International Journal of Computer Vision (2021) 129:1691–1711
https://doi.org/10.1007/s11263-020-01428-6

Successive Graph Convolutional Network for Image De-raining

Xueyang Fu[1] · Qi Qi[2] · Zheng-Jun Zha[1] · Xinghao Ding[2] · Feng Wu[1] · John Paisley[3]

Received: 9 December 2019 / Accepted: 31 December 2020 / Published online: 2 March 2021

Abstract
Deep convolutional neural networks (CNNs) have shown their advantages in the single image de-raining task. However, most existing CNNs-based methods utilize only local spatial information without considering long-range contextual information. In this paper, we propose a graph convolutional networks (GCNs)-based model to solve the above problem. We specifically design two graphs to extract representations from new dimensions. The first graph models the global spatial relationship between pixels in the feature, while the second graph models the interrelationship across the channels. By integrating conventional CNNs and our GCNs into a single framework, the proposed method is able to explore comprehensive feature representations from three aspects, i.e., local spatial patterns, global spatial coherence and channel correlation. To better exploit the explored rich feature representations, we further introduce a simple yet effective recurrent operations to perform the de-raining process in a successive manner. Benefiting from the rich information exploration and exploitation, our method achieves state-of-the-art results on both synthetic and real-world data sets.

Keywords Image de-raining · Graph convolutional networks · Deep learning · Image processing

1 Introduction

Images captured on rainy days are often contaminated by rain streaks, which can blur and block background objects. This phenomenon significantly degrades the visual quality and negatively affects outdoor vision systems, such as detection (Mordan et al. 2019; Liu et al. 2019), recognition (Zhang and Patel 2016), tracking (Zhang et al. 2013), segmentation (Wojna et al. 2019), scene understanding (Sakaridis et al. 2018; Chen et al. 2017) and person re-identification (Wang et al. 2018; Li et al. 2018). Since single image de-raining can potentially improve both subjective perception quality and objective vision system performance, it has been attracted more and more attention in the computer vision community (Narasimhan and Nayar 2002; Li et al. 2019c; Wang et al. 2020; Li et al. 2019b; Yang et al. 2020).

Communicated by Vishal Patel.

✉ Zheng-Jun Zha
zhazj@ustc.edu.cn

[1] School of Information Science and Technology, University of Science and Technology of China, Hefei, China

[2] School of Informatics, Xiamen University, Xiamen, China

[3] Department of Electrical Engineering and Data Science Institute, Columbia University, New York,, USA

In the past several years, various approaches have been introduced to remove rain streaks form both videos and single images. For both issues, existing methods either manually design algorithms to solve it in a model-driven fashion, or automatically learn de-raining functions based on the data-driven deep learning methodology. Since there is no temporal information for rain streaks detection, single image de-raining is obviously more challenging than video. Moreover, the success in single image de-raining can be directly extended to video. Therefore, in this paper, we focus on removing rain streaks from single images.

Due to the powerful nonlinear modeling capacity, deep learning-based methods have recently dominated the field of single image de-raining. Current state-of-the-art methods are all adopting convolutional neural networks (CNNs) as backbones. However, conventional CNNs only capture local spatial information and thus ignore long-range relationships between image pixels. Since rain streaks and object structures are long in space and similar in geometry, several methods (Li et al. 2018b; Yang et al. 2019) adopt the dilated convolution (Yu and Koltun 2016) to obtain larger receptive fields without increasing parameter burdens. However, since the convolution operation is essentially a process of weighted summation, what we get from dilated convolution is still local spatial information, i.e., obtaining one pixel value from one

finite spatial image area. Recently, a method based on a self-attention mechanism (Li et al. 2018) is proposed to extract spatial neighbor information for single image de-raining. However, this method requires a large amount of memory consumptions due to the non-local operation (Wang et al. 2018), which limits its practical values. Another direction is to combine CNNs and recurrent neural networks (RNNs) (Li et al. 2018b; Ren et al. 2019). Due to the repeated features utilization, the de-raining performance can be boosted. However, these methods focus on propagating spatial information without considering the correlation between channels. Therefore, for the specific image de-raining task, the aspect of fully exploring global spatial coherence and channel correlation has not been noticed.

To address the above limitations, this paper utilizes graph convolutional networks (GCNs) (Kipf and Welling 2017) to extract and complement new contextual information for the single image de-raining. Specifically, we first design a dilated convolution fusion block to extract multi-scale local spatial patterns. Then, we propose two lightweight graph convolutional networks to explore global spatial coherence and channel correlation of features from the dilated block. The dilated block and graph convolutional networks comprise the basic unit of our network, which is able to extract both local spatial and long-range contextual information. To fully utilize extracted features, we perform the de-raining process in a successive manner by introducing simple recurrent operations. Therefore, our model is able to explore and exploit multi-dimension features representations for the tough single image de-raining problem.

In summary, our contributions are fourfold:

- We propose a unified deep model based on graph convolution networks for single image de-raining. Our model integrates the advantages of CNNs, GCNs and RNNs and applies them to this specific task.
- We design two lightweight graph convolutional networks to model global spatial coherence and channel correlation. By combining dilated CNNs and proposed GCNs, our model is able to explore rich representations of both local spatial patterns and global contextual information.
- We further introduce a simple yet effective recurrent operations to perform the de-raining in a successive manner. The extracted rich representations in current stages can be well exploited to help subsequent stages for boosting de-raining performance.
- Our network has the advantages of easy implementation and efficient calculation. Experiments demonstrate that our model favorably performs against the state-of-the-art methods on both synthetic and real-world datasets.

2 Related Work

2.1 Video-Based Methods

We first briefly review the rain removal from video frames, in which both spatial and temporal information can be utilized. The first study on video de-raining is proposed by Garg and Nayar (2004), in which rain streaks are removed from a static background using average intensities from the neighboring frames. The authors (Garg and Nayar 2005, 2007) then reduce rain effect by increasing the exposure time or reducing the depth of camera field. Santhaseelan and Asari (2015) minimize registration error between frames and use phase congruency to detect and remove the rain streaks. Other methods focus on de-raining in the Fourier domain (Barnum et al. 2010), using Gaussian mixture models (GMMs) (Bossu et al. 2011), low-rank approximations (Chen et al. 2013), and via matrix completions (Kim et al. 2015). Ren et al. (2017) divide rain streaks into sparse ones and dense ones, then a matrix decomposition based algorithm is proposed for de-raining. Based on motion segmentation of dynamic scene, (Chen and Chau 2013) use photometric and chromatic constraints for rain detection and removal. Kim et al. (2015) adopt support vector machine to decompose rain streaks and outliers, and use low rank matrix completion to achieve rain removal. By exploring the intrinsic characteristics of clean frames and rain streaks, (Jiang et al. 2017, 2018) introduce a novel tensor-based method for video rain removal. An alternation direction method of multipliers is adopted to solve the objective function. Recently, a novel patch-based mixture of Gaussians (Wei et al. 2017) is firstly proposed for video de-raining removal. The authors proposed a concise P-MoG model by integrating the spatio-temporal smoothness of moving objects and low rank background structures. Li et al. (2018a) propose a multi-scale convolutional sparse coding model (Zhang and Patel 2017) to model the characteristics of video rain streaks. Similar to Wei et al. (2017), both ℓ_1 and TV regularization are utilized to solve the model.

Very recently, deep convolutional neural networks have also been explored for the video de-raining. To deal with highly dynamic scenes, (Chen et al. 2018) utilize deep CNNs and super-pixel for content alignment and rain removal. Liu et al. (2018) design a hybrid rain model and utilize recurrent neural networks to explore temporal redundant information. This method is able to achieve rain degradation classification, rain steaks removal and object details reconstruction. Later, the authors further propose a dynamic routing recurrent network (Liu et al. 2018) with residue learning for video de-raining.

2.2 Single-Image Methods

Different with the video de-raining that can utilize temporal information, single image de-raining is more ill-posed and challenging. Therefore, the research on single image de-raining has drawn more and more attention. In general, single image de-raining methods can be categorized into model-driven and data-driven methods.

Model-driven methods are manually designed by utilizing physical characteristics of rain streaks, or exploring prior knowledge to constrain the ill-posed problem. For example, (Kim et al. 2013), apply a nonlocal mean smoothing filter on the rainy image to achieve image de-raining. Zhang et al. (2013) modify the guided filtering (He et al. 2013) to obtain the de-rained result in an iterative manner. Several model-driven methods adopt various priors to separate rain streaks and content form rainy images. Kang et al. (2012) adopt morphological component analysis and utilize histogram of oriented gradients features to cluster learned dictionary, and then remove rain streaks in high frequency regions. To recognize rain streaks, (Huang et al. 2014) propose a self-learning based image decomposition method. By coding over a learned dictionary with mutual exclusivity, (Luo et al. 2015) propose a discriminative sparse coding to distinguish rain streaks from non-rain content. Chang et al. (2017) employ the low-rank assumptions to model and separate rain streaks. Li et al. (2016) exploits a GMM based patch prior to isolate rain streaks and preserve background details. Wang et al. (2017) introduce a hierarchical scheme combined with guided filtering and dictionary learning to progressively remove rain and snow. To efficiently separate background and rain streaks, (Zhang and Patel 2017) proposed a method to learn generic sparsity-based and low-rank representation-based convolutional filters. Then, an optimization problem is introduced to obtain de-rained results by using the learned filters. Gu et al. (2017) utilize analysis sparse representation and synthesis sparse representation to model large-scale structures and fine-scale textures, respectively. A joint convolutional analysis and synthesis sparse representation model is proposed to extract and remove rain streaks. Zhu et al. (2017) explore three priors, i.e., local and nonlocal sparsity, gradients derivation and visual similarity, for separating rain streaks from background. The authors then integrate these priors into a joint bi-layer optimization process for image rain removal.

In recent years, deep learning has achieved significant success in several image processing tasks, such as image de-noising (Zhang et al. 2017), super-resolution (Dong et al. 2016; Tai et al. 2017), de-blurring (Li et al. 2019), de-hazing (Ren et al. 2019; Zhang and Patel 2018a), image generation (Zhang et al. 2019), raindrop removal (Eigen et al. 2013; Qian et al. 2018), etc. With the powerful nonlinear modeling capacity, deep learning is also utilized to solve the tough image de-raining problem. For example, (Fu et al. 2017) additionally utilize domain knowledge and train a 3-layer CNNs on high frequency parts to simplify the learning processing. This method is improved in Fu et al. (2017) by combining ResNet (He et al. 2016) to reduce the mapping range for easing the learning process. Other deep CNNs-based methods focus on designing advanced network structure to improve de-raining performance. Based on the generative adversarial networks (GANs) (Goodfellow et al. 2014; Zhang et al. 2019) integrate quantitative, visual, and discriminative performance into loss function for image de-raining. To adaptively utilize the rain-density information, the authors (Zhang and Patel 2018b) further presented a multi-stream dense network to perform a density-aware image de-raining process. Yang et al. (2017) combine a RNN and dilated convolutions (Yu and Koltun 2016) to build a multi-task network structure, which is able to learn rain streaks appearance, rain streaks location and de-rained image. Later, a detail preserving process (Yang et al. 2019) is added into the above model for a better de-raining performance. To mitigate the problem of large rain streaks and rain streak accumulation, the wavelet transform is embedded into a recurrent learning framework (Yang et al. 2019) to exploit hierarchical features for challenging heavy rainy scenes. Li et al. (2018b) also utilize recurrent neural networks and combine it with squeeze-and-excitation (SE) blocks (He et al. 2018) for rain removal. The SE block is designed to automatically assign weights to various rain streak layers. To capture abstract features for accurate rain streaks estimation, (Li et al. 2018) proposed a non-locally enhanced encoder-decoder network based on the non-local neural networks (Wang et al. 2018). Fu et al. (2019) adopt the classical image pyramid decomposition to handle the image de-raining in a divide-and-conquer manner. To handle heavy rainy images with hazy effect, the authors of Li et al. (2019a) design a deep network based on the physical model and insert auxiliary losses to train it. Similar with (Li et al. 2019a), the authors of Hu et al. (2019) formulate the rain imaging process based on scene depth, and then introduce a depth-guided attention mechanism to remove heavy rain streaks. To take the location information of rain drops into consideration, uncertainty guided multi-scale residual learning network is proposed in Yasarla and Patel (2019, 2020) to learn the rain content at different scales. By repeatedly unfolding a shallow ResNet, (Ren et al. 2019) propose a simple yet effective de-raining network with progressive recurrent operations. To handle the problem of lacking real-world training pairs, (Wang et al. 2019) semi-automatically built a large-scale real-world data set that covers various natural rain scenes. The authors further propose a spatial attentive network to remove rain streaks in a local-to-global fashion. To make the trained network better generalize to real-world scenarios, (Wei et al. 2019) introduce semi-supervised learning paradigm for unseen rain types. The authors adopt ResNet

as the supervised part, and feed real-world rainy images for unsupervised regularization with Gaussian Mixture Model. In this semi-supervised manner, the method alleviates the issues of generalization ability and over-fitting. Wang et al. (2019) propose an entangled representation learning formulation to solve this intrinsic de-raining problem. Recently, to effectively aggregate the contextual information, (Wang et al. 2020) design a novel context-aware feature extraction module. A new reverse parametric rectified linear unit is further proposed to improve the representation capacities. This method shows promising results on both de-raining and dehazing tasks. Based on the pyramid architecture, (Jiang et al. 2020) introduce a novel multi-scale progressive fusion network to explore the multi-scale representations to modeling rain streaks. To fully integrate conventional model-driven optimization methods and data-driven deep learning technology, (Wang et al. 2020) propose a novel interpretable network architecture for image de-raining. The proximal gradient descent technique is utilized to design the network.

