Searching Efficient Model-Guided Deep Network for Image Denoising

Qian Ning, Weisheng Dong®, Member, IEEE, Xin Li®, Fellow, IEEE, and Jinjian Wu®, Member, IEEE

Abstract—Unlike the success of neural architecture search (NAS) in high-level vision tasks, it remains challenging to find computationally efficient and memory-efficient solutions to low-level vision problems such as image restoration through NAS. One of the fundamental barriers to differential NAS-based image restoration is the optimization gap between the super-network and the sub-architectures, causing instability during the searching process. In this paper, we present a novel approach to fill this gap in image denoising application by connecting model-guided design (MoD) with NAS (MoD-NAS). Specifically, we propose to construct a new search space under a model-guided framework and develop more stable and efficient differential search strategies. MoD-NAS employs a highly reusable width search strategy and a densely connected search block to automatically select the operations of each layer as well as network width and depth via gradient descent. During the search process, the proposed MoD-NAS remains stable because of the smoother search space design under the model-guided framework. Experimental results on several popular datasets show that our MoD-NAS method has achieved at least comparable even better PSNR performance than current state-of-the-art methods with fewer parameters, fewer flops, and less testing time. “The code associate with this paper is available at: https://see.xidian.edu.cn/faculty/wsdong/Projects/Mod-NAS.htm”.

Index Terms—Image denoising, model-guided design (MoD), neural architecture search (NAS), and efficient deep neural network.

I. INTRODUCTION

T he field of image restoration, especially image denoising, has advanced rapidly in recent years. Many deep learning-based methods have achieved great performance in image denoising applications such as Trainable Nonlinear Reaction Diffusion Network (TNRD) [1], Denoising Convolutional Neural Network (DnCNN) [2], Memory Network for Image Restoration (MemNet) [3], Non-local recurrent network (NLRN) [4], Evolutionary search for convolutional auto-encoders (E-CAE) [5], Dual residual networks (DuRN) [6], and Neural nearest neighbors networks (N3Net) [7]. Most recently, the neural architecture search (NAS) based approach [8] has been proposed and demonstrated highly competitive performance for the image denoising task.

NAS has been proposed to overcome the difficulty of manually designing neural architectures for deep learning and has achieved remarkable performance in various high-level tasks. Some early works have adopted reinforcement learning (RL) [9], [10] and evolutionary algorithm (EA) [11], [12] as search strategies. However, both RL-based and EA-based methods require tremendous GPU resources and running time. For example, RL-based NASNet [10] and EA-based AmoebaNet [12] take 48k and 76k GPU hours, respectively. Given such prohibitive complexity, a differential search strategy, such as DARTS [13], was proposed to relax the discrete search space using a differentiable proxy, so that the NAS can be optimized by gradient descent. Many recent works, including ours and [8], [14], [15], [16] have been inspired by this strategy of differential NAS.

NASNet has proposed a cell-based search space [10], where the cell is defined by a directed acyclic graph with several nodes. Cell-based methods [10], [12], [13] search for the operations between nodes (the so-called feature maps) in a cell and repeat the cell to obtain a complete network architecture. However, networks found in cell-based search spaces often suffer from long inferring times. To improve efficiency, many works such as ProxylessNAS [17], FBNet [15], DenseNAS [16] have proposed a new search space based on MobileNetV2 [18]. When searching for expansion ratios and kernel sizes of MBConv layers, these NAS methods often achieved better results than previous methods [9], [10], [12].

These works inspire us to construct a new search space based on another popular architecture (i.e., U-net) and automatically search the layer operations, network width, and depth, achieving lightweight and low inferring time simultaneously. Despite the extensive study of NAS [19], the mainstream applications of NAS have been limited to middle- to high-level vision tasks such as image classification [9], [13], semantic segmentation [14], and object detection [20], [21]. Only a handful of works on the use of NAS for low-level vision tasks have been published so far (e.g., image superresolution [22], [23], image restoration [24], [25], and image denoising [5], [8], [26]). So far, there have been few works on the application of NAS to image

Authorized licensed use limited to: XIDIAN UNIVERSITY. Downloaded on January 09,2023 at 09:57:11 UTC from IEEE Xplore. Restrictions apply.
restoration. E-CAE [5] exploits convolutional autoencoders for image inpainting and denoising employing EA as a search strategy, which requires enormous computational resources and consumes a lot of time. HiNAS [8] employs a gradient-based search strategy in the cell-based search space to denoise and derain images with less search time. Hierarchical NAS has been studied for various low-level vision tasks such as image denoising [8] and superresolution [22]; however, they have adopted a search space similar to that of high-level vision tasks [13]. Similarly to DARTS [13], those NAS methods [8], [22] for low-level tasks have only searched the layers’ operations, ignoring flexibility in terms of network width and depth. Similarly to HiNAS, we adopt a gradient-based search strategy but design a new search space under a model-guided framework for low-level tasks. Besides, our method can automatically search the operations of each layer, network width, and depth; while HiNAS has only searched the operations of each layer.

The motivation behind this work is mainly two-fold. On the one hand, inspired by recent work of model-guided networks for image restoration [27], [28], [29], [30], we propose to construct a new search space by leveraging the domain knowledge implied in the model-guided neural architecture. To our knowledge, this is the first work to incorporate the domain knowledge of image restoration into NAS to design a new search space for low-level tasks. On the other hand, recent works on densely connected search spaces [16] and network pruning [31] have motivated us to pursue more stable and flexible NAS strategies specifically tailored for the newly constructed search spaces. Unlike previous NAS methods [8], [13], [22] that only searched for the operations of each layer, our proposed method can automatically select not only the operations of each layer but also the width and depth of the network, achieving the efficiency objective. The key contributions of this paper are summarized as follows.

- We propose to search for efficient model-guided deep networks for image denoising. Our approach consists of a newly constructed search space and a customized search strategy to automatically select the width and depth of the network, achieving a lightweight and low inferring time simultaneously. The new search space is built under model-guided framework, in which the deep denoiser is based on U-net. The construction of a search space by model-guided design provides a smoother search space, remaining stable during the search process.
- Selecting the operations of each layer, as well as the width and depth of the network, is based on more flexible search strategies. By combining highly reusable width search with densely connected blocks for depth search, efficient and effective networks have been searched for image denoising and compression artifact reduction.
- We have conducted extensive benchmark experiments on our searched network for image denoising and compression artifact reduction. Experimental results on several popular datasets show that our MoD-NAS performs comparably or even better than current state-of-the-art methods with fewer parameters, a lower number of flops, and less amount of running time.

II. MODEL-GUIDED DESIGN WITH NEURAL ARCHITECTURE SEARCH (MoD-NAS)

We first briefly review the model-guided design method for image denoising. Then we elaborate on the proposed search space constructed under the model-guided framework and present the strategies of searching operations for each layer, as well as the network width and depth. Finally, we summarize the overall search procedure of MoD-NAS. The whole process of this work can be divided into two steps: the first step, called the “Search procedure” searches for candidate networks from the super-net that we construct from the search space, while the second step, called the “Training procedure”, aims to train the candidate networks searched from scratch to obtain the best denoising performance.

A. Model-Guided Design (MoD) for Image Denoising

Assuming that a noisy image is modeled with the most commonly used additive Gaussian noise, the degradation process can be formulated as

\[ y = x + n, \]

where \( y, x \) denotes the clean/noisy image pair and \( n \) denotes the additive white Gaussian noise. Based on the above model, we can obtain the clean image \( x \) by the maximum a posteriori (MAP) estimator as

\[ \hat{x} = \arg\max_x \log P(y|x) = \arg\max_x \log P(y|x) + \log P(x), \]

where \( P(y|x) \) and \( P(x) \) denote Gaussian likelihood and prior terms, respectively, which can be expressed as

\[ P(y|x) \propto \exp(-\frac{1}{\sigma_n^2}||y - x||_2^2), \]

\[ P(x) \propto \exp(-\eta R(x)), \]

where \( \sigma_n^2 \) is the noise variance and \( R(x) \) is the regularization function (e.g., sparsity-based [32], [33], nonlocal self-similarity-based [34], [35], [36]). As mentioned in [36] and [27], we can solve the image denoising problem using a deep denoising network as a plug-and-play prior [37]. Compared to hand-crafted priors, deep denoising priors better characterize image prior distributions. Substituting the Gaussian likelihood of Eq. (2) and prior term of Eq. (3) into the MAP estimator of Eq. (1), the following objective function for image denoising can be obtained

\[ x = \arg\min_x \frac{1}{\sigma_n^2}||y - x||_2^2 + \eta R(x). \]

For the prior term \( P(x) \), instead of using an explicitly mathematical representation of \( R(x) \), a DCNN-based denoising prior [27] is introduced by using an auxiliary variable \( v \), which can be expressed as

\[ (x, v) = \arg\min_{x,v} \frac{1}{\sigma_n^2}||y - x||_2^2 + \eta R(v), \quad s.t. \ v = x. \]
The constrained optimization problem can be solved by the half quadratic splitting (HQS) method formulated as

\[
  \min_{x, v} \frac{1}{\sigma_n^2} ||y - x||_2^2 + \lambda ||x - v||_2^2 + \eta R(v),
\]  

(6)

where \( \lambda \) is a penalty parameter varying in each iteration in a non-descending order. With the HQS method, the original problem of Eq. (5) can be solved by alternatively optimizing the following two subproblems:

\[
  v^{(t+1)} = \arg\min_v \lambda ||x^{(t)} - v||_2^2 + \eta R(v),
\]  

(7a)

\[
  x^{(t+1)} = \arg\min_x \frac{1}{\sigma_n^2} ||y - x||_2^2 + \lambda ||x - v^{(t+1)}||_2^2.
\]  

(7b)

The \( v \)-subproblem is a proximity operator of \( R(v) \), which can be solved by a deep denoising network with current estimate \( x^{(t)} \). The \( x \)-subproblem admits a closed-form solution. Then, Eq. (6) can be solved by alternatively computing

\[
  v^{(t+1)} = f_{DN}(x^{(t)}),
\]  

(8a)

\[
  x^{(t+1)} = \frac{y + \lambda \sigma_n^2 v^{(t+1)}}{1 + \lambda \sigma_n^2} = \delta y + (1 - \delta)v^{(t+1)},
\]  

(8b)

where \( f_{DN}(\cdot) \) denotes a deep denoiser and \( \delta = \frac{1}{1 + \lambda \sigma_n^2} \).

The basic idea of model-guided design (MoD) is to unfold conventional model-based iterative algorithms Eq. (8) into the implementation by cascaded DNN. Despite the excellent performance achieved by model-guided methods [27], [36], [38], those networks are still hand-crafted, and their optimality remains questionable. As advocated in [8], NAS serves as an appealing remedy for optimizing neural architectures. Meanwhile, we focus on searching computationally efficient and lightweight denoising networks with additional domain knowledge of image denoising.

Even though U-net was originally proposed for medical image segmentation, it has been demonstrated to have great performance in image restoration domain (e.g., image denoising [26], image dehazing [40], video deraining [41], video deblurring [42]). Inspired by the success of U-net in image restoration, we propose to construct a new search space under model-guided framework, in which deep denoiser \( f_{DN}(x) \) is based on U-net structure. When using U-net as a deep denoising prior, the one-step iteration Eq. (8) can be unfolded into a deep network implementation as shown in Fig. 1(a). The addition operator (red module) faithfully sums up the two terms in Eq. (8b).

Note that the denoising network within the dashed orange box only represents a one-step implementation after unfolding; its concatenation into multiple stages unfolds the iterative solution to Eq. (5). Specifically, all layers of both encoding block (EB) and decoding block (DB) are searchable (highlighted by blue and orange modules). Taking DB as

Fig. 1. (a) The overall architecture of the proposed network; (b)-(d) a list of candidate operations to be searched for NL, DSL, and USL, respectively; (e) network depth searching; (f) layer operations searching; (g) network width searching.
an example (refer to the right part of Fig. 1(e)), except for the last DB, each block consists of three normal layers and one upsampling layer. The last DB consists of three normal layers only. Similarly, each EB consists of three normal layers and one down-sampling layer. During the search, there are seven, four and five candidate operations to be searched in normal layer, down-sampling layer and up-sampling layer respectively. The detailed candidate operations for each layer are listed in Fig. 1(b-d). The network width and depth can also be automatically selected via proposed searching strategies. By relaxing all discrete architectures of network space into differential formulations, we can search the architectures with gradient descent algorithms such as ADAM [43].

