Multi-scale Attentive Image De-raining Networks via Neural Architecture Search

Lei Cai, Yuli Fu, Wanliang Huo, Youjun Xiang, Tao Zhu, Ying Zhang, Huanqiang Zeng, Senior Member, IEEE, and Delu Zeng

Abstract—Multi-scale architectures and attention modules have shown effectiveness in many deep learning-based image de-raining methods. However, manually designing and integrating these two components into a neural network requires a bulk of labor and extensive expertise. In this article, a high-performance multi-scale attentive neural architecture search (MANAS) framework is technically developed for image de-raining. The proposed method formulates a new multi-scale attention search space with multiple flexible modules that are favorite to the image de-raining task. Under the search space, multi-scale attentive cells are built, which are further used to construct a powerful image de-raining network. The internal multi-scale attentive architecture of the de-raining network is searched automatically through a gradient-based search algorithm, which avoids the daunting procedure of the manual design to some extent. Moreover, in order to obtain a robust image de-raining model, a practical and effective multi-to-one training strategy is also presented to allow the de-raining network to get sufficient background information from multiple rainy images with the same background scene, and meanwhile, multiple loss functions including external loss, internal loss, architecture regularization loss, and model complexity loss are jointly optimized to achieve robust de-raining performance and controllable model complexity. Extensive experimental results on both synthetic and realistic rainy images, as well as the down-stream vision applications (i.e., objection detection and segmentation) consistently demonstrate the superiority of our proposed method. The code is publicly available at https://github.com/lcai-gz/MANAS.

Index Terms—Image de-raining, multi-scale attentive neural architecture search, multi-to-one training strategy.

I. INTRODUCTION

When people take photos in an outdoor environment, the captured images could suffer from visibility degradation owning to various bad weather conditions such as rain [1], [2], haze [3], [4], or snow [5]. Particularly, on a rainy day, the presence of rain can seriously degrade the visual quality of outdoor images, and as a result affects the visual authenticity of human perception and the performance of many outdoor vision systems [6]. In this context, image de-raining becomes very necessary and meaningful, it can serve as an essential pre-step for various computer vision tasks such as object detection [7], object segmentation [8], autonomous driving [9], and more. However, image de-raining remains a formidable challenge due to its intractability and complexity, e.g., the rain streaks in the real world could present different shapes, sizes, density, orientations, etc. Therefore, how to design an effective de-raining algorithm is crucial and has drawn much attention in the computer vision field.

The current research on rain removal can be divided into two directions, namely video de-raining methods [10]–[18] and image de-raining methods [19]–[50]. The former need to exploit the additional temporal correlation in adjacent multi-frames to help restore the pixel corrupted by rain. However, once the temporal information becomes unreliable (e.g., unstable video) or unavailable (e.g., single frame), the performance of the existing video de-raining methods could be severely degraded. Furthermore, for rain removal from videos acquired from a moving camera, the video de-raining methods could also produce poor performance [15]. Unlike video de-raining methods, image de-raining methods aim to remove rain directly from a given rainy image and restore its clean background, without the temporal information.

Over the past decade, a variety of successful algorithms, ranging from early hand-crafted prior-based methods [19]–[21], [26], [27] to the latest deep learning-based methods [6], [22], [23], [25], [28]–[38], [40]–[45], [52], [53], have been proposed to handle the image de-raining task. The prior-based de-raining methods first leverage effective regularizers to characterize the property of the background and rain streak layers, and then separate them by solving an objective function with proper optimization algorithms. The representative methods include dictionary learning [26], discriminative sparse coding [19], Gaussian mixture model [29], nonlocal means filter [21], and more. However, these conventional methods have limited capability to model and remove rain, thus cannot satisfy the increasingly complex de-raining task [46]. For deep learning-based de-raining methods, the major development over the years lies on the design of various effective neural network architectures. Among them, the multi-scale architecture [6], [23], [25], [30], [31], [37], [40], [53] and the attention...
Neural Architecture Search et al. technology. Ahn to a multi-scale Gaussian Laplacian pyramid decomposition proposed light pyramid image de-raining networks, according the multi-scale architecture and the attention module can architectures for image de-raining. The motivation behind explicitly search and integrate the multi-scale attentive neural mechanism was introduced into their network to learn the to-end de-raining network, where a depth-guided attention raining, Hu [42] successfully designed an end-to-end de-raining network, where a depth-guided attention context [56]. Regarding the image de-raining task, it would be better to combine them, rather than separately employing them, so that their merits are taken and their demerits are overcome. Unfortunately, to the best of our knowledge, the current multi-scale attentive neural networks (e.g., [32], [44], [54], [55]) for image de-raining are almost designed and integrated in a hand-crafted manner.

On the other hand, when a well-designed de-raining network is ready to be used for image de-raining, how to train it is also considerably important. Under the premise of sufficient training data, it is highly expected to design an appropriate training strategy to encourage the outputs of the network to be close to the real clean background [57]. In previous image de-raining methods, the most possible way is to estimate mean square error (MSE) or other intricate distance measurements on one-to-one image pairs. As shown in Fig. 1 (b) and (c), two recently-developed image de-raining methods, namely CLEARER [51] and DGNL-Net-fast [46], which were trained with MSE loss on one-to-one image pairs, are applied to remove the rain streaks and fog from three synthetic rainy images shown in Fig. 1 (a). As can be seen, although these input rainy images have the same background scene, parts of their corresponding de-rained results suffer from undesirable local artifacts or color distortion or preserve some rain streaks. This means that these two de-raining methods, which were trained by a one-to-one training paradigm, are not robust enough for image de-raining. See again Fig. 1 (a), as different degrees of rain (i.e., the rain density, intensity, shapes, sizes, and orientations vary) could be presented in multiple images with the same static background, thus we know that these input rainy images should have the same clean background after de-raining. To this end, based on this prior knowledge, we present a multi-to-one training strategy to allow our

module [42], [45], [46], [52], [54] or their combination [32], [44], [54], [55] have been demonstrated effective in boosting the performance. Take a few examples, Zhang et al. [25] developed a multi-stream densely connected de-raining network to efficiently learn features at different scales. Fu et al. [40] proposed light pyramid image de-raining networks, according to a multi-scale Gaussian Laplacian pyramid decomposition technology. Ahn et al. [45] recently designed an image de-raining network by employing the elementwise attentive gating block as a basic unit. To achieve depth-attentive image de-raining, Hu et al. [42] successfully designed an end-to-end de-raining network, where a depth-guided attention mechanism was introduced into their network to learn the depth-attentional features. Rather than separately exploiting the multi-scale architecture or the attention module, Jiang et al. [32] constructed a multi-scale pyramid structure and further introduced U-shaped residual attention blocks to obtain a multi-scale progressive fusion network for image de-raining.

Even though multi-scale architectures and attention modules are demonstrated to be helpful for image de-raining, it is difficult to design their neural architectures and most of them rely heavily on human design. Such a hand-crafted manner has the following limitations. Firstly, manual design is very labor-intensive, especially for those networks with multi-scale architectures and attention modules. Secondly, one more daunting issue of the manual design is unknown when to fuse the low-scale and high-scale features or what kind of attention operation should be applied. Thirdly, the networks with multi-scale architectures and attention modules are generally more complex than the plain ones, thus it is necessary to find an elegant trade-off between the performance and the model complexity [51]. Obviously, it is very difficult to tackle the above limitations through human design.

In this work, we propose a novel Multi-scale Attentive Neural Architecture Search (MANAS) framework to automatically search and integrate the multi-scale attentive neural architectures for image de-raining. The motivation behind our MANAS method is based on the consideration that both the multi-scale architecture and the attention module can strengthen the representation ability of a neural network, but from a different aspect. The multi-scale architecture can not only capture the global structure of input images but also retain their local details [51], while attention modules can well handle long-range dependencies which enables the neural network to give more attention to useful information within a context [56]. Regarding the image de-raining task, it would be better to combine them, rather than separately employing them, so that their merits are taken and their demerits are overcome. Unfortunately, to the best of our knowledge, the current multi-scale attentive neural networks (e.g., [32], [44], [54], [55]) for image de-raining are almost designed and integrated in a hand-crafted manner.

Figure 1. Three examples of rainy images (a) come from RainCityscapes [46] dataset, as well as their corresponding de-rained images derived by CLEARER [51] (b), DGNL-Net-fast [46] (c), and our MANAS (d). One can see that our MANAS method can deliver a more robust de-raining effect.
de-raining network to get sufficient background information from multiple rainy images with the same background, and simultaneously impose external and internal constraints on the de-raining model to direct it to be robust.