2.3 Graph Convolutional Networks

Recent deep learning-based image de-raining methods are all designed based on convolutional neural networks. The limited receptive fields of CNNs prevent them from taking all the contextual image information into account. Although the dilated convolution (Yu and Koltun 2016) are utilized to increase the receptive field, it still capture a limited local spatial information.

On the other hand, graph convolutional neural networks (Kipf and Welling 2017) are proposed to effectively model long-range contextual information. GCNs have been used for various high-level vision tasks (Qi et al. 2017; Johnson et al. 2018; Li et al. 2018c), and for image and video, the most widely used form of GCNs is non-local networks (Wang et al. 2018). For example, for object detection and video understanding, non-local operations learn an adjacency matrix to character the spatial coherence between all pixels in the image. Recently, the non-local operator has been applied to single image de-raining (Li et al. 2018). However, this method directly utilizes the original non-local operation and thus have a huge memory cost. In addition, there are still few methods to explore the potential value of GCNs for the image de-raining.

Different from existing CNNs-based methods, we specifically design two GCNs to model the global contextual information of input features. One GCN is used to explore global spatial coherence and the other one aims to measure channel correlation. Compared with the method (Li et al. 2018), our well designed GCNs can calculate more efficiently and generate features from the new channel dimension. The obtained two context-aware features contain rich global information and provide more comprehensive representations for the specific single image de-raining task.

3 Methodology

In Fig. 1, we present our proposed model for single-image de-raining. To summarize at a high level, our network mainly contains ten basic units and each is comprised by one dilated convolution block and two graph convolution blocks. The direct network output is the residual map, which is a common technique used in existing deep learning-based methods (Li et al. 2018b; Yang et al. 2019) to ease learning. The whole de-raining process is performed in a successive manner to fully

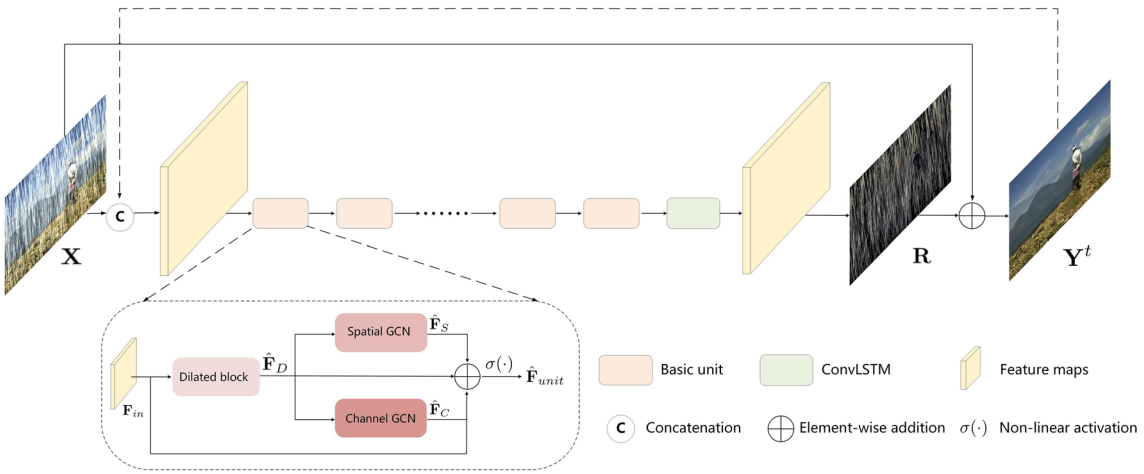


Fig. 1 The framework of the proposed successive graph convolutional network for single image de-raining. Our network contains two standard convolutional layers, ten basic units and one ConvLSTM layer. Each basic unit consists of one dilated block, one spatial GCN block and one channel GCN block. t indicates the current stage

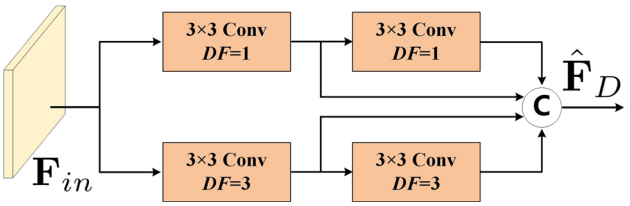


Fig. 2 The structure of our dilated block. $\mathbf{F}_{in}$ is the input features and DF indicates the dilation factor. The generated multi-scale features are concatenated to produce the output features $\hat{\mathbf{F}}_D$

exploit the extracted rich features. Below, we describe the network design, loss function, and implementation details of our method.

3.1 Network Architecture

Our network architecture mainly contains three components: one feature extraction layer, ten basic units and one reconstruction layer. The feature extraction layer is designed to extract basic features from the rainy input. The operation of this layer is defined as

$$\mathbf{F} = \sigma(\mathbf{W} * \mathbf{X} + \mathbf{b}), \tag{1}$$

where $\mathbf{X}$ is the input, $\mathbf{F}$ is the feature map, $*$ indicates the convolutional operation, $\mathbf{W}$ and $\mathbf{b}$ are the kernel and bias, $\sigma(\cdot)$ is the non-linear activation. The basic unit, which is the core of our model, aims to explore both local and global information. This unit will be detailed at the following section. The reconstruction layer is used to generate the de-rained image $\mathbf{Y}$ by

$$\mathbf{Y} = \mathbf{X} + \mathbf{R} = \mathbf{X} + f(\mathbf{X}), \tag{2}$$

where $\mathbf{R}$ is the residual generated by the network $f(\cdot)$.

3.2 Basic Unit

We design one dilated convolution block and two graph convolution blocks to form the basic unit. The former block aims to capture local spatial patterns while the later blocks focus on modeling global contextual information. In this way, our model is able to explore effective information from multiple dimensions to serve the de-raining process.

3.2.1 Dilated Convolution Block

Since rain streaks and object structures are spatially long, we therefore use dilated convolutions (Yu and Koltun 2016) to capture multi-scale spatial patterns. Dilated convolutions is able to increase the contextual area while preserving resolution and reducing parameters burden. The features obtained by dilated convolutions are defined as

$$\mathbf{F}_{DF} = \mathbf{W}_{DF} * \mathbf{F}_{in} + \mathbf{b}, \tag{3}$$

where DF is the dilation factor, $\mathbf{F}_{in}$ is the input feature, $\mathbf{F}_{DF}$ is the output feature of convolution with DF.

To obtain multi-scale spatial information, we further propose a parallel fusion strategy as shown in Fig. 2. Specifically, we design two convolutional paths in each dilated block. The upper path consists of two normal convolutional layers to capture small-scale spatial patterns, while the lower one contains two dilated convolutional layers for large-scale spatial information. The output of this block is generated by fusing the four features with a 1×1 convolutional operation. In this way, the fused features $\hat{\mathbf{F}}_D$ contains information of different receptive fields, i.e., 3×3, 5×5, 7×7 and 13×13. This enables the dilated convolution block to effectively extract multi-scale local spatial representations.

3.2.2 Graph Convolution Blocks

Although the dilated convolution is utilized to obtain a large receptive field, the information contained in generated features is still from a local spatial region. Different with CNNs, the graph convolution allows long-range information interaction. For images, a typical form of graph convolution is the non-local operation (Wang et al. 2018), which is able to measure the similarity between one pixel and all pixels in the image. Let a feature map be $\mathbf{F} \in \mathbb{R}^{HW \times N}$, where N is the number of channel, H and W are the height and width of $\mathbf{F}$, respectively. The graph convolution is defined as Kipf and Welling (2017)

$$\hat{\mathbf{F}} = \mathbf{AFW}, \tag{4}$$

where $\mathbf{A}$ is the adjacency matrix and $\mathbf{W}$ is the weight matrix. For the specific image de-raining task, we propose two graph convolution blocks to learn representations of global spatial coherence and channel correlation, respectively.

Spatial graph convolution block. We first build a fully-connected graph to model the global spatial coherence. In our spatial graph convolution network, the nodes of the graph are pixels and the adjacency matrix measures the similarity between these pixels. Similar with the non-local network Wang et al. (2018), we use three 1×1 convolution layers, $\theta(\cdot)$, $\nu(\cdot)$ and $\xi(\cdot)$, on the input feature to reduce channels and aggregate spatial information. As shown in Fig. 3, the new feature is defined by

$$\begin{aligned}\hat{\mathbf{F}}_S &= \mathbf{A}_S\mathbf{F}_S\mathbf{W}_S \\ &= \theta(\hat{\mathbf{F}}_D)\nu(\hat{\mathbf{F}}_D)^\top\xi(\hat{\mathbf{F}}_D)\mathbf{W}_S,\end{aligned} \tag{5}$$

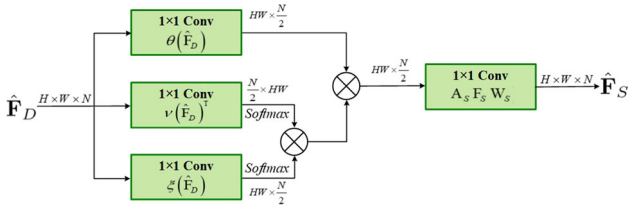


Fig. 3 The structure of our spatial GCN block. The sizes of both input and output feature maps are $H \times W \times N$ and $\otimes$ denotes matrix multiplication

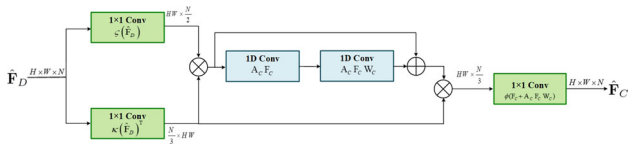


Fig. 4 The structure of our channel GCN block. The green box is the 2D convolution using a kernel with size of 1×1. The blue box is the 1D convolution and $\bigoplus$ denotes element-wise addition

where $\hat{\mathbf{F}}_D$ is output of the dilated block, $\top$ is the transpose operation. $\theta(\cdot)\nu(\cdot)^\top$ is performed by matrix multiplication and can be seen as the adjacency matrix $\mathbf{A}_S$. Note that the term $(\theta(\cdot)\nu(\cdot)^\top)\xi(\cdot)$ can be re-ordered to $\theta(\cdot)(\nu(\cdot)^\top\xi(\cdot))$ according to the associative rule (Chen et al. 2018). Compared with the original non-local network (Wang et al. 2018; Li et al. 2018) which calculates a large adjacency matrix with size of $(HW)^2$, using the re-ordering can significantly reduce computation complexity. We also introduce the softmax operation to avoid numerical instabilities. The weighting process of $\mathbf{W}_S$ is performed by using one 1×1 convolution layer on $\mathbf{A}_S\mathbf{F}_S$ to perform a hidden-to-output operation. By deploying the proposed spatial GCN into the basic unit, our model is allowed to produce coherent predictions that consider all pixels of the input feature.

Channel graph convolution block. In addition to exploring global spatial coherence, we also design a channel graph convolution block to take the channel correlation into consideration. This channel GCN aims to model interdependencies along the channel dimensions of the input features.

