# Journal Pre-proof


Fine-grained neural architecture search for image super-resolution


Heewon Kim, Seokil Hong, Bohyung Han, Heesoo Myeong, Kyoung Mu Lee



PII: S1047-3203(22)00174-2
DOI: https://doi.org/10.1016/j.jvcir.2022.103654
Reference: YJVCI 103654

To appear in: *J. Vis. Commun. Image R.*

Received date : 10 January 2022
Revised date : 2 September 2022
Accepted date : 24 September 2022

Please cite this article as: H. Kim, S. Hong, B. Han et al., Fine-grained neural architecture search for image super-resolution, *J. Vis. Commun. Image R.* (2022), doi: https://doi.org/10.1016/j.jvcir.2022.103654.

## Highlights for "Fine-Grained Neural Architecture Search for Image Super-Resolution"

- A new search space that allows for fine-grained architectural changes with the number of channels per layer and the activation function per channel.
- A new algorithm that automatically finds computationally-efficient architectures in the proposed search space through gradient descent.
- A new finding that multiple heterogeneous activation functions in a layer can improve the model accuracy.
- Performance improvement over the models that are automatically determined by existing Neural Architecture Search (NAS) algorithms.

Journal of Visual Communication and Image Representation (2022)

Contents lists available at ScienceDirect

Journal of Visual Communication and Image Representation

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

# Fine-Grained Neural Architecture Search for Image Super-Resolution

Heewon Kim[a], Seokil Hong[a], Bohyung Han[a,b], Heesoo Myeong[c], Kyoung Mu Lee[a,b,*]

[a]*The Department of Electrical and Computer Engineering, ASRI, Seoul National University, Seoul, Korea*
[b]*Interdisciplinary Program in Artificial Intelligence, Seoul National University, Seoul, Korea*
[c]*Qualcomm Korea YH, Seoul, Korea*

ARTICLE INFO

*Article history*:

*Keywords:* Image super-resolution, Neural architecture search, Convolutional neural network

ABSTRACT

Designing efficient deep neural networks has achieved great interest in image super-resolution (SR). However, exploring diverse network structures is computationally expensive. More importantly, each layer in a network has a distinct role that leads to the design of a specialized structure. In this work, we present a novel neural architecture search (NAS) algorithm that efficiently explores layer-wise structures. Specifically, we construct a supernet allowing flexibility in choosing the number of channels and per-channel activation functions according to the role of each layer. The search process runs efficiently via channel pruning since gradient descent jointly optimizes the Mult-Adds and the accuracy of the searched models. We facilitate estimating the model Mult-Adds in a differentiable manner using relaxations in the backward pass. The searched model, named FGNAS, outperforms the state-of-the-art NAS-based SR methods by a large margin.



## 1. Introduction

Image super-resolution (SR) is one of the conventional computer vision tasks, increasing the resolution of an image by restoring its original details. Early SR methods using convolutional neural networks (CNNs) [1, 2, 3, 4] dramatically improved the restoration accuracy through deeper and wider CNNs. Still, their high computational cost is a considerable obstacle for resource-constrained applications, such as 4K image SR in mobile phones. Many recent works [5, 6, 7, 8, 9, 10] present efficient SR networks by manually exploring the computational redundancy and bottlenecks of the architectures. However, the hand-crafted architectures are still suboptimal, and finding them requires extreme training resources and the efforts of many researchers.

Neural Architecture Search (NAS) is an AutoML technique that automatically designs the structure of a neural network. Search space design in this area is of crucial importance for two reasons: (1) Search space should contain the final model for a given objective (*e.g.*, optimizing efficiency-accuracy trade-off) as one of the candidates. (2) Search space with too many incompetent candidates makes the search process unfeasible. However, search spaces of the existing NAS methods may have two limitations for the optimal efficiency of the SR network.

First, previous NAS-based SR models are incorporated by large search units that incur computational redundancy. MoreMNAS [11] and FALSR [12] search for seven cells as basic building blocks to construct full models through stacking. HNAS [13] and ESRN [14] allow each block of networks to select one of two and three searched cells, respectively. By contrast, recent NAS algorithms for image classification [15, 16, 17, 18, 19] adopt small search units that find layer-wise configurations to reduce the redundancy from repeated blocks.

Second, the small search units designed for image classification may not be suitable for SR. Different outputs between classification and SR lead a custom macro-level architecture for

搜索单元大? 大→计算开销大

SR太大 IC太小? 小→不适合SR.

*Corresponding author.
*e-mail:* kyoungmu@snu.ac.kr (Kyoung Mu Lee)

each task; networks for classification reduce the input dimension (or resolution), while SR networks increase the dimension. In this regard, NAS algorithms have developed the micro-level search spaces for each task. For instance, the works for classification [20, 15, 21, 22, 23, 16, 17, 18, 19] adopt large kernel sizes up to 7×7 to reduce feature resolution, whereas the search units for SR [11, 12, 14] only contain the small kernel sizes of 1×1 and 3×3.

In this paper, we present a fine-grained search space for SR networks that allows candidates to have layer-wise configurations. Our search space incorporates the number of channels in each convolution and the activation function per channel. Specifically, we present a searchable block that includes a point-wise convolution and multiple activation functions in a residual block. Each block of the entire network can have an individual structure, reducing redundancy due to structural repetition. The point-wise convolution allows large channels for both the activation and the skip connection with the small computation costs, and channel-specific activation functions improve network accuracy.

Our search process runs efficiently via a differentiable channel pruning method. Instead of training and evaluating every model candidate individually, we construct a supernet consisting of the searchable blocks and prune the supernet for exploration. We adopt a straight-through (ST) estimator per channel to learn where to prune by gradient descent during training. To this end, we introduce an auxiliary relaxation to count channels for activation search. This relaxation allows estimating the model Mult-Adds in a differentiable manner to optimize the efficiency-accuracy trade-off by gradient descent.

Experimental results show that our search algorithm finds multiple heterogeneous activation functions of a layer and efficient channels per convolution. Our final model, named FG-NAS, outperforms the state-of-the-art NAS-based SR models with a PSNR score of over 0.21 dB on the standard benchmark while using the lower Mult-Adds and parameters.

## 2. Related Work

### 2.1. Image Super-Resolution

Image super-resolution (SR) is a classical topic in computer vision which aims at restoring the original high-resolution image from its low-resolution versions. Early deep-learning-based approaches have achieved breakthroughs in accurate restoration performances. SRCNN [1] is the first neural network for SR that uses three convolution layers. VDSR [2] allows training a deep convolutional network with 20 layers using residual learning. SRGAN [4] adopts residual blocks [24] and adversarial training [25] for realistic texture generation. EDSR [3] uses large numbers of convolutional layers and channels by enhanced residual blocks for stable training. RDN [26] consists of densely connected residual blocks to achieve better accuracy.

However, these accurate models are computationally expensive (*e.g.*, Mult-Adds). More importantly, convolution neural networks (CNNs) for SR consume more computations for high-resolution images, making them difficult to be used for resource-constrained applications. To this end, recent works focus on designing efficient deep SR models. The early works [27, 4] use low-resolution image input to reduce the computations of convolution. AdaDSR [28] saves computation costs via adaptive inference. CARN [7] uses residual blocks with multiple skip connections for a fast and lightweight architecture. MXDSIR [29] adopts depth-wise convolutions [30] to reduce the number of parameters and PReLU activation to improve accuracy. WDSR [10] and BTSRN [31] use point-wise convolution and 3×3 convolution to design a residual block with a large number of channels for activation. Different from WDSR, BTSRN increases feature resolution in the middle of the network for accurate training.

On the other hand, improving the efficiency of neural networks has been researched in various applications with intuitive approaches. The work in [32] recently proposed a video SR using an event camera, which can save network computations for still scenes. DIN [33] proposed a dynamic instance normalization method that enables networks based on efficient architectures (*e.g.*, MobileNet [30]) to perform flexible style transfer. Works in [34, 35] quantize weights and feature maps of image super-resolution networks. While SLS [36] prunes image restoration networks with $N$:$M$ sparsity for the next generation hardware, TASNet [37] adaptively prunes network channels specialized in a restoration task. For graph neural networks, knowledge distillation [38], knowledge amalgamation [39], and graph factorization [40] have been proposed for efficient inferences by learning small networks or disentangled representations. ClassSR [41] proposed a general framework for efficient image super-resolution by adaptively selecting efficient models depending on the restoration difficulty of each image patch.