C. The Search Strategies

1) Layer Operations: As shown in Fig. 1(b-d), there are seven, four and five candidate operations to be searched in the normal layer (NL), the down-sampling layer (DSL), and the up-sampling layer (USL), respectively. It is worth mentioning that each convolution operation starts with a ReLU activation function, but is not followed by a batch normalization layer since it requires more GPU memory [44]. And each interpolation operation is followed by a $1 \times 1$ convolution for channel conversion. Taking the normal layer as an example, let $O$ denote the set of candidate operations listed in Fig. 1(b). Each candidate operation $o \in O$ of each layer $\ell$ in block $b$ has been assigned an architecture parameter $\omega_o^{\ell,b}$. We have adopted the function $\text{softmax}$ to compute the architecture weight for every operation layer $\ell$ in block $b$:

$$
\omega_o^{\ell,b} = \frac{\exp(\omega_o^{\ell,b})}{\sum_{o' \in O} \exp(\omega_o^{\ell,b})}.
$$

Finally, the output of operations in layer $\ell$ in block $b$ (refer to Fig. 1(f)) can be expressed as

$$
z^{\ell,b} = \sum_{o \in O} \omega_o^{\ell,b} \cdot o(L^{\ell-1,b}),
$$

where $L^{\ell-1,b}$ denotes the input of layer $\ell$ in block $b$. Similarly, all other types of layer can also be relaxed.

In summary, the task of choosing the best operation for the layer $\ell$ in the block $b$ has been translated into the problem of optimizing the architecture parameters $\alpha^{\ell,b}$, which can be solved by gradient descent algorithms [13]. After the supernet is trained, we only need to choose the operation $\alpha^{\ell,b}$ with the largest architecture weight $\omega_o^{\ell,b}$, computed by $\alpha^{\ell,b}$ with Eq. (9) and discard the others. In other words, the selection of layer operations can be formulated as follows.

$$
\alpha^{\ell,b} = \underset{o \in O}{\text{argmax}} \omega_o^{\ell,b}.
$$

As shown in Fig. 1(f), only one operational pathway highlighted by solid red arrows has been selected.

2) Network Width: HiNAS [8] and Autolab [14] search for the width of a cell by stacking cells with different widths side by side (i.e., 1W, 2W and 4W, where W is the basic width that must be set manually before the search). Such a strategy suffers from a prohibitive cost of computation, GPU memory, and search time. In addition, this search strategy can only search for a finite number of width parameters (as determined by W), and all layers in one cell must share the same width. It follows that many potential architectures (e.g., with varying layer width in one cell) are excluded from the search, implying a lack of flexibility.

Inspired by rapid advances in network pruning [31], we propose a new highly reusable width search method to search the width of every layer. As shown in the left part of Fig. 1(g), every channel of operation $o$ of layer $\ell$ in block $b$ has been assigned an architecture parameter $\beta_c^{o,\ell,b}$, where $c$ denotes the $c$-th channel of total $C$ channels. During the search process, the width and other architecture parameters will be optimized together using the gradient descent algorithm. Once the supernet has been trained, the probability distribution of the width architecture parameters $\beta$ generally observes a heavy-tailed distribution with a single peak at the origin (as shown in Fig. 2(a)). This observation implies that we can discard the channel with small architecture parameters $\beta_c^{o,\ell,b}$ around zero. Specifically, we have chosen the top-$M$ channels with the largest architecture parameters $\beta_c^{o,\ell,b}$. The criterion for our selection can be written as follows.

$$
\sum_{i=1}^{M} |\beta_i^{o,\ell,b}| \geq 90\% \cdot \sum_{j=1}^{C} |\beta_j^{o,\ell,b}|, \quad (12a)
$$

$$
\hat{M} \mod 2^n = 0, \quad (12b)
$$

where Eq. (12a) reflects the idea of preserving large parameters only and Eq. (12b) is used for GPU acceleration (we have set $n = 3$ in our experiment) [16].

After the pruning, preserved channels are shown in the right part of Fig. 1(g). Note that the actual number of preserved channels is a variable, which makes our architecture more flexible than fixed-width searching such as HiNAS [8] and E-CAE [5].

3) Network Depth: Searching for the suitable depth of a network is crucial to the success of NAS-related applications. Previous works [15], [17], [39] usually search the depth of network by adding a skip operation in the candidates. Once a skip operation has been selected, it is equivalent to the reduction of network depth by one layer. An undesirable consequence of such search strategy is that the produced network might be too shallow when many skip operations have been selected. One possible explanation of this phenomenon is that the skip has the same probability as the other operations, and skip is often more frequently selected during search because it does not have any parameter. In this paper, we propose a new densely connected block to address this issue by utilizing dense connections [45] to implement the strategy of searching the suitable depth of the network.

As shown in Fig. 1(e), every candidate connection highlighted by orange color has been assigned an architecture parameter. Namely, the path from layer $i$ to layer $\ell$ in block $b$ has a parameter $\gamma_i^{\ell,b}$. Let $L_i^{\ell,b}$ denotes the outputs of layer $\ell$ of block $b$. The output of operations $z_i^{\ell,b}$ has to be compared with output of former layers $L_i^{\ell,b}$ to decide whether to be kept or not. Similar to the relaxation of layer operations, we adopt $\text{softmax}$ function to compute the probability $p_i^{\ell,b}$ of each
path. Then, the selection process can be formulated as:

\[
L_{\ell,b} = \sum_{i=0}^{\ell-1} p_{i}^{\ell,b} \cdot L_{i}^{\ell,b} + p_{\ell}^{\ell,b} \cdot z_{\ell,b},
\]

\[
p_{i}^{\ell,b} = \frac{\exp(y_{i}^{\ell,b})}{\sum_{j=0}^{\ell} \exp(y_{j}^{\ell,b})},
\]

where \(z_{\ell,b}\) denotes the result of the operations in the layer \(\ell\) of block \(b\). After training the supernet, we only choose the paths with the highest probability \(p_{i}^{\ell,b}\) in each layer and discard the others, as shown in Fig. 2(b).