First, to aggregate information from different channels, we adopt two 1×1 convolutions $\kappa(\cdot)$ and $\zeta(\cdot)$ on the input feature $\hat{\mathbf{F}}_D$, where $\kappa(\hat{\mathbf{F}}_D) \in \mathbb{R}^{HW \times \frac{N}{3}}$ and $\zeta(\hat{\mathbf{F}}_D) \in \mathbb{R}^{HW \times \frac{N}{2}}$. Then a new feature $\mathbf{F}_C$ that represents the channel correlation is given by

$$\mathbf{F}_C = \kappa(\hat{\mathbf{F}}_D)^\top \zeta(\hat{\mathbf{F}}_D). \tag{6}$$

Here, the size of this new feature $\mathbf{F}_C$ is $\frac{N}{3} \times \frac{N}{2}$, that is, it contains $\frac{N}{2}$ nodes, and the dimension of each node is $\frac{N}{3}$. For this channel GCN, we construct a fully-connected graph with the adjacency matrix $\mathbf{A}_C \in \mathbb{R}^{\frac{N}{3} \times \frac{N}{3}}$ and the weights $\mathbf{W}_C \in \mathbb{R}^{\frac{N}{2} \times \frac{N}{2}}$ on the new feature $\mathbf{F}_C$ to describe the channel correlation. As shown in Fig. 4, our channel graph convolution is defined by

$$\begin{aligned}\hat{\mathbf{F}}_C &= \phi(\mathbf{F}_C + \mathbf{A}_C\mathbf{F}_C\mathbf{W}_C) \\ &= \phi(\mathbf{F}_C + \mathbf{A}_C(\kappa(\hat{\mathbf{F}}_D)^\top \zeta(\hat{\mathbf{F}}_D))\mathbf{W}_C),\end{aligned} \tag{7}$$

where the adjacency matrix $\mathbf{A}_C$ and the weight $\mathbf{W}_C$ are implemented by the 1D convolution (Chen et al. 2019) and learned from training data. We add $\mathbf{F}_C$ for better gradient back prorogation during training. Since the size of generated graph is $\frac{N}{3} \times \frac{N}{2}$, we design a operation to perform the hidden-to-output operation. Specifically, we first multiply $\kappa(\hat{\mathbf{F}}_D)$ by the generated graph and then utilize one convolutional layer $\phi(\cdot)$ to increase the number of channel to N. In this way, the size of $\hat{\mathbf{F}}_C$ is $HW \times N$, so that it can participate in subsequent operations. By deploying the proposed channel GCN into the basic unit, our model is allowed to capture correlations between channels of the input feature.

Finally, by taking the generated features of all blocks into consideration, the output feature of our basic network unit is defined by

$$\hat{\mathbf{F}}_{unit} = \sigma(\mathbf{F}_{in} + \hat{\mathbf{F}}_D + \hat{\mathbf{F}}_S + \hat{\mathbf{F}}_C). \tag{8}$$

The refined feature thus contains rich representations from multiple dimensions.

It is worth noting that our proposed unit is similar to the inception block (Szegedy et al. 2015, 2016, 2017). Both our unit and the inception block are constructed in the fashion of network in network (Lin et al. 2014). This makes our unit the same as the inception block and can be used as the basic component of deep networks. However, our proposed unit has several advantages in dealing with image de-raining. First, the inception block utilizes parallel filters with different kernel sizes to extract multi-scale representations. Our dilated block has the same goal, but it can get a larger receptive field. Although the depth of the inception block (Szegedy et al. 2015) and our dilated block are both 2, the largest receptive field of the former and the latter in a single block is 5 and 13, respectively. Second, the inception block only focuses on exploring multi-scale features of local spatial regions. While our proposed two GCN blocks are able to explore global information from the spatial and channel dimensions. Third, since the inception block is designed for high-level vision tasks, pooling operations are incorporated into the block to obtain abstract features. While we remove all pooling operations to reduce the loss of detailed information. Therefore, compared to the inception block, the above mentioned advantages can effectively improve the network representation capability and make it more suitable for the image de-raining task.

3.2.3 Successive Image De-raining

To fully exploit the extracted multi-dimension information, we take the basic network structure one step further and preform the de-raining process in a successive manner. First, we incorporate a recurrent layer to propagate fused rich features to facilitate de-raining. We choose the convolutional long short-term memory (ConvLSTM) (Xingjian et al. 2015) as our recurrent layer due to its empirical superiority.

At stage t, the concatenation of the current feature $\mathbf{F}^t$ and previous state $\mathbf{h}^{t-1}$ is sent into ConvLSTM. It includes an input gate $\mathbf{i}^t$, a forget gate $\mathbf{f}^t$, an output gate $\mathbf{o}^t$ and a cell state $\mathbf{c}^t$. The current operations are expressed as

$$\begin{aligned}
\mathbf{F}^t &= concat(\mathbf{F}^t, \mathbf{h}^{t-1}),\\
\mathbf{i}^t &= sigmoid(\mathbf{W}_{iF} * \mathbf{F}^t + \mathbf{W}_{is} * \mathbf{h}^{t-1} + \mathbf{b}_i),\\
\mathbf{f}^t &= sigmoid(\mathbf{W}_{fF} * \mathbf{F}^t + \mathbf{W}_{fs} * \mathbf{h}^{t-1} + \mathbf{b}_f),\\
\mathbf{o}^t &= sigmoid(\mathbf{W}_{oF} * \mathbf{F}^t + \mathbf{W}_{os} * \mathbf{h}^{t-1} + \mathbf{b}_o),\\
\mathbf{g}^t &= tanh(\mathbf{W}_{gF} * \mathbf{F}^t + \mathbf{W}_{gs} * \mathbf{h}^{t-1} + \mathbf{b}_g),\\
\mathbf{c}^t &= \mathbf{f}^t \odot \mathbf{c}^{t-1} + \mathbf{i}^t \odot \mathbf{g}^t,\\
\mathbf{h}^t &= \mathbf{o}^t \odot tanh(\mathbf{c}^t),
\end{aligned} \tag{9}$$

where $\odot$ is element-wise product, $concat(,)$ is the concatenation. $sigmod(\cdot)$ and $tanh(\cdot)$ are sigmoid function and hyperbolic tangent function, respectively. To alleviate the memory burden, we only deploy this layer on the output feature of the last unit.

Inspired by the *Strengthen-Operate-Subtract* (SOS) method (Romano and Elad 2015; Chen et al. 2019), which aims to improve the image de-noising performance of the boosting framework, we further concatenate the previous de-rained result and the rainy image as the input of the current stage. In each stage of the original SOS algorithm, the input of the de-noising *operator* $f(\cdot)$ is the *strengthened* image $\mathbf{X} + \mathbf{Y}^t$. To obtain faithful de-noised results, the output of $f(\cdot)$ *subtracts* $\mathbf{Y}^t$ and the calculation at stage t is defined by

$$\mathbf{Y}^t = f(\mathbf{X} + \mathbf{Y}^{t-1}) - \mathbf{Y}^{t-1}. \tag{10}$$

Since $\mathbf{Y}^t$ is a de-noised version, according to the Cauchy-Schwarz inequality, we have $SNR(\mathbf{X} + \mathbf{Y}^{t-1}) > SNR(\mathbf{X})$ (Romano and Elad 2015), where SNR denotes signal-to-noise ratio (SNR). This SOS operation makes the de-noising process more effectively due to the improved SNR. Therefore, we utilize both the previous de-rained result and the rainy image to strengthen the signal. We remove the subtraction in Eq. (10) and use concatenation to let the network automatically learn its parameters to predict the residual image. At stage t, Eq. (2) can be formulated as

$$\mathbf{Y}^t = \mathbf{X} + \mathbf{R} = \mathbf{X} + f(concat(\mathbf{X}, \mathbf{Y}^{t-1})), \tag{11}$$

where $\mathbf{R}$ denotes the residual generated by our network (operator) $f(\cdot)$, and $\mathbf{Y}^T$ is the final de-rained result at the last stage T. Compared to using only the rainy input or only the previous de-rained result (Li et al. 2018b), adding both of them provides more useful information to help boost the de-raining performance.

3.2.4 Loss Function

The most widely used loss function for training a network is mean squared error (MSE). However, MSE usually generates over-smoothed results because of its ℓ_2 penalty. To address this drawback, we adopt the SSIM (Ren et al. 2019) as our loss function to balance rain removal and detail preservation

$$\mathcal{L} = \frac{1}{M}\sum_{i=1}^{M} 1 - SSIM(\mathbf{Y}_i^T, \mathbf{Y}_{gt,i}), \tag{12}$$

where M is the number of training data and $\mathbf{Y}_{gt}$ is the ground truth.

3.2.5 Training Details

We set the kernel size of the dilated convolution block as 3×3. The number of feature maps is 18 for all convolutions and the non-linear activation $\sigma(\cdot)$ is ReLU (Krizhevsky et al. 2012). The dilated factors are set 1 and 3 within each dilated block. We found 10 basic units are enough to generate good results. We use TensorFlow (Abadi et al. 2016) and Adam (Kingma and Ba 2014) with a mini-batch size of 8 to train our network. We initialize the learning rate to 0.001, divide it by 10 at 30 and 70 epoch, and terminate training after 100 epoch. We randomly select 100×100 patch pairs from training image data sets as inputs. All experiments are performed on a server with Intel Core i7-8700K CPU and NVIDIA GTX 1080Ti GPU.

4 Experiments

We quantitatively evaluate our model on both and real-world synthetic data sets. We compare our method with one model-based deraining method: Joint Convolutional Analysis and Synthesis (JCAS) (Gu et al. 2017), and seven recent deep learning-based methods: Deep Detail Networks (DDN) (Fu et al. 2017), Density-aware Image De-raining (DID) (Zhang and Patel 2018b), RECurrent Squeeze-and-excitation Context Aggregation Net (RESCAN) (Li et al. 2018b), JOint Rain DEtection and Removal with detail Enhancement (JORDER-E) (Yang et al. 2019), Semi-supervised Image Rain Removal (SIRR) (Wei et al. 2019), Progressive Recur-

rent Network(PReNet) (Ren et al. 2019) and SPatial Attentive Network (SPANet) (Wang et al. 2019).

4.1 Synthetic Data

We use five public synthetic data sets provided by the method (Li et al. 2016), JORDER-E (Yang et al. 2019), DDN (Fu et al. 2017) and DID (Zhang and Patel 2018b). These five data sets were generated using different synthetic strategies. The method (Li et al. 2016) contains 12 testing light rainy images, which generated by using Matlab and the additive composite model expressed as:

$$\mathbf{Y} = \mathbf{X} + \mathbf{R}. \tag{13}$$

The JORDER-E dataset contains two subsets, each with 200 testing images, which have light rain streaks and challenging heavy rain streaks, respectively. This dataset is synthesized by using Matlab and the rain accumulation model:

$$\mathbf{Y}{=}\alpha\left(\mathbf{X} + \sum_{l=1}^{L}\mathbf{R}_l\right) + (1-\alpha)\mathbf{A}, \tag{14}$$

where l denotes the rain streak layer and L is the total number of rain streak layers. $\mathbf{A}$ is the global atmospheric light, and α is the atmospheric transmission. The rest two datasets contain 1400 and 1200 testing images, respectively. The authors adopt Photoshop,[1] in which the process of rainy image generation is based on the screen blending model:

$$\mathbf{Y} = \mathbf{X} + \mathbf{R} - \mathbf{X}\mathbf{R}. \tag{15}$$

Although these rainy imaging models are heuristic, the synthetic datasets are at least somewhat helpful in removing rain streaks, as reported in the literature (Yang et al. 2020). For these five synthetic datasets, we call them, 'Rain12', 'Rain200L', 'Rain200H', 'DDN-Data' and 'DID-Data'. The models trained on JORDER-E light rainy dataset are used to test 'Rain12'.