The main contributions of this article are:

- We make the first attempt to incorporate the multi-scale architecture search and the attention search into a unified neural architecture search (NAS) framework to automatically discover the high-performance multi-scale attentive image de-raining networks. This proposed framework, called MANAS, is expected to get rid of the daunting designing procedure of the multi-scale attentive neural networks to some extent.
- We propose a novel multi-scale attention search space and integrate it into a differentiable form through a continuous relaxation operation. Moreover, we also introduce a gradient-based search algorithm to search for the best paths in the de-raining network to determine when to fuse the high-scale and low-scale features and what kind of attention operation should be applied.
- We provide a practical and effective multi-to-one training strategy for image de-raining, where multiple loss functions, including external loss, internal loss, architecture regularization loss, and model complexity loss, are jointly used to train our model to achieve robust de-raining performance and controllable model complexity. Both quantitative and qualitative results show that our proposed method outperforms multiple state-of-the-art image de-raining methods.

II. RELATED WORK

A. Image De-raining Methods

Existing image de-raining methods can be broadly divided into two categories: 1) hand-crafted prior-based de-raining methods and 2) hand-crafted neural network-based de-raining methods.

1) Hand-crafted Prior-based De-raining Methods: The hand-crafted prior-based de-raining methods mainly rely on the statistic analysis of rain streaks and background scenes. Such methods usually use effective regularizers to characterize the rain streak and background layers, and then separate them by solving an objective function with proper optimization algorithms. Specifically, Kang et al. [25] decomposed high frequency parts of rainy images into rain and rain-free components by conducting dictionary learning and sparse coding. Luo et al. [19] presented a discriminative sparse coding framework for layer decomposition. Li et al. [20] used a Gaussian mixture model to approximate priors of the background and rain-streak layers and then decomposed them with a Maximum A Posterior (MAP). Kim et al. [21] proposed a nonlocal means filter-based method to remove rain streaks within the detected rain streak regions. However, the above prior-based methods are insufficient in characterizing the background and rain-streak layers, thus cannot clearly remove rain from diverse rainy images.

2) Hand-crafted Neural Network-based De-raining Methods: The major development of deep-learning-based de-raining methods lies on the manual design of various neural networks [6, 23, 30, 32, 37, 40, 42, 44–46, 52–55]. In these de-raining networks, multi-scale architecture [6, 23, 25, 30, 31, 37, 40, 53, 58] and attention module [45, 52, 59] or their combination [32, 44, 54, 55] have been widely incorporated into the design of various image de-raining networks and have achieved the promising performance. The related works are succinctly described as follows: 1) Image de-raining using multi-scale neural networks: Wang et al. [53] developed a modeling Haze-Like effect-based deep neural network for image de-raining, in which a SSP module [60] is introduced to extract multi-scale features to help remove the haze-like effect. Zhang et al. [25] proposed a multi-stream densely connected network to effectively exploit multi-scale features to characterize rain streaks with different scales and shapes. Fu et al. [40] developed light-weight Pyramid networks by introducing multi-scale decomposition of Gaussian Laplacian pyramid, which simplifies the learning process of image de-raining. Yasarla et al. [58] presented an uncertainty guided multi-scale residual learning network to learn the rain content at different scales.

2) Image de-raining using attentive networks: Jiang et al. [1] proposed an improved attention-guided de-raining network for rain streak removal, where a mixed attention block was designed to guide the fusion of rain layers by focusing on the local and global overlaps. Zhu et al. [2] developed a non-local guided attention module in their de-raining network to learn attentional non-local features for the global residual image prediction. Ahn et al. [45] leveraged the elementwise attentive gating block as a basic unit to construct an elementwise attentive gating network for single image de-raining. Que et al. [52] proposed an attentive composite residual network for image de-raining, where a channel-wise attention mechanism is built using a squeeze-and-excitation (SE)-Res2Net. To be well adapted to the stochastic distribution of real rain streaks, Wang et al. [61] developed a spatial attentive ResNet for the stochastic distribution of real rain streaks. Qian et al. [54] proposed an improved generative adversarial network for raindrop removal, where visual attention was injected into both the multi-scale generative and discriminative networks to guide the removal of raindrops at different scales. Bao et al. [55] designed a multi-scale pyramid structure and an iterative attention mechanism. Zamir et al. [44] designed a multi-scale progressive network architecture for image de-raining. In their method, an encoder-decoder is applied in the earlier stage to learn multi-scale contextual information. And a supervised attention module is plugged between very two

...
stages to enable progressive learning.

Although the above de-raining networks can provide a promising solution for image de-raining, their internal neural architectures are all designed and integrated in a hand-crafted manner, which requires a bulk of labor and extensive expertise. Different from these hand-crafted de-raining networks, we employ neural architecture search (NAS) to construct the high-performance multi-scale attentive neural networks for image de-raining, where the internal multi-scale attentive neural architecture of our de-raining network can be searched and integrated automatically through a gradient-based search algorithm.

B. Neural Architecture Search

NAS aims to automate the procedure of discovering neural architectures using advanced search algorithms such as evolutionary algorithm (EA), reinforcement learning (RL), gradient-based algorithm, and more. Benefit from the development of neural architecture search [51], [56], [62]–[64], [64], [65]. Some works make attempts to handle the image restoration task by automatically searching effective neural architectures. The seminal work of NAS-based image restoration is E-CAE [62]. It exploits an evolutionary algorithm to search for good architectures of the Convolutional Auto-Encoders (CAEs). Later, Gou et al. [51] designed a multi-resolution search space, and used a data-driven strategy to search their image restoration networks. Zhang et al. [64] proposed the Hierarchical NAS (HiNAS) for image de-noising. Their method adopted a gradient-based search algorithm and created a hierarchical search space by employing operations with adaptive receptive field. Similarly, Guo et al. [63] developed a Hierarchical Neural Architecture Search (HNAS) method for image super-resolution. Different from the above-mentioned methods, we make the first attempt to integrate the multi-scale architecture search and the attention search into a unified NAS framework to find the high-performance multi-scale attentive neural network for image de-raining. Moreover, we specifically design a practical and effective multi-to-one training strategy for training our model. Such a training paradigm is also the first trial in the neural architecture search.

III. PROPOSED MULTI-SCALE ATTENTION NEURAL ARCHITECTURE SEARCH FOR IMAGE DE-RAINING

In this section, we first define a multi-scale attention search space that involves three basic terms, namely multi-scale module, multi-scale attentive cell, and multi-scale attentive

Figure 2. The proposed multi-scale attentive neural architecture search (MANAS) framework. In brief, multiple typical modules, including the multi-scale transition module, multi-scale parallel module, multi-scale fusion module, and multi-scale attention module, are utilized to build multi-scale attentive cells. These built cells are further employed to construct a multi-scale attentive image de-raining network (Bottom). We take the first cell in the de-raining network as a showcase (Upper). It starts with a multi-scale transition module, followed by a structure with 4 columns (each column contains a multi-scale fusion module represented by the pink box or a multi-scale parallel module represented by the yellow box), and finally ends with a multi-scale attention module represented by the cyan box. In the first cell, there are two scales of resolution, e.g., 1x represents the original resolution, whereas 0.5x represents the original resolution reduced by 0.5. The red arrows in the cell denote the paths to be searched.
image de-raining network. Then, we elaborate the method of the multi-scale attentive neural architecture search. Finally, we provide a multi-to-one training strategy for image de-raining. The framework of the proposed multi-scale attentive neural architecture search (MANAS) method is shown in Fig. 2.

A. Multi-scale Attention Search Space

We introduce four typical multi-scale modules, namely multi-scale transition module, multi-scale parallel module, multi-scale fusion module, and multi-scale attention module, into our search space. Their generalized structures are depicted in Fig. 3.

