RCDNet: An Interpretable Rain Convolutional Dictionary Network for Single Image Deraining

Hong Wang®, Qi Xie®, Qian Zhao®, Yuexiang Li®, Yong Liang®, Yefeng Zheng®, Fellow, IEEE, and Deyu Meng®, Member, IEEE

Abstract—As common weather, rain streaks adversely degrade the image quality and tend to negatively affect the performance of outdoor computer vision systems. Hence, removing rains from an image has become an important issue in the field. To handle such an ill-posed single image deraining task, in this article, we specifically build a novel deep architecture, called rain convolutional dictionary network (RCDNet), which embeds the intrinsic priors of rain streaks and has clear interpretability. In specific, we first establish a rain convolutional dictionary (RCD) model for representing rain streaks and utilize the proximal gradient descent technique to design an iterative algorithm only containing simple operators for solving the model. By unfolding it, we then build the RCDNet in which every network module has clear physical meanings and corresponds to each operation involved in the algorithm. This good interpretability greatly facilitates an easy visualization and analysis of what happens inside the network and why it works well in the inference process. Moreover, taking into account the domain gap issue in real scenarios, we further design a novel dynamic RCDNet, where the rain kernels can be dynamically inferred corresponding to input rainy images and then help shrink the space for rain layer estimation with few rain inputs. By end-to-end training such an interpretable network, all involved rain kernels and proximal operators can be automatically extracted, faithfully characterizing the features of both rain and clean background layers and, thus, naturally leading to better deraining performance. Comprehensive experiments implemented on a series of representative synthetic and real datasets substantiate the superiority of our method, especially on its well generality to diverse testing scenarios and good interpretability for all its modules, compared with state-of-the-art single image derainers both visually and quantitatively. Code is available at https://github.com/hongwang01/DRCDNet.

Index Terms—Dictionary learning, generalization performance, interpretable deep learning (DL), single image rain removal.

I. INTRODUCTION

Images and videos captured in rainy scenes always suffer from noticeable visual degradations, tending to adversely affect outdoor computer vision tasks, such as automatic driving and video surveillance [1]. As a hot research topic, rain removal from images and videos has brought considerable attention to the research community [2], [3], [4]. In this work, we focus on the single image deraining task.

Recent years have witnessed significant progress in single-image deraining, which can be mainly categorized into two research lines. One is the traditional unsupervised (i.e., prior-based) method, which focuses on exploring the prior structures of background and rain layers to constrain the solution space of a carefully designed optimization model. Typically presented priors include frequency information [5], sparse representation [6], [7], [8], [9], and local patch low rankness [9]. Very recently, researchers explore that rain streaks repeatedly appear at different locations over a rainy image with similar local patterns, such as shape, thickness, and direction [9], [10]. They formulate such an understanding (i.e., nonlocal self-similarity) as a convolutional dictionary learning model, where rain kernels are imposed on sparse rain maps, as intuitively depicted in Fig. 1. This idea has achieved state-of-the-art (SOTA) performance in video deraining when background frames are well extracted based on the temporal information and low-rankness prior in surveillance video sequences [11]. Albeit effective in certain specific scenarios, the rationality of these conventional deraining approaches largely depends on the reliability of such manually designed prior assumptions on the unknown background and rain streaks. However, with subjective and relatively simple forms, these handcrafted priors cannot always comprehensively and adaptively reflect the complex and variant structures underlying real rainy images collected from different resources.

The other popular approach on this task is based on deep learning (DL). The main idea of current deep derainers is to utilize the precollected training samples to learn the mapping function from a rainy image to its corresponding rain-removed background layer with diverse network architectures, including CNN [12], [13], [14], adversarial learning [15], [16], [17],

Manuscript received 14 July 2021; revised 25 August 2022, 1 November 2022, and 23 November 2022; accepted 12 December 2022. Date of publication 9 January 2023; date of current version 4 June 2024. This work was supported in part by the National Key Research and Development Program of China under Grant 2022YFA1004100; in part by the Major Project Key Project of Peng Cheng Laboratory (PCL) under Grant PCL-2021A12; in part by the Macao Science and Technology Development Fund under Grant 061/2020/A2; and in part by the China NSFC Projects under Contract 61721002, Contract U1811461, and Contract 62206214. (Corresponding authors: Qi Xie; Deyu Meng.)

Hong Wang, Yuexiang Li, and Yefeng Zheng are with the Tencent Jarvis Lab, Shenzhen 518057, China (e-mail: hazelhwang@tencent.com; vicxyi@tencent.com; yefengzheng@tencent.com).
Qi Xie and Qian Zhao are with the School of Mathematics and Statistics and the Ministry of Education Key Lab of Intelligent Networks and Network Security, Xi’an Jiaotong University, Xi’an, Shaanxi 710049, China (e-mail: xie.qi@mail.xjtu.edu.cn; timmy.zhaqian@mail.xjtu.edu.cn).
Yong Liang is with the Faculty of Information Technology, Macau University of Science and Technology, Macau (e-mail: yliang@must.edu.mo).
Deyu Meng is with the School of Mathematics and Statistics and the Ministry of Education Key Lab of Intelligent Networks and Network Security, Xi’an Jiaotong University, Xi’an, Shaanxi 710049, China, also with the Peng Cheng Laboratory, Shenzhen 518066, China, and also with the Macao Institute of Systems Engineering, Macau University of Science and Technology, Macau (e-mail: dymeng@mail.xjtu.edu.cn).

This article has supplementary downloadable material available at https://doi.org/10.1109/TNNLS.2022.3231453, provided by the authors.

Digital Object Identifier 10.1109/TNNLS.2022.3231453
The convolutional dictionary learning mechanism for rain layer.

First, we utilize the intrinsic convolutional dictionary learning mechanism to encode rain shapes and propose a concise rain convolutional dictionary (RCD) model for a single rainy image. To solve it, we adopt the proximal gradient technique [33] to develop an optimization algorithm. Different from conventional solvers made up of complicated operators (e.g., Fourier transform), the proposed algorithm only consists of simple computations [see Fig. 2(a)] easy to be implemented by general network modules. This novel manner not only explicitly incorporates the intrinsic prior structures of rain streaks but also facilitates us to easily unfold this algorithm into a deep network architecture.

Second, by unfolding every step of the algorithm, we construct an interpretable network for single image deraining, named as RCDNet. Every module in this network corresponds to the implementation operator of the proposed algorithm, and thus, all network modules have clear physical interpretability, as demonstrated in Fig. 2. Specifically, the RCDNet successively consists of M-net and B-net, updating the rain map $\mathcal{M}$ and the background layer $\mathcal{B}$, respectively. All the operators in these two subnetworks are easy to understand and suitable for extracting rain layers since they are consistent with the corresponding algorithm. Moreover, the rain layer extracted

---

\[ \text{For } x = 0.5 \text{ do} 
\]

\[
\%
\text{Updating } \mathcal{M} \\
1. \quad \mathcal{M}(0) = \mathcal{M}^{(0)} \\
2. \quad \mathcal{M}(i+1) = \mathcal{M}(i) + \lambda_i \nabla \mathcal{M}(0) \\
3. \quad \mathcal{M}(i+1) = \mathcal{M}(i) + \lambda_i \nabla \mathcal{M}(0) \\
4. \quad \mathcal{M}(i+1) = \text{proj}_{\mathcal{M}}(\mathcal{M}(i) + \lambda_i \nabla \mathcal{M}(0)) \]

\[
\%
\text{Updating } \mathcal{B} \\
5. \quad \mathcal{B}(0) = \mathcal{B}^{(0)} \\
6. \quad \mathcal{B}(i+1) = \mathcal{B}(i) - \lambda_i \nabla \mathcal{B}(0) \\
7. \quad \mathcal{B}(i+1) = \text{proj}_{\mathcal{B}}(\mathcal{B}(i) - \lambda_i \nabla \mathcal{B}(0)) \]

---

1Compared with our conference paper [32], the work has made substantial extensions. Specifically, a novel network with fine interpretability and generalization ability is designed. More model analysis, methodology expansions, visualization verifications, and experimental evaluations are provided. Especially, a core extension is that rain kernels (see Fig. 3) are adaptively inferred according to the input rainy image. This dynamic prediction mechanism makes it possible to achieve better generalization performance even when the rain patterns are different between training and testing samples.