We show four representative visual results with different rain orientations and magnitudes from Figs. 5, 6, 7, 8, 9, and their corresponding references are shown in Fig. 10. For the light rainy images from *Rain200L* data set, except for JORDER-E, PreNet and our method, the remaining methods can not effectively remove rain streaks in bright areas in the red rectangles. While our model has a better structure preservation than JORDER-E and PreNet, as shown in the blue rectangles in Fig. 5. For the challenging *Rain200H* data set, it is clear that JCAS fails to process heavy rain streaks due to modeling limitations. Since unsupervised training samples makes the data distribution deviate from synthetic data, the semi-supervised method SIRR generates obvious artifacts. As shown in the red and blue rectangles, DDN, DID, RESCAN and SPANet are able to remove the rain streaks while tending to over-smooth the results. JORDER-E, PreNet and our model have similar global visual performance and outperform other methods. However, as shown in Figs. 6j and 7j, our method is able to preserve more informative details. For the relatively light rainy data sets *DDN-Data* and *DID-Data*, all methods generate de-rained result with close visual qualities, while our model contains less artifacts, which achieves the best PSNR value. Moreover, we calculate average PSNR and SSIM values for quantitative evaluation. By following the strategy of JORDER-E (Yang et al. 2019), we calculate both PSNR and SSIM values in the Y channel of YCbCr space. As shown in Tables 1 and 2, our method has the best overall results on both PSNR and SSIM, which is consistent with visual results.

4.2 Real-World Data

To demonstrate the generalization ability of our method, we conduct experiments on the recent public real-world rainy dataset *SPA-Data* (Wang et al. 2019), which contains 1000 testing images with labels. Specifically, we first retrain all deep learning-based methods on a disjoint synthetic data set (Li et al. 2019c), and then directly test them on the *SPA-Data*. Figure 11 shows one visual result and the rectangles indicate that our network can simultaneously remove rain and preserve object edges. Since this real-world data set contains ground truth, we can quantitatively evaluate the de-rained performance. As shown in Table 3, our method consistently achieves the best results. We also show another comparison on a real-world rainy image from the Google image searching engine in Fig. 12. As can be seen, our model can remove rain streaks, preserve feather details and reduce artifacts on the bird leg. This experiment shows that our learned network, which is trained on synthetic data, translates well to real-world rainy images.

To provide realistic feedback and quantify the subjective evaluation, we also constructed an independent user study. In this experiment, we randomly select 200 de-rained results from *SPA-Data* and display them on a screen. We then separately asked 20 participants to rank each image from 1 to 5 subjectively according to five measurements, i.e., the overall rating, de-raining effect, artifacts reduction, clarity perception and color preservation. The participants are given instructions that color distortion and over de-raining artifacts should decrease the quality. The score 1 represents the worst quality image and 5 represents the best quality image. We show the radar chart of the five measurements in Fig. 13. It is clear that our method outperforms other methods on de-raining performance and has promising results on

[1] https://www.photoshopessentials.com/photo-effects/photoshop-weather-effects-rain/

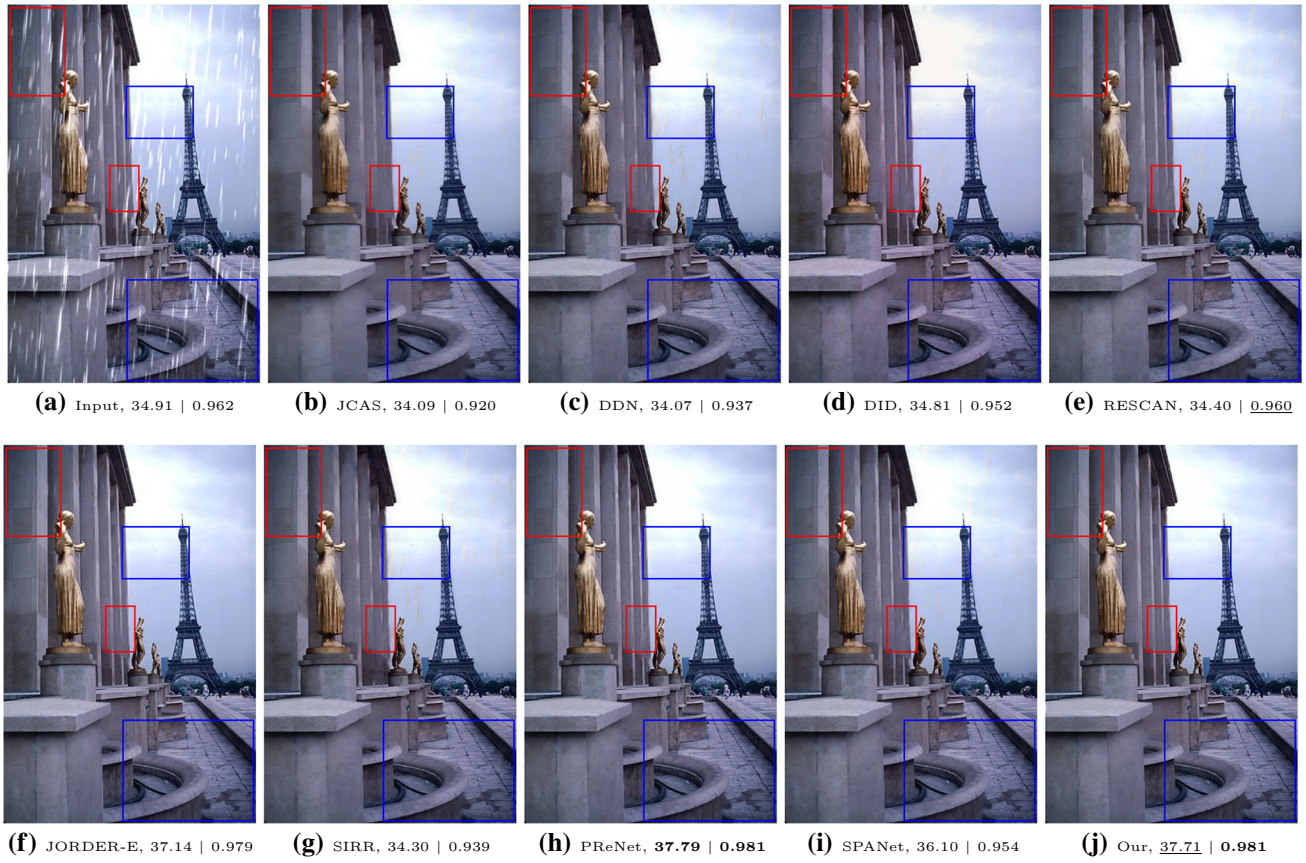


Fig. 5 One visual comparisons with PSNR | SSIM values from the synthetic data set '*Rain200L*'

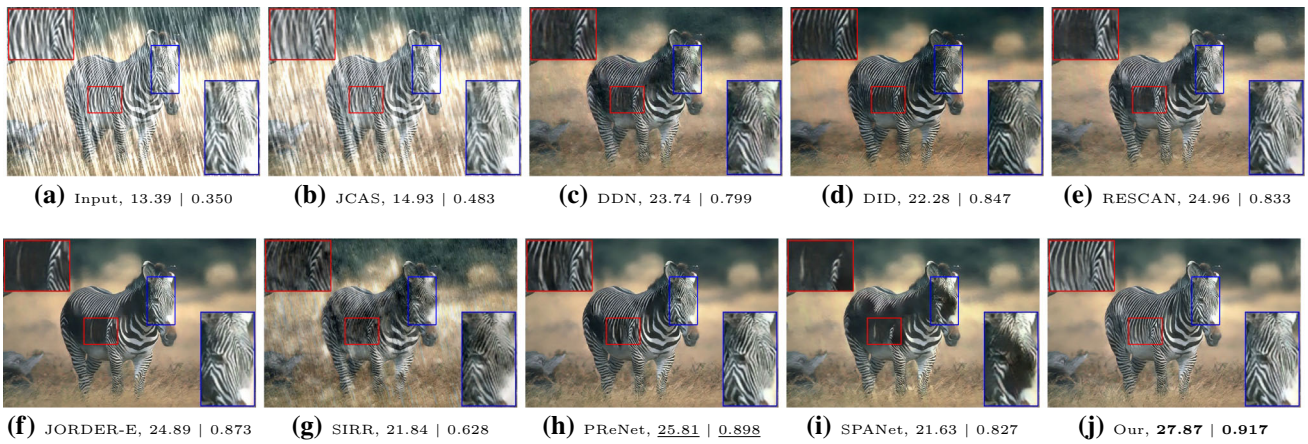


Fig. 6 One visual comparisons with PSNR | SSIM values from the synthetic data set '*Rain200H*'

other measurements. This gives additional support that our GCNs-based network is able to improve the subjective visual qualities of real-world data.

When dealing with more complex imaging conditions, an intuitive solution is to cascade our model and other existing algorithms. Figure 14 shows five examples by directly combining our model with one image de-hazing method (Ren et al. 2019). As can be seen, both rain streaks and hazy effects are significantly eliminated. We believe that the performance can be further improved by integrating physical imaging models into our network.

4.3 Runtime and Parameter Numbers

The proposed algorithm is efficiency on both calculation and storage. We use 100 images with a size of 1000 × 1000 for

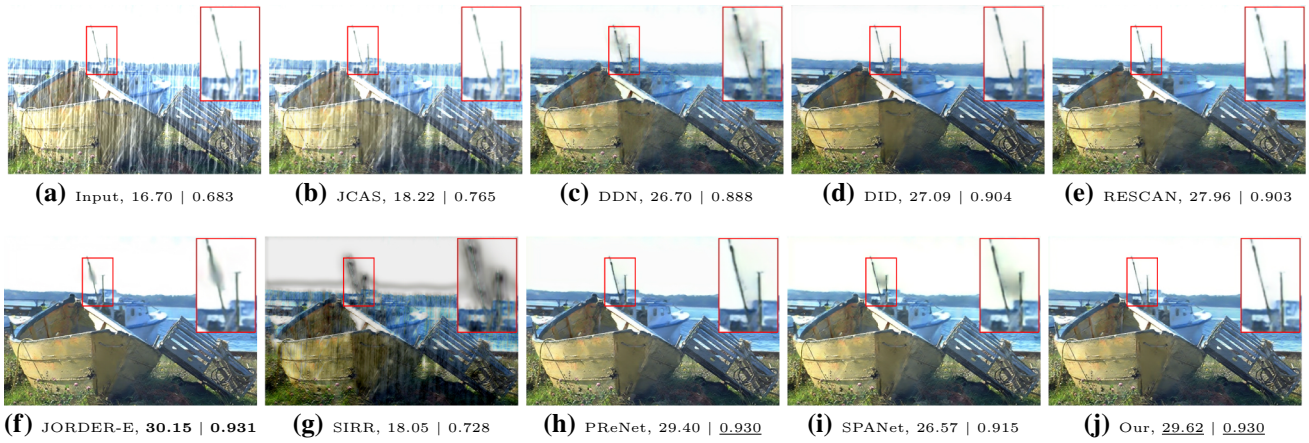


Fig. 7 One visual comparison with PSNR | SSIM values from the synthetic data set '*Rain200H*'

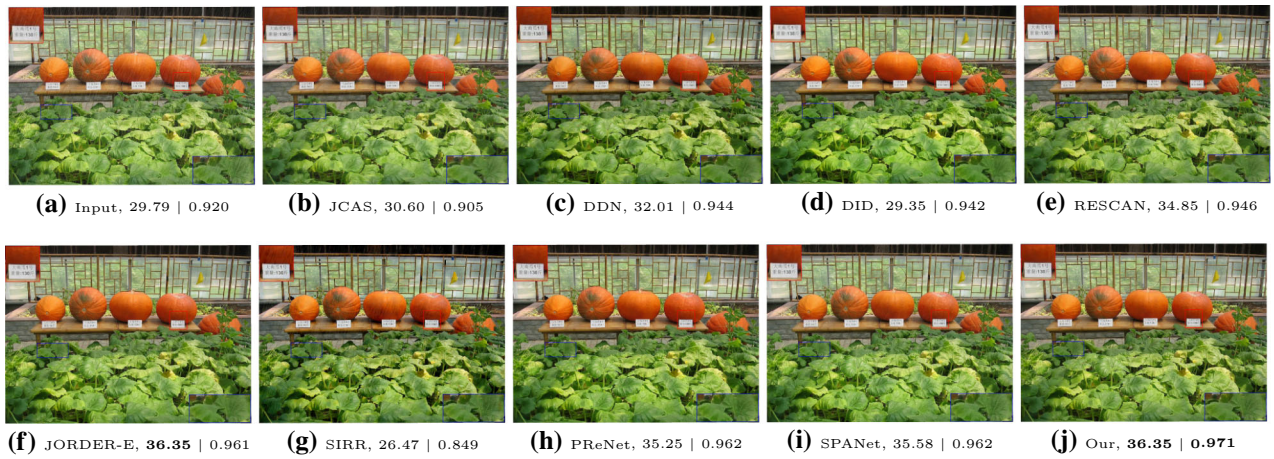


Fig. 8 One visual comparison with PSNR | SSIM values from the synthetic data set '*DDN-Data*'

evaluation. Comparisons on average runtime and parameter numbers are shown in Table 4. According to the provided codes, the JCAS and SPANet are tested on CPU and GPU, respectively. Other deep learning-based methods are tested on both CPU and GPU. Our method has a comparable GPU runtime compared with other deep learning methods, and is significantly faster than several deep models on a CPU. This is because our model is able to explore more effective representations, which leads to comparable results with tolerable resource consumption.