D. Overall Search Procedure

Putting things together, we can see how the search space constructed by MoD can be seamlessly integrated with the NAS strategy, as follows (see Algorithm 1). The unfolding process can be summarized into the procedure of **Forward inferring** of Algorithm 1. During this procedure, the mapping \(f_{DN}^{(t)}\) corresponds to the denoising network in Fig. 1(a) at \(t\)-th stage. Our MoD-NAS adopts a differentiable search strategy, where the search process can be optimized using gradient descent algorithms such as ADAM [43]. We train the supernet with the following MSE loss:

\[
(\mathcal{W}, \mathcal{A}) = \arg \min_{\mathcal{W}, \mathcal{A}} \sum_{i=1}^{N} \| \mathcal{F}(y_{i}; \mathcal{W}, \mathcal{A}) - x_{i} \|_{2}^{2},
\]

where \(y_{i}\) and \(x_{i}\) denote the \(i\)-th pair of original and degraded image patches, respectively, and \(\mathcal{F}(y_{i}; \mathcal{W}, \mathcal{A})\) denotes the reconstructed image patch by the supernet with the set of operation weight parameters \(\mathcal{W}\) and the set of architecture parameters \(\mathcal{A}\). We alternately optimize the operation weights by decreasing \(\nabla_{\mathcal{W}} L_{\text{train}}(\mathcal{W}, \mathcal{A})\) formulated as Eq. (14) in the training set and optimize the architecture parameters by decreasing \(\nabla_{\mathcal{A}} L_{\text{val}}(\mathcal{W}, \mathcal{A})\) formulated as Eq. (14) in the validation set, as shown in the procedure **Backpropagation** of Algorithm 1. When the supernet has been trained, we derive the final architecture based on the parameters \(\alpha, \beta, \gamma\) as shown in Algorithm 1. An example of the search for the final denoising network architecture is shown in Fig. 2(c).

III. EXPERIMENTAL RESULTS

In this section, we first describe the experimental settings for search and training. We then provide ablation studies on MoD-NAS and searching network width and depth, showing the superiority of our proposed searching space and searching strategies. Next, we present the Gaussian image denoising experiment comparisons with state-of-the-art methods and discuss the main advantages of our approach over other NAS methods. Finally, we provide experiments on real image denoising and image compression artifact reduction to verify the robustness of our proposed searching approach.

A. Experimental Settings

1) Training and Testing Datasets: In this paper, we have conducted three different experiments to demonstrate the robustness of our proposed search approach and the resulting network in varying denoising scenarios.

- Gaussian image denoising. We have randomly selected 4000 images from the Waterloo [46] dataset for training. Following [2], [7], [27], [47], three standard benchmark datasets (Set12 [2], BSD68 [48], Urban100 [49]) are used for the test. Noisy images are generated by adding white Gaussian noise to the corresponding clean images with \(\sigma = 15, 25, 50\) following [2], [27], [47]. It is worth

---

**Algorithm 1** Proposed MoD-NAS Algorithm

- **Initialization:**
  1. Initialize \(x\) as \(x^{(0)} = y\).

- **For** epoch = 1 to N **do**
  1. **Forward inferring:** for \(t = 1, 2, \ldots, T\), do
     a. Compute \(v_{t} = f_{DN}^{(t)}(x^{(t-1)})\);
     b. Compute \(x^{(t)}\) via Eq. (8b);
     c. \(t = t + 1\).
  **End for**

- **Backpropagation:**
  a. Update operation weights and \(\delta^{(t)}\) by descending \(\nabla_{\mathcal{W}} L_{\text{train}}(\mathcal{W}, \mathcal{A})\);
  b. Update architecture parameters by descending

---

---
mentioning that the search experiments are carried out with $\sigma = 25$ and the training experiments are conducted with $\sigma = 15, 25, 50$.

- Real image denoising. The SIDD [50] dataset has provided a medium training set (320 image pairs) and a validation set (40 image pairs) for fast training and evaluation, but the results of the tests can only be obtained by online submission. The training set and the test set of the medium SIDD data set are used for training and testing, respectively.

- Image compression artifact reduction. The training dataset consists of 4000 randomly selected images from the Waterloo dataset [46]. The LIVE1 [51] dataset is used for testing. The Matlab JPEG encoder [52] is used to generate compressed images. Search and training experiments are conducted with $q = 20$.

2) Search Settings: The training dataset [46] has been equally divided into two non-overlapping parts: one for updating the weights of network operations (Training $W$) and the other for updating the architecture parameters (Training $A$). The batch size is 12 and the patch size is 64 $\times$ 64. Two ADAM optimizers [43] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$ are adopted to optimize the parameter sets $W$ and $A$, respectively. The learning rate of two optimizers decreases from $10^{-3}$ to $10^{-5}$ with the cosine annealing schedule [53] within 140 epochs. The stages of the super-net are set to $T = 2$ and the initial number of channels is set to $C = 48$. In the first 40 epochs, we update only the parameter set $W$ of the operations, and the parameter sets $W$ and $A$ are optimized alternately in the remaining epochs. Our network is implemented by the PyTorch framework and the total search time is about 7 hours using one NVIDIA RTX 2080Ti GPU. The total epoch of searching $N$ is 140.

3) Training Settings: We train the network that was searched by our method with MSE loss. We randomly selected 32 noisy patches sized by 128 $\times$ 128 as input. The ADAM algorithm [43] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$ is adopted to optimize the network. The learning rate decreases from $10^{-3}$ to $10^{-5}$ with the cosine annealing schedule [53] within 600 epochs. The searched denoising network is shown in Fig. 2(c). Our network is implemented by PyTorch and the total training time takes about 8, 10, and 12 hours for $T = 1, 2$ and 3, respectively, using one NVIDIA RTX 2080Ti GPU.

