on Kodak24, BSD68, and Urban100 datasets.

Method	Parameters	Kodak24				BSD68				Urban100			
		10	30	50	70	10	30	50	70	10	30	50	70
BM3D [50]	N/A	34.39	29.13	26.99	25.73	33.31	27.76	25.62	24.44	34.47	28.75	25.94	24.27
TNRD [51]	N/A	34.41	28.87	27.20	24.95	33.41	27.66	25.97	23.83	33.78	27.49	25.59	22.67
DnCNN [52]	0.56M	34.90	29.62	27.51	26.08	33.88	28.36	26.23	24.90	<u>34.73</u>	28.88	26.28	24.36
MemNet [12]	0.67M	N/A	29.72	27.68	<u>26.42</u>	N/A	28.43	<u>26.35</u>	<u>25.09</u>	N/A	<u>29.10</u>	<u>26.65</u>	<u>25.01</u>
IRCNN [42]	0.12M	34.76	29.53	27.45	N/A	33.74	28.26	26.15	N/A	34.60	28.85	26.24	N/A
FFDNet [6]	N/A	34.81	29.70	27.63	26.34	33.76	28.39	26.29	25.04	34.45	29.03	26.52	24.86
A-CubeNet [15]	1.37M	**35.06**	**29.84**	**27.77**	**26.44**	**33.94**	**28.50**	**26.37**	**25.10**	N/A	N/A	N/A	N/A
JSNet (Ours)	0.79M	<u>35.04</u>	<u>29.80</u>	<u>27.69</u>	26.39	<u>33.93</u>	<u>28.48</u>	<u>26.35</u>	25.05	**35.17**	**29.58**	**26.95**	**25.21**

Table 10
Gray image denoising performance of three NAS-based methods on BSD200.

Methods	Parameters	BSD200		
		$\sigma = 30$	$\sigma = 50$	$\sigma = 70$
E-CAE [30]	1.05M	28.23	26.17	24.83
CLEARER [29]	6.31M	28.54	26.40	**25.06**
JSNet (Ours)	**0.79M**	**28.64**	**26.43**	**25.06**

so it is reasonable to perform pretty well on this dataset. Visual results of various methods under noise level $\sigma = 50$ are shown in Fig. 8. Compared with other methods, our method not only removes noise well but keeps texture structure as much as possible.

To further prove the superiority of JSNet, we also compare our method with recent NAS-based denoising methods E-CAE [30] and CLEARER [29], with the quantitative results on the BSD200 dataset shown in Table 10. We can see that compared with E-CAE, our method obtains better performance in all noise levels. Compared with CLEARER, we achieve competitive results, and simultaneously, the number of trainable parameters of our method is much lower than it.

4.4.4. JPEG image deblocking

We also apply our JSNet to reduce JPEG image deblocking. For a fair comparisons, we perform training and evaluation both on Y channel of YCbCr color space. We compare our method with SA-DCT [53], ARCNN [7], TNRD [51], DnCNN [52], and A-CubeNet [15]. As shown in Table 11, our JSNet achieves the second-best results with all JPEG quality settings under the evaluation of both PSNR and SSIM, and our method presents an attractive performance with fewer parameters. We also show visual results under very low JPEG quality (q = 10). As shown in Fig. 9, our method not only removes artifacts but also preserves more details. The superiority of our method benefits from the fact that the proposed JSNet integrates various attention features from spatial-wise, channel-wise, and pixel dimensions to improve the representational capability of models.

Table 11
Quantitative results about JPEG image deblocking on LIVE1 and Classic5 datasets.