Despite the manually designed CNNs improve the efficiency-accuracy trade-off, exploring candidate architectures is laborious and time consuming. A few studies have been proposed in automation to find network architectures for SR. MoreMNAS [11] is the first work in this area that automatically determines the configurations of seven convolution blocks in the predetermined search space. The automation is possible by learning the architecture controller through reinforcement learning and evolution algorithms. Designing the search space and search algorithm is critical to finding the desired architecture in a feasible time. FALSR [12] proposed enhanced search space than MoreMNAS [11] and presented better efficiency-accuracy trade-off. ESRN [14] determines one of three convolution blocks, including pooling and upsampling feature maps, to build a network through stacking. Similarly, HNAS [13] determines the position of upsampling layers using LSTM-based architecture controllers. However, the searched architectures in existing works might still redundant due to large search units; they select one of several predetermined numbers of channels for each convolution block or stack the same convolution block to build an entire network. By contrast, our search space has a channel-level search unit that determines where to prune and which activation function to use on each channel.

### 2.2. Neural Architecture Search (NAS)

Automatic architecture search techniques have achieved breakthroughs in image classification. Early works develop

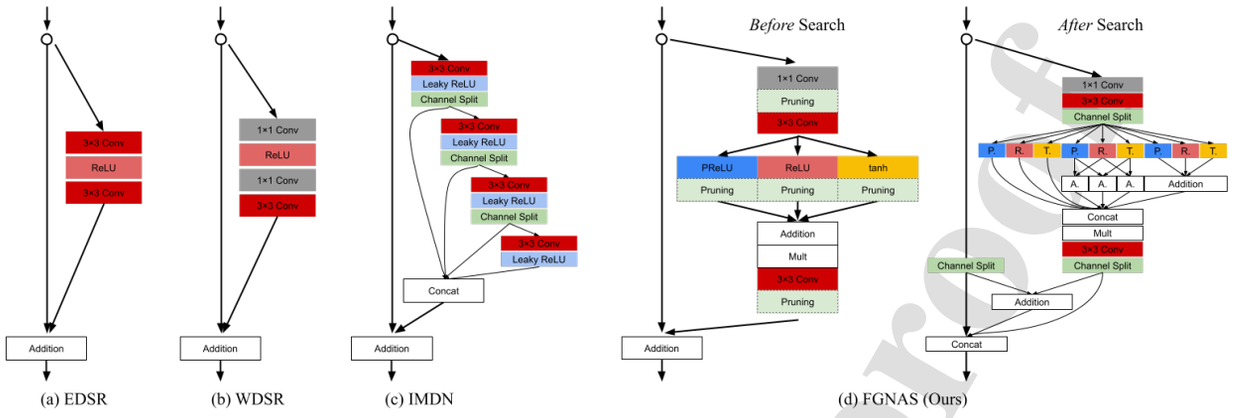



**Fig. 1. Comparison of residual blocks in EDSR [3], WDSR [10], IMDB [9], and the proposed FGNAS. While existing methods use predetermined activation functions and channels, our searchable block (*Before* Search) allows FGNAS to search for multiple heterogeneous activation functions and the efficient number of channels (*After* Search) through channel pruning.**

the search algorithm to find accurate models. NASNet [42] and MetaQNN [43] adopt reinforcement learning to optimize the non-differentiable process of searching for accurate architectures. ENAS [44] learns a RNN-based architecture controller that draws a series of sample models while maximizing their expected reward. PNAS [45] performs a progressive search process by predicting the accuracy of candidate models. Evolutionary search [46] employs tournament selection and found the state-of-the-art model for accurate image classification. DARTS [47] relaxes the discrete architecture representation to a continuous one and makes the objective function differentiable. However, the searched models require a significant amount of computational resources.

Recently, improving the efficiency of searched models has become the main topic. Search algorithms in this area incorporate resource consumption, such as Mult-Adds, run-time, and the number of parameters, in the search objectives, and the search space has become granular to reduce redundant computations. MnasNet [15] consists of seven searched convolution blocks that allow repeating each block through stacking. MobileNetV3 [22] adopts layer-wise pruning [48] to remove the redundancy in the block-wise search [15] with a novel architecture design using squeeze-and-excitation [49]. ProxylessNAS [16] and FBNet [17] search for efficient convolution in each layer. EfficientNet [23] employs scaling the number of layers and channels and image resolution of the backbone network as the architecture search process. MixNet [19] provides multiple kernel sizes of depth-wise convolution in a single layer. AtomNAS [18] proposed a channel-level search unit to determine the kernel size of depth-wise convolution for each channel via channel pruning.

In this work, we propose a channel-level search unit for image super-resolution. The efficient architecture for SR inevitably does not match the one for image classification. We design the search space based on the residual block of EDSR [3] and formulate the search process as channel pruning. To optimize the accuracy and Mult-Adds of searched models differently, we relax the discrete channel counting function to a continuous one.

**Table 1. Supernet Architecture.**

| Input shape | Block | #Channels | n | Upscale | Global skip |
|---|---|---|---|---|---|
| $H \times W \times 3$ | 3×3 Conv | 64 | 1 | 1 | from |
| $H \times W \times 64$ | Searchable Block | 64 | 16 | 1 | - |
| $H \times W \times 64$ | 3×3 Conv | 64 | 1 | 1 | to |
| $H \times W \times 64$ | 3×3 Conv | 12 | 1 | 1 | - |
| $H \times W \times 12$ | PixelShuffle | 3 | 1 | 2 | - |

All convolutions in a Searchable block have 64 channels. 'n' denotes the number of repetitions of the block. Image resolution is upscaled at the last block of PixelShuffle. The global skip connects the feature maps denoted by 'from' and 'to.'

## 3. Proposed Method

Supernet-based NAS approaches [47, 16, 17, 18] identify a neural network architecture from a larger network called supernet. Their search spaces are designed by the architectures of the supernet and its removable subnetworks. Accordingly, we propose a novel supernet for image super-resolution (SR) and assume its channels as the removable subnetworks. This assumption leads to our search process: pruning channels of the supernet. Our search space allows the number of the channel search for each convolution and activation function search for each channel described in Section 3.1. Section 3.2 illustrates our search process, which is a differentiable channel pruning method specialized in our supernet.

### *3.1. Fine-Grained Search Space for Image Super-Resolution*

In general, deep SR models improve the network efficiency by designing novel residual blocks as visualized in Figure 1. EDSR [3] removes unnecessary ReLU activations for accurate network training. WDSR [10] expends the number of channels for activation using efficient point-wise convolutions. IMDN [9] splits channels with Leaky ReLU activation to utilize information from skip connections without additional computations. However, staking a single block structure for the entire network is still suboptimal. To find efficient SR networks,

we present a supernet of which candidates can have the following structural benefits: (1) non-repeating structures of residual blocks to reduce network redundancy, (2) a large number of channels for activation and skip connection while the channels for the others are relatively small, and (3) multiple heterogeneous activation functions of a layer where each function is chosen to maximize the accuracy.

Table 1 illustrates the structure of our supernet. All convolutions except for the last have 64 channels. The last convolution has 12 output channels which are rearranged into a 3-channel RGB image after PixelShuffle [6]. This approach saves computation costs of auxiliary convolutions without accuracy drop [10]. Our search space includes the number of channels in all convolutions, which naturally finds non-repeating structures of residual blocks.

For the other structural benefits, we design a searchable residual block, described in Figure 1(d), consisting of an 1×1 (point-wise) convolution, two 3×3 convolutions, a ReLU activation, a PReLU activation, and a tanh activation with skip connection and addition (element-wise sum) operations. We prune the channels of feature maps from 1×1 convolution, three activation functions, and the last 3×3 convolution. The 1×1 convolution can squeeze channels between skip connection and activation function while keeping their numbers of channels large. This channel squeezing dramatically reduces the number of floating-point operations (Mult-Adds) of convolution, which is a linear function of the multiplication between input and output channels. The outputs of three activation functions are pruned individually, and the element-wise sum of the three generates heterogeneous activation functions of a channel. Each channel can have $2^3 - 1$ different types of activation that are combinations of ReLU, PReLU, and tanh. We empirically observe that the element-wise sum of different activation functions makes the network training unstable. To alleviate this problem, we adopt the scaling of $\frac{1}{3}$ and the activation functions with the same values for the zero inputs. The last 3×3 convolution can have small channels again, and each channel performs concatenation and element-wise sum to the channels of the skip connection.