5 Ablation Studies and Discussions

We provide ablation studies to explore the effect of each part of our model over the challenging *Rain200H* (Yang et al. 2019).

5.1 Effect of GCN Blocks

Since introducing the GCN is the core of this paper, we first test the effect of proposed two GCN blocks by comparing to the baseline network which only contains dilated blocks. As shown in Table 5, adding the spatial GCN block achieves an improvement of 1.51% and 3.02% on average PSNR and SSIM, respectively. Similarly, the channel GCN improves the baseline by 2.21% and 3.83%. The best results are obtained by combining the two blocks together, resulting in an improvement of 3.40% and 4.99%, respectively.

We also show a visual comparison in Fig. 15. As can be seen in Fig. 15a, using only dilated blocks show powerless for preserving spatially long structures, i.e., the stripes on zebra. In Fig. 15b, adding spatial GCN blocks can generate sharp results, but obvious white artifacts appear on the zebra's body. While adding channel GCN blocks is able to avoid white artifacts and generate clearer results than dilated blocks, it still has a blur effect at the junction of black and white stripes

Fig. 9 One visual comparison with PSNR | SSIM values from the synthetic data set '*DID-Data*'

Fig. 10 References of Figs. 5,6, 7, 8, 9,11

Table 1 Average PSNR values of de-rained results on the three synthetic data sets

	JCAS	DDN	DID	RESCAN	JORDER-E	SSIR	PreNet	SPANet	Ours
Rain12	33.10	35.74	36.25	36.54	**36.73**	35.71	36.61	35.92	36.53
Rain200L	31.42	34.68	35.40	36.09	37.25	34.75	**37.80**	35.79	37.65
Rain200H	14.69	25.68	27.91	27.75	**29.34**	20.37	29.03	26.27	29.13
DDN-Data	26.80	29.99	30.84	31.18	**32.79**	28.44	32.63	31.35	32.13
DID-Data	25.16	30.95	31.65	32.35	32.48	31.97	33.01	31.94	**33.53**

The best and the second best results are **boldfaced** and underlined

in Fig. 15c. The best visual quality is obtained by jointly utilizing the two GCN blocks, as shown in Fig. 15d.

In Fig. 16, we visualize some of the feature maps of Fig. 17 to see which representations our proposed blocks have learned. Here we show five representative features of each block from the last unit. To highlight specific feature representations learned by different blocks, we use the absolute values of the image, normalize them to 0 to 1, and only display pixels with values greater than 0.5. Since the dilated convolution block receives the output of the previous unit directly, the features $\hat{\mathbf{F}}_D$ from this block contain comprehensive representations of both rain streaks and the animal. This is consistent with our intention to capture and fuse multi-scale local spatial patterns. The features $\hat{\mathbf{F}}_S$ of the spatial GCN block have a great response to the similar and repeated rain streaks, which have obvious long connection structures in space. This is in line with our expectations for the spatial GCN, that is, to effectively capture long-range spatial structure information. The features $\hat{\mathbf{F}}_C$ from the channel GCN block explore correlations across channels to highlight object parts, i.e., the animal's body and trunk in the image. This is similar with high-level vision tasks that capture semantic features. Therefore, unlike most existing deep CNNs-based methods that only utilize local spatial features, introducing GCNs can extract features from new dimensions, which allows our network to learn and emphasize specific characteristics.

Table 2 Average SSIM values of de-rained results on the three synthetic data sets

	JCAS	DDN	DID	RESCAN	JORDER-E	SSIR	PreNet	SPANet	Ours
Rain12	0.930	0.951	0.956	0.957	0.963	0.950	0.960	0.958	**0.964**
Rain200L	0.917	0.967	0.962	0.970	0.975	0.969	0.981	0.965	**0.983**
Rain200H	0.499	0.857	0.872	0.864	0.890	0.599	0.898	0.865	**0.902**
DDN-Data	0.848	0.892	0.912	0.915	0.924	0.890	**0.934**	0.915	0.924
DID-Data	0.813	0.915	0.924	0.917	0.923	0.896	0.926	0.902	**0.932**

The best and the second best results are **boldfaced** and underlined

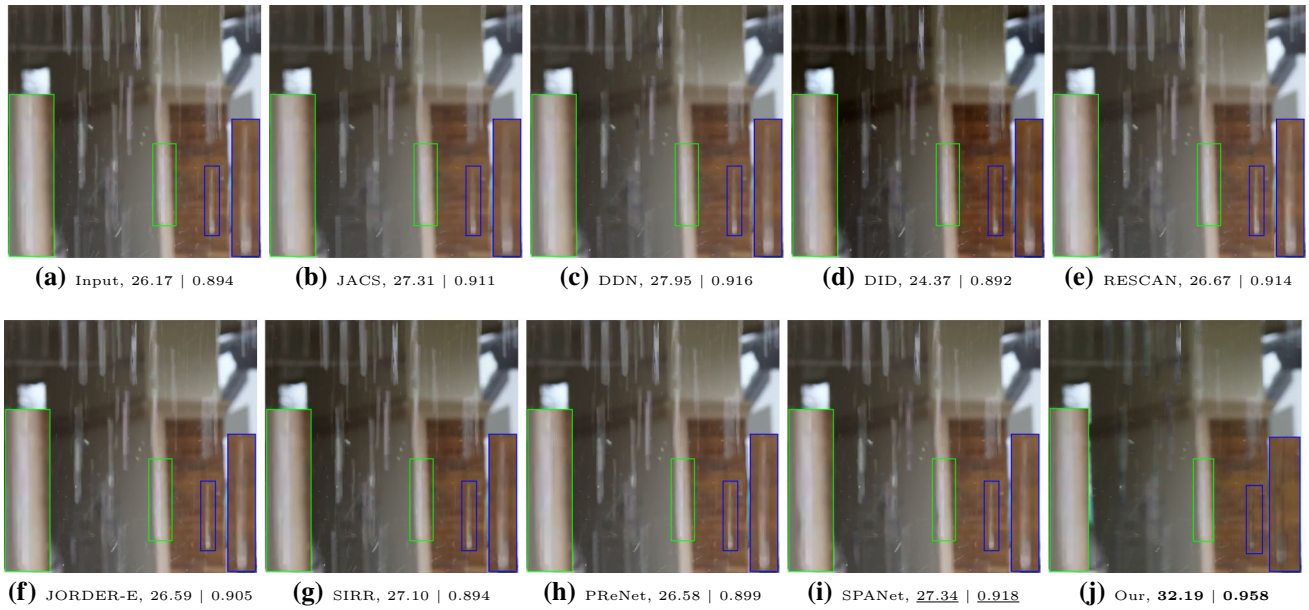


Fig. 11 One visual comparison with PSNR | SSIM values from the real-world data set '*SPA-Data*'

Table 3 Average PSNR and SSIM of de-rained results on the real-world *SPA-Data* data set (Wang et al. 2019)

	JCAS	DDN	DID	RESCAN	JORDER-E	SSIR	PreNet	SPANet	Ours
PSNR	34.95	31.05	31.98	34.71	34.55	30.25	34.68	34.26	**35.12**
SSIM	0.945	0.949	0.936	0.952	0.950	0.938	0.950	0.952	**0.957**

Note that we train all the deep learning-based methods on the synthetic data set (Li et al. 2019c) and test on the real-world SPA-Data

5.2 Effect of Successive Operations

Due to the effective propagation and reuse of information, it is common to add more recurrent stages to bring better performance. As shown in Fig. 17, with the increase of T, rain streaks are gradually removed and the image quality is improved. We also test our model with different stage numbers and the PSNR and SSIM results are shown in Fig. 18. It is clear that keeping increasing the stage number eventually brings only limited improvement. Therefore, to balance the effectiveness and efficiency, we select $T = 5$ as the default stage number.

5.3 Kernel Number Versus Unit Number

We also test the impact of kernel number and unit number. Specifically, we test the kernel numbers $K \in \{9, 18, 36\}$ and dilated block numbers $L \in \{8, 10, 12\}$. Quantitative results are shown in Tables 6 and 7. It is clear that increasing kernels and units can generate higher SSIM and PSNR. Adding basic unit results in larger modeling capacity, which has a greater advantage over increasing the number of kernels. However, increasing K and L eventually brings only limited improvement at the cost of storage and computation. Thus, to balance the trade-off between performance and speed, we choose $K = 18$ and $L = 10$ as our default setting.

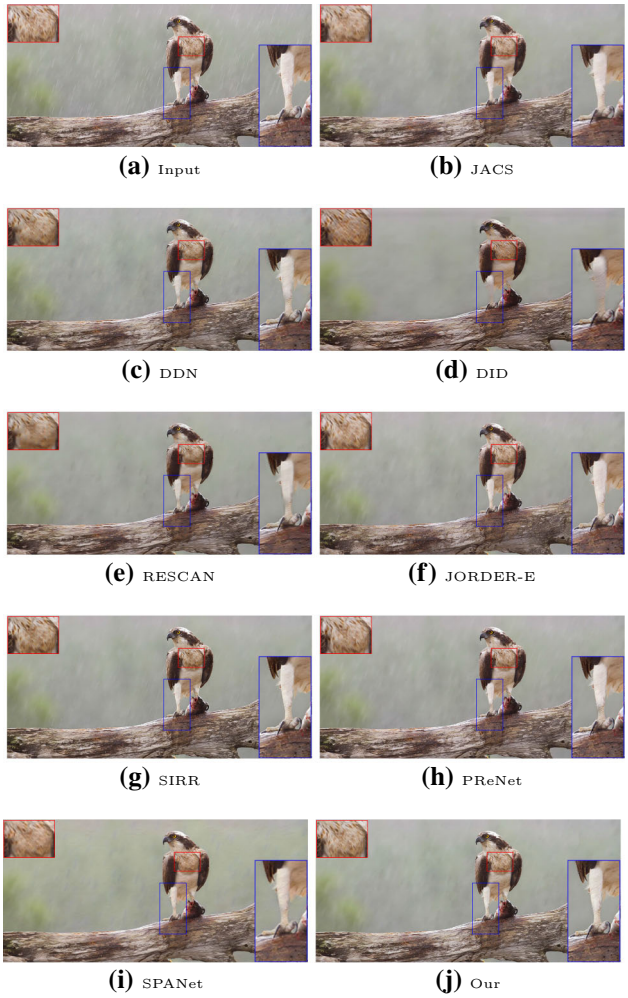


Fig. 12 One visual comparison on a real-world rainy image from Google image searching engine. Please zoom in for a better visualization

5.4 Effect of Loss Functions

We also test the impact of using the MSE and SSIM loss functions. Figure 19 shows one visual comparison on the *Rain200H* data set. As shown in the red rectangles, using only MSE loss generates a de-rained image with obvious artifacts. While the SSIM penalty is appropriate for simultaneously achieve artifacts reduction and rain removal. In Table 8, we show the quantitative evaluations of using different loss functions. We observe that SSIM loss can significantly improve both PSNR and SSIM. Therefore, we choose SSIM as the default loss function. Other advanced loss functions, such as GANs (Goodfellow et al. 2014) loss and perceptual loss (Johnson et al. 2016), can also be utilized to further improve the de-raining performance.

6 Discussions

6.1 Improvements

Our network is mainly constructed by the proposed basic unit and successive operation, and these two components complement each other. On the one hand, the basic unit is able to explore multi-dimension representations. This can implicity alleviate the resource consumptions of successive operations, e.g., using relative less parameters and successive operations can generate good results. On the other hand, by adopting successive operations, the extracted rich features can be fully exploited to boost the de-raining performance. Moreover, in this paper, we choose relatively simple calculations to perform the proposed basic unit and successive operations. However, other advanced technologies, e.g., squeeze-and excitation networks (He et al. 2018) and network acceleration (Zhang et al. 2018), could also be incorporated into our model for better effectiveness and efficiency.