B. Ablation Studies

1) Benefits of MoD-NAS: During the search process for differentiable NAS, there is a phenomenon in which the performance of the supernet drops greatly when the number of search epochs becomes large. This phenomenon known as mode collapse has been observed in both high-level tasks [54] and low-level tasks [8]. DARTS+ [54] for image classification and HiNAS [8] for image denoising have employed a similar way called early stopping. This early stopping strategy stops the searching process early before mode collapse occurs.

In this paper, by incorporating model-guided design with differentiable NAS (MoD-NAS), the performance of the supernet remains stable and even increases slightly when the number of search epochs becomes large. To demonstrate this advantage, we have compared the performance of our MoD-NAS against a common differentiable NAS [13] during the search process evaluated on the Set12 dataset as shown in Fig. 3. Note that we have removed the model-guided design part (removed the auxiliary variable $v = f(x)$) by only searching one denoising network as the baseline control; while all other settings are kept the same as MoD-NAS. From Fig. 3, we can see that the performance of the proposed MoD-NAS remains stable during the entire search process, while the baseline method suffers from apparent collapses, especially when Epochs $> 100$. In addition, during the search process, our proposed MoD-NAS demonstrates improved stability, showing a much smoother PSNR curve than the baseline. We argue that such a benefit of the proposed MoD-NAS method can be explained away by MoD-NAS providing a smoother search space than differentiable NAS [13].

2) Benefits of Searching for Network Width: To verify the effectiveness of the width of the search network, we carried out the MoD-NAS experiment ($T = 3$) ($C = 64$), which changes all channels of MoD-NAS ($T=3$) to 64. The experimental results of different widths of MoD-NAS ($T=3$) have been shown in Table I, from which we can see that MoD-NAS ($T=3$) achieves almost the same result on Set12 dataset. Therefore, MoD-NAS ($T=3$) achieves a better trade-off between the number of parameters or flops and the accuracy in searching for the network width, achieving the goal of efficiency.

3) Benefits of Searching for Network Depth: We also have conducted experiments to verify the validity of the proposed densely connected block to determine the depth of the network.
performance in three datasets is slightly different in terms of datasets with stages. We have shown the quantitative results of different parameter that controls the capacity of the searched network to obtain more efficient networks. It is not mandatory, but searching both items simultaneously would with comparable and even better results than not. It is not depths of networks can obtain a more lightweight network flops than not. Therefore, both searching for widths and depths of networks can obtain better performance with fewer parameters and flops than not. To gain a deeper understanding of the deep unfolding framework, we have considered three different $T$ values and compared their PSNR performance. We have shown the experimental results of $T = 1$, $T = 3$, and $T = 6$ in Tab. V and Tab. VI.

![Fig. 4. The benefit of searching for network depth.](image)

**TABLE II**

| Widths | Depths | PSNR | Parameters | Flops |
|--------|--------|------|------------|-------|
| 1      | ✓      | 30.83| 1559k      | 1.62G |
| 2      | ✓      | 30.89| 1883k      | 2.00G |
| 3      | ✓      | 30.88| 1253k      | 1.45G |

As shown in Fig. 4, the blue line shows the search with a densely connected search strategy; the orange line shows the search by adding skips to select the depth of the network. It can be seen that the search procedure with our proposed densely connected search strategy (blue line) is more stable than adding skips (orange line), and the derived network converges rapidly (while adding skips in candidates converges to a very shallow network).

We conducted an ablation study to investigate whether both width and depth search are mandatory and the results are shown in Table II. Comparing case 1 and case 2, searching depths of networks can obtain better performance with fewer parameters and flops than not. Comparing case 2 and case 3, searching the widths of networks can obtain almost the same performance with fewer parameters and flops than not. Therefore, both searching for widths and depths of networks can obtain a more lightweight network with comparable and even better results than not. It is not mandatory, but searching both items simultaneously would obtain more efficient networks.

4) The Stage Number $T$: The stage number $T$ is the parameter that controls the capacity of the searched network model. We have shown the quantitative results of different stages $T$ ranging from one to seven on three benchmark datasets with $\sigma = 25$ in Table III, from which we can see that performance in three datasets is slightly different in terms of trend, but the improvement in PSNR rapidly saturates when $T > 6$. Therefore, we can choose the appropriate number $T < 7$ to balance performance and computational complexity according to actual demand.

**TABLE III**

The average performance of the PSNR as a function of the parameter $T$ (the total number of stages of unfolding) for Gaussian image denoising on three benchmark datasets with $\sigma = 25$. The best performance is shown in bold, and the second-best performance is shown in underline.

| Stages | 1   | 2   | 3   | 4   | 5   | 6   | 7   |
|--------|-----|-----|-----|-----|-----|-----|-----|
| Set12  | 30.73 | 30.84 | 30.88 | 30.89 | 30.90 | 30.95 | 30.97 |
| BSD68  | 29.40 | 29.45 | 29.47 | 29.48 | 29.49 | 29.50 | 29.50 |
| Urban100 | 30.38 | 30.60 | 30.73 | 30.76 | 30.78 | 30.83 | 30.91 |
| params | 418k | 535k | 1253k | 1681k | 2088k | 2500k | 2923k |
| flops  | 6.48G | 6.97G | 1.45G | 1.94G | 2.42G | 2.91G | 3.39G |

5) The Initial Value $C$: We conducted experiments using different initial values of $C$, and the results are shown in Table IV. As shown in Table IV, as the initial number of channels increased, the results of the searched networks improved, and the best results were achieved when the number of initial channels was 48, confirming the choice of our search setting.

**C. Comparisons With State-of-the-Art Methods**

For image denoising, we have compared MoD-NAS with several current state-of-the-art methods: BM3D [35], TNRD [1], DnCNN [2], N3Net [7], MemNet [3], DPDNN [27], FOCNet [47], RDN [55], and DCDicL [56] and two NAS methods to denoise images: E-CAE [5], HiNAS [8]. The average PSNR and SSIM results of the comparing methods in Tab. V and Tab. VI are either directly cited from the original papers or reproduced by running the officially released source codes. The testing time is shown in the Tab. V is the total time to evaluate the entire Urban100 dataset when $\sigma = 25$. To gain a deeper understanding of the deep unfolding framework, we have considered three different $T$ values and compared their PSNR performance. We have shown the experimental results of $T = 1$, $T = 3$, and $T = 6$ in Tab. V and Tab. VI.