### 3.2. Search Process via Channel Pruning

*Overview.* Our search algorithm identifies a model by pruning channels of our novel supernet, described in Section 3.1. We formulate the pruning process differentiable to optimize the searched network for the trade-off between the Mult-Adds and the reconstruction error. To this end, we introduce an auxiliary learnable parameter $\psi_{\ell,c,i} \in \boldsymbol{\psi}$ that determines whether the corresponding channel is pruned or not, where $\ell$, $c$, and $i$ denote the $\ell$-th layer, $c$-th channel, and $i$-th sub-feature map of the supernet. We adopt a straight-through (ST) estimator that predicts a binary value from the auxiliary learnable parameter. The binary values play two roles in our search process: (1) pruning channels by multiplying them to the channel tensors of sub-feature maps. (2) estimating Mult-Adds of networks by counting the number of not pruned (or *alive*) channels. The ST estimator and the channel counting function relax their discrete functions in the forward pass to be differentiable in the backward pass. The relaxation allows the learning objective of an efficiency-accuracy trade-off to be optimized via gradient descent.

*Differences from existing NAS algorithms.* The proposed search algorithm is able to find the per-channel activation function that the existing NAS algorithms cannot do. Specifically, our algorithm is differentiabe while counting the alive channels for any activation functions that share a feature map. In contrast, the existing NAS algorithms [18, 19] *individually* count the alive channels for *each* activation function. The number calculated here does not match the number of alive channels in the shared feature map.

#### 3.2.1. Straight-Through (ST) Estimator

While ST estimator predicts a binary value in the forward pass, it is relaxed into a sigmoid function in the backward pass, formally,

$$g(\psi) = \begin{cases} \mathbb{I}[\psi > 0.5] & \text{if forward} \\ \text{sigmoid}(\psi) & \text{if backward,} \end{cases} \tag{1}$$

where $\mathbb{I}[\cdot]$ is an indicator function that returns 1 when the input is true and 0 otherwise, and $\psi$ is an auxiliary learnable parameter.

#### 3.2.2. Objective Function

Our approach aims to maximize the accuracy of a target task (SR) and minimize the identified model's resource usage (Mult-Adds). Hence, our objective function comprises two terms; one is the task-specific loss, and the other is the regularizer penalizing the overhead of networks. Let $\mathcal{L}(\cdot,\cdot)$ denote the $L1$ loss function between the model output and ground truth images and $\mathcal{R}(\cdot)$ be a regularizer to estimate model Mult-Adds. Then, the objective function is formally expressed as

$$\min_{\theta,\psi} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\psi}) + \lambda \cdot \mathcal{R}(\boldsymbol{\psi}), \tag{2}$$

where $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$ are the learnable parameters of the supernet and ST estimator $g(\cdot)$, respectively, and $\lambda$ is the hyper-parameter balancing the two terms.

#### 3.2.3. Resource Regularizer

Specifically, $\mathcal{R}(\boldsymbol{\psi})$ estimates Mult-Adds of current models while $\boldsymbol{\psi}$ learns where to prune:

$$\mathcal{R}(\boldsymbol{\psi}) = \sum_{\ell=1}^{L} K_\ell^2 \cdot H_\ell \cdot W_\ell \cdot \sum_{c=1}^{C_{\ell-1}} \gamma(\boldsymbol{\psi}_{\ell-1,c}) \cdot \sum_{c=1}^{C_\ell} \gamma(\boldsymbol{\psi}_{\ell,c}), \tag{3}$$

where $C_\ell$, $K_\ell$, $H_\ell$, and $W_\ell$ are the number of channels, the kernel size, the output height, and the output width of the $\ell$-th convolution, respectively. $L$ is the number of layers in the supernet. $\sum \gamma(\cdot)$ represents the numbers of input/output *alive* channels of the $\ell$-th convolution.

#### 3.2.4. Channel Counting Function

Our supernet generates multiple sub-feature maps by multiple activation functions from a single convolution (See Figure 1). The convolution should generate the feature maps under the union of the *alive* channels while the channel pruning is performed in these feature maps individually. Since counting the

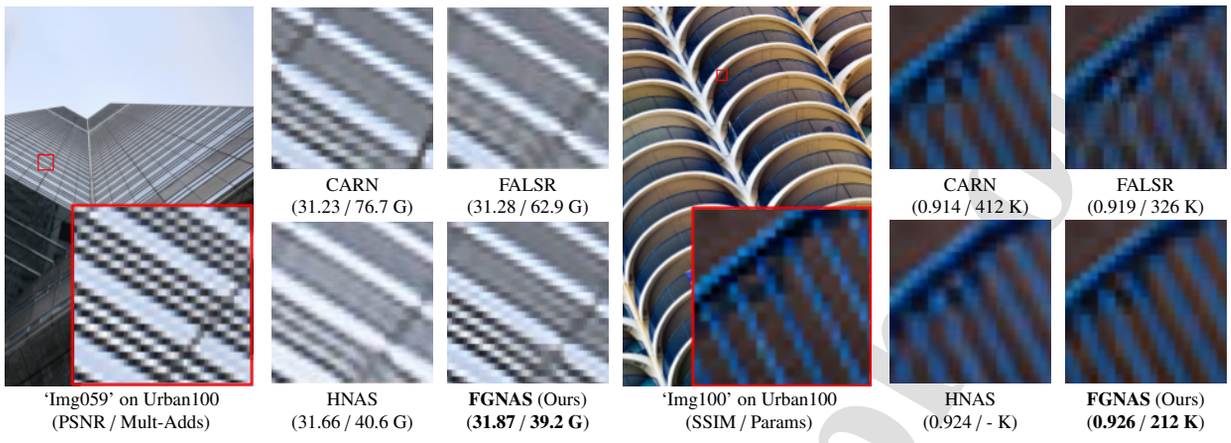



**Fig. 2. Qualitative comparisons on image super-resolution with scale factor of 2. PSNR and SSIM report average scores on Urban100. Mult-Adds is calculated to generate an HD image. Params denotes the number of model parameters.**

*alive* channels in the union is a non-differentiable process, we relax it into a summation in the backward pass by the following equation,

$$\gamma(\boldsymbol{\psi}_{\ell,c}) = \begin{cases} \mathbb{I}[\sum_{i=1}^{M_\ell} g(\psi_{\ell,c,i}) \geq 1] & \text{if forward} \\ \sum_{i=1}^{M_\ell} g(\psi_{\ell,c,i}) & \text{if backward,} \end{cases} \tag{4}$$

where $\mathbb{I}[\cdot]$ is the indicator function at Equation 1 and $M_\ell$ is the number of sub-feature maps (or activation functions).

We modify the channel counting function in following two cases; First, we regard that the 1×1 convolution in our searchable block and the first 3×3 convolution of our supernet have a sub-feature map ($M_\ell = 1$). Second, we regard a skip connection as a sub-feature map of its residual (*e.g.*, the last 3×3 convolution in our searchable block) to count the number of input channels for the next convolution.

## 4. Experiment

We use DIV2K [50] as a training dataset, which contains 800 2K images. We compare methods on the standard benchmark datasets of Set5 [51], Set14 [52], B100 [53], and Urban100 [54]. The measures of image restoration performance are PSNR (dB) and SSIM on the Y channel of the YCbCr color space. The reported Mult-Adds and latency of networks are measured to produce an HD image (1280×720 resolution). Latency is measured with a CPU on an Intel Xeon Gold 6130 processor. We will make training and evaluation codes available on Github.

### 4.1. Implementation Details

We use the Adam optimizer [63], and $\beta_1$, $\beta_2$, and $\epsilon$ are set to 0.9, 0.999, and $1.0 \times 10^{-8}$, respectively. The mini-batch size and the learning rate are set to 16 and $1.0 \times 10^{-4}$, respectively. The training patch sizes are 192×192 pixels. Hyper-parameter $\lambda$ is set to $1.0 \times 10^{-9}$. The supernet is trained for $2.0 \times 10^6$ iterations with the search process. The learning rate is halved after $1.0 \times 10^6$ iterations. For thorough comparisons and experiments on ClassSR [41], we searched models with different resources and scaling factors. We first search FGNAS-S, FGNAS-M, and FGNAS-L, consisting of 16, 32, and 64 searchable blocks, at the scaling factor of 2. At other scaling factors, we change the last convolution of the networks searched at a scaling factor of 2 to generate $3 \times s^2$ channels, where $s$ denotes the scaling factor. Then, we retrain the networks from scratch. FGNAS denotes FGNAS-S at scaling factor of 2, unless otherwise specified.