2This interpretable network design greatly facilitates us to analyze what happens during the network training and understand the implementation mechanisms (see model visualization in Section VII-B).
by RCDNet naturally complies with the prior constraints and can better exclude the background details, as shown in Fig. 10.

Third, we further construct a dynamic rain convolutional dictionary network, called DRCDNet, for better generalization capability. Unlike RCDNet that estimates a large rain dictionary network, called DRCDNet, for better generalization, DRCDNet can better exclude the background details, as shown in Fig. 10.

Here, DRCDNet is trained on the synthetic Rain100L dataset. For ease of understanding, we introduce some necessary notations and preliminaries as follows.

### II. Notations and Preliminaries

For ease of understanding, we introduce some necessary notations and preliminaries as follows.

Denote \( A \in \mathbb{R}^{H_1 \times W_1 \times \ldots \times H_B} \) as a tensor of order \( N \). The unfolding matrix \( U^d_n(a) \in \mathbb{R}^{k \times (H_1-1) \times (H_2-1) \times \ldots \times (H_B-1)} \) is composed by taking the mode-\( n \) vector of \( A \) as its columns. This matrix can also be seen as the mode-\( n \) flattening of \( A \). The vectorization of \( A \) is \( \text{vec}(A) \in \mathbb{R}^{k \times L_2 \times \ldots \times L_N} \). All these can be easily achieved by the “reshape” function in PyTorch [34].

The symbol \( \otimes \) represents the 2-D convolutional operation. It can be extended to the convolution in the form of tensor in deep networks as

\[
Y = C \otimes X
\]

where \( C \in \mathbb{R}^{k \times k \times N_x \times N_y} \), \( X \in \mathbb{R}^{H \times W \times N_x \times N_y} \), and \( Y \in \mathbb{R}^{H \times W \times N_x \times N_y} \). At the mode-3 of \( Y \), \( Y[i,:,j] = \sum_{n=1}^{N_y} C[i,:,j,n] \otimes X[i,:,;n] \), \( j = 1, 2, \ldots, N_y \). The notation \( \otimes \) between \( C[i,:,;i,n] \) and \( X[i,:,;n] \) is a 2-D convolutional computation. The convolutional operation in (1) can be easily achieved by the off-the-shelf function “torch.nn.Conv2d” in PyTorch.

The symbol \( \otimes^d \) represents the depthwise convolutional operation. Specifically,

\[
Z = C \otimes^d M
\]

where \( M \in \mathbb{R}^{H \times W \times N} \) and \( Z \in \mathbb{R}^{H \times W \times N_x \times N_y} \). Specifically, \( Z[i,:,;j,k,n] = C[i,:,;j,k] \otimes M[i,:,;n], j = 1, 2, \ldots, N_y, k = 1, 2, \ldots, N_x, n = 1, 2, \ldots, N \). The depthwise convolutional operation in (2) can be easily performed through the group convolution by setting the parameter “group” in the function “torch.nn.Conv2d.”

### III. RCD Model for Single Image Deraining

#### A. Model Formulation

Given \( O \in \mathbb{R}^{H \times W \times 3} \) as an observed color rainy image, we can rationally separate it as

\[
O = B + R
\]

where \( H \) and \( W \) are the height and width of the image, respectively; \( B \) and \( R \) are the clear backgrounds and rain layers, respectively.\(^3\) To recover the background, most of the current deep derainers focus on establishing complex network architectures to learn the mapping function between \( O \) and \( B \) (or \( R \)).

Instead of designing complex networks heuristically, we first consider the traditional rain generation model, which reflects the intrinsic prior structures of rain streaks [7], [9], [11]. Specifically, with the RCD physical mechanism, as visually illustrated in Fig. 1, the rain layer can be rationally expressed as

\[
R^c = \sum_{n=1}^{N} K^c_n \otimes M_n, \quad c = 1, 2, 3 \tag{4}
\]

where \( R^c \) denotes the \( c \)-th color channel of \( R \); \( \{K^c_n\}_{n=1}^{N} \subset \mathbb{R}^{k \times k} \) is a set of rain kernels with the size \( k \times k \) representing the repetitive local patterns of rain streaks; \( \{M_n\}_{n=1}^{N} \subset \mathbb{R}^{H \times W} \) is the rain maps representing the locations where local patterns repeatedly appear; \( N \) is the number of rain kernels; and \( \otimes \)

\(^3\)Note that (3) is an approximate model, which provides a rough direction for network learning. During the network implementation in Section IV, we add an adjustment module to flexibly deal with complicated rainy images. Section VII validates the effectiveness of our method in diverse rain scenarios.
is the 2-D convolutional operation. For simplicity, throughout this article, we rewrite (4) as

$$\mathcal{R} = \sum_{n=1}^{N} \mathcal{K}_n \otimes \mathcal{M}_n = \mathcal{K} \otimes \mathcal{M}$$

(5)

where $\mathcal{K}_n \in \mathbb{R}^{K \times K \times 3}$, $\mathcal{K} \in \mathbb{R}^{K \times K \times 3 \times N}$, and $\mathcal{M} \in \mathbb{R}^{M \times W \times N}$ are stacked by $\mathcal{K}_n$’s, $\mathcal{K}$’s, and $\mathcal{M}_n$’s, respectively. The 2-D convolutional operation $\otimes$ between $\mathcal{K}_n$ and $\mathcal{M}_n$ is executed in a channelwise manner, and the computation $\mathcal{K} \otimes \mathcal{M}$ is the extension of $\otimes$ from 2-D to tensor form.

We can rewrite the single rainy image model in (3) as

$$\mathcal{O} = \mathcal{B} + \mathcal{K} \otimes \mathcal{M}.$$  

(6)

It is clear that our goal is to estimate $\mathcal{K}$, $\mathcal{M}$, and $\mathcal{B}$ from $\mathcal{O}$. With sparse constraints on $\mathcal{M}$, it is easy to see that (5) can well model the sparsity and nonlocal similarity of rains.