Dataset	q	JPEG		SA-DCT [53]		ARCNN [7]		TNRD [51]		DnCNN [52]		A-CubeNet [15]		JSNet (Ours)	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
LIVE1	10	27.77	0.7905	28.65	0.8093	28.98	0.8217	29.15	0.8111	29.19	0.8123	**29.54**	**0.8216**	29.49	0.8206
	20	30.07	0.8683	30.81	0.8781	31.29	0.8871	31.46	0.8769	31.59	0.8802	**31.93**	**0.8859**	31.88	0.8853
	30	31.41	0.9000	32.08	0.9078	32.69	0.9166	32.84	0.9059	32.98	0.9090	**33.35**	**0.9136**	33.31	0.9130
	40	32.35	0.9173	32.99	0.9240	33.63	0.9306	N/A	N/A	33.96	0.9247	**34.36**	**0.9289**	34.33	0.9284
Classic5	10	27.82	0.7800	28.88	0.8071	29.04	0.8111	29.28	0.7992	29.40	0.8026	**29.84**	**0.8147**	29.74	0.8128
	20	30.12	0.8541	30.92	0.8663	31.16	0.8694	31.47	0.8576	31.63	0.8610	**32.04**	**0.8677**	31.94	0.8663
	30	31.48	0.8844	32.14	0.8914	32.52	0.8967	32.78	0.8837	32.91	0.8861	**33.30**	**0.8919**	33.25	0.8910
	40	32.43	0.9011	33.00	0.9055	33.34	0.9101	N/A	N/A	33.77	0.9003	**34.16**	**0.9048**	33.95	0.9030
Parameters		N/A		N/A		0.12M		N/A		0.56M		1.37M		0.67M	

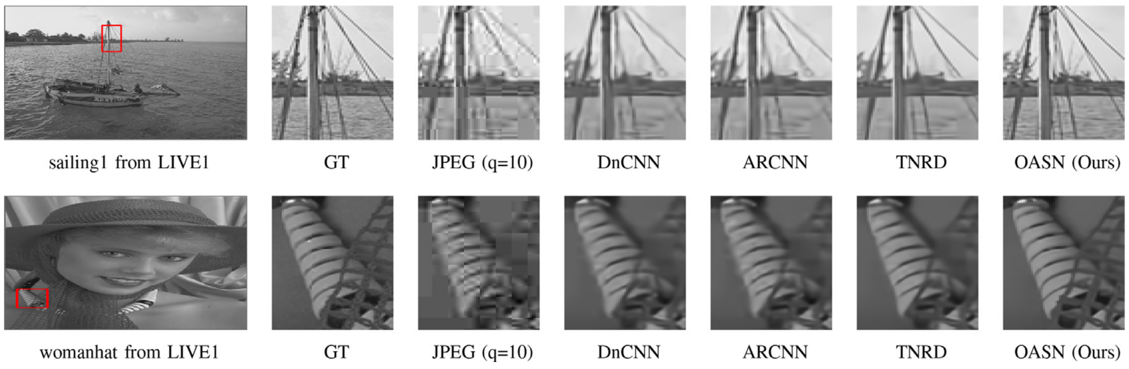

Fig. 9. Visual comparison of JPEG image deblocking of various methods with JPEG quality q = 10.

5. Conclusions

In this paper, we propose a joint operation and attention block search algorithm that aims to exploit the potential of multiple connections from the pre-defined operation blocks and attention blocks. The operation search module (OSM) and attention search module (ASM) are found by employing differentiable architecture search strategies. Then, we combine OSM with ASM to build the joint search module (JSM) and stack them to build the overall architecture. In addition, an effective cross-scale fusion module is also developed to enhance multi-scale features while reducing the number of parameters in the reconstruction process. Many experimental results demonstrate the proposed new method is able to achieve better performance compared with state-of-the-art methods for image super-resolution, gray image denoising, and JPEG image deblocking.

Although the proposed method achieves superior performance for lightweight image restoration tasks. In the future, there are still some works worth exploring. Firstly, the proposed method only constructs the cell-level search space to save search time. We can construct the cell-level and network-cell search space jointly to pursue better performance. Secondly, the JSNet may further benefit from adversarial training, which can help to alleviate over-smoothing artifacts and produce more detailed features. Thirdly, this paper mainly solves lightweight image restoration tasks. However, it is well known the performance gap between deep networks and lightweight networks still exists. Therefore, we can employ attention-based search space into some representative building blocks to pursue better performance. Fourthly, we believe that the proposed search algorithm can be further applied to other image restoration tasks, such as image derain and dehazing.

Declaration of Competing Interest

We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work, there is no professional or other personal interest of any nature or kind in any product, service and/or company that could be construed as influencing the position presented in, or the review of, the manuscript entitled.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61976079), the Anhui Key Research and Development Program (No. 202004a05020039), the Anhui High-level Talents Program (No. T000642), the National Key R&D Program of China (No. 2018AAA0100100), Science and Technology Innovation 2030 Major Projects of China (No. 2021ZD0201904), Introduction Plan of High-end Foreign Experts (No. G2021033002L), the Key Project of Science and Technology of Guangxi (No. AA22068057 and 2021AB20147), the Natural Science Foundation of Guangxi (No. 2021JJA170204 and 2021JJA170199), and the Guangxi Science and Technology Base and Talents Special Project (No. 2021AC19354 and 2021AC19394).