It can be observed that our MoD-NAS is consistently superior to other competing methods except for DCDicL [56] in terms of PSNR and SSIM performance for all datasets. More importantly, our single-stage model (MoD-NAS(1)) has achieved comparable and even better performance than most benchmark methods with the fewest parameters, the lowest number of flops, and the least test time. For larger models, MoD-NAS ($T = 3$) and MoD-NAS ($T = 6$) have achieved better results than other benchmark methods with a larger margin, as shown in Tab. V.

Although the number of parameters in MoD-NAS($T = 3$) and MoD-NAS($T = 6$) is comparable to that of previous methods such as N3Net [7], DPDNN [27], MoD-NAS ($T = 3$), and MoD-NAS ($T = 6$) has a great advantage over others in terms of flops and testing time. For example, our MoD-NAS ($T = 3$)
Fig. 5. Gaussian image denoising visual quality comparison. The first and second rows show the comparison of ‘Img_027’ from BSD68 with $\sigma = 25$; the third and fourth rows show the comparison of ‘Img_013’ from Urban100 with $\sigma = 50$; the fifth and sixth rows show the comparison of ‘Img_017’ from Urban100 with $\sigma = 50$ (zoom in for better view).

has 1253k parameters, which are only 5.7% those of RDN [55] and 43% MemNet [3]. Compared to RDN [55], our MoD-NAS(T=3) reduces the testing time on the Urban100 dataset by up to 95.3% with even better PSNR results. Compared to the DPDNN model-guided method [27] in which the denoising network was manually designed based on U-net, our single
Fig. 6. Gaussian image denoising visual quality comparison. The first row shows the comparison of ‘Img_002’ from Set12 with \( \sigma = 15 \); the second row shows the comparison of ‘Img_001’ from BSD68 with \( \sigma = 25 \); the third row shows the comparison of ‘Img_018’ from Urban100 with \( \sigma = 50 \) (zoom in for better view).

**TABLE V**

| Methods       | Set12 | BSD68 | Urban100 | params | flops | testing time |
|---------------|-------|-------|----------|--------|-------|--------------|
|               | 15    | 25    | 50       | 15     | 25    | 50           |
| BM3D [35]     | 32.37 | 29.97 | 26.72    | 31.07  | 28.57 | 25.62        |
| TNRD [11]     | 32.50 | 30.06 | 26.81    | 31.42  | 28.92 | 25.97        |
| DnCNN [2]     | 32.86 | 30.44 | 27.18    | 31.73  | 29.23 | 26.23        |
| N3Net [7]     | -     | 30.55 | 27.43    | -      | 29.30 | 26.39        |
| MemNet [3]    | 32.96 | 30.60 | 27.03    | 30.76  | 29.19 | 25.25        |
| DPDNN [27]    | 32.91 | 30.54 | 27.50    | 31.83  | 29.27 | 26.40        |
| ROCNet [47]   | 33.07 | 30.73 | 27.68    | 31.81  | 29.38 | 26.30        |
| RDN [56]      | 32.95 | 30.66 | 27.60    | 31.74  | 29.29 | 26.41        |
| DCDicL [57]   | 33.34 | 31.03 | 28.00    | 31.95  | 29.52 | 26.63        |
| HiNAS [8]     | 32.50 | 30.35 | 27.25    | 31.16  | 28.92 | 26.04        |
| E-CAE [5]     | -     | 26.33 | -        | -      | 25.86 | -            |
| MoD-NAS(T=1)  | 33.09 | 30.73 | 27.62    | 31.86  | 29.40 | 26.50        |
| MoD-NAS(T=3)  | 33.21 | 30.88 | 27.83    | 31.91  | 29.47 | 26.59        |
| MoD-NAS(T=6)  | 33.28 | 30.98 | 27.87    | 31.94  | 29.50 | 26.61        |
| **MoD-NAS(T=3)** | **33.28** | **30.98** | **27.87** | **31.94** | **29.50** | **26.61** |

* DPDNN can be viewed as a baseline method, as it is a model-guided method and the denoising network is designed manually based on U-net.

**We have also compared two NAS methods (E-CAE [5] and HiNAS [8]) for image denoising in Tab. V. When comparing the smallest network found by MoD-NAS(T=1) and HiNAS [8], our searched network MoD-NAS(T=1) has gained 0.91 dB over HiNAS [8] on average with fewer parameters, 19.6% flops, and 7.8% testing time, which verifies the effectiveness of our proposed search space and customized search strategies. Besides, we have shown the visualized comparison between the cost and performance in Fig. 7 for easy comparison. It can be observed that our proposed network could achieve at least comparable and often even
TABLE VI
AVERAGE SSIM RESULTS FOR GAUSSIAN IMAGE NOISE ON THREE BENCHMARK DATASETS. THE BEST PERFORMANCE IS SHOWN IN BOLD, AND THE SECOND-BEST PERFORMANCE IS SHOWN IN UNDERLINE.

| Methods   | Set12 15 | BSD68 15 | Urban(100) 15 |
|-----------|----------|----------|---------------|
|           | 0.8952   | 0.8504   | 0.7676        |
| BMD [35]  | 0.8769   | 0.8093   | 0.8117        |
| TNRD [5]  | 0.9031   | 0.8622   | 0.7829        |
| DnCNN [2] | 0.8907   | 0.8278   | 0.8278        |
| N3Net [7] | -        | 0.7957   | 0.6545        |
| MemNet [3] | 0.8848   | 0.7966   | 0.6466        |
| DPDNN [27]| 0.8738   | 0.8123   | 0.7095        |
| RDN [56]  | -        | 0.7942   | 0.6431        |
| E-CAE [5] | 0.8840   | 0.8067   | 0.7849        |
| HiNAS [8] | 0.8940   | 0.8350   | 0.7325        |
| MoG-NAS(T=1) | 0.9020   | 0.8689   | 0.7999        |
| MoG-NAS(T=3) | 0.9087   | 0.8720   | 0.8065        |
| MoG-NAS(T=6) | 0.9101   | 0.8729   | 0.8070        |

D. Comparisons With Other NAS Methods

Only a few NAS methods (e.g., image super-resolution [22], [23] and denoising [5], [8]) have been proposed for low-level vision tasks. Here, we compare our proposed MoD-NAS with E-NAS [5] and HiNAS [8] in Tab. V and Tab. VII. We also conducted experiments that replace the softmax function with the Gumbel-softmax function in the selection of operators for our approach. The Gumbel-softmax functions are formulated in Eq. (9) and Eq. (15) respectively.

$$\alpha_{\ell,b} = \frac{\exp((\alpha_{\ell,b} + g_{\ell,b})/\tau)}{\sum_{\sigma \in \mathcal{O}} \exp((\alpha_{\ell,b} + g_{\ell,b})/\tau)}.$$  

where $\tau$ denotes the temperature parameter and $g_{\ell,b}$ denotes a random variable i.i.d sampled from $\text{Gumbel}(0, 1)$. Apparently, in Tab. VII, methods with a gradient descent-based search strategy have advantages on the cost of GPU memory and searching/training time. Compared to HiNAS [8], our proposed method has the following main advantages.

- We have proposed a new efficient search space under a model-guided framework to address the problem that networks found by cell-based search spaces adopted by HiNAS often suffer from long testing times. As shown in Tab. V, MoD-NAS (T = 1) takes the HiNAS 7.8% testing time and has fewer parameters than HiNAS, demonstrating that MoD-NAS achieves a lightweight and low inferring time simultaneously with more competent performance than HiNAS.
- Compared to HiNAS and E-CAE in Tab. V and Tab. VII, our searched MoD-NAS networks (T = 1, 3, 6) achieve much better performance in terms of PSNR, indicating the superiority of MoD-NAS.
- Using a new highly reusable width search strategy to search the width of the network, our search process requires only one 11 GB 2080Ti GPU, while HiNAS [8] requires one 32 GB V100 GPU; E-CAE [5] requires four 16 GB P100 GPU.
The network searched using the Gumbel-softmax function also achieves great performance. When comparing softmax, the results of Gumbel-softmax are comparable, but Gumbel-softmax involves an additional hyper-parameter \( \tau \) that must be manually adjusted. Therefore, for basic use, we recommend using the softmax function, since it has fewer hyper-parameters.

### E. Real Image Denoising on SIDD Dataset

To demonstrate the generalizability of our searched network, we evaluated the performance of MoD-NAS(T=3) on a real blind denoising task with the SIDD [50] benchmark. The SIDD dataset has provided one medium training set (320 image pairs) and a validation set (40 image pairs) for fast training and evaluation, but test results can only be obtained by online submission. We have trained MoD-NAS (T = 3) in the medium training set and obtained the results of the test sets by online submission. We have compared six different state-of-the-art methods: CBM3D [35], CBDNet [57], AINDNet [60], VDN [58], DANet [59], InvDN [61]. Tab. VIII shows the PSNR and SSIM results of different methods in the SIDD testing set. Note that the results of the test sets are cited from the official website.\(^1\) From Tab. VIII, we can see that MoD-NAS(T=3) can achieve promising results in terms of PSNR and SSIM compared to other methods with much fewer parameters. For example, compared to CBDNet [57], our finalized MoD-NAS (T = 3) gains an improvement over 6 dB in terms of PSNR with only 28.8\% parameters of CBDNet. Compared to VDN [58] and DANet [59], our MoD-NAS(T=3) achieves comparable and even better results with only the parameters 16\% of VDN [58] and the parameters 13.7\% of DANet [59]. We have shown the visual comparison of denoising the real image in the SIDD dataset in Fig. 8, from which we can see that our method has achieved a better result than other methods (e.g., more effective noise suppression).

### F. Image Compression Artifact Reduction

We apply the proposed MoD-NAS search method to the reduction of image compression artifacts to further evaluate the generalizability of our method. We employ the same searching and training setting as the denoising experiments. We have compared four different state-of-the-art methods: TNRD [1],

---

\(^1\)https://www.eecs.yorku.ca/ kamel/sidd/benchmark.php
TABLE IX

| Methods    | JPEG   | TNRD [1] | DnCNN [2] | MemNet [3] | RDN [55] | MoD-NAS-AR |
|------------|--------|----------|-----------|------------|----------|------------|
| PSNR       | 30.07  | 31.46    | 31.59     | 31.83      | 32.07    | 32.30      |
| SSIM       | 0.8683 | 0.8769   | 0.8802    | 0.8846     | 0.8882   | 0.8945     |
| params     | -      | 668K     | 2905K     | 21970K     | 1670K    |            |

Fig. 9. Image compression artifact reduction visual quality comparison on LIVE1 dataset (zoom in for better view).

DnCNN [2], MemNet [3], RDN [55]. The JPEG quality results $q = 20$ are listed in Tab. IX and the visual results are shown in Fig. 9. As shown in Tab. IX, the MoD-NAS-AR searched by MoD-NAS achieves much better performance than other methods. Compared to RDN, MoD-NAS-AR gains an improvement of $0.23$ $dB$ in terms of PSNR with only $7.6\%$ parameters of RDN, demonstrating the superiority of our proposed efficient search deep network method. From Fig. 9, we can observe that MoD-NAS-AR recovers the compressed image with more texture details (see the stripes of the wooden window in the first row and the steps in the second row).

G. Limitations

This work is based on the differentiable NAS [13]. Although this work has achieved satisfactory performance with great convergence, as shown in Fig. 3, it lacks the guarantee of theory. Besides, the search space is built on a widely-used U-net, which may exclude some potential networks with high performance, faster inferring time, and low computational costs, because the structure is not based on U-net. Additionally, this paper mainly focuses on searching efficient networks with faster inferring times; it is still worth searching larger denoising networks with the highest denoising performance, ignoring computational costs. We leave this for future work.