The rain kernel $\mathcal{K}$ can be viewed as a set of convolutional dictionaries [10] for representing the repetitive and similar local patterns underlying rain streaks. In the training–testing domain match scenario where the rain patterns between training data and testing data are similar, a small number of rain kernels can finely represent a wide range of rain shapes [11]. Thus, they are the common knowledge for representing different rain types across all rainy images and can be learned from abundant training samples by virtue of the strong learning ability of CNN with an end-to-end training manner (see more details in Section IV). Thus, for predicting the clean background from an input rainy image, the key issue is to output $\mathcal{M}$ and $\mathcal{B}$ from $\mathcal{O}$ with $\mathcal{K}$ fixed. Correspondingly, the optimization problem is

$$\min_{\mathcal{M}, \mathcal{B}} \| \mathcal{O} - \mathcal{B} - \mathcal{K} \otimes \mathcal{M} \|^2_F + \lambda_1 p_1(\mathcal{M}) + \lambda_2 p_2(\mathcal{B})$$

(7)

where $\lambda_1$ and $\lambda_2$ are tradeoff parameters; $p_1(\cdot)$ and $p_2(\cdot)$ denote the penalty functions (i.e., regularizers) to deliver the prior structures of $\mathcal{M}$ and $\mathcal{B}$, respectively.

The first term of the problem (7) is a rational approximate model for rain streak generation, which well encodes the sparsity and nonlocal similarity of the rain layer. Motivated by this, we believe that, based on the solver of the problem (7), the constructed deep network modules are able to embed the prior of rain streak and constrain the space for rain layer estimation.

### B. Optimization Algorithm

The deep unfolding technique is an intuitive way to combine the solver of optimization models with DL methods. This technique releases us from manually designing penalty terms but also brings new challenges. The first one is how to develop an optimization algorithm that only contains simple computations easy to be transformed to network modules.

Confronted with the convolutional dictionary representation model (7), the traditional solvers usually contain complex computations, e.g., the Fourier transform and the inverse Fourier transform [10], [11], [78], tending to make this unfolding task difficult. We, thus, prefer to build a new algorithm where the to-the-estimated variables $\mathcal{M}$ and $\mathcal{B}$ are alternately updated by the proximal gradient technique [33]. In this way, the solution process only consists of simple computations, making it possible to easily achieve the transformation from the algorithm to network architectures. The details are given as follows.

Updating $\mathcal{M}$: At the $s$th iteration, the rain map $\mathcal{M}$ can be updated by solving the quadratic approximation [33] of the problem (7) with regard to $\mathcal{M}$ as

$$\min_{\mathcal{M}} \frac{1}{2} \| \mathcal{M} - (\mathcal{M}^{(s-1)} - \eta_1 \nabla f(\mathcal{M}^{(s-1)})\|^2_F + \lambda_1 \eta_1 p_1(\mathcal{M})$$

(8)

where $\lambda_1$ is the stepsize parameter, and $f(\mathcal{M}) = \| \mathcal{O} - \mathcal{B}^{(s-1)} - \mathcal{K} \otimes \mathcal{M}^{(s-1)}\|^2_F$. Under general regularization terms [79], the solution of (8) is expressed as

$$\mathcal{M}^{(s)} = \text{prox}_{\lambda \eta_1}\{ \mathcal{M}^{(s-1)} - \eta_1 \nabla f(\mathcal{M}^{(s-1)}) \}$$

(9)

where $\text{prox}_{\lambda \eta_1}(\cdot)$ is the proximal operator dependent on the regularization term $p_1(\cdot)$ with respect to $\mathcal{M}$. By substituting

$$\nabla f(\mathcal{M}^{(s-1)}) = \mathcal{K} \otimes ^T (\mathcal{K} \otimes \mathcal{M}^{(s-1)} + \mathcal{B}^{(s-1)} - \mathcal{O})$$

(10)

where $^T$ denotes the transposed convolution; we can obtain the updating formula for $\mathcal{M}$ as

$$\mathcal{M}^{(s)} = \text{prox}_{\lambda \eta_1}(\mathcal{M}^{(s-1)} - \eta_1 \mathcal{K} \otimes ^T (\mathcal{K} \otimes \mathcal{M}^{(s-1)} + \mathcal{B}^{(s-1)} - \mathcal{O}))$$

(11)

Instead of being derived from manually designed regularizer as in traditional methods, the form of the implicit proximal operator $\text{prox}_{\lambda \eta_1}(\cdot)$ can be expressed through a convolutional network module and automatically adapted from training data in an end-to-end manner, which is described in Section IV.

Updating $\mathcal{B}$: Similarly, the quadratic approximation of the problem (7) with respect to $\mathcal{B}$ is

$$\min_{\mathcal{B}} \frac{1}{2} \| \mathcal{B} - (\mathcal{B}^{(s-1)} - \eta_2 \nabla g(\mathcal{B}^{(s-1)})\|^2_F + \lambda_2 \eta_2 p_2(\mathcal{B})$$

(12)

where $g(\mathcal{B}^{(s-1)}) = \| \mathcal{O} - \mathcal{B}^{(s-1)} - \mathcal{K} \otimes \mathcal{M}^{(s)}\|^2_F$. By substituting

$$\nabla g(\mathcal{B}^{(s-1)}) = \mathcal{K} \otimes \mathcal{M}^{(s)} + \mathcal{B}^{(s-1)} - \mathcal{O},$$

it is easy to deduce that the final updating rule for $\mathcal{B}$ is

$$\mathcal{B}^{(s)} = \text{prox}_{\lambda \eta_2}(1 - \eta_2)\mathcal{B}^{(s-1)} + \eta_2 (\mathcal{O} - \mathcal{K} \otimes \mathcal{M}^{(s)})$$

(13)

where $\text{prox}_{\lambda \eta_2}(\cdot)$ is the proximal operator correlated with the regularization term $p_2(\cdot)$ with respect to $\mathcal{B}$.

Based on this iterative algorithm, we can then construct our deep unfolding network as follows.

### IV. Rain Convolutional Dictionary Network

Inspired by the recent deep unfolding techniques in various tasks, e.g., deconvolution [80], compressed sensing [81], image super-resolution [82], CT metal artifact reduction [50], [83], [84], [85], low light enhancement [48], and pansharpening [86], we build a novel network structure for this single image deraining task by separating and transforming each iterative step of the aforementioned algorithm as a specific network module.
form of network connection. Its specificity is that all network modules correspond to the algorithm operators, and thus, the entire network has clear interpretability.

As shown in Fig. 4(a), the proposed network consists of $S$ stages, representing $S$ iterations of the algorithm for solving (7). Each stage achieves sequential updates of $M$ and $B$ by the M-net and the B-net, respectively. Specifically, as displayed in Fig. 4(b), in each stage of the network, the M-net takes the rainy image $O$ and the previous outputs $B^{(s-1)}$ and $M^{(s-1)}$ as inputs and outputs an updated $M^{(s)}$, and then, the B-net takes $O$ and $M^{(s)}$ as inputs and outputs an updated $B^{(s)}$.

From the updating rules (11) and (13), it is easily understood that the involved concise iterative computations can be naturally performed with commonly used operators in normal networks [34]. The key issue of unrolling the algorithm is how to represent the two proximal operators $\text{prox}_K$ and $\text{prox}_O$. In this work, we adopt the deep residual network (ResNet) [87] to construct the operator as many as possible. M-net takes the rainy image $B$ and $M^{(s-1)}$ at the $s$th stage of the entire network has clear interpretability.

8672 IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, VOL. 35, NO. 6, JUNE 2024

image $B$ from training data (i.e., the paired clean image and rainy image $B$), by transforming the operators in (14) and (15) step-by-step. All the parameters involved can be automatically fit from training data (i.e., the paired clean image $B$ and the rainy image $O$) in an end-to-end manner, including $\{\theta_m, \theta_b\}_{s=1}^S$, rain kernels $K$, $\eta_1$, and $\eta_2$. Consider that, in some scenarios, the composition of rainy images is complicated. Thus, we further refine the reconstructed result $B^{(s)}$ by feeding it into an extra ResNet, which has the same structure as CRCDNet (CRCDNet) whenever necessary. The details of DRCDNet are given as follows.

V. Dynamic RCDNet

As seen, the large random dictionary $K$ in RCDNet is shared among the entire dataset. Such settings would be more applicable for the consistent case that training and testing datasets are with similar rain patterns. To further enhance the generalization capability, we construct a DRCDNet. Specifically, in DRCDNet, the rain kernel $K$ is dynamically inferred for each rainy image. In this way, the number of the to-be-estimated rain map $M$ can be greatly reduced, and the hidden solution space for estimating the rain layer is also greatly shrunk, which naturally improves the generalization ability. For clarity, we also refer the RCDNet in Section IV as consistent RCDNet (CRCDNet) whenever necessary. The details of DRCDNet are given as follows.

A. Model Formulation

For DRCDNet, we reformulate the rain kernel $K_n$ in (5) as

$$\begin{align*}
K_n &= D \alpha_n
\end{align*}$$

where $D \in \mathbb{R}^{k \times k \times 3 \times d}$ is the rain kernel dictionary representing common knowledge for conveying variant rain types across the entire training set; $d$ is the number of rain kernels in this dictionary; and $\alpha_n \in \mathbb{R}^d$ denotes the weighting coefficient. Instead of pretraining and then fixing rain kernels $K_n$’s for any testing rainy image as CRCDNet does, DRCDNet can flexibly infer the rain kernels $K_n$’s for every rainy sample by dynamically updating $\alpha_n$’s. One can refer to Fig. 5 for easy understanding. This motivation is finely verified in Section VII-B.

More details about network design are described in the Supplementary Material.

$^7$More details about network design are described in the Supplementary Material.

$^8$Please refer to the Supplementary Material for more analysis.
Section III-B, we can easily derive the updating rules of

\[ S \]

\[ \text{ANet et al.: RCDNet: AN INTERPRETABLE RAIN CONVOLUTIONAL DICTIONARY NETWORK} \]

\[ \text{involvement of } \alpha \text{ for DRCDNet is prelearned like the kernel } K \text{ for RCDNet. However, the involvement of } \alpha \text{ helps the DRCDNet possess adaptive learning capability, with dynamically predicted rain kernels } K \text{ for different testing rainy images } O. \]

Then, the rainy image in (6) is rewritten as

\[ O = B + D \alpha \otimes M \]  \hspace{1cm} (17)

where \( \alpha \in \mathbb{R}^{d \times N} \) is stacked by \( a_n \)'s and \( D \alpha \in \mathbb{R}^{d \times k \times 3 \times N} \).

Similar to the dictionary \( K \) in RCDNet, the common dictionary \( D \) in this dynamic case can be automatically learned from training samples in an end-to-end manner by virtue of the strong nonlinear fitting ability of the deep network. Our goal is to estimate the unknown \( M, B, \) and \( \alpha \) from \( O \). Thus, the corresponding optimization problem is formulated as

\[ \begin{align*}
\min_{M, B, \alpha} & \quad \| O - B - D \alpha \otimes M \|_F^2 + \lambda_1 p_1(M) + \lambda_2 p_2(B) + \lambda_3 p_3(\alpha) \\
\text{s.t.} & \quad \| a_n \|_2 = 1, n = 1, 2, \ldots, N \quad (18)
\end{align*} \]

where the explicit constraint, i.e., \( \| a_n \|_2 = 1 \), is used to control the energy of weighting coefficient \( a_n \), so as to avoid affecting the learning of rain kernels. Similar to \( p_1(\cdot) \) and \( p_2(\cdot) \), we also prefer to automatically fit the regularizer \( p_3(\cdot) \) for \( \alpha \) via deep unrolling network modules.

**B. Optimization Algorithm**

With the similar algorithm for the problem (7) given in Section III-B, we can easily derive the updating rules of \( M \) and \( B \) for the problem (18) as

\[ M^{(s)} = \text{prox}_{s_1 \eta_1} \left( M^{(s-1)} - \eta_2 (D \alpha^{(s-1)} \otimes M^{(s-1)}) \right) \]

\[ B^{(s)} = \text{prox}_{s_2 \eta_2} \left( 1 - \eta_2 B^{(s-1)} + \eta_2 (O - D \alpha^{(s-1)} \otimes M^{(s)}) \right). \]

(19)

(20)

As for \( \alpha \), the quadratic approximation of the problem (18) with respect to \( \alpha \) is derived as

\[ \min_{\alpha \in \Omega} \frac{1}{2} \| \alpha - (\alpha^{(s-1)} - \eta_3 \nabla h(\alpha^{(s-1)})) \|_F^2 + \lambda_3 \eta_3 p_3(\alpha) \]

\[ (21) \]

where \( \Omega = \{ \alpha \| a_n \|_2 = 1, n = 1, 2, \ldots, N \}; h(\alpha^{(s-1)}) = \| O - B^{(s)} - D \alpha^{(s-1)} \otimes M^{(s)} \|_F^2 \). Then, we can derive that

\[ \frac{\partial h(\alpha^{(s-1)})}{\partial a_n} = U^T_d (D \otimes \eta D^{(s)}) \]

\[ \times \text{vec} \left( D \alpha^{(s-1)} \otimes M^{(s)} + B^{(s)} - O \right) \]

(22)

where the computed result of \( D \otimes \eta D^{(s)} \) has the size of \( H \times W \times 3 \times d; U^T_d (\cdot) \) represents unfolding the result at the fourth mode; and the resulted shape is \( d \times 3HW \).

It is clear that the updating rule for \( \alpha \) is, finally, derived as

\[ \alpha^{(s)} = \text{prox}_{s \eta} \left( \alpha^{(s-1)} - \eta \nabla h(\alpha^{(s-1)}) \right) \]

(23)

where \( \nabla h(\alpha^{(s-1)}) = \{ (\partial h(\alpha^{(s-1)})/\partial a_1), (\partial h(\alpha^{(s-1)})/\partial a_2), \ldots, (\partial h(\alpha^{(s-1)})/\partial a_N) \} \).

Such concise iterative rules (19), (20), and (23) facilitate us to unfold this iterative algorithm into a deep interpretable network as follows. Note that the constraint space \( \Omega \) can be easily achieved by embedding a normalization operation into the implicit proximal operator \( \text{prox}_{s \eta} \).

**C. Network Design**

Similar to Section IV, we subsequently decompose these updating rules (19), (20), and (23) into substeps and achieve

**Fig. 4.** (a) Proposed RCDNet with 5 stages. The network takes a rainy image \( O \) as input and outputs the learned rain map \( M \) and background image \( B \). (b) Illustration of the network architecture at the \( r \)th stage. Each stage consists of M-net and B-net to accomplish the update of rain map \( M \) and background layer \( B \), respectively. The images are better to be observed by zooming in on the screen. (a) Illustration of the entire RCDNet. (b) Design of a single stage.
the following procedures for the $s$th stage of the proposed DRCDNet:

\[
\begin{align*}
\text{M-net :} & \\
\hat{R}^{(s)} &= O - B^{(s-1)} \\
\overline{R}^{(s)} &= D_s^{(s-1)} \otimes \mathcal{M}^{(s-1)} \\
G^{(s)} &= \eta_1 D_s^{(s-1)} g^{(s)} \left( \hat{R}^{(s)} - \overline{R}^{(s)} \right) \\
\mathcal{M}^{(s)} &= \text{proxNet}_{\theta_s}^{(s)} \left( \mathcal{M}^{(s-1)} - G^{(s)} \right) \\
\end{align*}
\]

\[ (24) \]

\[
\begin{align*}
\text{B-net :} & \\
R^{(s)} &= D_s^{(s-1)} \otimes \mathcal{M}^{(s)} \\
\overline{B}^{(s)} &= O - R^{(s)} \\
B^{(s)} &= \text{proxNet}_{\theta_s}^{(s)} \left( (1 - \eta_2)B^{(s-1)} + \eta_2 \overline{B}^{(s)} \right) \\
\end{align*}
\]

\[ (25) \]

\[
\begin{align*}
\text{A-net :} & \\
\hat{R}^{(s)} &= O - B^{(s)} \\
\overline{R}^{(s)} &= D_s^{(s-1)} \otimes \mathcal{M}^{(s)} \\
G_a^{(s)} &= \eta_3 \left( U_4 \left( D \otimes d \mathcal{M}^{(s)} \right) \right) \vec{\alpha} \left( \overline{R}^{(s)} - \hat{R}^{(s)} \right) \\
\alpha^{(s)} &= \text{proxNet}_{\theta_s}^{(s)} \left( \alpha^{(s-1)} - G_a^{(s)} \right) \\
\end{align*}
\]

\[ (26) \]

where $G_a^{(s)} = \{G_{\theta_s}^{(s)}, G_{\rho_s}^{(s)}, \ldots, G_{\eta_s}^{(s)}\}$. The parameters in (24) and (25) have been explained in (14) and (15), respectively. For proxNet$_{\theta_s}^{(s)}$, it is a ResNet only consisting of one Resblock with the parameters $\theta_s^{(s)}$. Specifically, the Resblock simply contains two linear layers followed by a normalization operation at the second dimension of $\alpha$.\footnote{Please refer to the Supplementary Material for more details about DRCDNet.}

Then, by transforming the operators in (24)–(26) step-by-step, we can construct the DRCDNet. It is clear that, at each step, the DRCDNet is composed of three subnetworks, i.e., M-net, B-net, and A-net. Specifically, by comparing (14) and (15) with (24) and (25), respectively, we can directly construct the M-net and B-net by replacing the rain kernel dictionary $\mathcal{D}$, $\eta_1$, $\eta_2$, and $\eta_3$, can be automatically learned from training data (i.e., the paired clean background image $\mathcal{B}$ and the rainy image $\mathcal{O}$) in an end-to-end manner.

Remark 2: Similar to the CRCDNet, all the network modules in DRCDNet are correspondent to the iterative computations (19), (20), and (23), and thus, the DRCDNet also has clear interpretability. Compared with CRCDNet, DRCDNet has specific merits. First, at the testing phase, although the common rain kernel dictionary $\mathcal{D}$ is pretrained and fixed, the dynamic inference of $\alpha$ makes it possible to achieve the flexible prediction of rain kernel $\mathcal{D}a$ according to the rain types of variant testing rainy images. Besides, in CRCDNet, the rain kernels $\mathcal{K}$ are utilized to represent the entire dataset. Compared with the entire dataset that contains more rain types, depicting a specific rainy image should need fewer rain kernels $\mathcal{D}a$. Hence, we can choose a smaller $N$ for DRCDNet. Equivalently, the channel number of rain map $\mathcal{M}$ is also smaller than that in CRCDNet. Under this setting, the hidden space for estimating the rain layer is greatly shrunk, which naturally helps improve the generalization ability. This is comprehensively substantiated in Section VII-D.

Remark 3: Compared with the general channel attention mechanism, the $\alpha$-net has specific characteristics. First, instead of weighting feature maps on the channel dimension, we focus on weighting the rain kernel dictionary $\mathcal{D}$, which would save the computational cost. Second, the $\alpha$-net is built based on an optimization algorithm, and thus, it has clear physical interpretability. Third, as shown in Fig. 5, the obtained rain kernel $\mathcal{K}$ has obvious physical meanings, which validates the effectiveness of such weighting operators.

VI. NETWORK TRAINING

A. Training Loss

For simplicity, we adopt the mean square error (mse) [55] for the learned background and the rainy layer at every stage as the training objective function

\[
L = \sum_{s=0}^{S} \rho_s \left\| B^{(s)} - \mathcal{B}^{(s)} \right\|_F^2 + \sum_{s=1}^{S} \gamma_s \left\| \mathcal{O} - \mathcal{R}^{(s)} \right\|_F^2
\]

(27)

where $B^{(s)}$ and $R^{(s)}$ separately denote the denoised result and extracted rain layer at the $s$th stage ($s = 0, 1, \ldots, S$), as expressed in (15) for CRCDNet and (25) for DRCDNet. $B^{(0)}$ is initialized by a convolutional operator on $\mathcal{O}$. $\rho_s$ and $\gamma_s$ are tradeoff parameters and simply set as $\rho_S = \gamma_S = 1$ and others as 0.1 for all experiments to make the outputs at the final stage play a dominant role. More parameter settings are discussed in the Supplementary Material.

B. Implementation Details

We use PyTorch [34] to implement our method, and the network is trained based on an NVIDIA GeForce GTX 1080Ti GPU. For both CRCDNet and DRCDNet, we adopt the Adam optimizer [90] with the batch size of 10 and the patch size of $64 \times 64$. The initial learning rate is 0.001 and divided by 5 every 25 epochs. The total epoch number is 100. It is worth mentioning that we use these same parameter settings for all experiments. This would show the favorable robustness and generality of our method.

VII. EXPERIMENTAL RESULTS

We first conduct model verification to verify the working mechanisms of the proposed network. Then, we evaluate the superiority of CRCDNet by comparing it with other SOTA
single image derainers based on synthetic datasets. Finally, the performance of DRCDNet is verified by generalization experiments where rain patterns are obviously different between training samples and testing ones.

A. Details’ Explanations

1) Benchmark Datasets: Eight datasets are adopted, as listed in Table I, including five synthesized ones and three real ones. Similar to other supervised methods, during the training process, what we explicitly need are the paired clean image $B$ and the rain-affected image $O$.\(^{10}\)

2) Comparison Methods: We compare our network with current SOTA single image derainers, including the following.\(^{11}\)

Prior-Based Methods: DSC \[6\] and JCAS \[7\].
DL Methods: Clear \[12\], DDN \[13\], RESCAN \[18\], PReNet \[19\], SPAnet \[26\], and JORDER_E \[20\].
Semisupervised Method: SIRR \[73\].

3) Performance Metrics: For paired data, the classical metrics are PSNR \[91\] and SSIM \[92\]. Since the human visual system is sensitive to the luminance channel ($Y$) channel of a color image in the YCbCr space, similar to \[16\], \[19\], and \[60\], we also compute PSNR and SSIM based on the luminance channel, while, for unlabel data, we adopt the nonreference indicators, i.e., naturalness image quality evaluator (NIQE) \[93\] and blind/referenceless image spatial quality evaluator (BRISQUE) \[94\]. Specifically, PSNR and SSIM generally measure the intensity fidelity and structure quality, respectively. NIQE and BRISQUE aim to quantify the quality of a distorted image in a way that matches the human judgments of visual quality as closely as possible. Higher PSNR and SSIM, as well as lower NIQE and BRISQUE, indicate the better result.

B. Model Verification

Here, we utilize Rain100L to execute the model verification.

1) Stage Number $S$: Table II reports the effect of stage number $S$ on the deraining performance of the proposed CRCDNet. Here, $S = 0$ represents the fact that, without adopting the RCD mechanism, the initialization $B^{(0)}$ is directly regarded as the final rain-removed result. Taking $S = 0$ as a baseline, it is easily seen that, with only two stages, our method already achieves significant rain removal performance improvement, substantiating the essential role of the constructed M-net and B-net. We also find that, when $S = 20$, there are no further obvious performance gains since larger $S$ would make the gradient propagation more difficult. Based on this observation, we set $S$ as 17 for the CRCDNet throughout all our experiments. More discussions are listed in the Supplementary Material.

2) Network Visualization: We then visually show how the interpretability of CRCDNet facilitates an easy analysis of the working mechanism inside the network modules. Fig. 7 presents the extracted background $B^{(s)}$, $\hat{B}^{(s)}$ of CRCDNet and the rain layer $R^{(s)}$ at different stages. The total stage number $S$ is 17. PSNR/SSIM is listed below the corresponding results for easy reference.

![Fig. 7. Visualization of the recovery background $B^{(s)}$, $\hat{B}^{(s)}$ of CRCDNet and the rain layer $R^{(s)}$ at different stages. The total stage number $S$ is 17. PSNR/SSIM is listed below the corresponding results for easy reference.](image)

![Fig. 8. At the final stage $s = 17$, the extracted rain layer, rain kernels $K_{a,s}$, and rain maps $M_{a}$ for the input $O$ in Fig. 7. The lower left is the rain kernels $K$ learned by CRCDNet based on Rain100L training pairs. In CRCDNet, $N = 32$.](image)

3) RCD Model Visualization: For the input $O$ in Fig. 7, the rain kernels and the rain maps learned by CRCDNet are presented in Fig. 8. It is clear that the CRCDNet finely extracts proper rain layers explicitly complying with the RCD model (5). This not only verifies the reasonability of our method but also manifests the peculiarity of our proposal.
which validates the reliability of embedding the RCD prior.

DRCDNet both contain fewer unexpected background details, easily observe that the rain layers extracted by CRCDNet and streaks and finely preserving image details. DRCDNet perform better in sufficiently removing the rain leaving obvious rain streaks, and deep derainers lose certain ing methods on a rainy image from Rain100L. As shown, for consistent with that of training data, based on the benchmark DRCDNet in the case that the rain types of testing data are

C. Training-Test Domain Match Experiments

In this section, we evaluate the proposed CRCDNet and DRCDNet in the case that the rain types of testing data are consistent with that of training data, based on the benchmark datasets, including Rain100L, Rain100H, and Rain1400.

Fig. 10 illustrates the deraining performance of all competing methods on a rainy image from Rain100L. As shown, for background recovery, traditional model-based DSC and JCAS leave obvious rain streaks, and deep derainers lose certain useful image textures. However, the proposed CRCDNet and DRCDNet perform better in sufficiently removing the rain streaks and finely preserving image details. Moreover, we can easily observe that the rain layers extracted by CRCDNet and DRCDNet both contain fewer unexpected background details, which validates the reliability of embedding the RCD prior constraints into the network design.\footnote{More visual experimental results on Rain100H and Rain1400 are provided in the Supplementary Material.}

Table III reports the average PSNR and SSIM computed on the entire testing data of each synthesized dataset. It is seen that, in this training–testing domain match case, our CRCDNet attains significant deraining performance on each evaluation dataset, and DRCDNet performs comparable to CRCDNet.

D. Training-Test Domain Mismatch Experiments

Here, we evaluate the CRCDNet and DRCDNet in the case that rain types are inconsistent between training and testing.

1) Performance Comparison on Dense10 and Sparse10: We first adopt Dense10 and Sparse10 to evaluate the generalization capability of all DL competing methods trained on Rain100H. From the quantitative results in Table IV, we can find that CRCDNet is still competing with the SOTA, and DRCDNet even obtains higher PSNR and SSIM. This tells us that the proper embedding of prior constraints is helpful to alleviate the overfitting issue, and the dynamic inference model would make the space for rain layer estimation tighter and then help further improve the generalization performance.\footnote{More visual experimental results are provided in the Supplementary Material.}

2) Performance Comparison on SPA-Data: This real dataset is composed of complicated rain patterns, diverse shooting scenes, and rich background details. All these factors bring great challenges to accurate rain layer extraction and the model
generalization performance on the dataset. Fig. 11 displays the reconstructed images where deep methods are trained on Rain100L. It is clear that the proposed DRCDNet performs better on both rain removal and detail preservation.

Table V provides the quantitative comparisons under different testing scenarios. As for the generalization case from Rain100L to SPA-Data, although DRCDNet achieves the higher PSNR and SSIM than CRCDNet, due to the simplicity of rain types in Rain100L and the complexness of rainy samples in SPA-Data, the generalization performance of DRCDNet is not prominent. However, by utilizing Rain100L and Rain1400 with 14 rain types as training data, the generalization performance of DRCDNet is largely improved.

3) Performance Comparison on Internet-Data: Table VI listed the quantitative comparisons on the real Internet-Data where all the DL-based deraining models are trained on Rain100H and both Rain100L and Rain100H, respectively. As seen, our DRCDNet consistently achieves the lowest BRISQUE and NIQE, showing better generalization performance.14

E. Downstream Tasks

In this section, to comprehensively substantiate the effectiveness of our proposed method in rain-removed image restoration, we conduct a series of experiments on the downstream tasks, including object detection and semantic segmentation to investigate the potential of different deraining methods in helping improve the high-level vision performance.

1) Object Detection: For the objection detection task, we select the widely adopted benchmark COCO val2017 [95] which consists of 5000 images with bounding box annotations. Three popular detection algorithms are adopted, including Faster RCNN [97], Mask RCNN [98], and YOLOv3 [99]. Following [12] and [20], we synthesize the corresponding 5000 rainy images (called synthesized COCO val2017) by using Photoshop,15 which contains various rain types with different directions, scales, and magnitudes. To execute the downstream task, we first utilize all the comparing deraining methods to restore the rain-removed results of the synthesized COCO val2017. Then, we adopt the publicly available pretrained models [100] of Faster RCNN, Mask RCNN, and YOLOv3 to perform the object detection task on these restored images. For quantitative comparison, we evaluate the mean Average Precision (mAP) averaged for IOU ∈ [0.5 : 0.05 : 0.95] (COCO’s standard metric, simply denoted as mAP@[.5, .95]).