1) Multi-scale Transition Module: As shown in Fig. 3, the multi-scale transition module is responsible for transmitting the input features to lower-scale ones through a strided convolution operation and simultaneously keeping the original resolution scale unchanged in the horizontal direction via none operation. Mathematically, it can be formulated as:

\[
\begin{align*}
\hat{X}_{1 \times} &= X_{1 \times}, \\
\vdots \\
\hat{X}_{0.5^{t-1} \times} &= X_{0.5^{t-1} \times}, \\
\hat{X}_{0.5^t \times} &= f_s(X_{0.5^t \times}),
\end{align*}
\]

where \(X_{1 \times}\) represents the features at the original resolution, \(X_{0.5^{t-1} \times}\) denotes the features whose resolution is only 0.5\(^{t-1}\) times to the original resolution, and \(t \in \{1, \ldots, T\}\) denotes the index of the multi-scale attentive cell in the de-raining network. \(f_s(\cdot)\) denotes a strided convolution operation to reduce the resolution of input features. For example, for the first cell (i.e., \(t = 1\)), the input features are with single resolution scale, but after the multi-scale transition module, there will produce two resolution scales, i.e., \(\hat{X}_{1 \times} = X_{1 \times}\) and \(\hat{X}_{0.5 \times} = f_s(X_{1 \times})\). Likewise, for the second cell (i.e., \(t = 2\)), its input features come from the first cell and have two resolution scales, but after the multi-scale transition module, there will bring in three resolution scales, i.e., \(\hat{X}_{1 \times} = X_{1 \times}\), \(\hat{X}_{0.5 \times} = X_{0.5 \times}\), \(\hat{X}_{0.25 \times} = f_s(X_{0.5 \times})\).

2) Multi-scale Parallel Module: The multi-scale parallel module is responsible for processing the multi-scale features in parallel, where the resolution of features in each parallel line remains unchanged via a residual block [66], which can be formulated as:

\[
\begin{align*}
\hat{X}_{1 \times} &= \mathcal{F}(X_{1 \times}) + X_{1 \times}, \\
\vdots \\
\hat{X}_{0.5^{t-1} \times} &= \mathcal{F}(X_{0.5^{t-1} \times}) + X_{0.5^{t-1} \times}, \\
\hat{X}_{0.5^t \times} &= \mathcal{F}(X_{0.5^t \times}) + X_{0.5^t \times},
\end{align*}
\]  

where \(\mathcal{F}(\cdot)\) represents a residual function which is learned by 3 \(\times\) 3 convolution layers.

3) Multi-scale Fusion Module: The multi-scale fusion module is responsible for fusing the features at different resolution scales via strided convolution, upsampling, and none operation. The formulation of this module can be written as:

\[
\begin{align*}
\hat{X}_{1 \times} &= \sigma(X_{1 \times} + \text{US}(X_{0.5^t \times}) + \cdots + \text{US}(X_{0.5^t \times})), \\
\vdots \\
\hat{X}_{0.5^{t-1} \times} &= \sigma(f_s(X_{1 \times}) + \cdots + f_s(X_{0.5^{t-2} \times}) + \text{US}(X_{0.5^t \times})) + X_{0.5^{t-1} \times}, \\
\hat{X}_{0.5^t \times} &= \sigma(f_s(X_{1 \times}) + \cdots + f_s(X_{0.5^{t-1} \times}) + X_{0.5^t \times}),
\end{align*}
\]

where \(\text{US}(\cdot)\) represents an up-sampling operation to the input features, and \(\sigma(\cdot)\) is a Rectified Linear Unit (ReLU) [67] for non-linearity.

4) Multi-scale Attention Module: The multi-scale attention module is responsible for adaptively attending to important regions within a context at different resolution scales. As shown in Fig. 3, the multi-scale attention module first divides the input features \(X\) (here we omit its scale subscript for brevity) into two groups along the channel dimension, leading to two groups of split features, i.e., \(\{X_1, X_2\}\). After that, a series of attention operations are applied on \(\{X_1, X_2\}\), producing the corresponding intermediate features \(\{Y_1, Y_2\}\):

\[
\begin{align*}
Y_1 &= \Phi(X_1), \\
Y_2 &= \Phi(X_2) + \Phi(Y_1),
\end{align*}
\]
Various attention operations in attention search sub-space, where $Z_i$ denotes the $i$-th split group features $X_i$ or its corresponding intermediate features $Y_i$, $Z_{i,\text{avg}}$ denotes the global spatial average pooled features from the input $Z_i$, $\otimes$ denotes the Kronecker products, $Z_{i,\text{max}}$ denotes the global spatial max pooled features, $\delta(\cdot)$ denotes a multi-layer perceptron, $\delta(\cdot)$ denotes the sigmoid activation function, $f_\delta(\cdot)$ denotes a mapping function learned by a $3 \times 3$ depth-wise convolutional layer.

| Name                              | Definition                                                                 |
|-----------------------------------|---------------------------------------------------------------------------|
| Channel Attention V1              | $\Phi_0(Z_i) = \delta(\phi(Z_{i,\text{avg}})) \otimes T_i$              |
| Channel Attention V2              | $\Phi_1(Z_i) = \delta(\phi(Z_{i,\text{avg}}) + \phi(Z_{i,\text{max}})) \otimes Z_i$ |
| Spatial Attention                 | $\Phi_{2}(Z_i) = \delta(f_{1c}(\phi(Z_{i,\text{avg}}, Z_{i,\text{max}})) \otimes Z_i$ |
| Normalization Attention           | $\Phi_3(Z_i) = \delta(f_{1c}(Z_i)) \otimes Z_i$                         |
| Convolutional Block Attention     | $\Phi_4(Z_i) = \Phi_3(f_{1c}(Z_i))$                                      |
| Identity Attention                | $\Phi_5(Z_i) = Z_i \otimes 1$                                            |
| Zero Attention                    | $\Phi_6(Z_i) = Z_i \otimes 0$                                            |

where $\Phi(\cdot)$ denotes an attention operation. Finally, the resulting intermediates features $\{Y_1, Y_2\}$ are concatenated along the channel axis to yield the attentive output:

$$\hat{X} = f_{1c}([Y_1; Y_2; \overline{Y}],)$$

where $\overline{Y} = Y_1 + Y_2$, $[;]$ represents the concatenation along the channel axis, and $f_{1c}(\cdot)$ represents the mapping function learned by $1 \times 1$ convolutional layers.

For the multi-scale paradigm, the attentive outputs are:

$$\hat{X}_{1 \times} = f_{1c}([Y_{1,1 \times}; Y_{1,1 \times}; \overline{Y}_{1 \times}],)$$

$$\vdots$$

$$\hat{X}_{0.5^{t-1} \times} = f_{1c}([Y_{1,0.5^{t-1} \times}; Y_{2,0.5^{t-1} \times}; \overline{Y}_{0.5^{t-1} \times}],)$$

$$\hat{X}_{0.5^{t} \times} = f_{1c}([Y_{1,0.5^{t} \times}; Y_{2,0.5^{t} \times}; \overline{Y}_{0.5^{t} \times}],)$$

It is worthwhile of mentioning that all of the attention operations in this module can be searched automatically, thus an attention search sub-space used in [56] is also incorporated into our search space. The formulation of each attention operation in the sub-space is demonstrated in Table I.

5) Multi-scale Attentive Cells: With the above-defined multi-scale modules, we further use them to build our multi-scale attentive cells. As the upper part of Fig. 2 shows, where we take the first cell as a showcase, the multi-scale attentive cell starts with a multi-scale transition module, followed by a structure with 4 columns (each column contains a multi-scale fusion module or a multi-scale parallel module), and finally ends with a multi-scale attention module. The motivation to build our cell with such a layout is based on the consideration that it can bring in a series of high-scale to low-scale attentive sub-networks, as the color-filled boxes in the upper part of Fig. 2 show, this could effectively strengthen the representation ability of the de-raining network.

6) Multi-scale Attentive Image De-raining Network: With the multi-scale attentive cells, we further use them to construct a powerful multi-scale attentive image de-raining network, as the bottom part of Fig. 2 shows. Following the idea in [51], we first exploit two residual blocks [66] cascaded together to receive the rainy image $I_{1 \times}$, and keep its original scale unchanged. After that, we introduce several multi-scale attentive cells connected one-by-one to extract the desirable multi-scale attentive features. The last cell is followed by a multi-scale fusion module which is responsible for fusing the multi-scale attentive features into the single-scale ones. And at the end of our de-raining network, a $1 \times 1$ convolutional layer is employed to output the final de-rained image $O_{1 \times}$. Note that, as we increase the number of cells, more and more high-scale to low-scale attentive sub-networks will be added into the de-raining network, this can significantly facilitate the learning of highly representative and discriminative features for image de-raining.

B. Multi-scale Attentive Neural Architecture Search

With the comprehensive coverage of the multi-scale attention search space, we then give a method for the multi-scale attentive neural architecture search. Our ultimate objective in searching is to find the best path in each multi-scale attentive cell, so as to determine the internal architecture of the cell. For convenience, we still take the first cell in the de-raining network as an example, which is shown in the upper part of Fig. 2. As shown, we need to search two types of path to determine the internal architecture of the cell. Among them, the first type of path is searched to determine whether the multi-scale parallel module or the multi-scale fusion module is chosen at each column, while the second type of path is searched to determine what kind of attention operations should be applied to our multi-scale attention module.

Specifically, let $X^m$ be the input of the multi-scale parallel module or the multi-scale fusion module at the $m$-th column, where $m \in \{1, 2, 3, 4\}$, $P^m(\cdot)$ and $F^m(\cdot)$ be the multi-scale parallel module and the multi-scale fusion module at the $m$-th column. Then, we have the first type of path search:

$$X^{m+1} = P^m(X^m | \alpha^m_0) + F^m(X^m | \alpha^m_1),$$

where $\alpha^m_0, \alpha^m_1 \in \{0, 1\}$ denote the architecture parameters in the $P^m(\cdot)$ and $F^m(\cdot)$, respectively. If $\alpha^m_0 = 1, \alpha^m_1 = 0$, then the path to the multi-scale parallel module $P^m(\cdot)$ is chosen. Otherwise, $\alpha^m_0 = 0, \alpha^m_1 = 1$ means that the path to the multi-scale fusion module $F^m(\cdot)$ is chosen.

Regarding the multi-scale attention module, let $\{X_1, X_2\}$ be the split group features, then we have the second type of path search:

$$Y_1 = \Phi_k(X_1 | \beta_k)$$

$$Y_2 = \Phi_k(X_2 | \beta_k) + \Phi_k(Y_1 | \beta_k)$$

where $\beta_k$ represents the architecture parameters in the multi-scale attention module. Likewise, if their values are equal to one, then their corresponding attention operations, i.e., $\Phi_k(\cdot)$, are chosen, as the red arrows in the upper right part of Fig. 2 show. $\{Y_1, Y_2\}$ denotes the intermediate features resulted from the attention operations. $k \in \{0, 1, \ldots, 6\}$ is the index of the specific attention operations listed in Table I.

Obviously, all of the architecture parameters are limited to a binary constraint, i.e., $\alpha_0, \alpha_1, \beta_k \in \{0, 1\}$, this means that they are discrete. In order to integrate our search method into a differentiable manner, like [51], [56], [68], we conduct
continuous relaxation on the architecture parameters using softmax, such that:
\[
\alpha_i = \frac{\exp(\mu_i)}{\sum_{i=0}^{N-1} \exp(\mu_i)} \quad \text{and} \quad \beta_k = \frac{\exp(\nu_k)}{\sum_{k=0}^{N} \exp(\nu_k)},
\]
where \(\mu_i\) and \(\nu_k\) denote the learnable architecture weights corresponding to \(\alpha_i\) and \(\beta_k\), respectively.

For brevity, we use \(\theta\) to represent all the architecture weights \(\mu_i\) and \(\nu_k\) in the de-raining network. Accordingly, our search problem turns to optimizing the architecture weights \(\theta\) and the network weights \(\omega\), as follows:
\[
\min_{\theta} L_{\text{trainB}}(\omega^*(\theta), \theta)
\]
\[
s.t. \quad \omega^*(\theta) = \arg\min_{\omega} L_{\text{trainA}}(\omega, \theta),
\]
where \(L_{\text{trainA}}\) denotes the loss function for optimizing the network weights \(\omega\), and \(L_{\text{trainB}}\) denotes the loss function for optimizing the architecture weights \(\theta\). To solve this optimization problem, we adopt bi-level optimization, as shown in Algorithm 1.

Finally, when finishing the architecture search training, all of architecture parameters \(\{\alpha_i, \beta_k\}\) are encoded into binary, such that:
\[
\alpha_i = \begin{cases} 
1, & i = \arg\max_i \mu_i, \\
0, & \text{otherwise},
\end{cases}
\]
\[
\beta_k = \begin{cases} 
1, & k = \arg\max_k \nu_k, \\
0, & \text{otherwise},
\end{cases}
\]

At this point, the best paths in the de-raining network are determined.

### C. Multi-to-one Training Strategy

The schematic illustration of the proposed multi-to-one training strategy is depicted in Fig. 1. As shown, the training data set \(S\) is first partitioned into two disjoint training sets \(\text{trainA}\) and \(\text{trainB}\), both of them consist of multi-to-one image pairs and each pair have multiple rainy images \(\{I_1, I_2, \ldots, I_N\}\) with the same clean background (i.e., the ground truth image \(G\)). Then, the de-raining network is driven to process the multi-to-one image pairs to calculate the loss \(L_{\text{trainA}}\) on the set \(\text{trainA}\) and the loss \(L_{\text{trainB}}\) on the set \(\text{trainB}\). Finally, the resulting losses \(L_{\text{trainA}}\) and \(L_{\text{trainB}}\) are fed back to the de-raining network to optimize the architecture weights \(\theta\) and the network weights \(\omega\) alternately through bi-level optimization. In the following, we will elaborate the formulation of the losses \(L_{\text{trainA}}\) and \(L_{\text{trainB}}\), respectively.

1) \(L_{\text{trainA}}\) for Optimizing \(\omega\): Since we train our de-raining network in a supervised manner, we can measure the distance between the de-rained outputs \(\{O_1, O_2, \ldots, O_N\}\) and the ground truth \(G\) to form an external loss \(L_{\text{ext}}\). Specifically, \(L_{\text{ext}}\) is defined as a composition of the MSE loss and SSIM loss [69], as follows:
\[
L_{\text{ext}} = \frac{1}{N} \sum_{i \in N} \|O_i - G\|^2_2 + \frac{1}{N} \sum_{i \in N} (1 - \text{SSIM}(O_i, G)).
\]

2) \(L_{\text{trainB}}\) for Optimizing \(\theta\): The performance of image de-raining is sensitive to neural architectures, according to the previous study on image de-raining. Thus, when formulating the loss \(L_{\text{trainB}}\) for optimizing the architecture weight \(\theta\), we also incorporate the external loss and the internal loss into \(L_{\text{trainB}}\) to facilitate the searching of the high-performance multi-scale attentive image de-raining networks. Meanwhile, besides the de-raining performance, a desirable de-raining network should also take into account the model complexity, since it is crucial to many resource-constrained scenarios such as mobile phones. To this end, we formulate a model complexity loss \(L_{\text{comp}}\) to control the complexity of our de-raining model, as follows:
\[
L_{\text{comp}} = \frac{1}{U + V} \sum (\alpha_0 \Omega_0 + \alpha_1 \Omega_1 + \beta_0 \Lambda_0 + \ldots + \beta_6 \Lambda_6),
\]
where \(\Omega_0\) and \(\Omega_1\) are the size of the multi-scale parallel module and the multi-scale fusion module, respectively, whereas \(\Lambda_0, \Lambda_1, \ldots, \Lambda_6\) are the size of the attention operations listed in Table II. \(U\) is the number of \(\alpha\), and \(V\) is the number of \(\beta\).

On the other hand, although we have made our search method differentiable, there still exists a problem caused by the continuous relaxation using softmax. To be more specific, it could lead to a trivial solution about the architecture parameters, such that \(\alpha_0 = \alpha_1\) and \(\beta_0 = \beta_1 = \ldots = \beta_6\). This means that these searched candidate modules as well as the searched candidate attention operations are indistinguishable.
To tackle this problem, like [51], we introduce the architecture regularization loss $L_{\text{arch}}$ to regularize the architecture parameters, as follows:

$$
L_{\text{arch}} = - \frac{1}{U} \sum_{\alpha \in \{\alpha_0, \alpha_1\}} (\alpha \log \alpha + (1 - \alpha) \log(1 - \alpha)) \\
- \frac{1}{V} \sum_{\beta \in \{\beta_0, \beta_1, \ldots, \beta_n\}} (\beta \log \beta + (1 - \beta) \log(1 - \beta)). 
$$

(17)

Accordingly, based on Eq. 9 and Eq 17 we relax the discrete architecture parameters into a continuous distribution which approximates to either 0 or 1.

By jointly considering the external loss, internal loss, architecture regularization loss, and model complexity loss, the overall formulation of $L_{\text{trainB}}$ for optimizing the architecture weight $\theta$ is expressed as:

$$
L_{\text{trainB}} = L_{\text{ext}} + L_{\text{int}} + \lambda_{\text{arch}} L_{\text{arch}} + \lambda_{\text{comp}} L_{\text{comp}},
$$

(18)

where $\lambda_{\text{arch}}$ and $\lambda_{\text{comp}}$ denote the hyper-parameters to balance the corresponding losses $L_{\text{arch}}$ and $L_{\text{comp}}$.

IV. EXPERIMENTS

A. Experimental Protocol

1) Datasets: In this research, we adopt two different synthetic datasets to validate the superiority of our proposed MANAS method for image de-raining.

The first dataset, called DID-MDN [25], was created according to the additive composite model. The rainy image in this dataset is a simple superimposition of the background layer and the rain streak layer, without considering the rain accumulation that could produce veiling/fog effect. The DID-MDN dataset consists of 4400 three-to-one rainy/clean image pairs, i.e., every three rainy images correspond to a common clean background image, and each rainy image in the pair represents a rain-density level such as light rain, medium rain, and heavy rain, as shown in Fig. 5 We utilized 4000 three-to-one rainy/clean image pairs ($\text{trainA} = 2000$ and $\text{trainB} = 2000$) to train and find the best multi-scale attentive neural architecture for the de-raining network, and then test it with 400 three-to-one rainy/clean image pairs to report the final de-raining performance. Note that our MANAS was still trained and tested according to the original training and testing split in the DID-MDN dataset, i.e., 12000 rainy images and 4000 clean background images (equal to 4000 three-to-one...
rainy/clean image pairs) for training, and 1200 rainy images and 400 clean background images (equal to 400 three-to-one rainy/clean image pairs) for testing.

The second dataset, called RainCityscapes [46], was created according to the depth-aware rain model, in which the depth information is used to distribute different degrees of rain streaks and fog in the images. The RainCityscapes dataset consists of 10620 rainy images with 9432 for training and 1188 for testing. In this work, we partitioned 9432 training images and 1188 testing images into three fog-thickness levels, according to the fog-thickness attenuation coefficient provided by [46], i.e., light fog, medium fog, and heavy fog, as shown in Fig. 6. Likewise, we still followed the original training and testing split in the RainCityscapes dataset. That is we employed 3144 three-to-one rainy/clean image pairs (trainA = 1572 and trainB = 1572), corresponding to 9432 rainy images and 3144 clean background images in the training set of RainCityscapes dataset, to train and find the best multi-scale attentive neural architecture for the de-raining network, and 396 three-to-one rainy/clean image pairs, corresponding to 1188 rainy images and 396 clean background images in the testing set of RainCityscapes dataset, to report the corresponding results.

2) Evaluation Metrics: For the performance evaluation, we adopted three commonly-used metrics including Peak Signal to Noise Ration (PSNR), Structure Similarity Index (SSIM) [69], and Natural Image Quality Evaluator (NIQE) [70], to evaluate the performance of rain removal on synthesized datasets. Generally speaking, higher PSNR and SSIM while lower NIQE values indicate better de-raining results. Due to the fact that the real-world rainy images have no corresponding clean ground truth images, we thus adopted a blind de-raining quality assessment model B-FEN [71] to estimate scores for the de-rained images. A higher score for the de-rained image generally means it has better visual quality.

B. Implementation Details

Our MANAS method was trained by the specifically-designed multi-to-one training strategy in a stage-wise manner, which involves two training stages, namely the architecture search stage and the model training stage. In the architecture search stage, the proposed MANAS method was trained to optimize the architecture weights ω and the network weights θ. For optimizing ω, we used a standard SGD optimizer with the momentum of 0.9 and the weight decay of 3 \times 10^{-4}. And the learning rate is automatically decayed from 2 \times 10^{-3} to 1 \times 10^{-4} by the cosine annealing strategy [72]. For optimizing θ, we adopted the Adam optimizer [73] with the basic learning rate of 3 \times 10^{-4} and the weight decay of 1 \times 10^{-3}. To make a fair comparison with other competing methods, we ignored the model complexity by simply picking the hyperparameter λ_{comp} at 0. In our later experiments, we will show the effect of this hyper-parameter on balancing the de-raining performance and the model complexity. For λ_{arch}, we fixed it at 0.01. In the model training stage, we encoded the architecture parameters into binary and froze them, and then fine-tune the network weights ω with the overall training data set S. We adopted the Adam optimizer to optimize the network weights ω. And the learning rate is decayed from 1 \times 10^{-3} to 0 via the cosine annealing strategy. We assigned three multi-scale attentive cells (i.e., T = 3) in the de-raining network to have a comparison with other competing methods. In our later ablation experiments, we will study the influence of the cell number on de-raining performance.

For DID-MDN [25] dataset, we followed the same practice in [51], [62], randomly cropping the 512 \times 512 original resolution images to 64 \times 64 image patches to augment the data and improve the search efficiency. For RainCityscapes [46] dataset, since the scene depth plays an important role in distributing different degrees of rain streaks and fog in the image, randomly cropping the original resolution image will inevitably destroy the scene depth, and in turn breaks the spatial distribution of rain streaks and fog in the image. With this consideration, we followed the practice in [42], [46], scaling the 1024 \times 2048 original resolution images to the 128 \times 256 versions to maintain the distribution pattern of the rain streaks and fog in the image. In addition, during the training procedure, we employed the image flip to augment the training data. All experiments are conducted on a server with NVIDIA GTX 2080Ti and Intel(R) Xeon(R) CPU E5-2630 2.20GHz. In our single GPU implementation, the training of the proposed MANAS on DID-MDN and RainCityscapes datasets respectively take 102 and 91 hours. The training convergence curves of MANAS on these two datasets are shown in Fig. 7. As can be seen, the architecture search and the model training both converge within 200 epochs, and the model training can present a faster convergence rate and lower training loss.

C. Comparison with the State-of-the-Art

1) Results on Synthetic Rain Removal: Table II shows the quantitative results of the proposed method and previous image de-raining methods on two synthetic datasets, i.e., DID-MDN [25] and RainCityscapes [46]. Regarding the previous image de-raining methods, if authors have publicly released their de-raining models which had been trained with the training set of DID-MDN [25] and RainCityscapes [46] datasets,

1Note that the fog in the images can be viewed as the rain, due to the fact that the fog effect in the real world is caused by the rain streak accumulation [30].

2The training includes the architecture search and the model training.
Table II
Comparison of PSNR, SSIM, and NIQE results by different methods on the test set of DID-MDN \cite{25} and RainCityscapes \cite{46}, respectively. The best results are marked in bold.

| Method          | Dataset DID-MDN \cite{25} | Year | PSNR ↑ | SSIM ↑ | NIQE ↓ | PSNR ↑ | SSIM ↑ | NIQE ↓ |
|-----------------|---------------------------|------|--------|--------|--------|--------|--------|--------|
| DSC \cite{19}   |                           | 2015 | 21.44  | 0.790  | 5.56   | 16.25  | 0.775  | 5.94   |
| GMM \cite{20}   |                           | 2016 | 22.75  | 0.835  | 5.37   | 17.80  | 0.817  | 5.88   |
| JCAS \cite{74}  |                           | 2017 | 23.62  | 0.777  | 4.16   | 15.66  | 0.771  | 6.01   |
| DID-MDN \cite{25}|                           | 2018 | 27.95  | 0.909  | 4.04   | 28.43  | 0.935  | 4.40   |
| PreNet \cite{29} |                           | 2019 | 30.93  | 0.905  | 3.71   | 30.60  | 0.967  | 3.88   |
| MSPFN \cite{32} |                           | 2020 | 28.46  | 0.845  | 3.96   | 25.26  | 0.934  | 4.31   |
| CLEARER \cite{51}|                           | 2020 | 31.43  | 0.893  | 3.62   | 32.01  | 0.976  | 3.68   |
| DGNL-Net-fast \cite{46}|                     | 2021 | 31.21  | 0.877  | 3.64   | 30.59  | 0.956  | 4.15   |
| DGNL-Net \cite{46}|                           | 2021 | 32.41  | 0.910  | 3.52   | 32.44  | 0.973  | 4.01   |
| MANAS (Ours)    |                           | -    | 32.60  | 0.922  | 3.59   | 35.19  | 0.984  | 3.38   |

Figure 8. Comparison of qualitative and quantitative results on three synthetic rainy images with the same background from the test set of DID-MDN \cite{25}. The purple boxes remark the local details after de-raining. Quantitative results PSNR(dB)/SSIM are documented below each image, and the best results are marked in bold.

respectively, then they were directly used to evaluate the performance on the corresponding testing set. These pre-trained models were trained and tested according to the original settings (e.g., hyper-parameters and input size) mentioned in their corresponding published papers. On the other hand, for those de-raining methods whose authors did not provide the pre-trained models, we re-trained models from scratch with their publicly released codes and followed their original settings under these two datasets. After training, we used the well-trained models to evaluate the performance on the corresponding testing set.

As shown in Table II, our MANAS method exceeds other competing methods and achieves the best average performance. In particular, the proposed MANAS method is superior to the second best method, i.e., DGNL-Net \cite{46}, by 1.47 dB, 0.011, and 0.28 in terms of average PSNR, SSIM, and NIQE, respectively. In addition to quantitative comparisons, our method also outperforms other competing methods in qualitative comparisons. As shown in Fig. 8, three representative synthetic rainy images with the same background scene were selected from DID-MDN dataset to perform the rain streak removal. It can be observed that our proposed MANAS method removes rain streaks more clearly, and meanwhile preserve image details better than previous image de-raining methods. Furthermore, we also adopted three representative synthetic rainy images selected from RainCityscapes \cite{46} dataset to conduct the rain streak and fog removal. As shown in Fig. 9, our method better removes the rain streaks and fog in the line of sight, while other de-raining methods could still remain some rain streaks or fog, or produce undesirable local artifacts.
Figure 9. Comparison of qualitative and quantitative results on three synthetic rainy images with the same background from the test set of RainCistyscapes [46]. The green boxes remark the regions where remain some rain streaks or fog, the blue boxes remark the regions with color distortion, and the red boxes remark the regions with artifacts. Please zoom in to see the details.

Table III

| Method          | Rainy inputs | JCAS [74] | PReNet [29] | CLEARER [51] | DGNL-Net [46] | MANAS (Ours) |
|-----------------|--------------|-----------|-------------|--------------|---------------|--------------|
| Average Score ↑ | 0.2997       | 0.3202    | 0.3025      | 0.3216       | 0.3109        | 0.3281       |

Figure 10. Comparison of qualitative and quantitative results on real rainy images from DQA [71] database. All of the de-rained images were derived from the de-raining models that had been trained on the RainCityscapes [46] dataset, except JCAS [74]. Quantitative results (i.e., predicted quality score by B-FEN model [71]) are documented below each image, and the best results are marked in bold.

and color distortion in the de-rained images, please zoom in to see the details. The superior performance on quantitative and qualitative evaluations consistently demonstrates the effectiveness of our MANAS method on synthetic rain removal.

2) Results on Real-world Rain Removal: To demonstrate the practicality, we conduct real-world rain removal on the De-raining Quality Assessment (DQA) [71] database which contains 206 real rainy images. To make a fair comparison, except the hand-crafted prior-based de-raining methods requiring no training, all of the competing methods were trained on the RainCityscapes dataset and then applied to remove rain from the real rainy images. The quantitative results, i.e., average score over 206 de-rained images, are tabulated in Table III. As can be seen, our MANAS achieves a higher average score than other de-raining methods compared against. Besides, six real-world rainy images were selected from the DQA database and the Internet to conduct de-raining visual comparisons. As shown in Fig. 10 and Fig. 11, the JCAS [74], which belongs to the hand-crafted prior-based de-raining method, fails to remove the fog that comes with the rain. And other competing methods either cannot well operate the rain streaks in the images or more easily produce undesirable artifacts and color distortion. In contrast, our method can more sufficiently remove the rain streaks and fog in the real-world rainy images, leading to vivid and clear de-rained results. This implies that our MANAS method is also effective for real-world rain removal and outperforms previous de-raining methods.
Figure 11. Comparison of qualitative and quantitative results on real rainy images collected from the Internet. Likewise, quantitative results (i.e., predicted quality score by B-FEN model [71]) are documented below each image.

Table IV

Comparison of mAP (%) and mIoU (%) results on RainCityscapes [46].

| Method                        | mAP ↑ | mIoU ↑ |
|-------------------------------|-------|--------|
| Rainy images                  | 25.25 | 40.89  |
| De-rained images (by MANAS)   | 32.88 | 46.61  |
| Ground-truth images           | 54.79 | 48.39  |

Figure 12. Object detection and segmentation results on a (a) synthetic rainy image from the RainCityscapes [46], and on its corresponding (b) de-rained image produced by our MANAS method. The first row shows the reference images before conducting the object detection and segmentation task.

Figure 13. Object detection and segmentation results on a (a) real rainy image, and on its corresponding (b) de-rained image produced by our MANAS method. The first row shows the reference images before conducting the object detection and segmentation task.

3) Preprocessing for High-level Vision Tasks: Image de-raining can be applied to enhance the image content that has been degraded by the rain, thus such a technique has great potential for improving the visibility of objects under various rainy weather conditions [6]. In this section, we investigate how our MANAS method contributes to improving the performance of object detection and segmentation. Specifically, we downloaded the publicly released Mask R-CNN model [75] which had been pre-trained on the COCO [76] dataset, and then used it to detect and segment the objects in rainy images, de-rained images (by MANAS), and rain-free ground-truth images inside the RainCityscapes [46] dataset. Table IV reports the quantitative results in terms of mean Average Precision (mAP) and mean Intersection over Union (mIoU). It is obvious that, after applying our method for image de-raining, the mAP and mIoU of de-rained images show notable improvements over those of original rainy images by 7.63% and 5.72%, respectively. In addition, Fig. 12 and Fig. 13 show the visual results of the object detection and segmentation before and after applying our MANAS method for removing the rain streaks and fog in the images. It can be intuitively observed that our method helps improve the performance of object detection and segmentation. This study implies that the proposed MANAS has the potential to be used as a preprocessing step for the object detection and segmentation task.

D. Validation of the Proposed MANAS Method

To further validate the proposed MANAS method, we conduct deeper analyses on DID-MDN [25] dataset. Specifically, we have investigated the following three aspects: 1) the importance of the multi-scale attentive neural architecture for image de-raining; 2) the effectiveness of our multi-to-one training strategy for improving the robustness; 3) the role of
the model complexity loss \( L_{\text{comp}} \) on controlling the trade-off between performance and model size.

1) How Important Is the Multi-scale Attentive Neural Architecture for Image De-raining?: As mentioned earlier, all of the multi-scale attentive cells in the de-raining network integrate the novel multi-scale attentive neural architecture, and the more cells in the de-raining network, the more high-scale to low-scale attentive sub-networks will be incorporated into the de-raining network. Accordingly, to demonstrate the importance of the multi-scale attentive neural architecture for image de-raining, we have conducted an ablation experiment to evaluate the quantitative performance after we gradually reduce the number of cells in the de-raining network. The corresponding results are shown in Table. VII. We observe that as we reduced the number of multi-scale attentive cells, the performances on PSNR, SSIM, and NIQE are consistently degraded. This is due to the fact that the fewer multi-scale attentive cells will lead to fewer high-scale to low-scale attentive sub-networks, which in turn weakens the representation ability of the de-raining network to learn the highly representative and discriminative multi-scale attentive features to handle the image de-raining task. Furthermore, once we remove all of the multi-scale attentive cells \( (i.e., T = 0) \), the performance drops drastically, as shown in Table. VII only 25.66 (dB)/0.825/3.79 on PSNR/SSIM/NIQE, much lower than those with multi-scale attentive cells in the de-raining network. This study clearly demonstrates the importance of the multi-scale attentive neural architecture for image de-raining.

2) Whether the Multi-to-One Training Strategy Could Make Our Model Robust?: To answer this question, we also trained our MANAS method with a one-to-one training strategy which was commonly used in previous deep-learning-based de-raining methods. The one-to-one training strategy adopts one-to-one image pairs for training, and each pair contains only one rainy image and one ground truth image, thus this training strategy cannot use the internal loss \( L_{\text{int}} \) like our multi-to-one training strategy. On the other hand, the one-to-one training paradigm does not mean that fewer rainy samples were used for training. Instead, all of the rainy images in the training set were used. The quantitative comparisons between one-to-one and multi-to-one training strategies are shown in Table. VII. As can be seen, better performances are obtained by our multi-to-one training strategy. Furthermore, in Fig. 14 we showcase three synthetic rainy images that have the same background but are degraded by light rain, medium rain, and heavy rain, respectively (from top to bottom), as well as their corresponding de-rained versions resulting from the one-to-one and our multi-to-one training strategies. We observe that the one-to-one training strategy could lead to parts of the de-rained images presenting serious distortions, \( i.e., \) color inconsistency and local artifacts, while the proposed multi-to-one training strategy can well avoid these issues, see the second and third columns in Fig. 14 for a comparison. Intuitively, this could be due to the fact that the internal loss \( L_{\text{int}} \) in the multi-to-one training strategy can well constrain the de-rained images that have the same background to be close with each other, thus this to some extent prevents the distortion after de-raining. This study shows that our multi-to-one training strategy can indeed make our de-raining model robust, and outperforms the previously commonly-used one-to-one training strategy.

3) How Does \( \lambda_{\text{comp}} \) Control the Trade-off Between Performance and Model Size?: To study this point, we set \( \lambda_{\text{comp}} \) to different values to verify the role of the model complexity loss, \( i.e., L_{\text{comp}} \), on balancing the performance and the model size. The corresponding results are shown in Table. VII. It is interesting to see that as we increase the value of \( \lambda_{\text{comp}} \), the model size will become smaller, and meanwhile the performance will be worse but still maintain a relatively high quantitative results.

### Table V
PSNR, SSIM, AND NIQE COMPARISON FOR DIFFERENT CELL NUMBERS
\( T = 3, 2, 1, 0. \)

| Method   | PSNR | SSIM | NIQE |
|----------|------|------|------|
| MANAS\(_T=3\) | 32.60 | 0.922 | 3.59 |
| MANAS\(_T=2\) | 31.98 | 0.913 | 3.61 |
| MANAS\(_T=1\) | 31.19 | 0.904 | 3.65 |
| MANAS\(_T=0\) | 25.66 | 0.825 | 3.79 |

### Table VI
COMPARISON OF PSNR, SSIM, AND NIQE PERFORMANCES USING ONE-TO-ONE AND MULTI-TO-ONE TRAINING STRATEGIES.

| Method | PSNR | SSIM | NIQE |
|--------|------|------|------|
| One-to-one training | 31.52 | 0.905 | 3.62 |
| Multi-to-one training | 32.60 | 0.922 | 3.59 |

Figure 14. Three synthetic rainy images with the same background from DID-MDN [25] dataset, and their corresponding de-rained images resulting from one-to-one and multi-to-one training strategies.

### Table VII
PSNR, SSIM, AND NIQE COMPARISON FOR DIFFERENT \( \lambda_{\text{comp}} \) VALUES.

| Configurations | PSNR | SSIM | NIQE | Model Size (M) |
|----------------|------|------|------|----------------|
| \( \lambda_{\text{comp}} = 0.00 \) | 32.60 | 0.922 | 3.59 | 8.19 |
| \( \lambda_{\text{comp}} = 0.01 \) | 32.58 | 0.921 | 3.59 | 7.96 |
| \( \lambda_{\text{comp}} = 0.10 \) | 32.37 | 0.919 | 3.60 | 6.79 |
| \( \lambda_{\text{comp}} = 1.00 \) | 32.14 | 0.914 | 3.62 | 5.85 |
E. Limitations

Although the proposed MANAS framework can automatically search and integrate the multi-scale attentive neural architecture for image de-raining, there still exist some limitations in this proposed method. For example, the number and width of the multi-scale attentive cells in the de-raining network are required to be set manually. Recently, Fang et al. [65] proposed to automatically search block counts and block widths in a dense super network using neural architecture search (NAS). Inspired by their work, the number and width of the multi-scale attentive cells may be able to be searched alternatively or synchronously in the de-raining network. We leave this study as future work.

V. Conclusion

In this work, we proposed a novel multi-scale attentive neural architecture search (MANAS) framework for image de-raining. In brief, we formulated a new multi-scale attention search space consisting of multiple typical basic modules that are beneficial for image de-raining. Meanwhile, we integrated the multi-scale attention search space into a differentiable form through a continuous relaxation operation, and exploited a gradient-based search algorithm to automatically search the internal multi-scale attentive architecture for the de-raining network. On the other hand, to generate a robust de-raining model, we also specifically devised a practical and effective multi-to-one training strategy for image de-raining. We conducted extensive image de-raining experiments on both synthetic and realistic datasets, and also applied our MANAS method to high-level vision applications, i.e., object detection and segmentation. The experimental results consistently demonstrate the superiority of our method.

REFERENCES

[1] K. Jiang, Z. Wang, P. Yi, C. Chen, Z. Han, T. Lu, B. Huang, and J. Jiang. “Decomposition makes better rain removal: An improved attention-guided deraining network,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 31, no. 10, pp. 3981–3995, 2021.
[2] L. Zhu, Z. Deng, X. Hu, H. Xie, X. Xu, J. Qin, and P. A. Heng. “Learning gated non-local residual for single-image rain streak removal,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 31, no. 6, pp. 2147–2159, 2021.
[3] J. L. Yin, Y. C. Huang, B. H. Chen, and S. Z. Ye. “Color transferred convolutional neural networks for image dehazing,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 30, no. 11, pp. 3957–3967, 2020.
[4] S. C. Agrawal and A. S. Jalal. “Dense haze removal by nonlinear transformation,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 2, pp. 593–607, 2022.
[5] D. W. Jaw, S. C. Huang, and S. Y. Kuo. “Desnowgan: An efficient single image snow removal framework using cross-resolution lateral connection and gans,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 31, no. 4, pp. 1342–1350, 2021.
[6] L. Cai, Y. Fu, T. Zhu, Y. Xiang, Y. Zhang, and H. Zeng. “Joint depth and density guided single image de-raining,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 7, pp. 4108–4121, 2022.
[7] X. Chen, H. Li, Q. Wu, K. N. Ngan, and L. Xu. “High-quality r-cnn object detection using multi-path detection calibration network,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 31, no. 2, pp. 715–727, 2020.
[8] W. Liu, G. Lin, T. Zhang, and Z. Liu. “Guided co-segmentation network for fast video object segmentation,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 31, no. 4, pp. 1607–1617, 2020.
[9] D. Feng, C. Haase-Schütz, and et al. “Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges,” IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 3, pp. 1341–1360, 2021.
[10] J. Chen, C. H. Tan, J. Hou, L. P. Chau, and H. Li. “Detection and removal of rain from videos,” IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 520–528, 2004.
[11] K. Garg and S. K. Nayar. “Photo-realistic rendering of rain streaks,” ACM Transactions on Graphics, vol. 25, no. 3, pp. 996–1002, 2006.
[12] X. Zhang, H. Li, Y. Qi, W. K. Leow, and T. K. Ng. “Rain removal in video by combining temporal and chromatic properties,” IEEE International Conference on Multimedia and Expo, pp. 461–464, 2006.
[13] N. Brewer and N. Liu, “Using the shape characteristics of rain to identify and remove rain from video,” Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR), pp. 451–458, 2008.
[14] P. C. Barnum, S. Narasimhan, and T. Kanade. “Analysis of rain and snow in frequency space,” International Journal of Computer Vision, vol. 82, no. 2, pp. 256–274, 2010.
[15] J. Bossu, N. Hautiere, and J. P. Tarel. “Rain or snow detection in image sequences through use of a histogram of orientation of streaks,” International Journal of Computer Vision, vol. 93, no. 3, pp. 348–367, 2011.
[16] Y. L. Chen and C. T. Hsu. “A generalized low-rank appearance model for spatio-temporally correlated rain streaks,” IEEE International Conference on Computer Vision, pp. 1968–1975, 2013.
[17] D. Eigen, F. Krishnan, and R. Fergus. “Restoring an image taken through a window covered with dirt or rain,” IEEE International Conference on Computer Vision, pp. 633–640, 2013.
[18] W. Wei, L. Yi, Q. Xie, Q. Zhao, D. Meng, and Z. Xu. “Should we encode rain streaks in video as deterministic or stochastic?,” IEEE International Conference on Computer Vision, pp. 2516–2525, 2017.
[19] Y. Luo, Y. Xu, and H. Ji. “Removing rain from a single image via discriminative sparse coding,” IEEE International Conference on Computer Vision, pp. 3397–3405, 2015.
[20] Y. Li, R. T. Tan, X. Guo, J. Lu, and M. S. Brown. “Rain streak removal using layer priors,” IEEE Conference on Computer Vision and Pattern Recognition, pp. 2736–2744, 2016.
[21] J. H. Kim, C. Lee, J. Y. Sim, and C. S. Kim. “Single-image deraining using an adaptive nonlocal means filter,” IEEE International Conference on Image Processing, pp. 914–917, 2013.
[22] X. Fu, J. Huang, X. Ding, Y. Liao, and J. Paisley. “Clearing the skies: A deep network architecture for single-image rain removal,” IEEE Transactions on Image Processing, vol. 26, no. 6, pp. 2944–2956, 2017.
[23] W. Yang, R. T. Tan, J. Feng, J. Liu, Z. Guo, and S. Yan. “Deep joint rain detection and removal from a single image,” IEEE Conference on Computer Vision and Pattern Recognition, pp. 1357–1366, 2017.
[24] Z. Fan, H. Wu, X. Fu, Y. Huang, and X. Ding. “Residual-guide network for single image deraining,” ACM International Conference on Multimedia, pp. 1751–1759, 2018.
[25] H. Zhang and V. M. Patel. “Density-aware single image de-raining using a multi-stream dense network,” IEEE Conference on Computer Vision and Pattern Recognition, pp. 695–704, 2018.
[26] L. W. Kang, C. W. Lin, and Y. H. Fu. “Automatic single-image-based rain streaks removal via image decomposition,” IEEE Transactions on Image Processing, vol. 21, no. 4, pp. 1742–1755, 2012.
[27] D. Y. Chen, C. C. Chen, and L. W. Kang. “Visual depth guided color image rain streaks removal using sparse coding,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 24, no. 8, pp. 1430–1455, 2014.
[28] X. Li, J. Wu, Z. Lin, H. Liu, and H. Zha. “Recurrent squeeze-and-excitation context aggregation net for single image deraining,” European Conference on Computer Vision, pp. 254–269, 2018.
[29] D. Ren, W. Zuo, Q. Hu, P. Zhu, and D. Meng. “Progressive image deraining networks: A better and simpler baseline,” IEEE Conference on Computer Vision and Pattern Recognition, pp. 3937–3946, 2019.
[30] W. Yang, R. T. Tan, J. Feng, Z. Guo, S. Yan, and J. Liu. “Joint rain detection and removal from a single image with contextualized deep networks,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 42, no. 6, pp. 1377–1393, 2019.
[31] H. Zhang, V. Sindagi, and V. M. Patel. “Image de-raining using a conditional generative adversarial network,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 30, no. 11, pp. 3943–3956, 2019.
[32] K. Jiang, Z. Wang, P. Yi, C. Chen, B. Huang, Y. Luo, J. Ma, and J. Jiang. “Multi-scale progressive fusion network for single image deraining,”
S. Deng, M. Wei, J. Wang, Y. Feng, L. Liang, H. Xie, F. L. Wang, and M. Wang, “Detail-recovery image deraining via context aggregation networks,” *IEEE International Conference on Computer Vision*, pp. 14560–14569, 2020.

W. Yang, Q. Xie, Z. Zhao, and D. Meng, “A model-driven deep neural network for single image rain removal,” *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3103–3112, 2020.

W. Wang, M. Wei, J. Wang, Y. Feng, L. Liang, H. Xie, F. L. Wang, and M. Wang, “Scale-free single image deraining via visibility-enhanced recurrent wavelet learning,” *IEEE Transactions on Image Processing*, vol. 28, no. 6, pp. 2948–2961, 2019.

W. Yang, R. T. Tan, S. Wang, Y. Fang, and J. Liu, “Single image deraining: From model-based to data-driven and beyond,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 11, pp. 4059–4077, 2021.

Y. Wang, Y. Song, C. Ma, and B. Zeng, “Rethinking image deraining via rain streaks and vapors,” *European Conference on Computer Vision*, pp. 367–382, 2020.

H. Lin, Y. Li, X. Fu, X. Ding, Y. Huang, and J. Paisley, “Rain o’er me: Synthesizing real rain to derain with data distillation,” *IEEE Transactions on Image Processing*, vol. 29, pp. 7668–7680, 2020.

X. Lin, L. Ma, B. Sheng, Z. J. Wang, and W. Chen, “Utilizing two-phase processing with fbs for single image deraining,” *IEEE Transactions on Multimedia*, vol. 23, pp. 664–670, 2020.

X. Fu, B. Liang, Y. Huang, X. Ding, and J. Paisley, “Lightweight pyramid networks for image deraining,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 6, pp. 1794–1807, 2019.

B. Ma, J. Zhang, Y. Xia, and D. Tao, “Auto learning attention,” *IEEE Transactions on Image Processing*, vol. 29, no. 5, pp. 1650–1674, 2021.

Y. Guo, Y. Luo, Z. He, J. Huang, and J. Chen, “Hierarchical neural architecture search for single image super-resolution,” *IEEE Signal Processing Letters*, vol. 27, pp. 1255–1259, 2020.

H. Zhang, Y. Li, H. Chen, and C. Shen, “Memory-efficient hierarchical neural architecture search for image denoising,” *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3657–3666, 2019.

J. Fang, Y. Sun, Q. Zhang, Y. Li, W. Liu, and X. Wang, “Densely connected search space for more flexible neural architecture search,” *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10628–10637, 2020.

K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Advances in Neural Information Processing Systems*, 2012.

H. Liu, K. Simonyan, and Y. Yang, “Darts: Differentiable architecture search,” *arXiv preprint arXiv:1806.09055*, 2018.

Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

A. Mittal, R. Soudararajan, and A. C. Bovik, “Making a “completely blind” image quality analyzer,” *IEEE Signal Processing Letters*, vol. 20, no. 3, pp. 209–212, 2012.

P. Wu, X. Wang, K. N. Nguyen, H. Li, F. Meng, and L. Xu, “Subjective and objective de-raining quality assessment towards authentic rain image,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 11, pp. 3883–3897, 2020.
Lei Cai received the M.S. degree in communication and information system from the School of Information Science and Engineering, Huaqiao University, Xiamen, China, in 2018. He is currently pursuing the Ph.D degree at the School of Electronic and Information Engineering, South China University of Technology. His research interests include pedestrian gender recognition, compressed sensing, and image restoration.

Yuli Fu received the Ph.D. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2000. He is currently a Full Professor with the School of Electronic and Information Engineering, South China University of Technology, Guangzhou, China. He has authored or coauthored about 80 journal papers on control theory, neural networks and signal processing. His current research interests include adaptive signal processing, artificial intelligence, and pattern recognition.

Wanliang Huo received the B.E. degree from the South China University of Technology, Guangzhou, China, in 2020. He is currently pursuing the M.S. degree at the School of Electronic and Information Engineering. His research interests include machine learning and computer vision, particularly focusing on the image de-raining problem.

Youjun Xiang received the M.S. degree in pattern recognition and intelligent system from Xidian University, Xi’an, China, and the Ph.D. degree in signal and information processing from South China University of Technology, Guangzhou, China. She is currently an Associate Professor with the South China University of Technology. Her research interests include video coding, artificial intelligence, and pattern recognition.

Tao Zhu received the Ph.D degree in signal and information processing from the School of Electronic and Information Engineering, South China University of Technology, Guangzhou, China, in 2021. His research interests include inverse problem, dictionary learning, and independent component analysis.

Ying Zhang received the B.E. degree from the South China University of Technology, Guangzhou, China, in 2020. He is currently pursuing the M.S. degree at the School of Electronic and Information Engineering, South China University of Technology. His research interests include image de-raining and related image restoration problems.

Huanqiang Zeng (S’10-M’13-SM’18) received the B.S. and M.S degrees in electrical engineering from Huaqiao University, China, and the Ph.D. degree in electrical engineering from Nanyang Technological University, Singapore. He is currently a Full Professor at the School of Engineering and the School of Information Science and Engineering, Huaqiao University. Before that, he was a Postdoctoral Fellow at The Chinese University of Hong Kong, Hong Kong. He has published more than 100 papers in well-known journals and conferences, including three best poster/paper awards (in the International Forum of Digital TV and Multimedia Communication 2018 and the Chinese Conference on Signal Processing 2017/2019). His research interests include image processing, video coding, machine learning, and computer vision. He has also been actively serving as the General Co-Chair for IEEE International Symposium on Intelligent Signal Processing and Communication Systems 2017 (ISPACS2017), the Co-Organizer for ICME2020 Workshop on 3D Point Cloud Processing, Analysis, Compression, and Communication, the Technical Program Co-Chair for Asia-Pacific Signal and Information Processing Association Annual Summit and Conference 2017 (APSIPA-ASC2017), the Area Chair for IEEE International Conference on Visual Communications and Image Processing (VCIP2015 and VCIP2020), and a technical program committee member for multiple flagship international conferences. He has been actively serving as an Associate Editor for IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and Electronics Letters (IET). He has been actively serving as a Guest Editor for Journal of Visual Communication and Image Representation, Multimedia Tools and Applications, and Journal of Ambient Intelligence and Humanized Computing.

Delu Zeng received his Bachelor degree in applied mathematics and his Phd. degree in signal and information processing in South China University of Technology (SCUT) in June 2003 and June 2010, respectively. He has been visiting scholar in Columbia University, University of Waterloo, University of Oulu. And He is currently a full professor with the School of Electronic and Information Engineering in South China University of Technology (SCUT) in Guangzhou. His research interests focus on statistics learning, image and speech processing, computational intelligence, machine learning, fitting and approximation, and their applications to communications, industrial intelligence, etc.