Transformer-Based Multi-Branch Single Image Deraining Method Postprint

Authors: Tan Fuxiang, Qian Yurong, Kong Yuting, Zhang Hao, Zhou Daxin, Fan Yingying, Chen Long

Date: 2022-04-07T15:01:56+00:00

Abstract

Rain streaks severely degrade the quality of captured images, affecting subsequent computer vision tasks. To improve the quality of rainy images, we propose a Transformer-based single image deraining algorithm. First, the algorithm obtains a large receptive field through a transformer with a windowing mechanism, thereby acquiring contextual information of rain streak features and enhancing the model' s capability to extract rain streak features; second, the algorithm extracts and fuses different types and levels of features through a multi-branch module, improving the model' s representation capability for complex rain streak information; finally, shallow and deep features are fused through residual connections to complement the missing detail information in deep features and enhance network expressiveness. Experimental results on the public datasets Rain100L, Rain100H and the private dataset Rain3000 demonstrate that compared with existing algorithms, the proposed method can more effectively remove rain streaks while better recovering the lost background texture information in images. The Peak Signal-to-Noise Ratio and Structural Similarity (PSNR/SSIM) reach 38.33/0.9855, 28.42/0.9000, and 34.51/0.9643, respectively.

Full Text

Abstract

Rain streaks can seriously degrade the quality of captured images and affect subsequent computer vision tasks. To improve the quality of rainy images, this paper proposes a single-image deraining algorithm based on Transformer. First, the algorithm obtains a wide receptive field through a Transformer with a window mechanism, then acquires contextual information of rain streak features to enhance the model' s ability to extract rain streak features. Second, the algorithm employs a multi-branch module to extract and fuse features of different types and levels, improving the model' s capacity to characterize complex rain streak information. Finally, residual connections are used to combine shallow and deep features, completing the missing detail information lost in deep features and enhancing the network' s expressive capability. Experimental results on the public datasets Rain100L, Rain100H, and the proprietary dataset Rain3000 demonstrate that the proposed method outperforms existing algorithms, more effectively removing rain streaks while better recovering lost background texture information in images. The Peak Signal-to-Noise Ratio and Structural Similarity Index (PSNR/SSIM) reach 38.33/0.9855, 28.42/0.9000, and 34.51/0.9643 respectively.

Keywords: single image deraining; multi-branch; Transformer; feature fusion

0 Introduction

Rainy weather, as a common condition, reduces the quality of captured images or videos and limits the application scenarios of computer vision tasks such as image classification, object detection, and image segmentation. Compared with video, single images lack temporal information, making single-image deraining more challenging.

The single-image deraining task primarily aims to recover lost background information based on rain streaks and their surrounding pixel information. Methods can be broadly categorized into traditional approaches and deep learning methods [1,2]. Traditional methods design models based on prior knowledge of rain streaks. Chen et al. [3] constructed a low-rank representation method for rain streak removal based on the similarity of rain streak geometric dimensions. Li et al. [4] approached the problem from the sparsity of rain streak features, using a sparse discriminative dictionary for deraining. Li et al. [5] proposed a Gaussian mixture model for similar patch completion to achieve single-image deraining. Kang et al. [6] first decomposed images into high and low frequencies, then applied sparse coding to high-frequency information for rain streak removal.

Although these methods achieve certain effects, they suffer from insufficient deraining or over-deraining in areas with dense, complex rain streaks and difficult-to-identify backgrounds. Deep learning methods based on Convolutional Neural Networks (CNN) possess powerful feature representation capabilities and can effectively learn the nonlinear mapping from rainy to rain-free images. Fu et al. [7] proposed the DerainNet model, which first applied CNN to single-image deraining by separating input images into high-frequency detail layers and low-frequency base layers, training the deraining network on the high-frequency layer and performing image enhancement on the low-frequency layer. Du et al. [8] proposed a conditional variational single-image deraining network based on the observation that rain streaks differ across spatial positions and channels. Zhang et al. [9] also considered density and constructed a multi-stream density estimator for adaptive image deraining. He et al. [10] jointly considered rain streak density and drop size, proposing a multi-scale rain streak density estimation module to guide network deraining. Jiang et al. [11] further studied the effectiveness of multi-scale models for deraining tasks, proposing a multi-scale progressive fusion model. Wang et al. [12] also noted the importance of multi-scale information for deraining tasks, proposing to learn features at different scales through scale aggregation modules and self-attention modules.

Current CNN-based methods achieve certain results, but CNNs' reliance on convolutional layers for indirect correlation between local pixels results in limited receptive fields. Most existing deraining models expand receptive fields by stacking convolutional kernels, which still yields limited receptive fields and weakens long-term feature dependencies, causing insufficient or excessive deraining.

The recently popular Transformer [13] possesses global computational characteristics that effectively obtain global attention maps and long-range feature dependencies, and has been applied to image classification [14], image segmentation [15,16], and other fields. However, Transformer’ s unrestricted computational approach is not suitable for single-image deraining tasks. Therefore, inspired by Swin Transformer [17], this paper combines Transformer, window mechanisms, and the characteristics of deraining tasks to design a Multi-branch Window Transformer Network for Single Image Deraining (MBWTNet). The model’ s feature extraction module offers the advantages of large receptive fields and strong rain streak feature expression capability, while the multi-branch module can adaptively learn different types and levels of rain streak features, enriching feature representation. Experimental results demonstrate that the proposed method can effectively remove complex rain streaks while better recovering background textures obscured by rain, achieving state-of-the-art deraining performance compared with current mainstream single-image deraining models.

1.1 Transformer Model Introduction

Transformer was proposed by Vaswani et al. [13] to address the inability of Recurrent Neural Networks to process in parallel for Natural Language Processing (NLP). The standard model, shown in Figure 1, consists of an Encoder (left) and a Decoder (right). In the Encoder stage, words in a sentence are first converted into word embeddings; then global self-attention feature maps are obtained through self-attention modules, residual connections, and layer normalization; finally, the Encoder output is obtained through feed-forward networks, residual connections, and layer normalization. Compared with the Encoder, the Decoder only adds one attention module and normalization layer to receive the Encoder output.

The Decoder input includes both the Encoder output and the output from the previous Decoder. The Decoder outputs probability distributions for corresponding positions. Since parallel inputs lack word positional relationships, Transformer uses positional encoding to preserve positional information.

Dosovitskiy et al. [14] proposed the VIT model, which was the first to directly use the Encoder part of Transformer for image classification, laying the foundation for subsequent vision Transformers. To adapt to Transformer input, VIT first splits images into non-overlapping patches, then stretches and embeds positional encoding to obtain one-dimensional vectors. Most subsequent vision Transformer research uses this approach for inputting images or feature maps. For output, VIT processes the Encoder' s output features through a classifier to obtain predictions. Both VIT and MBWTNet use relative position encoding, but MBWTNet differs by adding position encoding in the self-attention module.

1.2 Multi-Head Self-Attention Mechanism Introduction

Multi-head self-attention is a crucial component of Transformer, with its structure shown in Figure 2. First, the Encoder's input matrix is transformed through three weight matrices to obtain query matrix Q, key matrix K, and value matrix V. