NAS-DIP: Learning Deep Image Prior with Neural Architecture Search

Yun-Chun Chen*, Chen Gao*, Esther Robb, and Jia-Bin Huang

Virginia Tech

Fig. 1: **Applications.** We propose to *learn* deep image prior using a neural architecture search. The resulting network can be applied to solve various inverse image problems *without* pre-training the model with a large-scale dataset with ground truth. Through extensive experimental evaluations, we show that our model compares favorably against existing hand-crafted CNN models for learning-free image restoration tasks. In some cases, our model even reaches competitive performance when compared with recent learning-based models.

Abstract. Recent work has shown that the structure of deep convolutional neural networks can be used as a structured image prior for solving various inverse image restoration tasks. Instead of using hand-designed architectures, we propose to search for neural architectures that capture stronger image priors. Building upon a generic U-Net architecture, our core contribution lies in designing new search spaces for (1) an upsampling cell and (2) a pattern of cross-scale residual connections. We search for an improved network by leveraging an existing neural architecture search algorithm (using reinforcement learning with a recurrent neural network controller). We validate the effectiveness of our method via a wide variety of applications, including image restoration, dehazing, image-to-image translation, and matrix factorization. Extensive experimental results show that our algorithm performs favorably against state-of-the-art learning-free approaches and reaches competitive performance with existing learning-based methods in some cases.

1 Introduction

Convolutional neural networks (CNNs) have been successfully applied to various computer vision tasks. Apart from visual recognition tasks, CNNs have also demonstrated strong performance in restoration and synthesis problems. The

* equal contribution

reason behind these successful stories is often attributed to the ability of CNNs to *learn priors* from large-scale datasets (i.e., the priors are embedded in the *parameters*/*weights* of the trained network). In contrast to existing supervised learning paradigms that require learning the network parameters from labeled datasets, recent studies have discovered that the *structure* of the network by itself is sufficient to capture rich low-level image statistics [69, 78]. Such structured image priors encoded in the network architecture are critical for image restoration (e.g., single image super-resolution [44, 45, 46], image denoising [12, 49, 82], joint filtering [50]), and image synthesis (e.g., image-to-image translation [37, 38, 47, 48, 91]) tasks.

While learning-free methods [78] have demonstrated competitive performance on image restoration tasks when compared with learning-based approaches [44,46], only conventional network architectures such as ResNet [31] or U-Net [68] have been evaluated. There are two important aspects of network designs for these image restoration problems. First, while the design of an encoder has been extensively studied [31,35,43,71], the design of a decoder [80,81] (the upsampling cell in particular) receives considerably less attention. Second, as the spatial resolution of the features is progressively reduced along the path of the feature encoder, it is crucial for the network to recover feature maps with higher spatial resolution. U-Net [68] is one popular design to address this issue by concatenating the encoded features at the corresponding encoder layer with the features in the decoder when performing a sequence of up-convolutions. Such skip connection patterns, however, are manually designed and fixed for each task at hand.

Our work. In this paper, we propose to *search* for both (1) the upsampling cells in the decoder and (2) the skip connection patterns between the encoder and the decoder (i.e., cross-level feature connections). To achieve this, we develop new search spaces for the two components. First, to search for the upsampling cell, we decompose a typical upsampling operation into two steps: i) ways of changing the spatial resolution (e.g., bilinear or bicubic [20]) and ii) ways of feature transformation (e.g., 2D convolution or 2D transposed convolution [86]). Second, to search for the cross-level connection patterns, we propose to search connection patterns shared across different feature levels in an encoder-decoder network.

Motivated by the Neural Architecture Search (NAS) algorithm [24, 26, 92] which has been shown effective in discovering networks with top performance in a large search space, we leverage reinforcement learning (RL) with a recurrent neural network (RNN) controller [24, 26, 92] and use the PSNR as the reward to guide the architecture search. By simultaneously searching in the two developed search spaces, our method is capable of discovering a CNN architecture that captures stronger structured image priors for the task of interest. We show the applicability of our method through four *learning-free* image restoration tasks, including single image super-resolution, image denoising, image inpainting, and image dehazing, and a *learning-based* unpaired image-to-image translation problem (see Figure 1). Our experimental results demonstrate that searching

for both the upsampling cell and the cross-level feature connections results in performance improvement over conventional neural architectures.

Our contributions. First, we present a decomposition based on several commonly used upsampling operators that allows us to search for a novel upsampling cell for each task. Second, we develop a search space that consists of patterns of cross-level feature connections in an encoder-decoder architecture. Third, extensive evaluations on a variety of image restoration and synthesis tasks demonstrate that our proposed algorithm compares favorably against existing learning-based methods in some cases and achieves state-of-the-art performance when compared with existing learning-free approaches.

2 Related Work

Upsampling cell. The design of the upsampling cell can be categorized into two groups: 1) non-learnable parameter-based methods and 2) learnable parameter-based approaches. *Non-learnable* parameter-based methods use interpolation to resize the feature maps from lower spatial resolutions to higher spatial resolutions. Example operators include bilinear/bicubic interpolation [20,62], nearest neighbor upsampling [9,39,62], and depth-to-space upsampling [46]. *Learnable* parameter-based approaches *learn* the mappings between feature maps of lower spatial resolutions and higher ones. Among the design choices, 2D transposed convolution is one of the popular choices for various dense prediction tasks, including semantic segmentation [56], optical flow estimation [21], depth prediction [15], image restoration [49,75], and synthesis [62] problems. Recent advances include bilinear additive upsampling [80,81] and CARAFE [79].

In contrast to these methods that manually design the upsampling operations, we develop a search space for the upsampling cell by decoupling several existing upsampling operations into methods of changing the spatial resolution and methods of feature transformation. We then adopt a Neural Architecture Search algorithm [24,26,92] (i.e., reinforcement learning with an RNN controller) to *automatically* discover the optimal upsampling cell for each individual task. We further demonstrate that the discovered upsampling cells can be transferred across tasks with favorable performance compared with the base network architectures.