References

[1] Y. Yang, Y. Qi, Image super-resolution via channel attention and spatial graph convolutional network, Pattern Recognit 112 (2021) 107798.
[2] L. Wang, K.-J. Yoon, Semi-supervised student-teacher learning for single image super-resolution, Pattern Recognit 121 (2022) 108206.
[3] S.W. Zamir, A. Arora, S. Khan, M. Hayat, F.S. Khan, M.-H. Yang, L. Shao, Multi-stage progressive image restoration, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2021, pp. 14821–14831.

[4] N. Ahn, B. Kang, K.A. Sohn, Fast, accurate, and lightweight super-resolution with cascading residual network, in: Proceedings of the European Conference on Computer Vision, 2018, pp. 252–268.
[5] K. Jiang, Z. Wang, P. Yi, J. Jiang, Hierarchical dense recursive network for image super-resolution, Pattern Recognit 107 (2020) 107475.
[6] K. Zhang, W. Zuo, L. Zhang, FFDNet: toward a fast and flexible solution for CNN-based image denoising, IEEE Trans. Image Process. (2018) 4608–4622.
[7] C. Dong, Y. Deng, C. Change Loy, X. Tang, Compression artifacts reduction by a deep convolutional network, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 576–584.
[8] J. Kim, J.K. Lee, K.M. Lee, Accurate image super-resolution using very deep convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1646–1654.
[9] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, Y. Fu, Residual dense network for image restoration, IEEE Trans Pattern Anal Mach Intell 43 (7) (2020) 2480–2495.
[10] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, Y. Fu, Image super-resolution using very deep residual channel attention networks, in: Proceedings of the European Conference on Computer Vision, 2018, pp. 286–301.
[11] J. Kim, J. Kwon Lee, K. Mu Lee, Deeply-recursive convolutional network for image super-resolution, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1637–1645.
[12] Y. Tai, J. Yang, X. Liu, C. Xu, Memnet: A persistent memory network for image restoration, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 4539–4547.
[13] Z. Hui, X. Gao, Y. Yang, X. Wang, Lightweight image super-resolution with information multi-distillation network, in: Proceedings of the 27th ACM International Conference on Multimedia, 2019, pp. 2024–2032.
[14] D. Liu, B. Wen, Y. Fan, C.C. Loy, T.S. Huang, Non-local recurrent network for image restoration, in: Advances in Neural Information Processing Systems, 2018, pp. 1680–1689.
[15] Y. Hang, Q. Liao, W. Yang, Y. Chen, J. Zhou, Attention cube network for image restoration, in: Proceedings of the 28th ACM International Conference on Multimedia, 2020, pp. 2562–2570.
[16] D. Song, C. Xu, X. Jia, Y. Chen, C. Xu, Y. Wang, Efficient residual dense block search for image super-resolution, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2020, pp. 12007–12014.
[17] X. Chu, B. Zhang, H. Ma, R. Xu, Q. Li, Fast, accurate and lightweight super-resolution with neural architecture search, in: 2020 25th International Conference on Pattern Recognition, 2021, pp. 59–64.
[18] H. Liu, K. Simonyan, Y. Yang, Darts: Differentiable architecture search, in: International Conference on Learning Representations, 2019.
[19] Y. Xu, L. Xie, X. Zhang, X. Chen, G.-J. Qi, Q. Tian, H. Xiong, PC-DARTS: Partial channel connections for memory-efficient architecture search, in: International Conference on Learning Representations, 2020.
[20] B. Li, X. Gao, Lattice structure for regular linear phase paraunitary filter bank with odd decimation factor, IEEE Signal Process Lett (2013) 14–17.
[21] W. Liao, Z.-Q. Zhao, H. Shen, W. Tian, Residual attention block search for lightweight image super-resolution, in: 2021 IEEE International Conference on Multimedia and Expo, 2021, pp. 1–6.
[22] Y. Wang, L. Wang, H. Wang, P. Li, H. Lu, Blind single image super-resolution with a mixture of deep networks, Pattern Recognit 102 (2020) 107169.
[23] C. Dong, C.C. Loy, K. He, X. Tang, Image super-resolution using deep convolutional networks, IEEE Trans Pattern Anal Mach Intell (2015) 295–307.
[24] Y. Quan, Y. Chen, Y. Shao, H. Teng, Y. Xu, H. Ji, Image denoising using complex-valued deep CNN, Pattern Recognit 111 (2021) 107639.
[25] B. Li, Y. Gou, S. Gu, J.Z. Liu, J.T. Zhou, X. Peng, You only look yourself: unsupervised and untrained single image dehazing neural network, Int J Comput Vis 129 (5) (2021) 1754–1767.
[26] B. Li, X. Liu, P. Hu, Z. Wu, J. Lv, X. Peng, All-in-one image restoration for unknown corruption, in: IEEE Conference on Computer Vision and Pattern Recognition, New Orleans, LA, 2022.
[27] B. Zoph, V. Vasudevan, J. Shlens, Q.V. Le, Learning transferable architectures for scalable image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8697–8710.