### 4.2. Comparisons to the State-of-the-Art Methods

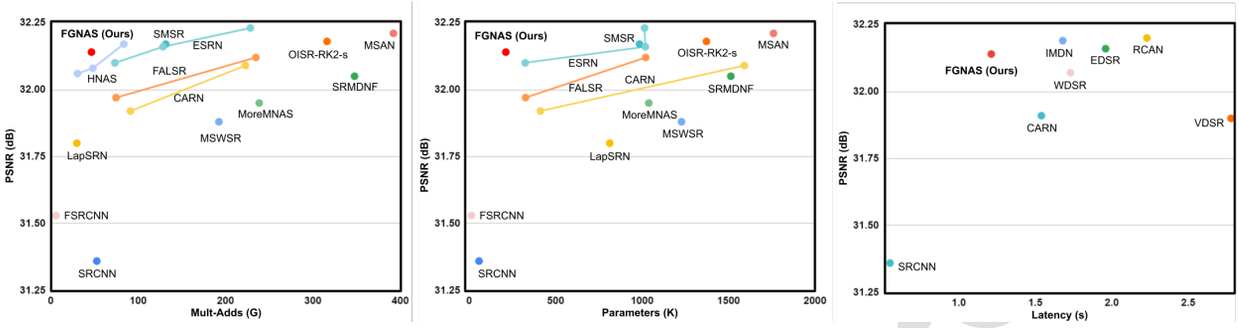
*Qualitative results.* Figure 2 presents the effectiveness of the restoration accuracy improvement by visual comparison. CARN [7] and FALSR [12] produce blurry patterns in both examples. HNAS [13] generates a white background mismatched to the original sky blue color in Figure 2(left). The color mismatch drops the scores in PSNR by a large margin. By contrast, FGNAS (or FGNAS-S at the scaling factor of 2) generates sharp and clear patterns with correct color compared to the other results.

*Quantitative results.* To present the effectiveness of our approach, we compare our final searched models, FGNAS-S, FGNAS-M, and FGNAS-L, with the state-of-the-art *efficient* SR approaches at scaling factor 2, 3, and 4. FGNAS-S, FGNAS-M, and FGNAS-L consist of 16, 32, and 64 searchable blocks, respectively. Table 2 presents quantitative comparisons. Since RCAN, SAN, and WDSR use over 70× parameters than FGNAS-S, we reproduce them with the reduced number of convolutional blocks using official codes. FGNAS-L outperforms all compared methods in PSNR, while requiring comparable FLOPs with VDSR. FGNAS-M, which performs the second best PSNR, uses fewer FLOPs and parameters compared to RCAN, SAN, and MSAN. FGNAS-S for the scaling factor of 2 achieves PSNR comparable to WDSR, SMSR, IMDN, and RFDN (0.05 dB difference in B100) using approximately 30% of FLOPs and parameters. Our NAS algorithm is the first gradient-based approach for SR and FGNAS-S outperforms all *NAS-based* methods in all measures (PSNR,

**Table 2.** ***Efficient*** **image super-resolution benchmark. Compared models have less than 3 M parameters. Red/Blue text: the best/second best performances.**

| Scale | Model | Search method | Set5 (PSNR/SSIM) | Set14 (PSNR/SSIM) | B100 (PSNR/SSIM) | Urban100 (PSNR/SSIM) | Params (K) | Mult-Adds (G) |
|---|---|---|---|---|---|---|---|---|
| ×2 | SRCNN [1] | manual | 36.66 / 0.954 | 32.42 / 0.906 | 31.36 / 0.888 | 29.50 / 0.895 | 57 | 52.7 |
| | VDSR [2] | manual | 37.53 / 0.959 | 33.03 / 0.912 | 31.90 / 0.896 | 30.76 / 0.914 | 665 | 612.6 |
| | BTSRN [31] | manual | 37.75 / - | 33.20 / - | 32.05 / - | 31.63 / - | 820 | 415.3 |
| | MemNet [55] | manual | 37.78 / 0.960 | 33.28 / 0.914 | 32.08 / 0.898 | 31.51 / 0.931 | 677 | 2,662.4 |
| | CARN [7]† | manual | 37.53 / 0.958 | 33.26 / 0.914 | 31.92 / 0.896 | 31.23 / 0.914 | 412 | 91.2 |
| | SRMDNF [56] | manual | 37.79 / 0.960 | 33.32 / 0.915 | 32.05 / 0.898 | 31.33 / 0.920 | 1,513 | 347.7 |
| | MSWSR [57] | manual | 37.49 / 0.958 | 33.23 / 0.912 | 31.88 / 0.893 | 31.14 / 0.917 | 1,228 | 192.5 |
| | EDSR [3]† | manual | 37.99 / 0.960 | 33.57 / 0.918 | 32.16 / 0.899 | 31.98 / 0.927 | 1,370 | 315.1 |
| | OISR-RK2-s [58] | manual | 37.98 / 0.960 | 33.58 / 0.917 | 32.18 / 0.900 | 32.09 / 0.928 | 1,372 | 316.2 |
| | RCAN [59]* | manual | 38.07 / 0.961 | 33.73 / 0.919 | 32.20 / 0.900 | 32.28 / 0.930 | 1,746 | 400.0 |
| | SAN [60]* | manual | 38.08 / 0.961 | 33.71 / 0.919 | 32.21 / 0.901 | 32.26 / 0.930 | 1,696 | 390.6 |
| | MSAN [61] | manual | 38.05 / 0.961 | 33.72 / 0.919 | 32.21 / 0.900 | 32.36 / 0.930 | 1,760 | 392.2 |
| | WDSR [10]* | manual | 37.84 / 0.960 | 33.42 / 0.916 | 32.07 / 0.899 | 31.64 / 0.924 | 724 | 166.8 |
| | SMSR [62] | manual | 38.00 / 0.960 | 33.64 / 0.918 | 32.17 / 0.899 | 32.19 / 0.929 | 985 | 131.6 |
| | IMDN [9] | manual | 38.00 / 0.961 | 33.63 / 0.918 | 32.19 / 0.900 | 32.17 / 0.928 | 694 | 159.4 |
| | RFDN [8] | manual | 38.05 / 0.961 | 33.68 / 0.918 | 32.16 / 0.899 | 32.12 / 0.928 | 534 | 123.3 |
| | MoreMNAS [11]† | RL+evolution | 37.63 / 0.958 | 33.23 / 0.914 | 31.95 / 0.896 | 31.24 / 0.919 | 1,039 | 238.6 |
| | FALSR [12]† | RL+evolution | 37.61 / 0.959 | 33.29 / 0.914 | 31.97 / 0.897 | 31.28 / 0.919 | 326 | 74.7 |
| | ESRN [14]† | evolution | 37.85 / 0.960 | 33.42 / 0.916 | 32.10 / 0.899 | 31.79 / 0.925 | 324 | 73.4 |
| | HNAS [13]† | RL | 37.92 / 0.960 | 33.46 / 0.917 | 32.08 / 0.898 | 31.66 / 0.924 | - | 48.2 |
| | **FGNAS-S** (Ours) | | 37.94 / 0.960 | 33.50 / 0.917 | 32.14 / 0.900 | 31.87 / 0.926 | 212 | 46.6 |
| | **FGNAS-M** (Ours) | gradient-based | 38.11 / 0.961 | 33.77 / 0.919 | 32.25 / 0.901 | 32.41 / 0.931 | 1,614 | 371.8 |
| | **FGNAS-L** (Ours) | | 38.16 / 0.961 | 33.86 / 0.920 | 32.29 / 0.901 | 32.60 / 0.933 | 2,802 | 645.6 |
| ×3 | SRCNN [1] | manual | 32.75 / 0.909 | 29.28 / 0.821 | 28.41 / 0.786 | 26.24 / 0.799 | 57 | 52.7 |
| | VDSR [2] | manual | 33.66 / 0.921 | 29.77 / 0.831 | 28.82 / 0.798 | 27.14 / 0.828 | 665 | 612.6 |
| | BTSRN [31] | manual | 34.03 / - | 29.90 / - | 28.97 / - | 27.75 / - | 820 | 352.4 |
| | MemNet [55] | manual | 34.09 / 0.925 | 30.00 / 0.835 | 28.96 / 0.800 | 27.56 / 0.838 | 677 | 2,662.4 |
| | CARN [7]† | manual | 33.99 / 0.924 | 30.08 / 0.837 | 28.91 / 0.800 | 27.55 / 0.839 | 412 | 46.1 |
| | SRMDNF [56] | manual | 34.12 / 0.925 | 30.04 / 0.837 | 28.97 / 0.803 | 27.57 / 0.840 | 1,246 | 166.4 |
| | EDSR [3]† | manual | 34.37 / 0.927 | 30.28 / 0.842 | 29.09 / 0.805 | 28.15 / 0.853 | 1,551 | 158.9 |
| | OISR-RK2-s [58] | manual | 34.43 / 0.927 | 30.33 / 0.842 | 29.10 / 0.805 | 28.20 / 0.853 | 1,557 | 160.1 |
| | RCAN [59]* | manual | 34.47 / 0.927 | 30.38 / 0.843 | 29.13 / 0.807 | 28.30 / 0.855 | 1,931 | 196.6 |
| | SAN [60]* | manual | 34.44 / 0.927 | 30.40 / 0.844 | 29.14 / 0.807 | 28.34 / 0.856 | 1,880 | 192.5 |
| | MSAN [61] | manual | 34.45 / 0.926 | 30.39 / 0.844 | 29.12 / 0.806 | 28.23 / 0.854 | 1,770 | 175.8 |
| | WDSR [10]* | manual | 32.27 / 0.896 | 28.67 / 0.784 | 27.64 / 0.738 | 26.26 / 0.791 | 1,196 | 121.4 |
| | SMSR [62] | manual | 34.40 / 0.927 | 30.33 / 0.841 | 29.10 / 0.805 | 28.25 / 0.854 | 993 | 67.8 |
| | IMDN [9] | manual | 34.36 / 0.927 | 30.32 / 0.842 | 29.09 / 0.805 | 28.17 / 0.852 | 703 | 71.9 |
| | RFDN [8] | manual | 34.41 / 0.927 | 30.34 / 0.842 | 29.09 / 0.805 | 28.21 / 0.853 | 541 | 55.6 |
| | ESRN [14]† | evolution | 34.23 / 0.926 | 30.27 / 0.840 | 29.03 / 0.804 | 27.95 / 0.848 | 324 | 36.2 |
| | **FGNAS-S** (Ours) | | 34.33 / 0.926 | 30.24 / 0.841 | 29.03 / 0.804 | 27.90 / 0.848 | 221 | 22.3 |
| | **FGNAS-M** (Ours) | gradient-based | 34.51 / 0.928 | 30.42 / 0.845 | 29.16 / 0.807 | 28.40 / 0.858 | 1,622 | 166.1 |
| | **FGNAS-L** (Ours) | | 34.63 / 0.929 | 30.45 / 0.845 | 29.20 / 0.809 | 28.55 / 0.861 | 2,811 | 287.8 |
| ×4 | SRCNN [1] | manual | 30.48 / 0.869 | 27.49 / 0.750 | 26.90 / 0.710 | 24.52 / 0.722 | 57 | 52.7 |
| | VDSR [2] | manual | 31.35 / 0.884 | 28.01 / 0.767 | 27.29 / 0.725 | 25.18 / 0.752 | 665 | 612.6 |
| | BTSRN [31] | manual | 31.85 / - | 28.20 / - | 27.47 / - | 25.74 / - | 820 | 330.3 |
| | MemNet [55] | manual | 31.74 / 0.889 | 28.26 / 0.772 | 27.40 / 0.728 | 25.50 / 0.763 | 677 | 2662.4 |
| | CARN [7]† | manual | 31.92 / 0.890 | 28.42 / 0.776 | 27.44 / 0.730 | 25.62 / 0.769 | 412 | 32.5 |
| | SRMDNF [56] | manual | 31.96 / 0.893 | 28.35 / 0.777 | 27.49 / 0.734 | 25.68 / 0.773 | 1,249 | 95.5 |
| | MSWSR [57] | manual | 32.01 / 0.891 | 28.47 / 0.778 | 27.48 / 0.731 | 25.78 / 0.774 | 1,228 | 217.1 |
| | SRResNet [4] | manual | 32.05 / 0.891 | 28.53 / 0.780 | 27.57 / 0.735 | 26.07 / 0.784 | 1,810 | 104.2 |
| | EDSR [3]† | manual | 32.09 / 0.894 | 28.58 / 0.781 | 27.57 / 0.736 | 26.04 / 0.785 | 1,810 | 104.2 |
| | OISR-RK2-s [58] | manual | 32.21 / 0.895 | 28.63 / 0.782 | 27.58 / 0.736 | 26.14 / 0.787 | 1,520 | 114.2 |
| | RCAN [59]* | manual | 32.23 / 0.894 | 28.62 / 0.782 | 27.57 / 0.737 | 26.13 / 0.787 | 2,189 | 125.5 |
| | SAN [60]* | manual | 32.24 / 0.894 | 28.64 / 0.782 | 27.58 / 0.737 | 26.14 / 0.787 | 2,138 | 123.1 |
| | MSAN [61] | manual | 32.32 / 0.896 | 28.66 / 0.783 | 27.59 / 0.737 | 26.12 / 0.788 | 1,784 | 99.8 |
| | WDSR [10]* | manual | 31.94 / 0.890 | 28.44 / 0.777 | 27.46 / 0.733 | 25.74 / 0.773 | 731 | 42.1 |
| | SMSR [62] | manual | 32.12 / 0.893 | 28.55 / 0.781 | 27.55 / 0.735 | 26.11 / 0.787 | 1,006 | 41.6 |
| | IMDN [9] | manual | 32.21 / 0.895 | 28.58 / 0.781 | 27.56 / 0.735 | 26.04 / 0.784 | 715 | 41.5 |
| | RFDN [8] | manual | 32.24 / 0.895 | 28.61 / 0.782 | 27.57 / 0.736 | 26.11 / 0.786 | 550 | 32.1 |
| | ESRN [14]† | evolution | 31.99 / 0.892 | 28.49 / 0.778 | 27.50 / 0.733 | 25.87 / 0.778 | 324 | 20.7 |
| | **FGNAS-S** (Ours) | | 31.97 / 0.890 | 28.44 / 0.778 | 27.46 / 0.734 | 25.76 / 0.775 | 233 | 12.8 |
| | **FGNAS-M** (Ours) | gradient-based | 32.32 / 0.895 | 28.67 / 0.784 | 27.62 / 0.738 | 26.23 / 0.791 | 1,633 | 94.1 |
| | **FGNAS-L** (Ours) | | 32.36 / 0.896 | 28.72 / 0.785 | 27.65 / 0.739 | 26.33 / 0.794 | 2,823 | 162.6 |

†: These works report multiple models for different resource usages. We compare scores of the models with the most similar Mult-Adds to FGNAS.
-: The scores were not reported in the reference papers and are impossible to reproduce since the reference papers missed some information of architecture details or source code is not available to access.
*: We reproduce light-weighted versions of these works, which have more than 15 M parameters, by reducing the number of convolutional blocks.