NAS applications and search space. NAS algorithms have been successfully applied to various tasks, including image classification [65,66,92,93], semantic segmentation [14,52,61], object detection [24], image restoration [17,74], and image generation [26]. In the context of object detection, NAS-FPN [24] develops a search space that allows the model to learn pyramidal representations by merging cross-scale features for improved detection of multiple objects with different scales and locations. In semantic segmentation, several methods focus on searching for the encoder architecture [52], the Atrous Spatial Pyramid Pooling module [14], or the decoder cell for a compact architecture [61]. In image restoration tasks, existing algorithms aim at discovering a better encoder structure for single image super-resolution [17] or asymmetric encoder-decoder architecture for image

inpainting and image denoising [74]. In image generation problems, AutoGAN [26] adopts a progressive scheme to search for a better generative adversarial network (GAN) architecture [27]. When searching for the decoder architecture, other methods focus on searching for more compact architectures [61] or optimizing cell structures with hand-designed upsampling cells and feature connections [26, 74].

Our focus differs from these methods in two aspects. First, we develop a search space for the *upsampling cell*, allowing us to discover an optimal upsampling choice for each task at hand. Second, we search for a pattern of *cross-level feature connections* and share it across different feature levels in an encoder-decoder architecture. Our cross-level feature connections are different from those in NAS-FPN [24] in that we aim at recovering feature maps with higher spatial resolution in the decoder, whereas NAS-FPN [24] aims at learning pyramidal feature representations for object detection. By simultaneously searching for both the upsampling cell and the cross-level feature connections, our searched architecture achieves the state-of-the-art performance when compared with existing learning-free approaches on a variety of image restoration tasks, and also reaches competitive results when compared with existing learning-based algorithms.

NAS algorithms. NAS methods can be grouped into several categories depending on the search algorithm. The primary methods are evolution [5, 53, 59, 65, 66, 72, 73, 83], reinforcement learning [7, 13, 77, 90, 92, 93], and differentiable search [2, 54]. Evolutionary search leverages evolutionary algorithms to discover network structures by randomly mutating high-performing candidates from a population of architectures. RL-based approaches adopt either Q-learning [7, 90] or policy gradient [13, 77, 92, 93] strategies to train a recurrent network which outputs a sequence of symbols describing a network architecture [92] or a repeatable cell structure [93]. Differentiable search methods develop a continuous search space and optimize the network architecture using gradient descent, providing time efficiency at the expense of being memory intensive.

In this work, we follow the RL-based approaches and adopt an RL-based search algorithm with an RNN controller [26, 92] to search for the upsampling cell and the cross-level connection patterns. We note that the search algorithm is *not* limited to RL-based approaches. Other alternatives, such as evolutionary-based methods or differentiable architecture search algorithms, can also be applied. The focus of our paper lies in the design of the two search spaces. We leave the development of the search algorithm as future work.

Test-time training. Training CNNs on *testing data* has been shown as an effective approach for improving performance [55, 57, 76]. Our work also trains a CNN at the test time. Our method differs in that we do not rely on any *external* data at the training stage.

3 Proposed Method

In this section, we first provide an overview of the proposed method. We then describe the two developed search spaces for the upsampling cell and the cross-scale residual connections, respectively.

3.1 Method overview

In contrast to existing learning-based methods that learn directly from large-scale datasets (Figure 2a), recent studies [78] have shown that by randomly mapping noise to a degenerated (e.g., noisy, low-resolution, or occluded) image, the *untrained* CNN can solve the image restoration problems with competitive performance (Figure 2b). To discover network structures that capture stronger image priors, we consider the task of searching for an upsampling cell and a pattern of cross-scale residual connections *without* learning from paired data. To achieve this, we present an algorithm consisting of two steps. As shown in Figure 2c, we first apply reinforcement learning with an RNN controller [92] (using the PSNR as the reward) to search for the best-performing network structure f_θ^* on a held-out training set (blue block). After the network architecture search step, for each image in the test set, we randomly reinitialize the weights of the best-performing network structure f_θ^* and optimize the mapping from the random noise to a degenerated image (green block).

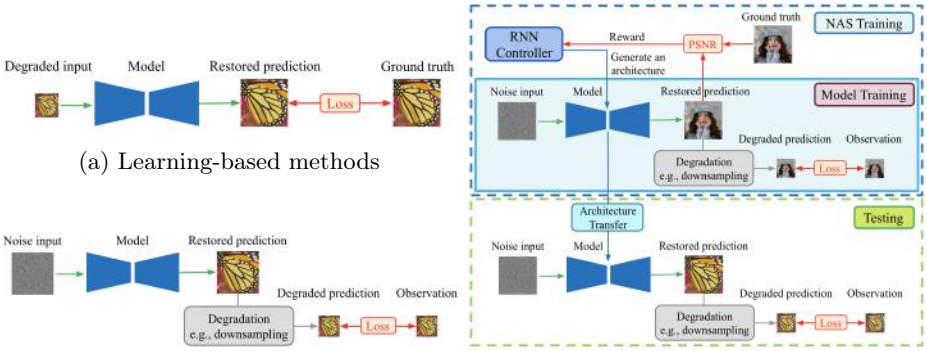
Searching for the best-performing network structure. Given an image $x \in \mathbb{R}^{H\times W\times 3}$ in the training set, we first generate a *degenerated* version x_0 by adding noise, downsampling, or dropping certain pixels from x depending on the task of interest. That is, for image denoising, $x_0 \in \mathbb{R}^{H\times W\times 3}$ denotes the noisy version of x, for single image super-resolution, $x_0 \in \mathbb{R}^{\frac{H}{r}\times\frac{W}{r}\times 3}$ denotes the low-resolution version of x where r represents the downsampling ratio, and for image inpainting, $x_0 \in \mathbb{R}^{H\times W\times 3}$ denotes the occluded version of x. We then sample a noise image $z \in \mathbb{R}^{H\times W\times C}$ and enforce the searched network f_θ to map the noise image z to the denoised, high-resolution, or inpainted version of x_0, i.e., map the noise image z to x.

To achieve this, we follow DIP [78] and optimize different objectives for different tasks. Please refer to the supplementary material for details.

As the ground-truth images in the training set are available, we can rank each of the searched network structure by computing the Peak Signal to Noise Ratio (PSNR) between the ground-truth image and the network's output (i.e., $f_\theta(z)$) and determine the best-performing network structure f_θ^* for the training set.

Determining the optimal stopping point t^*. We note that the optimal stopping point t^* (i.e., the number of iterations required) depends on the network structure. Since we have a held-out training set, we are able to estimate the best stopping point for each randomly generated network structure. We then rank all the sampled network structures by measuring the differences (i.e., computing the PSNR) between the recovered image and its corresponding ground truth (i.e., the original image from the training set). After that, we apply the best-performing network structure to the test set and report the results recorded at the optimal stopping point t^*.

Testing with the searched network structure f_θ^*. After searching on the training set, we apply the best-performing network structure f_θ^* with *random initialization* on each image in the test set at a time for optimization with t^* iterations using the same objective as DIP [78] for different tasks.



(a) Learning-based methods

(b) Deep Image Prior (DIP) [78]

(c) NAS-DIP (Ours)

Fig. 2: **Overview of the main workflow and comparison.** (a) **Learning-based methods**, e.g., super-resolution models such as LapSRN [44] or VDSR [41]. Given a dataset with labeled input/output pairs (e.g., low-resolution observation and the corresponding high-resolution image), this class of methods trains a deep CNN model to learn the mapping between the degraded input and its corresponding ground truth for the task of interest (e.g., super-resolution, denoising, etc.). (b) **Learning-free methods.** Given a noise input and an input observation, the DIP method [78] optimizes the model to produce a restored image that matches the given input observation after the specified image degradation model (e.g., downsampling). Here, the weights of the CNN model are *randomly initialized* (i.e., no need to train the model on a labeled dataset). (c) **NAS-DIP (Ours).** As DIP leverages CNN architectures as structured image priors, we explore ways to *learn* such priors. Specifically, we develop two search spaces (one for the upsampling cell and the other for the cross-level feature connections) and leverage existing neural architecture search techniques (an RNN-based controller trained with reinforcement learning) with PSNR as our reward to search for an improved network structure on a held-out training set (blue block). After the network architecture search, we then transfer the best-performing architecture and optimize the model the same way as DIP (green block).

3.2 Search space for the upsampling layer

We develop a search space for the upsampling layer by decomposing existing upsampling operations based on two steps: 1) methods of changing the spatial resolution of the feature map and 2) methods of feature transformation. Our search space of operations for changing the spatial resolution includes: bilinear upsampling, bicubic upsampling [20], nearest-neighbor interpolation, depth-to-space [46, 70], and stride 2 transposed convolution [21, 56, 86]. Our search space of operations for feature transformation includes: 2D convolution, add every N consecutive channels [80, 81], separable convolution [28, 80, 81], depth-wise convolution [23, 28, 29], and identity. To relax the degree of freedom during the network architecture search, our search space allows the operations that contain

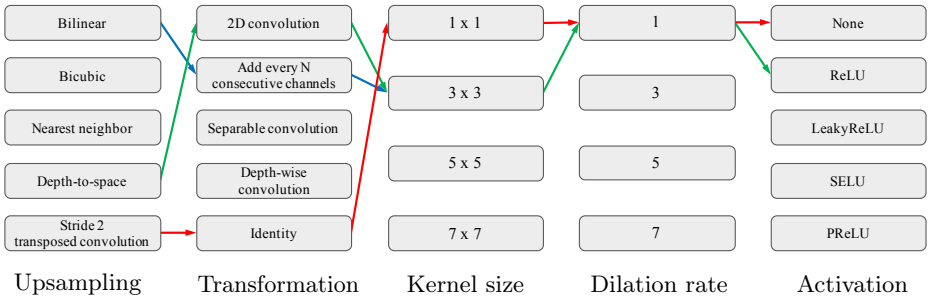


Fig. 3: **Search space for the upsampling cell.** Our search space consists of five main steps (i.e., spatial feature upsampling, feature transformation, kernel size, dilation rate, and activation layer). Each step has a set of discrete options. Our search space is expressive and covers many existing designs of upsampling operations. For example, the blue path indicates the bilinear additive upsampling operation [80,81]. The red path corresponds to the stride 2 transposed convolution used in [56, 86]. The green path represents the sub-pixel convolution [46, 70]. Searching in this space allows us to discover new and effective upsampling layers for capturing stronger image priors.

learnable parameters to search for the kernel size, the dilation rate, and whether to include an activation at the end. Our search space of operations for the activation function includes none, ReLU [60], LeakyReLU [84], SELU [42], and PReLU [30]. Figure 3 presents our developed search space for the upsampling cell. By decomposing several commonly used upsampling operators, our search space is more flexible and allows us to discover a novel upsampling cell for each task. Newly developed spatial upsampling operators (e.g., CARAFE [79]) can also be incorporated in our search space easily in the future.

3.3 Cross-scale residual connections

For the cross-scale residual connections, we develop a search space that covers cross-level feature connections. In contrast to U-Net [68] which *concatenates feature maps* at the same feature level, we adopt *residual connections* [31]. This allows us to merge feature maps extracted from different feature levels without the need to pre-define the number of input channels for each layer in the decoder because we design the number of input channels to always be the same.

Sharing cross-level patterns. The search space for the cross-scale residual connections can be extremely large. Assuming that the network depth is d (i.e., there are d feature levels in the encoder and d feature levels in the decoder), the number of possible connection patterns is 2^{d^2}. To constrain the search space, we enforce the connection patterns to depend only on *the difference of feature levels* (e.g., connecting all the feature maps to the feature maps one level higher).

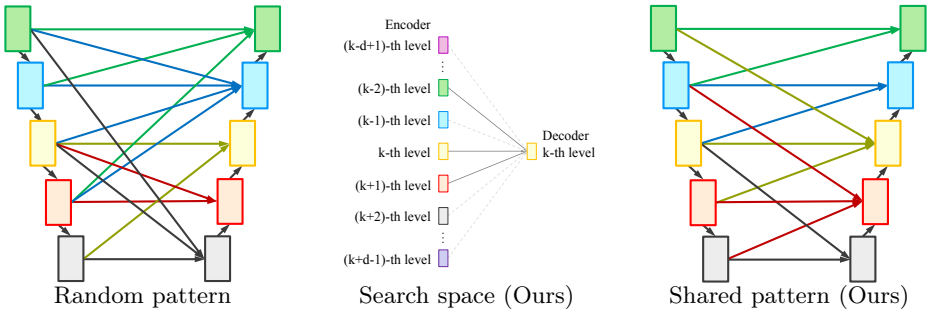


Fig. 4: **Illustration of the cross-level feature connections.** (*Left*) U-Net architecture with a random pattern of cross-level feature connections. Without any constraint, the search space is large. (*Middle*) Our proposed search space for the cross-level feature connections. To constrain the search space, we enforce the connection pattern to depend only on the level difference and share the pattern across different feature levels, e.g., each feature level in the decoder receives feature maps from two levels lower, the same level, and one level higher. With this constraint, the size of the search space is significantly reduced. (*Right*) U-Net architecture with the pattern of cross-level feature connections shown in the middle example shared across different feature levels.

For the k-th feature level in the decoder, it can receive feature maps from the $(k-d+1)$-th feature level up to the $(k+d-1)$-th feature level (i.e., there are in total 2^{2d-1} possible connection patterns). With this constraint, we then search for a pattern of feature connections and *share this pattern across different feature levels* in the decoder. Figure 4 illustrates the main idea of the proposed cross-level residual connections. The size of the search space can be significantly reduced from 2^{d^2} (without any constraints on the connection patterns) to 2^{2d-1}.

Progressive upsampling. For the cross-scale upsampling operation, as illustrated in Figure 5, we propose to *decouple* the $4\times$ upsampling operation in the left into two consecutive $2\times$ upsampling operations in the middle, and share the weights with the $2\times$ upsampling operations in the right at the same feature level. This allows us to define $2\times$ upsampling operations between each consecutive feature level only, and all other possible upsampling scales can be achieved by decoupling them into a series of $2\times$ upsampling operations. The cross-scale downsampling connections can be achieved similarly.

4 Experimental Results

In this section, we first describe the implementation details. We then present the quantitative and visual comparisons to existing methods as well as the ablation study. More visual results, implementation details, and the searched architectures are provided in the supplementary material.

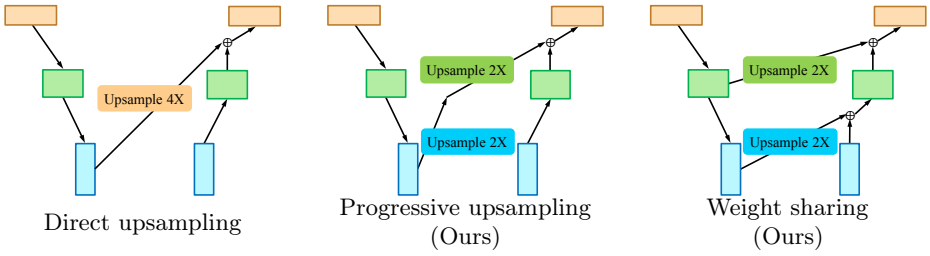


Fig. 5: **Decomposition and weight sharing of the upsampling operations.** To achieve cross-scale residual connections, we decouple the upsampling operations into a series of 2× upsampling operations, e.g., a 4× upsampling operation (*left*) can be realized by two consecutive 2× upsampling operations (*middle*). We adopt *weight sharing* for the upsampling operation at the same feature level, i.e., the weights of the 2× upsampling operations in the middle example are shared with those in the right example. After the cross-level feature upsampling, we add all the input feature maps at the same feature level. The resulting feature map then becomes the input of the decoder at the next layer. We note that the same level feature connections always exist in our model. We do not visualize them in this figure due to presentation clarity.

4.1 Implementation details

Here, we provide the implementation details regarding the neural architecture search, model training, and testing.

Neural architecture search. We implement our model using PyTorch. The network architecture of our RNN controller is the same as [26]. To create the training data for searching for the network architecture, we randomly sample 100 images from the DIV2K [1] training set. To search for the optimal network structure for the task of interest, the neural architecture search process is composed of two alternating phases. For the first phase, the RNN controller first samples a candidate network architecture with random initialization. We then optimize the sampled candidate model on the held-out training set (i.e., model training in our NAS-DIP framework). For the second phase, we first compute the PSNR between the restored prediction and the corresponding ground truth as the reward and use reinforcement learning to update the RNN controller (i.e., NAS training in our NAS-DIP framework). The training time required for each task varies. Specifically, finding the optimal network structure for the super-resolution task takes about 3 days, denoising about 3 days, and inpainting about 5 days (using an NVIDIA V100 GPU with 12GB memory). We refer the reader to the supplementary material for the details of the neural architecture search process.

Testing details. When applying the searched model for testing, there is a hyperparameter (number of iterations) that one can set to obtain the final

Table 1: **Comparison with existing methods on image restoration tasks.** For (a), (b), and (c), we report the average PSNR results. For (d), we follow the evaluation protocol in [6] and report the mean squared error (MSE). (a) Results of single image super-resolution on the Set5 [10] and Set14 [87] datasets with 2×, 4×, and 8× scaling factors. (b) Results of image inpainting on the dataset provided by [33] (*left*) and image denoising on the BM3D dataset [18] (*right*). (c) Comparison with Deep Decoder [32] on the dataset provided by [32] on single image super-resolution and image inpainting tasks. (d) Comparison with Latent Convolutional Models [6] on the Bedrooms dataset of LSUN [85]. For super-resolution in (c) and (d), we report the results with 4× scaling factor. Marker * indicates that the method uses the ground truth to get the best PSNR results. The **bold** and underlined numbers indicate the top two results, respectively.

(a)

Type	Method	Set5 [10] 2×	Set5 [10] 4×	Set5 [10] 8×	Set14 [87] 2×	Set14 [87] 4×	Set14 [87] 8×
Learning based	LapSRN [44]	37.52	31.54	26.14	33.08	28.19	24.44
	VDSR [41]	37.53	31.35	25.93	33.05	28.02	24.26
	EDSR [51]	38.11	32.46	26.96	33.92	28.80	24.91
	RDN [89]	38.24	32.47	-	34.01	28.81	-
	RCAN [88]	38.27	32.63	27.31	34.12	28.87	25.23
	SAN [19]	38.31	32.64	27.22	34.07	28.92	25.14
	EBRN [64]	**38.35**	**32.79**	**27.45**	**34.24**	**29.01**	**25.44**
Learning free	Bicubic [20]	33.66	28.44	24.37	30.24	26.05	23.09
	Glasner et al. [25]	-	28.84	-	-	26.46	-
	TV prior [8]	-	28.85	24.87	-	26.42	23.48
	RED [67]	-	30.23	25.56	-	27.36	23.89
	DeepRED [58]	-	30.72	26.04	-	27.63	24.28
	SelfExSR [36]	**36.60**	30.34	25.49	**32.24**	27.41	23.92
	DIP* [78]	33.19	29.89	25.88	29.80	27.00	24.15
	Ours	35.32	30.81	26.41	31.58	27.84	24.59
	Ours*	35.90	**31.09**	**27.03**	31.89	**28.37**	**25.17**

(b)

Method	Inpainting	Denoising
Papyan et al. [63]	31.19	-
DIP [78]	33.48	30.43
SGLD [16]	34.51	30.81
Ours	**34.72**	**31.42**

(c)

Method	SR 4×	Inpainting
Deep Decoder [32]	25.8	31.9
DIP [78]	26.9	32.3
Ours	**27.4**	**33.1**

(d)

Method	SR 4× ↓	Inpainting ↓
PGAN [40]	0.0183	0.0097
GLO [11]	0.0069	0.0085
LCM [6]	0.0071	0.0065
DIP [78]	0.0057	0.0063
Ours	**0.0054**	**0.0060**

prediction results. An early method [78] uses the ground truth of the test image and the PSNR to select the number of iterations with the best performance. However, this scheme may not be practical as the ground-truth image is not often available. To address this issue, we use the same training set for NAS training to find an optimal number of iterations that allow the model to reach the best performance. Specifically, we select model prediction at $4,500$ iterations for super-resolution, $3,500$ iterations for denoising, and $9,000$ iterations for inpainting.

4.2 Quantitative comparison

We validate the effectiveness of our searched model architecture by evaluating its performance when used as a deep image prior to solve various inverse problems in image restoration. In each task, we compare with state-of-the-art learning-free methods on benchmark datasets.

Fig. 6: **Qualitative results of single image super-resolution.** We present visual comparisons with learning-free methods (i.e., bicubic and DIP [78]) and a learning-based approach (i.e., LapSRN [44]) with 8× scaling factor.

Single image super-resolution. Following DIP [78], we adopt two standard benchmarks: the Set5 [10] and Set14 [87] datasets. We compare our approach with existing learning-free methods [8, 20, 25, 36, 58, 67, 78] and learning-based approaches [19, 41, 44, 51, 64, 88, 89] on three different upsampling scales (i.e., 2×, 4×, and 8× upsampling). We use the evaluation code provided by DIP [78]. Table 1a presents the experimental results.

Results on all three upsampling scales show that our algorithm performs favorably against state-of-the-art *learning-free* methods. Our results show that for a larger upsampling scale (e.g., 8× upsampling), our method achieves competitive or even better performance when compared with existing learning-based methods [41, 44] on both datasets. This is interesting because our model has *never seen any paired low-/high-resolution image pair*. The results also highlight the importance of our searched architecture, resulting in significant performance boost over existing methods that use hand-crafted priors (e.g., DIP [78]).

Image denoising and inpainting. We adopt the BM3D dataset [18] to evaluate the image denoising task. For a fair comparison, we follow DIP [78] and average the output of our network using an exponential sliding window (moving average) to obtain the final result. For evaluating the performance of the inpainting task, we follow DIP [78] and use the dataset provided by Heide et al. [33]. Here, we compare our results with DIP [78], Papyan et al. [63], and SGLD [16] using the 50% missing pixels setting. Table 1b reports the experimental results. Similarly, our method compares favorably against all competing approaches on both tasks.

Comparisons to recent CNN designs. There have been several recent methods that design the CNN architecture for improved performance on image restoration tasks. We first follow the same experimental setting as DeepDecoder [32] and evaluate our method on the 4× super-resolution and inpainting tasks. Table 1c reports the experimental results. Next, we follow the evaluation protocol in Latent Convolutional Models [6] and report our results on the 4× super-resolution and inpainting tasks in Table 1d.

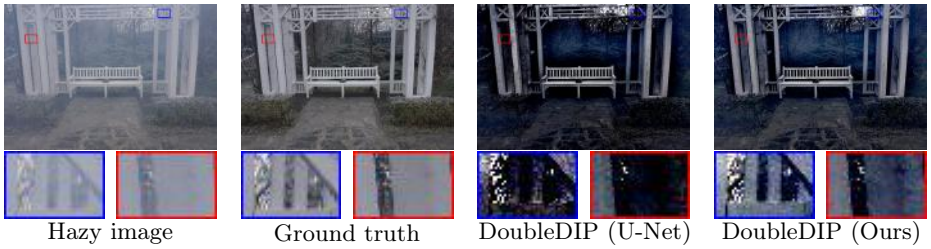


Fig. 7: **Qualitative results of image dehazing.** We present the visual comparisons with DoubleDIP [22] on the O-HAZE dataset [4].

From the extensive quantitative evaluations, we show that our model with both the searched upsampling cells and the cross-scale residual connections can serve as a stronger structured image prior to existing manual CNN architecture designs.

4.3 Visual comparison

Here, we show sample qualitative results of several image restoration tasks and compare them with the state-of-the-art approaches. We refer the reader to review the full resolution results to better perceive visual quality improvement.

Figure 6 presents the visual results of single image super-resolution. Generally, using our model as an image prior results in clearly visible improvement in terms of the visual quality. This improvement highlights the strength of *learning* a stronger structured image prior through neural architecture search.

In addition to standard image restoration tasks, we also experiment with *model transferability* to two different tasks. We use the dehazing application in DoubleDIP [22] and the matrix factorization task in CompMirror [3] for demonstration.

For dehazing, we follow the official implementation by DoubleDIP [22] for generating the dehazed results. To generate our results, we swap the standard U-Net model in DoubleDIP [22] with our searched model searched in the *denoising* task. Figure 7 shows an example of visual comparison with DoubleDIP [22] on the O-HAZE dataset [4].[1] Our results show that using our model produces dehazed images with better visual quality.

For matrix factorization, we use the official implementation by CompMirror [3] to generate the factorized results. To generate our results, we replace the upsampling layer in the CompMirror [3] model with our searched upsampling layer searched in the *denoising* task. Figure 8 presents an example of visual comparison

[1] We originally plan to conduct a quantitative evaluation using the O-HAZE dataset [4]. Unfortunately, using the provided source code and email correspondences with the authors, we were still unable to reproduce the results of DoubleDIP on this dataset. We thus did not report quantitative results on dehazing in this work.

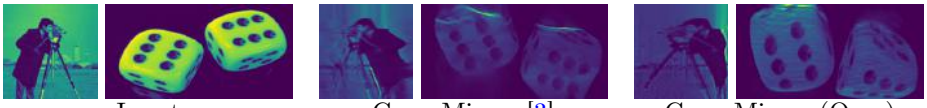


Fig. 8: **Qualitative results of the matrix factorization.** We present the visual comparisons with CompMirror [3] on the matrix factorization task.

Table 2: **Ablation study.** We report the average PSNR results with comparisons to our variant methods. For single image super-resolution, we report the results on the Set 14 dataset [87] with 4× and 8× scaling factors. For image inpainting, we report the results using the dataset provided by [33]. For image denoising, we use the BM3D dataset [18] for evaluation. We denote "S-U" for searching upsampling and "S-C" for searching connection. The numbers in the parenthesis denote the performance gain over DIP [78]. The **bold** and underlined numbers indicate the top two results, respectively.

Method	Search upsampling	Search connection	SR 4×	SR 8×	Inpainting	Denoising
DIP [78]	-	-	27.00	24.15	33.48	30.43
Ours w/o S-C	✓	-	27.54 (+0.54)	24.44 (+0.29)	34.01 (+0.53)	31.08 (+0.65)
Ours w/o S-U	-	✓	27.32 (+0.32)	24.29 (+0.14)	34.59 (+1.11)	31.16 (+0.73)
Ours	✓	✓	**27.84** (+0.84)	**24.59** (+0.44)	**34.72** (+1.24)	**31.42** (+0.99)

with CompMirror [3]. Our results show that using our model produces smoother factorized images with fewer visual artifacts.

4.4 Ablation study

We conduct an ablation study to isolate the contributions from individual components. Specifically, we aim to understand how much performance improvement can be attributed to each of our two technical contributions. As our method builds upon the U-Net architecture of DIP [78], we use their results as the baseline.

We report the results of our variant methods in Table 2. Our results demonstrate that searching for an upsampling cell and a pattern of cross-level residual connections consistently helps improve the performance over DIP [78] across multiple tasks. We also observe that the upsampling cell is particularly important for the single image super-resolution task. On the other hand, introducing the searched cross-scale residual connections offers a larger performance boost over upsampling cell for both inpainting and denoising tasks. The model with both components shows the best performance, highlighting the complementary nature of these two components.

4.5 Image-to-image translation

We also explore transferring the searched model (from the denoising task) to a different problem. Specifically, we aim to test if our searched model generalizes well

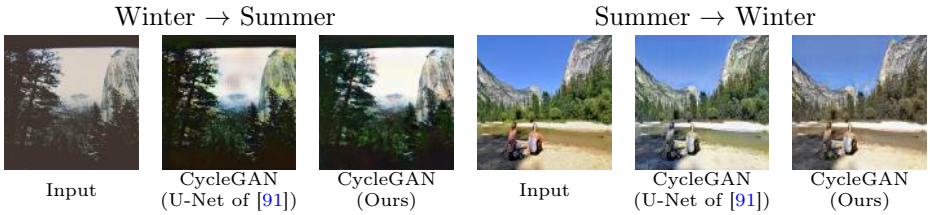


Fig. 9: **Qualitative results of the unpaired image-to-image translation task.** We present the visual comparisons with CycleGAN [91] on the Winter → Summer (*left*) and the Summer → Winter (*right*) translation tasks.

Table 3: **Quantitative results of unpaired image-to-image translation on the Summer ↔ Winter dataset.** (*Left*) The FID scores. (*Right*) The user study results.

Method	FID ↓	
	Summer → Winter	Winter → Summer
CycleGAN (U-Net of [91])	78.62	73.91
CycleGAN (U-Net of [78])	79.74	74.83
CycleGAN (Ours)	76.22	71.98

Method	User study	
	Summer → Winter	Winter → Summer
CycleGAN (U-Net of [91])	20.7%	27.82%
CycleGAN (Ours)	79.3%	72.18%

to image-to-image translation tasks. We take the official PyTorch implementation of CycleGAN [91] and train the Summer ↔ Winter translation. We compare our results with the standard U-Net based model of CycleGAN [91]. Figure 9 shows one sample result for each of the translation directions. To quantify the performance, we also compute the FID score [34] and perform a user study. We report the results in Table 3. Both objective (FID score) and subjective (user study) results indicate that our searched model improves the performance over the base CycleGAN [91] model.

5 Conclusions

In this paper, we propose to use neural architecture search techniques to discover stronger structured image priors captured by the CNN architecture. The core technical contributions of our work lie in (1) the search space design for the upsampling layer commonly used in a decoder and (2) the cross-level feature connections between the encoder and the decoder. We build our network design upon the standard U-Net architecture and search for an optimal upsampling cell and a pattern of cross-level feature connections for each task of interest. We validate the effectiveness of our model on four image restoration tasks, one matrix factorization application, and an unpaired image-to-image translation problem. Through extensive experimental evaluations, our results show consistent performance improvement over conventional network designs.

References

1. Agustsson, E., Timofte, R.: Ntire 2017 challenge on single image super-resolution: Dataset and study. In: CVPRW (2017)
2. Ahmed, K., Torresani, L.: Maskconnect: Connectivity learning by gradient descent. In: ECCV (2018)
3. Aittala, M., Sharma, P., Murmann, L., Yedidia, A., Wornell, G., Freeman, B., Durand, F.: Computational mirrors: Blind inverse light transport by deep matrix factorization. In: NeurIPS (2019)
4. Ancuti, C.O., Ancuti, C., Timofte, R., De Vleeschouwer, C.: O-haze: a dehazing benchmark with real hazy and haze-free outdoor images. In: CVPRW (2018)
5. Angeline, P.J., Saunders, G.M., Pollack, J.B.: An evolutionary algorithm that constructs recurrent neural networks. Transactions on Neural Networks (1994)
6. Athar, S., Burnaev, E., Lempitsky, V.: Latent convolutional models. In: ICLR (2019)
7. Baker, B., Gupta, O., Naik, N., Raskar, R.: Designing neural network architectures using reinforcement learning. In: ICLR (2017)
8. Beck, A., Teboulle, M.: Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. TIP (2009)
9. Berthelot, D., Schumm, T., Metz, L.: Began: Boundary equilibrium generative adversarial networks. arXiv (2017)
10. Bevilacqua, M., Roumy, A., Guillemot, C., Alberi-Morel, M.L.: Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In: BMVC (2012)
11. Bojanowski, P., Joulin, A., Lopez-Paz, D., Szlam, A.: Optimizing the latent space of generative networks. In: ICML (2018)
12. Burger, H.C., Schuler, C.J., Harmeling, S.: Image denoising: Can plain neural networks compete with bm3d? In: CVPR (2012)
13. Cai, H., Chen, T., Zhang, W., Yu, Y., Wang, J.: Efficient architecture search by network transformation. In: AAAI (2018)
14. Chen, L.C., Collins, M., Zhu, Y., Papandreou, G., Zoph, B., Schroff, F., Adam, H., Shlens, J.: Searching for efficient multi-scale architectures for dense image prediction. In: NeurIPS (2018)
15. Chen, Y.C., Lin, Y.Y., Yang, M.H., Huang, J.B.: Crdoco: Pixel-level domain transfer with cross-domain consistency. In: CVPR (2019)
16. Cheng, Z., Gadelha, M., Maji, S., Sheldon, D.: A bayesian perspective on the deep image prior. In: CVPR (2019)
17. Chu, X., Zhang, B., Ma, H., Xu, R., Li, J., Li, Q.: Fast, accurate and lightweight super-resolution with neural architecture search. arXiv (2019)
18. Dabov, K., Foi, A., Egiazarian, K.: Video denoising by sparse 3d transform-domain collaborative filtering. In: European Signal Processing Conference (2007)
19. Dai, T., Cai, J., Zhang, Y., Xia, S.T., Zhang, L.: Second-order attention network for single image super-resolution. In: CVPR (2019)
20. Dong, C., Loy, C.C., He, K., Tang, X.: Image super-resolution using deep convolutional networks. TPAMI (2015)
21. Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., Brox, T.: Flownet: Learning optical flow with convolutional networks. In: ICCV (2015)
22. Gandelsman, Y., Shocher, A., Irani, M.: "double-dip": Unsupervised image decomposition via coupled deep-image-priors. In: CVPR (2019)

23. Gao, H., Wang, Z., Ji, S.: Channelnets: Compact and efficient convolutional neural networks via channel-wise convolutions. In: NeurIPS (2018)
24. Ghiasi, G., Lin, T.Y., Le, Q.V.: Nas-fpn: Learning scalable feature pyramid architecture for object detection. In: CVPR (2019)
25. Glasner, D., Bagon, S., Irani, M.: Super-resolution from a single image. In: ICCV (2009)
26. Gong, X., Chang, S., Jiang, Y., Wang, Z.: Autogan: Neural architecture search for generative adversarial networks. In: ICCV (2019)
27. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NeurIPS (2014)
28. Guo, J., Li, Y., Lin, W., Chen, Y., Li, J.: Network decoupling: From regular to depthwise separable convolutions. In: BMVC (2018)
29. Guo, Y., Li, Y., Wang, L., Rosing, T.: Depthwise convolution is all you need for learning multiple visual domains. In: AAAI (2019)
30. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: ICCV (2015)
31. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
32. Heckel, R., Hand, P.: Deep decoder: Concise image representations from untrained non-convolutional networks. In: ICLR (2019)
33. Heide, F., Heidrich, W., Wetzstein, G.: Fast and flexible convolutional sparse coding. In: CVPR (2015)
34. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: NeurIPS (2017)
35. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: CVPR (2017)
36. Huang, J.B., Singh, A., Ahuja, N.: Single image super-resolution from transformed self-exemplars. In: CVPR (2015)
37. Huang, X., Liu, M.Y., Belongie, S., Kautz, J.: Multimodal unsupervised image-to-image translation. In: ECCV (2018)
38. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: CVPR (2017)
39. Jia, X., Chang, H., Tuytelaars, T.: Super-resolution with deep adaptive image resampling. arXiv (2017)
40. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. In: ICLR (2018)
41. Kim, J., Kwon Lee, J., Mu Lee, K.: Accurate image super-resolution using very deep convolutional networks. In: CVPR (2016)
42. Klambauer, G., Unterthiner, T., Mayr, A., Hochreiter, S.: Self-normalizing neural networks. In: NeurIPS (2017)
43. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NeurIPS (2012)
44. Lai, W.S., Huang, J.B., Ahuja, N., Yang, M.H.: Deep laplacian pyramid networks for fast and accurate super-resolution. In: CVPR (2017)
45. Lai, W.S., Huang, J.B., Ahuja, N., Yang, M.H.: Fast and accurate image super-resolution with deep laplacian pyramid networks. TPAMI (2018)
46. Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., Shi, W.: Photo-realistic single image super-resolution using a generative adversarial network. In: CVPR (2017)

47. Lee, H.Y., Tseng, H.Y., Huang, J.B., Singh, M., Yang, M.H.: Diverse image-to-image translation via disentangled representations. In: ECCV (2018)
48. Lee, H.Y., Tseng, H.Y., Mao, Q., Huang, J.B., Lu, Y.D., Singh, M., Yang, M.H.: Drit++: Diverse image-to-image translation via disentangled representations. International Journal of Computer Vision pp. 1–16 (2020)
49. Lefkimmiatis, S.: Non-local color image denoising with convolutional neural networks. In: CVPR (2017)
50. Li, Y., Huang, J.B., Ahuja, N., Yang, M.H.: Deep joint image filtering. In: ECCV (2016)
51. Lim, B., Son, S., Kim, H., Nah, S., Mu Lee, K.: Enhanced deep residual networks for single image super-resolution. In: CVPR (2017)
52. Liu, C., Chen, L.C., Schroff, F., Adam, H., Hua, W., Yuille, A., Fei-Fei, L.: Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In: CVPR (2019)
53. Liu, H., Simonyan, K., Vinyals, O., Fernando, C., Kavukcuoglu, K.: Hierarchial representations for efficient architecture search. In: ICLR (2018)
54. Liu, H., Simonyan, K., Yang, Y.: Darts: Differentiable architecture search. In: ICLR (2019)
55. Liu, Y.L., Lai, W.S., Yang, M.H., Chuang, Y.Y., Huang, J.B.: Learning to see through obstructions. In: CVPR (2020)
56. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR (2015)
57. Luo, X., Huang, J., Szeliski, R., Matzen, K., Kopf, J.: Consistent video depth estimation. ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH) (2020)
58. Mataev, G., Milanfar, P., Elad, M.: Deepred: Deep image prior powered by red. In: ICCVW (2019)
59. Miikkulainen, R., Liang, J., Meyerson, E., Rawal, A., Fink, D., Francon, O., Raju, B., Shahrzad, H., Navruzyan, A., Duffy, N., Hodjat, B.: Evolving deep neural networks. Artificial Intelligence in the Age of Neural Networks and Brain Computing (2019)
60. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: ICML (2010)
61. Nekrasov, V., Chen, H., Shen, C., Reid, I.: Fast neural architecture search of compact semantic segmentation models via auxiliary cells. In: CVPR (2019)
62. Odena, A., Dumoulin, V., Olah, C.: Deconvolution and checkerboard artifacts. Distill (2016)
63. Papyan, V., Romano, Y., Sulam, J., Elad, M.: Convolutional dictionary learning via local processing. In: ICCV (2017)
64. Qiu, Y., Wang, R., Tao, D., Cheng, J.: Embedded block residual network: A recursive restoration model for single-image super-resolution. In: ICCV (2019)
65. Real, E., Aggarwal, A., Huang, Y., Le, Q.V.: Regularized evolution for image classifier architecture search. In: ICML (2018)
66. Real, E., Moore, S., Selle, A., Saxena, S., Suematsu, Y.L., Tan, J., Le, Q.V., Kurakin, A.: Large-scale evolution of image classifiers. In: ICML (2017)
67. Romano, Y., Elad, M., Milanfar, P.: The little engine that could: Regularization by denoising (red). SIAM Journal on Imaging Sciences (2017)
68. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: MICCAI (2015)
69. Saxe, A.M., Koh, P.W., Chen, Z., Bhand, M., Suresh, B., Ng, A.Y.: On random weights and unsupervised feature learning. In: ICML (2011)

70. Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A.P., Bishop, R., Rueckert, D., Wang, Z.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: CVPR (2016)
71. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
72. Stanley, K.O., D'Ambrosio, D.B., Gauci, J.: A hypercube-based encoding for evolving large-scale neural networks. Artificial life (2009)
73. Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evolutionary computation (2002)
74. Suganuma, M., Ozay, M., Okatani, T.: Exploiting the potential of standard convolutional autoencoders for image restoration by evolutionary search. In: ICML (2018)
75. Sun, Q., Ma, L., Joon Oh, S., Van Gool, L., Schiele, B., Fritz, M.: Natural and effective obfuscation by head inpainting. In: CVPR (2018)
76. Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A.A., Hardt, M.: Test-time training for out-of-distribution generalization. In: ICML (2020)
77. Tan, M., Chen, B., Pang, R., Vasudevan, V., Le, Q.V.: Mnasnet: Platform-aware neural architecture search for mobile. In: CVPR (2019)
78. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Deep image prior. In: CVPR (2018)
79. Wang, J., Chen, K., Xu, R., Liu, Z., Loy, C.C., Lin, D.: Carafe: Content-aware reassembly of features. In: ICCV (2019)
80. Wojna, Z., Ferrari, V., Guadarrama, S., Silberman, N., Chen, L.C., Fathi, A., Uijlings, J.: The devil is in the decoder: Classification, regression and gans. IJCV (2019)
81. Wojna, Z., Uijlings, J.R.R., Guadarrama, S., Silberman, N., Chen, L.C., Fathi, A., Ferrari, V.: The devil is in the decoder. In: BMVC (2017)
82. Xie, J., Xu, L., Chen, E.: Image denoising and inpainting with deep neural networks. In: NeurIPS (2012)
83. Xie, L., Yuille, A.: Genetic cnn. In: ICCV (2017)
84. Xu, B., Wang, N., Chen, T., Li, M.: Empirical evaluation of rectified activations in convolutional network. In: ICMLW (2015)
85. Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., Xiao, J.: Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. arXiv (2015)
86. Zeiler, M.D., Taylor, G.W., Fergus, R.: Adaptive deconvolutional networks for mid and high level feature learning. In: ICCV (2011)
87. Zeyde, R., Elad, M., Protter, M.: On single image scale-up using sparse-representations. In: International conference on curves and surfaces (2010)
88. Zhang, Y., Li, K., Li, K., Wang, L., Zhong, B., Fu, Y.: Image super-resolution using very deep residual channel attention networks. In: ECCV (2018)
89. Zhang, Y., Tian, Y., Kong, Y., Zhong, B., Fu, Y.: Residual dense network for image super-resolution. In: CVPR (2018)
90. Zhong, Z., Yan, J., Wu, W., Shao, J., Liu, C.L.: Practical block-wise neural network architecture generation. In: CVPR (2018)
91. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: ICCV (2017)
92. Zoph, B., Le, Q.V.: Neural architecture search with reinforcement learning. In: ICLR (2017)
93. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning transferable arhictectures for scalable image recognition. In: CVPR (2018)