[28] H. Zhang, Y. Li, H. Chen, C. Shen, Memory-efficient hierarchical neural architecture search for image denoising, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 3657–3666.
[29] Y. Gou, B. Li, Z. Liu, S. Yang, X. Peng, Clearer: multi-scale neural architecture search for image restoration, Advances in Neural Information Processing Systems, volume 33, 2020.
[30] M. Suganuma, M. Ozay, T. Okatani, Exploiting the potential of standard convolutional autoencoders for image restoration by evolutionary search, in: International Conference on Machine Learning, PMLR, 2018, pp. 4771–4780.
[31] Y. Zhang, K. Li, K. Li, B. Zhong, Y. Fu, Residual non-local attention networks for image restoration, in: International Conference on Learning Representations, 2019.
[32] S. Woo, J. Park, J.-Y. Lee, I.S. Kweon, CBAM: Convolutional block attention module, in: Proceedings of the European conference on computer vision, 2018, pp. 3–19.
[33] J. Liu, W. Zhang, Y. Tang, J. Tang, G. Wu, Residual feature aggregation network for image super-resolution, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 2359–2368.
[34] H. Zhao, X. Kong, J. He, Y. Qiao, C. Dong, Efficient image super-resolution using pixel attention, in: Proceedings of the European conference on computer vision, 2020, pp. 56–72.
[35] A. Muqeet, J. Hwang, S. Yang, J. Kang, Y. Kim, S.-H. Bae, Multi-attention based ultra lightweight image super-resolution, in: Proceedings of the European conference on computer vision, 2020, pp. 103–118.
[36] W. Shi, J. Caballero, F. Huszár, J. Totz, A.P. Aitken, R. Bishop, D. Rueckert, Z. Wang, Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1874–1883.
[37] R. Timofte, E. Agustsson, L. Van Gool, M.-H. Yang, L. Zhang, Ntire 2017 challenge on single image super-resolution: Methods and results, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2017, pp. 114–125.
[38] M. Bevilacqua, A. Roumy, C. Guillemot, M.L. Alberi-Morel, Low-complexity single-image super-resolution based on nonnegative neighbor embedding, in: Proceedings of the British Machine Vision Conference, 2012, pp. 135.1–135.10.
[39] R. Zeyde, M. Elad, M. Protter, On single image scale-up using sparse-representations, in: International Conference on Curves and Surfaces, 2010, pp. 711–730.
[40] D. Martin, C. Fowlkes, D. Tal, J. Malik, et al., A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics, in: Proceedings Eighth IEEE International Conference on Computer Vision, volume 2, 2001, pp. 416–423.
[41] J.-B. Huang, A. Singh, N. Ahuja, Single image super-resolution from transformed self-exemplars, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5197–5206.
[42] K. Zhang, W. Zuo, S. Gu, L. Zhang, Learning deep CNN denoiser prior for image restoration, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 3929–3938.
[43] A.K. Moorthy, A.C. Bovik, Visual importance pooling for image quality assessment, IEEE J Sel Top Signal Process 3 (2) (2009) 193–201.
[44] A. Foi, V. Katkovnik, K. Egiazarian, Pointwise shape-adaptive dct for high-quality denoising and deblocking of grayscale and color images, IEEE Trans. Image Process. 16 (5) (2007) 1395–1411.
[45] Z. Li, J. Yang, Z. Liu, X. Yang, G. Jeon, W. Wu, Feedback network for image super-resolution, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 3867–3876.
[46] W. Li, K. Zhou, L. Qi, N. Jiang, J. Lu, J. Jia, Lapar: Linearly-assembled pixel-adaptive regression network for single image super-resolution and beyond, Advances in Neural Information Processing Systems, 2020.
[47] C. Dong, C.C. Loy, X. Tang, Accelerating the super-resolution convolutional neural network, in: Proceedings of the European conference on computer vision, 2016, pp. 391–407.
[48] Y. Tai, J. Yang, X. Liu, Image super-resolution via deep recursive residual network, in: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2017, pp. 3147–3155.
[49] Z. Hui, X. Wang, X. Gao, Fast and accurate single image super-resolution via information distillation network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 723–731.
[50] K. Dabov, A. Foi, V. Katkovnik, K. Egiazarian, Image denoising by sparse 3-d transform-domain collaborative filtering, IEEE Trans. Image Process. 16 (8) (2007) 2080–2095.
[51] Y. Chen, T. Pock, Trainable nonlinear reaction diffusion: a flexible framework for fast and effective image restoration, IEEE Trans Pattern Anal Mach Intell 39 (6) (2016) 1256–1272.
[52] K. Zhang, W. Zuo, Y. Chen, D. Meng, L. Zhang, Beyond a gaussian denoiser: residual learning of deep cnn for image denoising, IEEE Trans. Image Process. (2017) 3142–3155.
[53] A. Foi, V. Katkovnik, K. Egiazarian, Pointwise shape-adaptive DCT for high--quality denoising and deblocking of grayscale and color images, IEEE Trans. Image Process. 16 (5) (2007) 1395–1411.