IV. Conclusion

In this paper, we have presented a novel MoD-NAS-based approach to image denoising and reduction of image compression artifacts. By incorporating the strengths of model-guided design and NAS, we have constructed a new search space and designed flexible search strategies specially tailored for the task of image denoising. Through searching in the space of concatenated U-net, we demonstrate how joint consideration of layer operation, network width, and network depth can lead to an efficient network searching solution with excellent performance, including visual quality and convergence property. When comparing other NAS methods of image denoising, our proposed method requires much less GPU memory due to our newly constructed search space for image denoising and the highly reusable width search strategy. Our proposed network could achieve at least comparable and often even better PSNR results than current state-of-the-art methods with fewer parameters and flops, as well as less testing time.

REFERENCES

[1] Y. Chen and T. Pock, “Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1256–1272, Aug. 2017.
[2] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, “Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising,” IEEE Trans. Image Process., vol. 26, no. 7, pp. 3142–3155, Jul. 2017.
[3] Y. Tai, J. Yang, X. Liu, and C. Xu, “MemNet: A persistent memory network for image restoration,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 4549–4557.
[4] B. Liu, B. Wen, Y. Fan, C. Loy, and T. S. Huang, “Non-local recurrent network for image restoration,” in Proc. Adv. Neural Inf. Process. Syst., 2018, pp. 1673–1682.
[5] M. Suganuma, M. Ozay, and T. Okatani, “Exploiting the potential of standard convolutional autoencoders for image restoration by evolutionary search,” in Proc. 35th Int. Conf. Mach. Learn. (ICML), 2018, pp. 4771–4780.
[6] X. Liu, M. Suganuma, Z. Sun, and T. Okatani, “Dual residual networks leveraging the potential of paired operations for image restoration,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 7007–7016.
[7] T. Plötz and S. Roth, “Neural nearest neighbors networks,” in Proc. Adv. Neural Inf. Process. Syst., 2018, pp. 1087–1098.
[8] H. Zhang, Y. Li, H. Chen, and C. Shen, “Memory-efficient hierarchical neural architecture search for image denoising,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 3657–3666.
[9] B. Zoph and Q. Le, “Neural architecture search with reinforcement learning,” in Proc. Int. Conf. Learn. Represent., 2017, pp. 1–16.
[10] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, “Learning transferable architectures for scalable image recognition,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 8697–8710.
[11] H. Liu, K. Simonyan, O. Vinyals, C. Fernando, and K. Kavukcuoglu, “Hierarchical representations for efficient architecture search,” in Proc. Int. Conf. Learn. Represent. (ICLR), 2018, pp. 1–13.

Authorized licensed use limited to: XIDIAN UNIVERSITY. Downloaded on January 09,2023 at 09:57:11 UTC from IEEE Xplore. Restrictions apply.
[60] Y. Kim, J. W. Soh, G. Y. Park, and N. I. Cho, “Transfer learning from synthetic to real-noise denoising with adaptive instance normalization,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 3482–3492.

[61] Y. Liu et al., “Invertible denoising network: A light solution for real noise removal,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 13365–13374.

Qian Ning received the B.S. degree in electronic engineering from Xidian University, Xi’an, China, in 2019, where she is currently pursuing the Ph.D. degree in intelligent information processing. Her research interests include image restoration and deep learning.

Weisheng Dong (Member, IEEE) received the B.S. degree in electronic engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2004, and the Ph.D. degree in circuits and system from Xidian University, Xi’an, China, in 2010. He was a Visiting Student with Microsoft Research Asia, Beijing, China, in 2006. From 2009 to 2010, he was a Research Assistant with the Department of Computing, Hong Kong Polytechnic University, Hong Kong. In 2010, he joined the School of Electronic Engineering, Xidian University, as a Lecturer, where he has been a Professor since 2016. His research interests include inverse problems in image processing, deep learning, and parse representation. He was a recipient of the Best Paper Award at the SPIE Visual Communication and Image Processing (VCIP) in 2010. He has served as an Associate Editor for IEEE TRANSACTIONS ON IMAGE PROCESSING. He is currently an Associate Editor of SIAM Journal of Imaging Sciences.

Jinjian Wu (Member, IEEE) received the B.Sc. and Ph.D. degrees from Xidian University, Xi’an, China, in 2008 and 2013, respectively. From 2011 to 2013, he was a Research Assistant with Nanyang Technological University, Singapore, where he was a Postdoctoral Research Fellow, from 2013 to 2014. From 2015 to 2019, he was an Associate Professor with Xidian University, where he has been a Professor since 2019. His research interests include visual perceptual modeling, biomimetic imaging, quality evaluation, and object detection. He received the Best Student Paper Award at the ISCAS 2013. He served as the Special Section Chair for the IEEE Visual Communications and Image Processing (VCIP) 2017, and the Section Chair/an Organizer/a TPC Member for the ICME 2014–2015, PCM 2015–2016, ICIP 2015, VCIP 2018, and AAAI 2019. He has served as an Associate Editor for the journal of Circuits, Systems and Signal Processing (CSSP).

Xin Li (Fellow, IEEE) received the B.S. degree (Hons.) in electronic engineering and information science from the University of Science and Technology of China, Hefei, in 1996, and the Ph.D. degree in electrical engineering from Princeton University, Princeton, NJ, USA, in 2000. He was a member of Technical Staff with the Sharp Laboratories of America, Camas, WA, USA, from August 2000 to December 2002. Since January 2003, he has been a Faculty Member with the Lane Department of Computer Science and Electrical Engineering, where he has been an Associate Professor since 2006. He was elected as a fellow of IEEE in 2017, for his contributions to image interpolation, restoration, and compression.

Weisheng Dong (Member, IEEE) received the B.S. degree in electronic engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2004, and the Ph.D. degree in circuits and system from Xidian University, Xi’an, China, in 2010. He was a Visiting Student with Microsoft Research Asia, Beijing, China, in 2006. From 2009 to 2010, he was a Research Assistant with the Department of Computing, Hong Kong Polytechnic University, Hong Kong. In 2010, he joined the School of Electronic Engineering, Xidian University, as a Lecturer, where he has been a Professor since 2016. His research interests include inverse problems in image processing, deep learning, and parse representation. He was a recipient of the Best Paper Award at the SPIE Visual Communication and Image Processing (VCIP) in 2010. He has served as an Associate Editor for IEEE TRANSACTIONS ON IMAGE PROCESSING. He is currently an Associate Editor of SIAM Journal of Imaging Sciences.