6.2 Limitations

As our network is trained on synthetic data with limited types, the learned model might be less effective when input images contain extremely complex rain streaks. Figure 20 shows an example with extremely dense rain streaks in the input image. In this case, our learned network cannot fully remove all rain streaks due to the domain gap between training and testing data. Therefore, our model cannot restore the image well as shown in Fig. 20b. One possible solution is to apply the domain adaptation technology (Shao et al. 2020) to reduce the domain gap, which will be our future work.

6.3 Challenges

Another problem that needs to be solved in the computer vision community is the gap between low-level vision and high-level vision. In the recent article (Li et al. 2019c), the authors find that no existing de-raining algorithms can directly help high-level vision tasks. This may be due to the fact that existing de-raining method only trained towards the goal of low-level pixel regression, which ignore semantic representations. Therefore, we also test our model as a preprocessing for Clarifai,[2] which is a commercial image recognition system based on deep CNNs. We show two results, one is synthetic rainy image from *Rain200H* data set and the other is a real-world image from Google image searching engine, in Fig. 21. As can be seen, our results are recognized as "zebra" and "road" with the highest probability, respectively. These results are consistent with the main semantic information contained in the images. Moreover,

[2] https://www.clarifai.com/

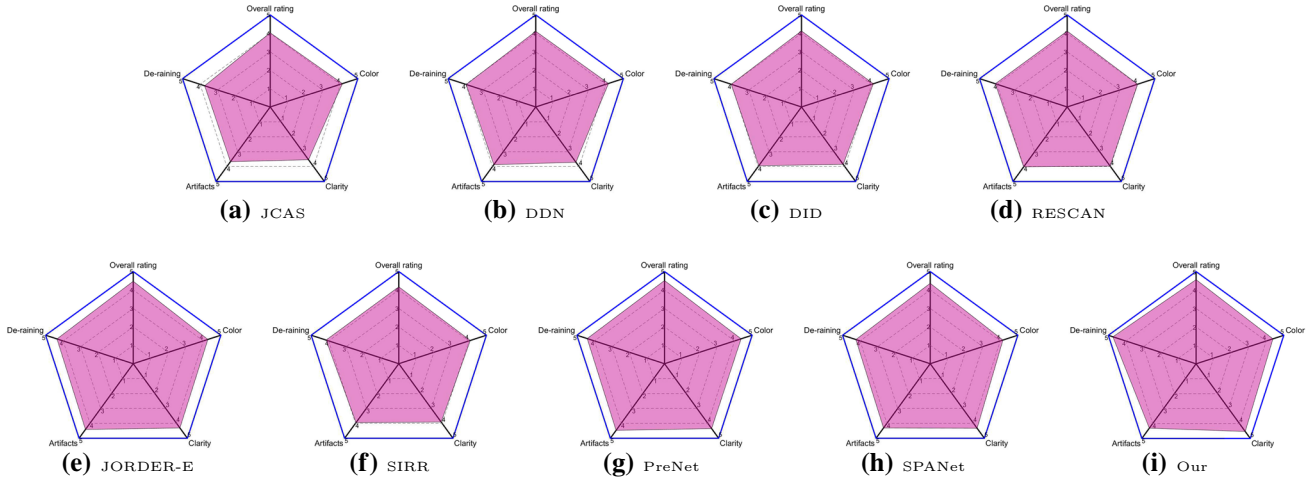


Fig. 13 Radar charts of average user study scores on overall rating, de-raining effect, artifacts reduction, clarity perception and color preservation. The score 1 represents the worst quality and 5 represents the best quality

Fig. 14 Visual results of real-world rainy image with hazy appearance. *Top* input rainy images; middle: de-hazed results by Ren et al. (2019), *bottom* de-hazed and de-rained results

Table 4 Comparisons on runtime (in seconds) and parameter numbers

	JCAS	DDN	DID	RESCAN	JORDER-E	SSIR	PreNet	SPANet	Ours
CPU	587.42	3.21	77.24	70.43	207.03	3.51	35.66	—	24.55
GPU	—	0.34	0.97	1.53	1.74	0.47	0.97	0.93	1.02
# Params	—	58,175	372,839	54,735	4,169,024	175,736	168,963	283,716	167,739

SPANet code is based on CuPy library and can only be run on GPU

as shown in Fig. 16d, the proposed channel GCN is able to extract correlations between more semantic features like animal's body and trunk. This implies that our network architecture has the potential value of connecting low-level and high-level vision tasks. We take this challenging problem as our future work.

Table 5 Comparison of different GCN blocks

	PSNR	SSIM
Baseline	28.54	0.861
Baseline + spatial GCN	28.97	0.887
Baseline + channel GCN	29.05	0.894
All	29.13	0.902

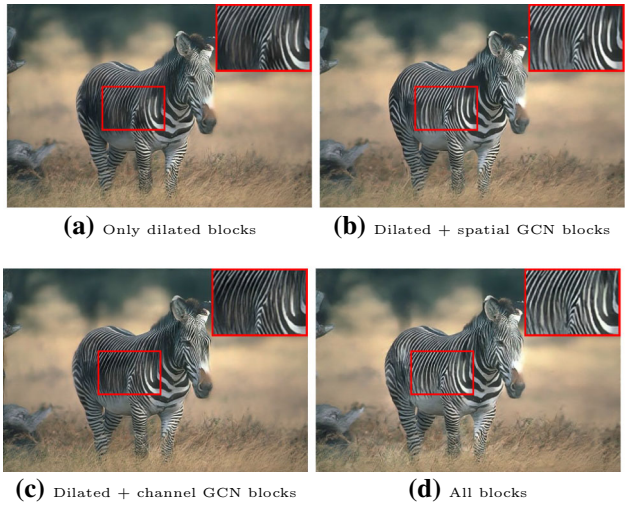


Fig. 15 Visual comparison on the effect of GCN blocks

7 Extension: JPEG Artifacts Reduction

Our model is perhaps more general than we presented it as being. Here we extend our model to another typical image restoration task: JPEG compression artifacts reduction. We use both the training set and testing set from BSD500 (Arbelaez et al. 2010) as our training data. The disjoint validation set is used for testing. We use the Matlab JPEG encoder to generate JPEG compressed images by setting the input quality value to 10, 20 and 30. Since our network is able to capture multi-scale and multi-dimension representations, we only train a single model to handle all three JPEG qualities to demonstrate the effectiveness of our method.

We compare our network with four learning-based methods, i.e., Artifacts Reduction Convolutional Neural Network (ARCNN) (Dong et al. 2015), Trainable Nonlinear Reaction Diffusion (TNRD) (Chen and Pock 2016), Denoising Convolutional Neural Network (DnCNN) (Zhang et al. 2017) and Learning Parameterized Image Operators (LPIO) (Fan et al. 2018). Figure 22 shows one visual comparison on a JPEG compressed image with quality = 10. As shown in the red rectangles, our model is able to simultaneously artifacts reduction and details preservation. We also calculate the SSIM and PSNR-B (Yim and Bovik 2010) for quantitative assessment. PSNR-B is recommended (Dong et al. 2015) for use in this problem since it is designed to be more sensitive to blocking artifacts. The quantitative results are shown in Tables 9 and 10. Our method has the best results on all JPEG qualities. We believe that our model will be motivated for other low-level vision tasks.

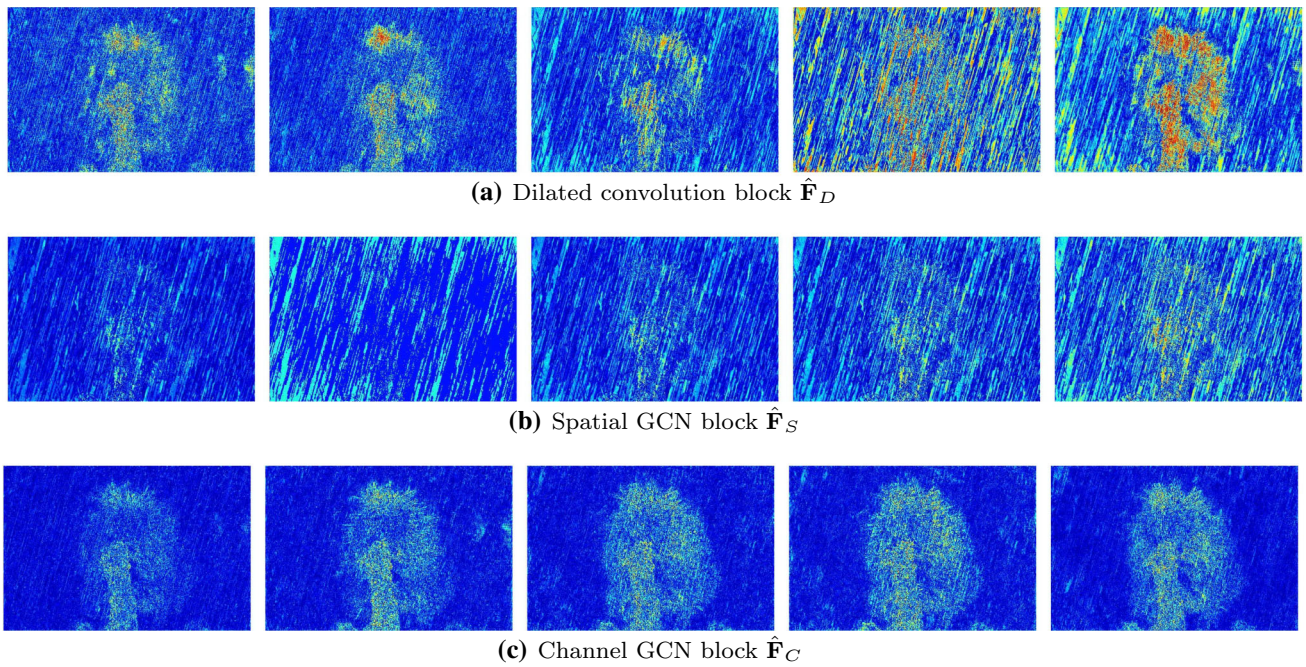
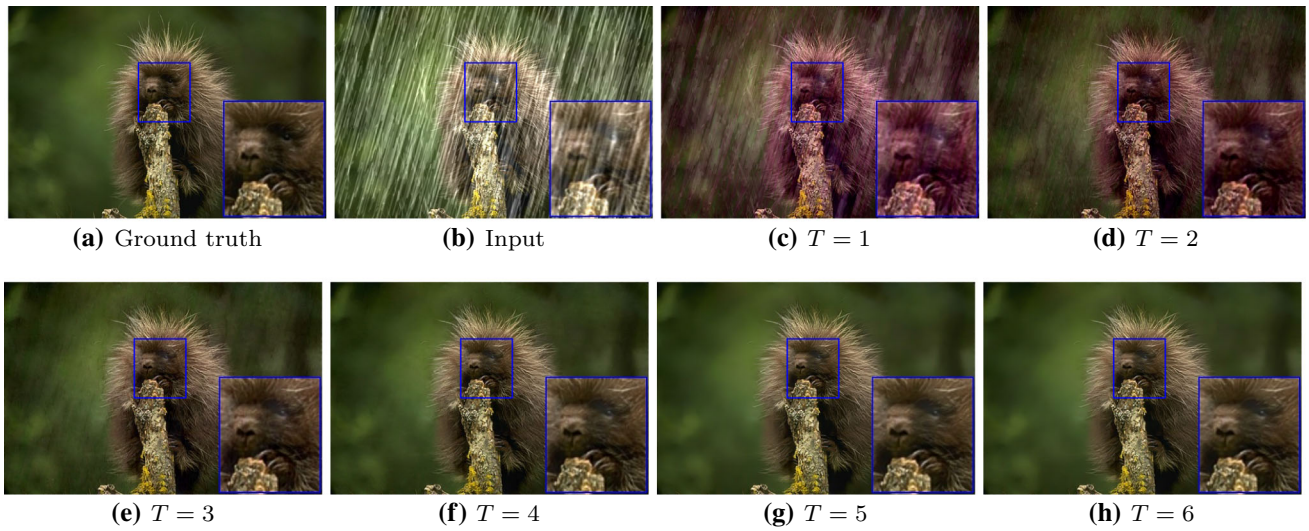