Hao Shen is currently pursuing the Ph.D. degree. His current research interests include image processing, computer vision, and deep-learning.

Zhong-Qiu Zhao (M'10) received the Ph.D. degree in pattern recognition and intelligent system from the University of Science and Technology of China, Hefei, China, in 2007. From 2008 to 2009, he held a post-doctoral position in image processing with the CNRS UMR6168 Lab Sciences de Information et des Systȿmes, La Garde, France. From 2013 to 2014, he was a Research Fellow in image processing with the Department of Computer Science, Hong Kong Baptist University, Hong Kong. He is currently a Professor with the Hefei University of Technology, Hefei. His current research interests include pattern recognition, image processing, and computer vision.

Wenrui Liao is currently pursuing the master's degree. His current research interests include image processing and computer vision.

Wei-Dong Tian is an associate professor at Hefei University of Technology, China. He obtained the Master's degree in Computer Science and Technology at School of Computer Science and Information Engineering, Hefei University of Technology, Hefei, China, in 1996. Now he works in Laboratory of Data Mining and Intelligent Computing, Hefei University of Technology, China. His research is about data mining, machine learning, image processing, and computer vision.

De-Shuang Huang (Fellow, IEEE) received the B.Sc. degree in electronic engineering from the Institute of Electronic Engineering, Hefei, China, in 1986, the M.Sc.

degree in electronic engineering from the National Defense University of Science and Technology, Changsha, China, in 1989, and the Ph.D. degree Xidian University, Xi'an, China, in 1993. From 1993 to 1997, he was a Postdoctoral Research Fellow with the Beijing Institute of Technology, National Key Laboratory of Pattern Recognition, Chinese Academy of Sciences, Beijing, China. In 2000, he joined the Institute of Intelligent Machines, Chinese Academy of Sciences, as the recipient of "Hundred Talents Program of CAS." In 2011, he entered into Tongji University, Shanghai, China, as a Chaired Professor. From 2000 to 2001, he worked as a Research Associate with the Hong Kong Polytechnic University, Hong Kong. In 2003, he visited the George Washington University, Washington, DC, USA, as a Visiting Professor. In 2004, he worked as the University Fellow with Hong Kong Baptist University, Hong Kong. From 2005 to 2006, he worked as a Research Fellow with the Chinese University of Hong Kong, Hong Kong. In 2006, he was a Visiting Professor with the Queens University of Belfast, Belfast, U.K. In 2007, 2008, and 2009, he worked as a Visiting Professor with Inha University, Incheon, South Korea. At present, he is with EIT Institute for Advanced Study. Dr. Huang is currently IAPR Fellow and IEEE Fellow. He has published over 220 journal papers. His current research interest includes bioinformatics, pattern recognition and machine learning.