Fig. 12 presents the objection detection results of Faster RCNN on the rain-removed images obtained by different deraining methods. The “Upper Bound” represents the detection result of Faster RCNN on the corresponding clean rain-free image. All the DL-based deraining methods are trained on Rain100L. It is obvious that rain streaks adversely degrade the detection accuracy and lead to the fake detection of the target. In addition, due to the corruption of rain streaks, all the methods cannot detect the tennis racket. However, compared with other deraining baselines, our proposed CRCDNet and DRCDNet achieve better visual effects in rain removal and detail preservation, which finely boosts the detection results.

Table VII lists the average quantitative results on the synthesized COCO val2017 of the three detection methods, i.e., Faster RCNN, Mask RCNN, and YOLOv3. As seen, the pro-

14More experiments are provided in the Supplementary Material.
15https://www.photoshopessentials.com/photo-effects/rain/
posed CRCDNet and DRCDNet consistently help these three different detection algorithms obtain higher detection accuracy and achieve about 8% improvement for mAP@[.5, .95] over the original input. This result substantiates that our proposed method indeed has the capability to accomplish the better restoration of rain-removed images and then help improve the performance of the downstream task in rainy weather conditions, which should be meaningful for practical applications. Besides, we can find that DRCDNet outperforms CRCDNet, showing the effectiveness of the proposed dynamic rain kernel inference mechanism.

2) Semantic Segmentation: For the semantic segmentation task, we adopt the Cityscapes validation set [96] as the benchmark, including 500 images with pixel-level annotations. Similar to the synthesized COCO val2017, we synthesize the rainy version of the Cityscapes validation set with Photoshop. FCN [101], PSPNet [102], and DeepLabv3+ [103] are utilized as the segmentation methods. The commonly used Mean Intersection over Union (MIOU) is taken as the performance metric for quantitative evaluation.

Based on an image selected from the synthesized Cityscapes validation set, Fig. 13 shows the visual segmentation results of PSPNet on the rain-removed images obtained by different rain-removal methods. It is clearly observed that the existence of rain streaks severely corrupts the image details, leading to a bad segmentation result. Attributed to the stronger deraining capability and the better generalization potential, our proposed CRCDNet and DRCDNet restore more credible contents with more details, which effectively promotes the semantic

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Fig. 12. Object detection results of Faster RCNN on the synthesized COCO val2017 dataset. All the DL-based deraining methods are trained on Rain100L.

Fig. 13. Visual results of joint deraining and semantic segmentation on the synthesized Cityscapes validation dataset. All the DL-based deraining methods are trained on Rain100L. For each method, the first row is the generalized deraining result, and the second row is the corresponding segmentation result with the PSPNet segmentation network.

Table VII

| Methods    | Input  | DSC | JCAS | Clear | DDN | RESCAN | PReNet | SPANet | JORDER_E | SIRR | CRCDNet | DRCNet | Upper Bound |
|------------|--------|-----|------|-------|-----|--------|--------|--------|----------|------|----------|--------|-------------|
| Faster RNN | 30.7   | 31.3| 32.0 | 34.7  | 35.1| 37.2   | 37.6   | 36.7   | 37.6     | 35.2 | 38.1     | 38.3   | 42.1        |
| Mask RNN   | 31.7   | 32.5| 33.4 | 36.1  | 36.8| 39.2   | 36.9   | 39.1   | 36.9     | 39.7 | 39.9     | 44.5   |             |
| YOLOv3     | 14.4   | 15.6| 16.3 | 18.0  | 18.5| 19.8   | 20.6   | 19.9   | 20.3     | 18.4 | 20.8     | 21.0   | 22.2        |

Table VIII

| Methods    | Input  | DSC | JCAS | Clear | DDN | RESCAN | PReNet | SPANet | JORDER_E | SIRR | CRCDNet | DRCNet | Upper Bound |
|------------|--------|-----|------|-------|-----|--------|--------|--------|----------|------|----------|--------|-------------|
| FCN        | 33.6   | 34.8| 37.1 | 46.6  | 52.7| 66.0   | 66.8   | 66.0   | 67.0     | 51.0 | 67.0     | 68.5   | 73.4        |
| PSPNet     | 33.6   | 36.2| 38.6 | 48.1  | 54.1| 69.8   | 70.3   | 68.1   | 70.4     | 70.9 | 72.6     | 72.6   | 78.5        |
| DeepLabv3+ | 34.0   | 38.8| 42.2 | 53.1  | 58.1| 71.8   | 72.1   | 71.7   | 72.3     | 55.9 | 72.6     | 74.2   | 79.8        |

F. Network Parameters and Inference Time

Table IX presents the comparisons including network parameters and average inference time on an NVIDIA GeForce GTX1080Ti GPU. This shows that the proposed CRCDNet and DRCNet are comparable to other competing methods.

Table IX

| Methods | Clear  | DDN   | RESCAN | PReNet | SPANet | Parameter # | Time (Seconds) | Methods | SIRR | CRCDNet | DRCNet | Time (Seconds) |
|---------|--------|-------|--------|--------|--------|-------------|----------------|---------|------|--------|--------|----------------|
| JORDER_E | 57,369 | 149,823 | 168,963 | 283,716 |       | 4,169,024 | 2,835,546 | 2,231,406 |       | 3.12 | 3.65 | 0.84 | 0.71 |
This makes the network have easily visualized interpretation for all its module elements and, thus, facilitates its easy analysis of what happens in the network. Furthermore, considering that the rain patterns of training data are inconsistent with testing data in most realistic scenarios, we have carefully designed a dynamic rain kernel inference mechanism and correspondingly built an interpretable DRCDNet, which can dynamically infer the corresponding rain kernels complying with diverse rain types of testing rainy images. This helps shrink the space for estimating rain layers and makes the network capable of being finely generalized to testing data even with rain patterns different from training data. All these superiorities have been comprehensively substantiated by a series of experiments, including model verification, network visualization, rain kernel visualization, and training/test domain match/mismatch evaluations. Besides, the extracted elements through the end-to-end learning by the network, such as the diverse rain kernels, are potentially useful for other related tasks on rainy images.

If the degraded images captured in rainy weather also contain nonstreaking rain types, such as mist which is often caused by heavy rains, our proposed RCD prior may not be able to finely represent this degradation form. How to finely execute the joint removal of rain streaks, rain mist, and raindrops is a more challenging-yet-meaningful practical problem and deserves further exploration in the future.

REFERENCES

[1] M. S. Shehata et al., “Video-based automatic incident detection for smart roads: The outdoor environmental challenges regarding false alarms,” IEEE Trans. Intell. Transp. Syst., vol. 9, no. 2, pp. 349–360, Jun. 2008.
[2] H. Wang, Y. Wu, M. Li, Q. Zhao, and D. Meng, “Survey on rain removal from videos or a single image,” Sci. China Inf. Sci., vol. 65, no. 1, pp. 1–23, Jan. 2022.
[3] H. Wang, Q. Xie, Y. Wu, Q. Zhao, and D. Meng, “Single image rain streaks removal: A review and an exploration,” Int. J. Mach. Learn. Cybern., vol. 11, no. 4, pp. 853–872, Apr. 2020.
[4] S. Li et al., “Single image deraining: A comprehensive benchmark analysis,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 3838–3847.
[5] L.-W. Kang, C.-W. Lin, and Y.-H. Fu, “Automatic single-image-based rain streaks removal via image decomposition,” IEEE Trans. Image Process., vol. 21, no. 4, pp. 1742–1755, Apr. 2012.
[6] Y. Luo, Y. Xu, and H. Ji, “Removing rain from a single image via discriminative sparse coding,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Dec. 2015, pp. 3397–3405.
[7] S. Gu, D. Meng, W. Zuo, and L. Zhang, “Joint convolutional analysis and synthesis sparse representation for single image layer separation,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 1708–1716.
[8] L. Zhu, C.-W. Fu, D. Lischinski, and P.-A. Heng, “Joint bi-layer optimization for single-image rain streak removal,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 2526–2534.
[9] H. Zhang and V. M. Patel, “Convolutional sparse and low-rank coding-based rain streak removal,” in Proc. IEEE Winter Appl. Comput. Vis. (WACV), Mar. 2017, pp. 1259–1267.
[10] F. Huang and A. Anandkumar, “Convolutional dictionary learning through tensor factorization,” in Proc. Feature Extraction, Mod. Questions Conf. (FEIQ), vol. 44, no. 15, 2015, pp. 116–129.
[11] M. Li et al., “Video rain streak removal by multiscale convolutional sparse coding,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 6644–6653.
[12] X. Fu, J. Huang, X. Ding, Y. Liao, and J. Paisley, “Clearing the skies: A deep network architecture for single-image rain removal,” IEEE Trans. Image Process., vol. 26, no. 6, pp. 2944–2956, Jun. 2017.
[13] X. Fu, J. Huang, D. Zeng, Y. Huang, X. Ding, and J. Paisley, “Removing rain from single images via a deep detail network,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 3855–3863.
[88] D. Yang and J. Sun, “Proximal Dehaze-Net: A prior learning-based deep network for single image dehazing,” in Proc. Eur. Conf. Comput. Vis., 2018, pp. 702–717.

[89] Q. Xie, M. Zhou, Q. Zhao, D. Meng, W. Zuo, and Z. Xu, “Multispectral and hyperspectral image fusion by MS/HS fusion net,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 1585–1594.

[90] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014, arXiv:1412.6980.

[91] Q. Huynh-Thu and M. Ghanbari, “Scope of validity of PSNR in image/video quality assessment,” Electron. Lett., vol. 44, no. 13, pp. 800–801, Jun. 2008.

[92] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” IEEE Trans. Image Process., vol. 13, no. 4, pp. 600–612, Apr. 2004.

[93] A. Mittal, R. Soundararajan, and A. C. Bovik, “Making a completely blind image quality analyzer,” IEEE Signal Process. Lett., vol. 20, no. 3, pp. 209–212, Oct. 2012.

[94] A. Mittal, A. K. Moorthy, and A. C. Bovik, “No-reference image quality assessment in the spatial domain,” IEEE Trans. Image Process., vol. 21, no. 12, pp. 4695–4708, Dec. 2012.

[95] T.-Y. Lin et al., “Microsoft COCO: Common objects in context,” in Proc. Eur. Conf. Comput. Vis., 2014, pp. 740–755.

[96] M. Cordts et al., “The cityscapes dataset for semantic urban scene understanding,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 3213–3223.

[97] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in Proc. Adv. Neural Inf. Process. Syst., vol. 28, 2015, pp. 1–9.

[98] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” in Proc. IEEE Int. Conf. Comput. Vis., Oct. 2017, pp. 2961–2969.

[99] J. Redmon and A. Farhadi, “YOLOv3: An incremental improvement,” 2018, arXiv:1804.02767.

[100] M Contributors. (2019). MM Detection: OpenMMLab Detection Toolbox and Benchmark. [Online]. Available: https://github.com/open-mmlab/mmdetection.

[101] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 3431–3440.

[102] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, “Pyramid scene parsing network,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 2881–2890.

[103] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, “Encoder–decoder with atrous separable convolution for semantic image segmentation,” in Proc. Eur. Conf. Comput. Vis. (ECCV), 2018, pp. 801–818.

[104] M Contributors. (2020). MM Segmentation: OpenMMLab Semantic Segmentation Toolbox and Benchmark. [Online]. Available: https://github.com/open-mmlab/mms segmentation.

Hong Wang received the Ph.D. degree from the School of Mathematics and Statistics, Xi’an Jiaotong University, Xi’an, China, in 2021, under the supervision of Prof. Deyu Meng.

She was a Research Intern with the Tencent Rhino Bird Elite Talent Program from 2020 to 2021. She is currently a Senior Researcher with Tencent, Shenzhen, China. Her current research interests focus on the design of model- and data-driven deep learning techniques for effective and interpretable image processing, including natural image restoration and medical image analysis.

Qi Xie received the B.Sc. and Ph.D. degrees from Xi’an Jiaotong University, Xi’an, China, in 2013 and 2020, respectively.

He was a Visiting Scholar with Princeton University, Princeton, NJ, USA, from 2018 to 2019. He is currently an Assistant Professor with the School of Mathematics and Statistics, Xi’an Jiaotong University. His current research interests include low-rank matrix factorization, tensor recovery, and sparse machine learning methods.

Yuexiang Li received the B.Eng. degree in telecommunications engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 2011, the M.S. degree in electronic engineering from The Hong Kong University of Science and Technology, Hong Kong, in 2012, and the Ph.D. degree in electronic engineering from the University of Nottingham, Nottingham, U.K., in 2016.

He joined the Tencent Jarvis Lab, Shenzhen, China, as a Senior Researcher, in 2018. He has published multiple academic papers in top-tier journals and conferences.

Yong Liang received the B.Sc. and M.Sc. degrees in applied mathematics from Xian Jiaotong University, Xian, China, in 1996 and 1999, respectively, and the Ph.D. degree in computer science from The Chinese University of Hong Kong, Hong Kong, in 2003.

From 2004 to 2006, he was a Post-Doctoral Researcher with the Department of Computer Science and Engineering, The Chinese University of Hong Kong. He later joined the Macau University of Science and Technology, Macau. He joined the Peng Cheng Laboratory, Shenzhen, China, in 2022, where he is currently a Professor. He has authored or coauthored more than 120 papers. His research interests are in computational intelligence, including machine learning, big data analysis, evolutionary computation, and bioinformatics.

Yefeng Zheng (Fellow, IEEE) received the B.E. and M.E. degrees from Tsinghua University, Beijing, China, in 1998 and 2001, respectively, and the Ph.D. degree from the University of Maryland, College Park, MD, USA, in 2005.

After graduation, he worked at Siemens Corporate Research, Princeton, NJ, USA, on medical image analysis before joining Tencent, Shenzhen, China, in 2018. He is currently a Distinguished Scientist and the Director of the Tencent Jarvis Lab, Shenzhen, leading the company’s initiative on medical AI.

He has published more than 250 papers and invented more than 70 U.S. patents. His work has been cited more than 10,000 times with an H-index of 57.

Dr. Zheng is a fellow of the American Institute for Medical and Biological Engineering (AIMBE) and an Associate Editor of IEEE TRANSACTIONS ON MEDICAL IMAGING.

Deyu Meng (Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees from Xi’an Jiaotong University, Xi’an, China, in 2001, 2004, and 2008, respectively.

From 2012 to 2014, he took his two-year sabbatical leave at Carnegie Mellon University, Pittsburgh, PA, USA. He is currently a Professor with the School of Mathematics and Statistics, Xi’an Jiaotong University, and an Adjunct Professor with the Faculty of Information Technology, Macau University of Science and Technology, Macau. His current research interests include model-based deep learning, variational networks, and metalearning.