(a) PSNR *vs.* Mult-Adds (b) PSNR *vs.* Parameters (c) PSNR *vs.* latency

**Fig. 3. Accuracy-efficiency trade-off comparisons on B100 with the scale factor of 2. Our model (FGNAS) demonstrates the better trade-off between (a) PSNR *vs.* Mult-Adds, (b) PSNR *vs.* Parameters, and (c) PSNR *vs.* latency than state-of-the-art methods.**

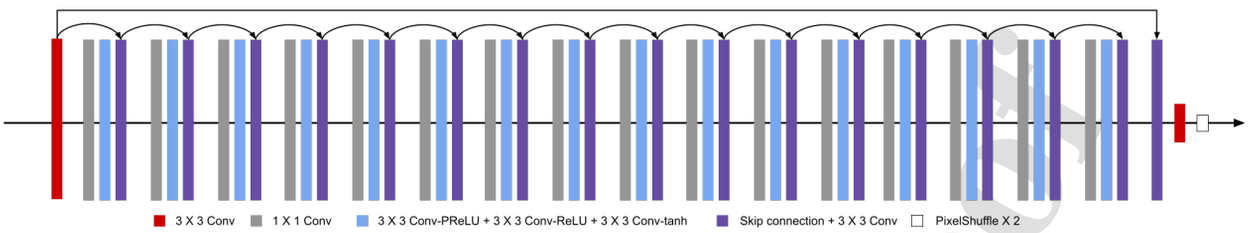
**Table 3. Performances for ClassSR-style networks on Test2K, Test4K, and Test8K. ClassSR-: the ClassSR-style network. Red/Blue text: best performances in PSNR/FLOPs.**

| Model | Parameters | Test2K | | Test4K | | Test8K | |
|---|---|---|---|---|---|---|---|
| | | PSNR | FLOPs | PSNR | FLOPs | PSNR | FLOPs |
| FSRCNN | 25 K | 25.61 dB | 468 M (100%) | 26.90 dB | 468 M (100%) | 32.66 dB | 468 M (100%) |
| ClassSR-FSRCNN | 113 K | 25.61 dB | 311 M (66%) | 26.91 dB | 286 M (61%) | 32.73 dB | 238 M (51%) |
| CARN | 295 K | 25.95 dB | 1.15 G (100%) | 27.34 dB | 1.15 G (100%) | 33.18 dB | 1.15 G (100%) |
| ClassSR-CARN | 645 K | 26.01 dB | 814 M (71%) | 27.42 dB | 742 M (64%) | 33.24 dB | 608 M (53%) |
| SRResNet | 1.5 M | 26.19 dB | 5.20 G (100%) | 27.65 dB | 5.20 G (100%) | 33.50 dB | 5.20 G (100%) |
| ClassSR-SRResNet | 3.1 M | 26.20 dB | 3.62 G (70%) | 27.66 dB | 3.30 G (63%) | 33.50 dB | 2.70 G (52%) |
| RCAN | 15.6 M | 26.39 dB | 32.60 G (100%) | 27.89 dB | 32.60 G (100%) | 33.76 dB | 32.60 G (100%) |
| ClassSR-RCAN | 30.1 M | 26.39 dB | 21.22 G (65%) | 27.88 dB | 19.49 G (60%) | 33.73 dB | 16.36 G (50%) |
| FGNAS-L (Ours) | 2.8 M | 26.30 dB | 5.78 G (100%) | 27.77 dB | 5.78 G (100%) | 33.64 dB | 5.78 G (100%) |
| ClassSR-FGNAS (Ours) | 4.7 M | 26.32 dB | 3.19 G (55%) | 27.79 dB | 2.77 G (48%) | 33.65 dB | 1.98 G (34%) |

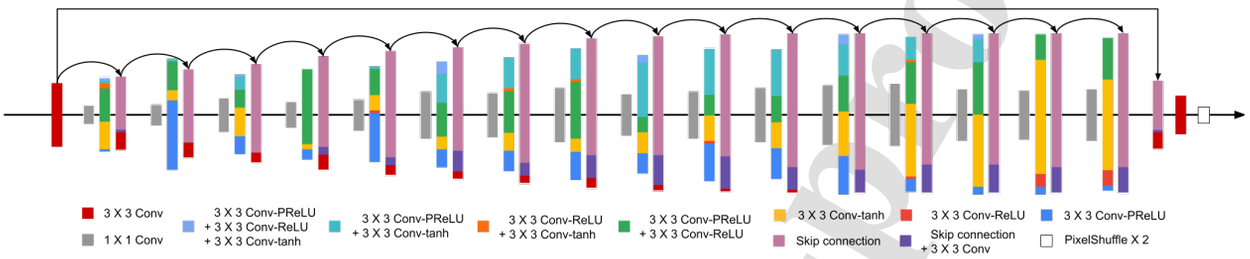
FLOPs, and parameters). Specifically, FGNAS-S outperforms HNAS [13] over 0.21 dB on Urban100 with similar Mult-Adds. Interestingly, FGNAS-S requires smaller Mult-Adds than SRCNN [1] which is the first deep-learning method for image super-resolution consisting of only 3 convolution layers. MXDSIR [29] is recently proposed for the lightweight neural network, but our FGNAS-S achieves the better image restoration performances with fewer parameters except Set14. Note that the performances on Set14 might be difficult to accept as the general image restoration accuracy of a model since Set14 only contains 14 images while B100 and Urban100 have 100 images in each dataset and Set14 also includes the 5 images on Set5 where FGNAS-S outperforms MXDSIR.

*Efficiency comparisons.* Recent state-of-the-art models report their best restoration accuracy (*e.g.*, PSNR and SSIM) with customized resources (*e.g.*, Mult-Adds and parameters). For thorough comparisons, we visualize the accuracy-efficiency trade-off of the models in Figure 3(a) and (b). SMSR [62], OISR-RK2-s [58], and MSAN [61] slightly outperform the PSNR result of FGNAS on B100 dataest by 0.03 dB, 0.04 dB, and 0.07 dB, respectively. However, they use 3×, 6×, and 9× more Mult-Adds with 4×, 6×, and 8× more parameters compared to FGNAS. In contrast, FSRCNN [27] and LapSRN [5] require smaller Mult-Adds, but PSNR results are 0.91 dB and 0.34 dB lower than FGNAS. FGNAS outperforms all compared method in accuracy-efficiency trade-offs. Note that HNAS [13] does not report the model parameters. Moreover, we compare the running time of several state-of-the-art methods in Figure 3(c). IMDN, EDSR, and RCAN achieve better PSNR (0.07 dB), but FGNAS runs 0.3 s ∼ 1.0 s faster. This phenomenon is because FGNAS searches for the activation functions which require negligible computation costs at the forward pass. We use official codes to measure running time for a feed-forward process. We measure the same models compared in Table 2.

*ClassSR results.* ClassSR [41] is a general framework for SR networks that improves efficiency by patch-wise inferences. A classifier in the framework identifies the restoration difficulty of each image patch and selects one of three SR models, which have different computational costs. While ClassSR designed the SR models by manually scaling the number of channels in the existing network, we searched three SR models, FGNAS-S, FGNAS-M, and FGNAS-L. Table 3 presents the ClassSR-style performances. FGNAS-L is a searched SR model with 64 searchable blocks, and ClassSR-FGNAS uses the three searched models to demonstrate ClassSR-style performances. ClassSR-FGNAS outperforms ClassSR-SRResNet in PSNR and FLOPs for all datasets. Notably, ClassSR-FGNAS achieves the lowest FLOPs ratios to the single SR model (34% in Test8K) while outperforming it in PSNR. This phenomenon is because our searched SR models have better PSNR-FLOPs efficiency than other models.




(a) Supernet




(b) FGNAS

**Fig. 4. Network architecture visualization. Our final model (FGNAS) is searched by pruning channels of (sub-)feature maps in the supernet. Each block represents the feature map of a layer. The height of blocks depicts the number of channels. "+" denotes element-wise sum (or addition) of sub-feature maps from the corresponding operations.**

**Table 4. Ablation study of search space.**

| Ex. # | #Channel search | Activation search | Urban100 (PSNR) | Mult-Adds (G) |
|---|---|---|---|---|
| (1) Supernet | - | - | 30.92 dB | 292.3 |
| (2) | ✓ | - | 30.84 dB | 50.0 |
| (3) | - | ✓ | **32.23 dB** | 264.5 |
| (4) FGNAS (Ours) | ✓ | ✓ | 31.87 dB | **46.2** |

#Channel search reduces FLOP significantly, and activation search enables accurate image restoration.

**Table 5. Ablation study of the residual block architecture.**

| Ex.# | Point-wise Conv | Multiple activation | Urban100 (PSNR) | Mult-Adds (G) |
|---|---|---|---|---|
| (1) Residual Block of EDSR [3] | - | - | 31.79 dB | 97.2 |
| (2) | ✓ | - | 31.71 dB | **46.2** |
| (3) | - | ✓ | **31.93 dB** | 97.2 |
| (4) Searchable Block (Ours) | ✓ | ✓ | 31.87 dB | **46.2** |

Point-wise conv further reduces Mult-Adds by #channel search, and multiple activation functions improve PSNR scores through activation search.