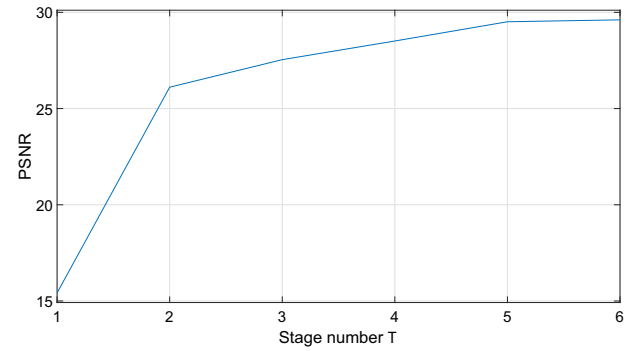
Fig. 16 Visualizations of learned feature maps from different blocks. We normalize all feature for a better visual effect. The features of dilated convolution blocks contain comprehensive representations. While the spatial GCN block and the channel GCN block respectively explore the representations with specific characteristics, that is, spatial structure information and object content information



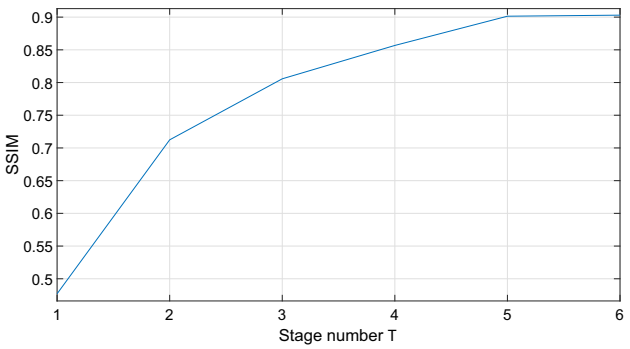
(a) Ground truth **(b)** Input **(c)** $T = 1$ **(d)** $T = 2$

(e) $T = 3$ **(f)** $T = 4$ **(g)** $T = 5$ **(h)** $T = 6$

Fig. 17 One visual example by using different stage number T



(a) PSNR performance curve



(b) SSIM performance curve

Fig. 18 Curves of PSNR and SSIM values of different stage number T over the *Rain200H* data set

Table 6 PSNR results using different kernel numbers and unit numbers

	$L = 8$	$L = 10$ (default)	$L = 12$
$K = 9$	27.93	28.10	28.70
$K = 18$ (default)	28.81	29.13	29.31
$K = 36$	28.96	29.24	29.41

Table 7 SSIM results using different kernel numbers and unit numbers

	$L = 8$	$L = 10$ (default)	$L = 12$
$K = 9$	0.879	0.882	0.887
$K = 18$ (default)	0.892	0.902	0.904
$K = 36$	0.897	0.905	0.907

8 Conclusions

In this paper, we handle the single image de-raining via a graph convolutional network-based model to extract multi-dimension representations. Specifically, we first introduce a dilated convolution block to capture multi-scale local patterns along the local spatial dimensions, and then propose two GCN blocks to extract contextual information along the global spatial dimensions and channel dimensions. The obtained rich features are further combined with recurrent

(a) Ground truth **(b)** Input **(c)** MSE loss **(d)** SSIM loss

Fig. 19 De-raining results by using different loss functions. Using SSIM loss is able to generate results with less artifacts as shown in (d)

Table 8 Quantitative results using different loss functions

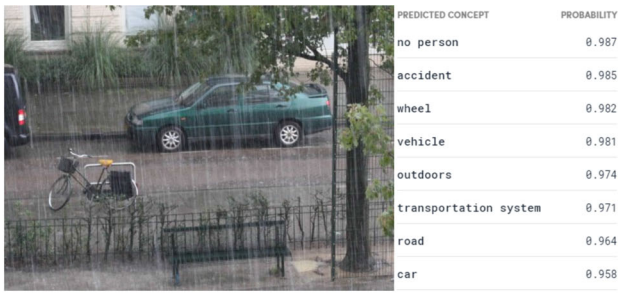
	MSE loss	SSIM loss
PSNR	28.43	29.13
SSIM	0.887	0.902

(a) Input **(b)** Our result

Fig. 20 Limitations of the proposed method. Our learned model is not effective for processing images with extremely heavy rain streaks

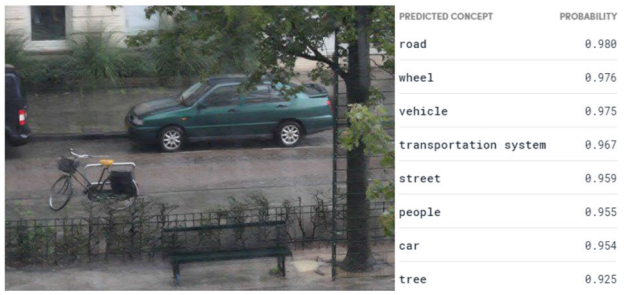
operations to jointly explore and exploit effective representations. Experiments verify the superiority of our method on both synthetic and real-world data sets. In addition, our network structure is easy to implement and has a high computational efficiency with promising de-raining performance.

(a) Synthetic rainy image



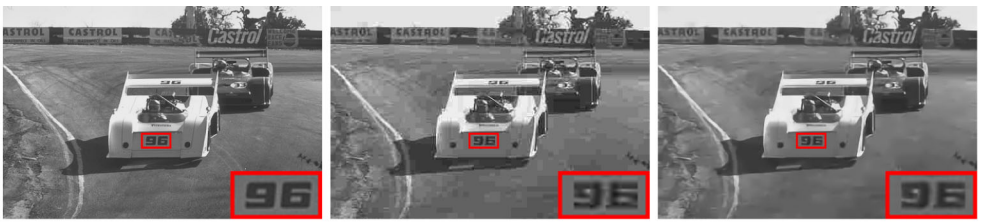
(b) Real-world rainy image

(c) Our result of (a)

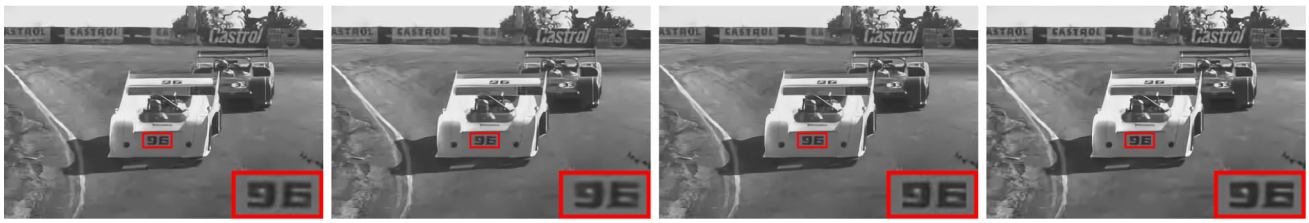


(d) Our result of (b)

Fig. 21 Preprocessing for image recognition on the Clarifai platform. **a** and **b** are recognized as "stripe" and "no person" with the highest probability, respectively. **c** and **d**, i.e., our results are accurately recognized as "zebra" and "road" with the highest probability, respectively



(a) Ground truth, PSNR-B|SSIM **(b)** JPEG, 27.72 | 0.738 **(c)** ARCNN, 28.91 | 0.768



(d) TNRD, 28.94 | 0.769 **(e)** DnCNN, 29.16 | 0.776 **(f)** LPIO, 27.17 | 0.764 **(g)** Our, **29.39** | **0.783**

Fig. 22 One visual comparison on a JPEG compressed image with quality = 10 from the BSD500 data set

Table 9 Average SSIM values comparisons on JPEG artifacts reduction

	ARCNN	TNRD	DnCNN	LPIO	Ours
Quality = 10	0.793	0.799	0.803	0.802	**0.824**
Quality = 20	0.852	0.858	0.861	0.857	**0.873**
Quality = 30	0.881	0.883	0.886	0.884	**0.891**

The best and the second best results are **boldfaced** and underlined

Table 10 Average PSNR-B values comparisons on JPEG artifacts reduction

	ARCNN	TNRD	DnCNN	LPIO	Ours
Quality = 10	28.76	29.04	29.13	29.04	**29.41**
Quality = 20	30.59	31.05	31.19	31.12	**31.53**
Quality = 30	31.98	32.24	32.38	32.28	**32.60**

The best and the second best results are **boldfaced** and underlined

Acknowledgements This work was supported by the National Key R&D Program of China under Grant 2020AAA0105702, the National Natural Science Foundation of China (NSFC) under Grants U19B2038, 61620106009 and 61901433, the USTC Research Funds of the Double First-Class Initiative under Grant YD2100002003.

Compliance with Ethical Standards

Conflict of interest The authors declare that they have no conflict of interest.

References

Abadi, M., Agarwal, A., & Barham, P., et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv: 1603.04467 (2016)

Arbelaez, P., Maire, M., Fowlkes, C., & Malik, J. (2010). Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *33*(5), 898–916.

Barnum, P. C., Narasimhan, S., & Kanade, T. (2010). Analysis of rain and snow in frequency space. *International Journal of Computer Vision*, *86*(2), 256–274.

Bossu, J., Hautière, N., & Tarel, J. P. (2011). Rain or snow detection in image sequences through use of a histogram of orientation of streaks. *International Journal of Computer Vision*, *93*(3), 348–367.

Chang, Y., Yan, L.,& Zhong, S. (2017). Transformed low-rank model for line pattern noise removal. In: ICCV

Chen, C., Xiong, Z., Tian, X., Zha, Z. J., & Wu, F. (2019). Real-world image denoising with deep boosting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*,. https://doi.org/10.1109/TPAMI.2019.2921548.

Chen, J., & Chau, L. P. (2013). A rain pixel recovery algorithm for videos with highly dynamic scenes. *IEEE Transactions on Image Processing*, *23*(3), 1097–1104.

Chen, J., Tan, C.H., Hou, J., Chau, L.P., & Li, H. (2018). Robust video content alignment and compensation for rain removal in a cnn framework. In: CVPR

Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2017). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *40*(4), 834–848.

Chen, Y., Kalantidis, Y., Li, J., Yan, S., Feng, J.: Aˆ2-nets: Double attention networks. In: NeurIPS (2018)

Chen, Y., & Pock, T. (2016). Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(6), 1256–1272.

Chen, Y., Rohrbach, M., Yan, Z., Shuicheng, Y., Feng, J., & Kalantidis, Y. (2019). Graph-based global reasoning networks. In: CVPR

Chen, Y.L., & Hsu, C.T. (2013) A generalized low-rank appearance model for spatio-temporally correlated rain streaks. In: ICCV

Dong, C., Deng, Y., Change Loy, C., & Tang, X. (2015). Compression artifacts reduction by a deep convolutional network. In: ICCV, pp. 576–584

Dong, C., Loy, C. C., He, K., & Tang, X. (2016). Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *38*(2), 295–307.

Eigen, D., Krishnan, D., & Fergus, R. (2013). Restoring an image taken through a window covered with dirt or rain. In: ICCV

Fan, Q., Chen, D., Yuan, L., Hua, G., Yu, N., & Chen, B. (2018). Decouple learning for parameterized image operators. In: ECCV, pp. 442–458

Fu, X., Huang, J., Ding, X., Liao, Y., & Paisley, J. (2017). Clearing the skies: A deep network architecture for single-image rain removal. *IEEE Transactions on Image Processing*, *26*(6), 2944–2956.

Fu, X., Huang, J., Zeng, D., Huang, Y., Ding, X., & Paisley, J. (2017). Removing rain from single images via a deep detail network. In: CVPR

Fu, X., Liang, B., Huang, Y., Ding, X., & Paisley, J. (2019). Lightweight pyramid networks for image deraining. *IEEE Transactions on Neural Networks and Learning Systems*,. https://doi.org/10.1109/TNNLS.2019.2926481.

Garg, K., & Nayar, S.K. (2004). Detection and removal of rain from videos. In: CVPR

Garg, K., & Nayar, S.K. (2005) When does a camera see rain? In: ICCV

Garg, K., & Nayar, S. K. (2007). Vision and rain. *International Journal of Computer Vision*, *75*(1), 3–27.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014) Generative adversarial nets. In: NeurIPS, pp. 2672–2680

Gu, S., Meng, D., Zuo, W., & Zhang, L. (2017). Joint convolutional analysis and synthesis sparse representation for single image layer separation. In: ICCV

He, K., Sun, J., & Tang, X. (2013). Guided image filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(6), 1397–1409.

He, K., Zhang, X., Ren, S., & Sun, J. (2016) Deep residual learning for image recognition. In: CVPR

Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. In: CVPR

Hu, X., Fu, C.W., Zhu, L., & Heng, P.A. (2019). Depth-attentional features for single-image rain removal. In: CVPR

Huang, D. A., Kang, L. W., Wang, Y. C. F., & Lin, C. W. (2014). Self-learning based image decomposition with applications to single image denoising. *IEEE Transactions on Multimedia*, *16*(1), 83–93.

Jiang, K., Wang, Z., Yi, P., Chen, C., Huang, B., Luo, Y., Ma, J., & Jiang, J. (2020). Multi-scale progressive fusion network for single image deraining. In: CVPR

Jiang, T.X., Huang, T.Z., Zhao, X.L., Deng, L.J., & Wang, Y. (2017). A novel tensor-based video rain streaks removal approach via utilizing discriminatively intrinsic priors. In: CVPR

Jiang, T. X., Huang, T. Z., Zhao, X. L., Deng, L. J., & Wang, Y. (2018). Fastderain: A novel video rain streak removal method using directional gradient priors. *IEEE Transactions on Image Processing*, *28*(4), 2089–2102.

Johnson, J., Alahi, A., & Fei-Fei, L. (2016). Perceptual losses for real-time style transfer and super-resolution. In: ECCV, pp. 694–711

Johnson, J., Gupta, A., & Fei-Fei, L. (2018). Image generation from scene graphs. In: CVPR

Kang, L. W., Lin, C. W., & Fu, Y. H. (2012). Automatic single image-based rain streaks removal via image decomposition. *IEEE Transactions on Image Processing*, *21*(4), 1742–1755.

Kim, J.H., Lee, C., Sim, J.Y., & Kim, C.S. (2013). Single-image deraining using an adaptive nonlocal means filter. In: IEEE ICIP

Kim, J. H., Sim, J. Y., & Kim, C. S. (2015). Video deraining and desnowing using temporal correlation and low-rank matrix completion. *IEEE Transactions on Image Processing*, *24*(9), 2658–2670.

Kingma, D.P., & Ba, J. (2014). Adam: A method for stochastic optimization. In: ICLR

Kipf, T.N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In: ICLR

Krizhevsky, A., Sutskever, I., & Hinton, G.E. (2012). ImageNet classification with deep convolutional neural networks. In: NeurIPS

Li, G., He, X., Zhang, W., Chang, H., Dong, L., & Lin, L. (2018). Non-locally enhanced encoder-decoder network for single image de-raining. In: ACM MM

Li, J., Ma, A. J., & Yuen, P. C. (2018). Semi-supervised region metric learning for person re-identification. *International Journal of Computer Vision*, *126*(8), 855–874.

Li, L., Pan, J., Lai, W. S., Gao, C., Sang, N., & Yang, M. H. (2019). Blind image deblurring via deep discriminative priors. *International Journal of Computer Vision*, *127*(8), 1025–1043.

Li, M., Xie, Q., Zhao, Q., Wei, W., Gu, S., Tao, J., & Meng, D. (2018). Video rain streak removal by multiscale convolutional sparse coding. In: CVPR

Li, R., Cheong, L.F., & Tan, R.T. (2019). Heavy rain image restoration: Integrating physics model and conditional adversarial learning. In: CVPR

Li, R., Tan, R.T., Cheong, L.F., Aviles-Rivero, A.I., Fan, Q., & Schonlieb, C.B. (2019). Rainflow: Optical flow under rain streaks and rain veiling effect. In: ICCV

Li, S., Araujo, I.B., Ren, W., Wang, Z., Tokuda, E.K., Junior, R.H., Cesar-Junior, R., Zhang, J., Guo, X., & Cao, X. (2019). Single image deraining: A comprehensive benchmark analysis. In: CVPR

Li, X., Wu, J., Lin, Z., Liu, H., & Zha, H. (2018). Recurrent squeeze-and-excitation context aggregation net for single image deraining. In: ECCV

Li, Y., Ouyang, W., Zhou, B., Shi, J., Zhang, C., & Wang, X. (2018). Factorizable net: an efficient subgraph-based framework for scene graph generation. In: ECCV

Li, Y., Tan, R.T., Guo, X., Lu, J., & Brown, M.S. (2016). Rain streak removal using layer priors. In: CVPR

Lin, M., Chen, Q., & Yan, S. (2014). Network in network. In: ICLR

Liu, J., Yang, W., Yang, S., & Guo, Z. (2018). D3R-Net: Dynamic routing residue recurrent network for video rain removal. *IEEE Transactions on Image Processing*, *28*(2), 699–712.

Liu, J., Yang, W., Yang, S., & Guo, Z. (2018). Erase or fill? deep joint recurrent rain removal and reconstruction in videos. In: CVPR

Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., et al. (2019). Deep learning for generic object detection: A survey. *International Journal of Computer Vision*,. https://doi.org/10.1007/s11263-019-01247-4.

Luo, Y., Xu, Y., & Ji, H. (2015). Removing rain from a single image via discriminative sparse coding. In: ICCV

Mordan, T., Thome, N., Henaff, G., & Cord, M. (2019). End-to-end learning of latent deformable part-based representations for object detection. *International Journal of Computer Vision*, *127*(11–12), 1659–1679.

Narasimhan, S. G., & Nayar, S. K. (2002). Vision and the atmosphere. *International Journal of Computer Vision*, *48*(3), 233–254.

Qi, X., Liao, R., Jia, J., Fidler, S., & Urtasun, R. (2017). 3d graph neural networks for rgbd semantic segmentation. In: ICCV

Qian, R., Tan, R.T., Yang, W., Su, J., & Liu, J. (2018). Attentive generative adversarial network for raindrop removal from a single image. In: CVPR

Ren, D., Zuo, W., Hu, Q., Zhu, P., & Meng, D. (2019). Progressive image deraining networks: A better and simpler baseline. In: CVPR

Ren, W., Pan, J., Zhang, H., Cao, X., & Yang, M. H. (2019). Single image dehazing via multi-scale convolutional neural networks with holistic edges. *International Journal of Computer Vision*,. https://doi.org/10.1007/s11263-019-01235-8.

Ren, W., Tian, J., Han, Z., Chan, A., & Tang, Y. (2017). Video desnowing and deraining based on matrix decomposition. In: ICCV

Romano, Y., & Elad, M. (2015). Boosting of image denoising algorithms. *SIAM Journal on Imaging Sciences*, *8*(2), 1187–1219.

Sakaridis, C., Dai, D., & Van Gool, L. (2018). Semantic foggy scene understanding with synthetic data. *International Journal of Computer Vision*, *126*(9), 973–992.

Santhaseelan, V., & Asari, V. K. (2015). Utilizing local phase information to remove rain from video. *International Journal of Computer Vision*, *112*(1), 71–89.

Shao, Y., Li, L., Ren, W., Gao, C., & Sang, N. (2020). Domain adaptation for image dehazing. In: CVPR

Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A.A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. In: AAAI

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. In: CVPR

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In: CVPR

Tai, Y., Yang, J., & Liu, X. (2017). Image super-resolution via deep recursive residual network. In: CVPR

Wang, C., Li, Z., Wu, J., Fan, H., Xiao, G., & Zhang, H. (2020). Deep residual haze network for image dehazing and deraining. *IEEE Access*, *8*, 9488–9500.

Wang, G., Sun, C., & Sowmya, A. (2019). Erl-net: Entangled representation learning for single image de-raining. In: ICCV

Wang, H., Li, M., Wu, Y., Zhao, Q., & Meng, D. (2020). A survey on rain removal from video and single image. *International Journal of Machine Learning and Cybernetics*,. https://doi.org/10.1007/s13042-020-01061-2.

Wang, H., Xie, Q., Zhao, Q., & Meng, D. (2020). A model-driven deep neural network for single image rain removal. In: CVPR

Wang, H., Zhu, X., Gong, S., & Xiang, T. (2018). Person re-identification in identity regression space. *International Journal of Computer Vision*, *126*(12), 1288–1310.

Wang, T., Yang, X., Xu, K., Chen, S., Zhang, Q., & Lau, R.W. (2019). Spatial attentive single-image deraining with a high quality real rain dataset. In: CVPR

Wang, X., Girshick, R., Gupta, A., & He, K. (2018). Non-local neural networks. In: CVPR

Wang, Y., Liu, S., Chen, C., & Zeng, B. (2017). A hierarchical approach for rain or snow removing in a single color image. *IEEE Transactions on Image Processing*, *26*(8), 3936–3950.

Wei, W., Meng, D., Zhao, Q., Wu, C., & Xu, Z. (2019). Semi-supervised transfer learning for image rain removal. In: CVPR

Wei, W., Yi, L., Xie, Q., Zhao, Q., Meng, D., & Xu, Z. (2017). Should we encode rain streaks in video as deterministic or stochastic? In: ICCV

Wojna, Z., Ferrari, V., Guadarrama, S., Silberman, N., Chen, L. C., Fathi, A., et al. (2019). The devil is in the decoder: Classification, regression and gans. *International Journal of Computer Vision*, *127*(11–12), 1694–1706.

Xingjian, S., Chen, Z., Wang, H., Yeung, D.Y., Wong, W.K., & Woo, W.c. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In: NeurIPS

Yang, W., Liu, J., Yang, S., & Guo, Z. (2019). Scale-free single image deraining via visibility-enhanced recurrent wavelet learning. *IEEE Transactions on Image Processing*, *28*(6), 2948–2961.

Yang, W., Tan, R.T., Feng, J., Liu, J., Guo, Z., & Yan, S. (2017). Deep joint rain detection and removal from a single image. In: CVPR

Yang, W., Tan, R. T., Feng, J., Liu, J., Yan, S., & Guo, Z. (2019). Joint rain detection and removal from a single image with contextualized deep networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(6), 1377–1393.

Yang, W., Tan, R. T., Wang, S., Fang, Y., & Liu, J. (2020). Single image deraining: From model-based to data-driven and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*,. https://doi.org/10.1109/TPAMI.2020.2995190.

Yasarla, R., & Patel, V.M. (2019). Uncertainty guided multi-scale residual learning using a cycle spinning cnn for single image de-raining. In: CVPR

Yasarla, R., & Patel, V. M. (2020). Confidence measure guided single image de-raining. *IEEE Transactions on Image Processing*, *29*, 4544–4555.

Yim, C., & Bovik, A. C. (2010). Quality assessment of deblocked images. *IEEE Transactions on Image Processing*, *20*(1), 88–98.

Yu, F., & Koltun, V. (2016). Multi-scale context aggregation by dilated convolutions. In: ICLR

Zhang, H., & Patel, V. M. (2016). Sparse representation-based open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(8), 1690–1696.

Zhang, H., & Patel, V.M. (2017). Convolutional sparse and low-rank coding-based rain streak removal. In: IEEE Winter Conference on Applications of Computer Vision

Zhang, H., & Patel, V.M. (2018). Densely connected pyramid dehazing network. In: CVPR

Zhang, H., & Patel, V.M. (2018). Density-aware single image de-raining using a multi-stream dense network. In: CVPR

Zhang, H., Riggan, B. S., Hu, S., Short, N. J., & Patel, V. M. (2019). Synthesis of high-quality visible faces from polarimetric thermal faces using generative adversarial networks. *International Journal of Computer Vision*, *127*(6–7), 845–862.

Zhang, H., Sindagi, V., & Patel, V. M. (2019). Image de-raining using a conditional generative adversarial network. *IEEE Transactions on Circuits and Systems for Video Technology*,. https://doi.org/10.1109/TCSVT.2019.2920407.

Zhang, K., Zuo, W., Chen, Y., Meng, D., & Zhang, L. (2017). Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Transactions on Image Processing*, *26*(7), 3142–3155.

Zhang, T., Ghanem, B., Liu, S., & Ahuja, N. (2013). Robust visual tracking via structured multi-task sparse learning. *International Journal of Computer Vision*, *101*(2), 367–383.

Zhang, X., Zhou, X., Lin, M., & Sun, J. (2018). Shufflenet: an extremely efficient convolutional neural network for mobile devices. In: CVPR

Zheng, X., Liao, Y., Guo, W., Fu, X., & Ding, X. (2013). Single-image-based rain and snow removal using multi-guided filter. In: International Conference on Neural Information Processing

Zhu, L., Fu, C.W., Lischinski, D., & Heng, P.A. (2017). Joint bi-layer optimization for single-image rain streak removal. In: ICCV