### *4.3. Searched Architecture Visualization*

We propose a supernet-based neural architecture search (NAS) algorithm that finds the number of channels for each layer and the activation function for each channel. Figure 4(a) visualizes our supernet architecture described in Table 1. The supernet consists of 16 searchable blocks using 1×1 convolution, 3×3 convolution with ReLU, PReLU, and tanh activation functions, and 3×3 convolution with skip connection (See Figure 1). Our search algorithm determines where to prune channels of feature maps from convolution and channels of sub-feature maps from activation functions as illustrated in Figure 1. As a result, the searched network, called FGNAS, has multiple heterogeneous activation functions of a layer and non-repeated residual block structures, as visualized in Figure 4(b). The searched block has large numbers of channels for activation layers and skip connections, while the other convolutions (*e.g.*, 1×1 Conv and 3×3 Conv+Skip connection) generate relatively small numbers of channels. The architecture of FGNAS has gradually increasing channels in searched blocks.

### *4.4. Ablation Study*

This section describes the contribution of the proposed search space and the searchable block architecture. Table 4 presents an ablation study of search space which includes the number of channel search and the activation search with the same supernet structure. The supernet without search process (See Table 4(1)) performs poor restoration accuracy since the heterogeneous activation function of ReLU, PReLU, and tanh make the training process unstable. #Channel search in Table 4(2) relocates the pruning processes from sub-feature maps for each activation function to the feature map for the 'Mult' operation in Figure 1 to prevent activation search. Table 4(2) performs low restoration accuracy due to the heterogeneous activation function while reducing the model Mult-Adds by chan-

所以本质上：
① 把 act 拍成卷积通道打孔的对
② 预定义拓扑为剪枝的通道

nel pruning. Activation search in Table 4(3) indicates activating the pruning processes for activation functions in the searchable block without resource regularizer described in Equation 2. Table 4(3) presents over 1 dB PSNR improvement than supernet since each layer can has the *multiple* heterogeneous activation functions. Using both #channel search and activation search, FGNAS achieves high reconstruction accuracy with low model Mult-Adds described in Table 4(4).

On the other hand, Table 5 illustrates an ablation study of the residual block structure using the same search process. Table 5(1) indicates the searched model using residual bock of EDSR [3] visualized in Figure 1(a). Based on this structure, point-wise convolution (or 1×1 convolution) achieves additional 50% Mult-Adds reduction by squeezing the number of channels for the following 3×3 convolution (See Table 5(2)) while multiple activation functions allow the searched model improving PSNR over 0.1 dB (See Table 5(3)). Our seachable block performs both Mult-Adds efficiency and accuracy improvement by using both point-wise convolution and multiple activation functions as visualized in Table 5(4).

## 5. Conclusion

We present a novel neural architecture search approach exploring a fine-grained search space for image super-resolution by pruning channels of our newly proposed supernet. The search space includes the number of channels per convolution for efficiency and the activation function per channel for accuracy. Our search process optimizes efficiency-accuracy tradeoff by gradient descent. To this end, we introduce the relaxation technique to estimate Mult-Adds of the searched models in a differentiable manner. The final model searched by the proposed algorithm, referred to as FGNAS, has layer-wise structures for efficiency, including multiple heterogeneous activation functions in a layer and many channels for activation and skip connection. FGNAS outperforms the state-of-the-art methods for image super-resolution in terms of Mult-Adds efficiency by a large margin.

## 6. Limitation and Future Research

The proposed NAS algorithm finds lightweight and computationally efficient neural networks for image super-resolution. However, the real-world scenario requires a single network for multiple tasks. For instance, image signal processor (ISP) in smartphones utilizes diverse low-level vision tasks, such as denoising, demosaicing, and tone mapping, with strict resource constraints of parameters, energy, and latency. Moreover, recent smartphones adopt neural networks for high-level vision tasks, such as object detection and segmentation. Finding a universal network for all tasks used in mobile devices could be an interesting future work, where each task might have a custom architecture to improve efficiency.

## References


[1] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *TPAMI*, 2014.

[2] J. Kim, J. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in *CVPR*, 2016.

[3] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *CVPRW*, 2017.

[4] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *CVPR*, 2017.

[5] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Deep laplacian pyramid networks for fast and accurate super-resolution," in *CVPR*, 2017.

[6] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *CVPR*, 2016.

[7] N. Ahn, B. Kang, and K.-A. Sohn, "Fast, accurate, and lightweight super-resolution with cascading residual network," *arXiv preprint*, vol. arXiv:1803.08664, 2018.

[8] J. Liu, J. Tang, and G. Wu, "Residual feature distillation network for lightweight image super-resolution," in *ECCVW*, 2020.

[9] Z. Hui, X. Gao, Y. Yang, and X. Wang, "Lightweight image super-resolution with information multi-distillation network," in *ACMMM*, 2019.

[10] J. Yu, Y. Fan, J. Yang, N. Xu, Z. Wang, X. Wang, and T. Huang, "Wide activation for efficient and accurate image super-resolution," in *BMVC*, 2018.

[11] X. Chu, B. Zhang, R. Xu, and H. Ma, "Multi-objective reinforced evolution in mobile neural architecture search," in *ECCVW*, 2020.

[12] X. Chu, B. Zhang, H. Ma, R. Xu, J. Li, and Q. Li, "Fast, accurate and lightweight super-resolution with neural architecture search," *arXiv preprint*, vol. arXiv:1901.07261, 2019.

[13] Y. Guo, Y. Luo, Z. He, J. Huang, and J. Chen, "Hierarchical neural architecture search for single image super-resolution," *SPL*, 2020.

[14] D. Song, C. Xu, X. Jia, Y. Chen, C. Xu, and Y. Wang, "Efficient residual dense block search for image super-resolution," in *AAAI*, 2020.

[15] M. Tan, B. Chen, R. Pang, V. Vasudevan, and Q. V. Le, "Mnasnet: Platform-aware neural architecture search for mobile," in *CVPR*, 2019.

[16] H. Cai, L. Zhu, and S. Han, "ProxylessNAS: Direct neural architecture search on target task and hardware," in *ICLR*, 2019.

[17] B. Wu, X. Dai, P. Zhang, Y. Wang, F. Sun, Y. Wu, Y. Tian, P. Vajda, Y. Jia, and K. Keutzer, "Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search," in *CVPR*, 2019.

[18] J. Mei, Y. Li, X. Lian, X. Jin, L. Yang, A. Yuille, and J. Yang, "Atomnas: Fine-grained end-to-end neural architecture search," in *ICLR*, 2020.

[19] M. Tan and Q. V. Le, "Mixconv: Mixed depthwise convolutional kernels," in *BMVC*, 2019.

[20] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *CVPR*, 2018.

[21] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *CVPR*, 2018.

[22] A. Howard, M. Sandler, G. Chu, L. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam, "Searching for mobilenetv3," in *ICCV*, 2019.

[23] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *ICML*, 2019.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[25] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," in *NIPS*, 2014.

[26] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image super-resolution," in *CVPR*, 2018.

[27] C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," in *ECCV*, 2016.

[28] M. Liu, Z. Zhang, L. Hou, W. Zuo, and L. Zhang, "Deep adaptive inference networks for single image super-resolution," in *ECCVW*, 2020.

[29] T. O. Wazir Muhammad, Supavadee Aramvith, "Multi-scale xception based depthwise separable convolution for single image super-resolutio," *PLOS ONE*, 2021.

[30] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint*, vol. arXiv:1704.04861, 2017.

[31] Y. Fan, H. Shi, J. Yu, D. Liu, W. Han, H. Yu, Z. Wang, X. Wang, and T. S. Huang, "Balanced two-stage residual networks for image super-resolution," in *CVPRW*, 2017.

[32] Y. Jing, Y. Yang, X. Wang, M. Song, and D. Tao, "Turning frequency to resolution: Video super-resolution via event cameras," in *CVPR*, 2021.

[33] Y. Jing, X. Liu, Y. Ding, X. Wang, E. Ding, M. Song, and S. Wen, "Dynamic instance normalization for arbitrary style transfer," in *AAAI*, 2020.

[34] C. Hong, H. Kim, S. Baik, J. Oh, and K. M. Lee, "Daq: Channel-wise distribution-aware quantization for deep image super-resolution networks," in *WACV*, 2022.

[35] C. Hong, S. Baik, H. Kim, S. Nah, and K. M. Lee, "Contents-aware dynamic quantization for image super-resolution," in *ECCV*, 2022.

[36] J. Oh, H. Kim, S. Nah, C. Hong, J. Choi, and K. M. Lee, "Attentive fine-grained structured sparsity for image restoration," in *CVPR*, 2022.

[37] H. Kim, S. Baik, M. Choi, J. Choi, and K. M. Lee, "Searching for controllable image restoration networks," in *ICCV*, 2021.

[38] Y. Yang, J. Qiu, M. Song, D. Tao, and X. Wang, "Distilling knowledge from graph convolutional networks," in *CVPR*, 2020.

[39] Y. Jing, Y. Yang, X. Wang, M. Song, and D. Tao, "Amalgamating knowledge from heterogeneous graph neural networks," in *CVPR*, 2021.

[40] Y. Yang, Z. Feng, M. Song, and X. Wang, "Factorizable graph convolutional networks," in *NeurIPS*, 2020.

[41] X. Kong, H. Zhao, Y. Qiao, and C. Dong, "Classsr: A general framework to accelerate super-resolution networks by data characteristic," in *CVPR*, 2021.

[42] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," in *ICLR*, 2017.

[43] B. Baker, O. Gupta, N. Naik, and R. Raskar, "Designing neural network architectures using reinforcement learning," in *ICLR*, 2017.

[44] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean, "Efficient neural architecture search via parameter sharing," in *PMLR*, 2018.

[45] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L.-J. Li, L. Fei-Fei, A. Yuille, J. Huang, and K. Murphy, "Progressive neural architecture search," in *ECCV*, 2018.

[46] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *AAAI*, 2019.

[47] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable architecture search," in *ICLR*, 2019.

[48] T.-J. Yang, A. Howard, B. Chen, X. Zhang, A. Go, M. Sandler, V. Sze, and H. Adam, "Netadapt: Platform-aware neural network adaptation for mobile applications," in *ECCV*, 2018.

[49] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *CVPR*, 2018.

[50] E. Agustsson and R. Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *CVPRW*, 2017.

[51] M. Bevilacqua, A. Roumy, C. Guillemot, and M. Alberi-Morel, "Low-complexity single-image super-resolution based on nonnegative neighbor embedding," in *BMVC*, 2012.

[52] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *ICCS*, 2010.

[53] D. R. Martin, C. C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *ICCV*, 2001.

[54] J. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *CVPR*, 2015.

[55] Y. Tai, J. Yang, X. Liu, and C. Xu, "Memnet: A persistent memory network for image restoration," in *ICCV*, 2017.

[56] K. Zhang, W. Zuo, and L. Zhang, "Learning a single convolutional super-resolution network for multiple degradations," in *CVPR*, 2018.

[57] H. Zhang, J. Xiao, and Z. Jin, "Multi-scale image super-resolution via a single extendable deep network," *JSTSP*, 2021.

[58] X. He, Z. Mo, P. Wang, Y. Liu, M. Yang, and J. Cheng, "Ode-inspired network design for single image super-resolution," in *CVPR*, 2019.

[59] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super-resolution using very deep residual channel attention networks," in *ECCV*, 2018.

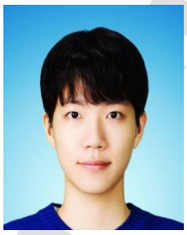
[60] T. Dai, J. Cai, Y. Zhang, S.-T. Xia, and L. Zhang, "Second-order attention network for single image super-resolution," in *CVPR*, 2019.

[61] L. Wang, J. Shen, E. Tang, S. Zheng, and L. Xu, "Multi-scale attention network for image super-resolution," *JVCIR*, 2021.

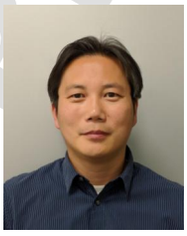
[62] L. Wang, X. Dong, Y. Wang, X. Ying, Z. Lin, W. An, and Y. Guo, "Exploring sparsity in image super-resolution for efficient inference," in *CVPR*, 2021.

[63] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

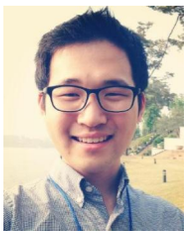

**Heewon Kim** received the BS degree in the Department of Electrical and Computer Engineering at Seoul National University, Korea, in 2014. Currently, he is working toward the PhD degree in the Department of Electrical and Computer Engineering at Seoul National University. He was a Lieutenant of Republic of Korea Army from 2014 to 2016 and a Research Intern at Qualcomm in 2018 and NVIDIA in 2022. He received several awards, in particular, Outstanding Reviewer Award in ECCV 2020, Runner-Up Award in AIM 2019 Challenge on Video Temporal Super-Resolution, and Winner Award and Best Paper Award in NTIRE 2017 Challenge Track. He served as a reviewer more than 20 times in refereed journals and conferences including TPAMI, CVPR, ECCV, ICCV, NeurIPS, ICLR, and AAAI.



**Seokil Hong** received the BS degree in the Department of Electrical and Computer Engineering at Seoul National University, Korea, in 2018. Currently, he is working toward the PhD degree in the Department of Electrical and Computer Engineering at Seoul National University. His research interests include computer vision, machine learning, and deep learning.



**Bohyung Han** received the BS and MS degrees from the Department of Computer Engineering at Seoul National University, Korea, in 1997 and 2000, respectively, and the PhD degree from the Department of Computer Science at the University of Maryland, College Park, MD, in 2005. He was with the research staff at the Samsung Electronics Research and Development Center, Irvine, CA, and Mobileye Vision Technologies, Princeton, NJ. He was an associate professor with the Department of Computer Science and Engineering at POSTECH, Pohang, Korea. He is a professor with the Department of Electrical and Computer Engineering at Seoul National University, Korea. He served (or will serve) as a general chair of ACCV 2022, a demo chair of ECCV 2022, a workshop chair of CVPR 2021 and ACCV 2929, a tutorial chair of ICCV 2019, an organizing chair of CVPR 2018 and ACCV 2012, a demo chair of ACCV 2014, and an area chair of NIPS/NeurIPS (2015, 2018, 2019), ICCV (2015, 2017, 2019), CVPR 2017, ECCV 2020, ACCV (2012, 2014, 2016, 2018), WACV (2014, 2017, 2021), and ACML 2016, AVSS 2018. His current research interests include computer vision, machine learning, pattern recognition, and computer graphics. More information can be found on his homepage http://cv.snu.ac.kr/bhhan.



**Heesoo Myeong** received the BS and PhD degrees in the Department of Electrical and Computer Engineering at Seoul National University, Korea, in 2009 and 2017, respectively. He is currently a machine learning engineer at Qualcomm Korea YH, Seoul, Korea. His research interests include computer vision and machine learning for autonomous driving systems.

**Kyoung Mu Lee** (Fellow, IEEE) is currently the Editor in Chief of the IEEE TPAMI. He received the B.S. and M.S. degrees in control and instrumentation engineering from Seoul National University (SNU), Seoul, South Korea, in 1984 and 1986, respectively, and the Ph.D. degree in electrical engineering from the University of Southern California, in 1993. He is the director of the Interdisciplinary Graduate Program in Artificial Intelligence at SNU. He is an Advisory Board Member of the Computer Vision Foundation (CVF). He was a Distinguished Lecturer of the Asia-Pacific Signal and Information Processing Association (APSIPA), from 2012 to 2013. He has received several awards, in particular, the Medal of Merit and the Scientist of Engineers of the Month Award from the Korean Government, in 2018 and 2020, respectively; the Most Influential Paper Over the Decade Award by the IAPR Machine Vision Application, in 2009; the ACCV Honorable Mention Award, in 2007; the Okawa Foundation Research Grant Award, in 2006; the Distinguished Professor Award from the College of Engineering of SNU, in 2009; and the SNU Excellence in Research Award in 2020. He has also served as a General Chair for ICCV2019, ACMMM2018, and ACCV2018; a Program Chair for ACCV2012; a Track Chair for ICPR2020 and ICPR2012; and an Area Chair for CVPR, ICCV, and ECCV many times. He has served as an Associate Editor-in-Chief (AEIC) and an Associate Editor for the Machine Vision and Application (MVA) journal, the IPSJ Transactions on Computer Vision and Applications (CVA), and the IEEE SIGNAL PROCESSING LETTERS (SPL); and an Area Editor for the Computer Vision and Image Understanding (CVIU). He is the founding member and served as the President of the Korean Computer Vision Society (KCVS). Prof. Lee is a Fellow of IEEE, a member of the Korean Academy of Science and Technology (KAST) and the National Academy of Engineering of Korea (NAEK). More information can be found on his homepage http://cv.snu.ac.kr/kmlee.

**Declaration of interests**

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐ The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: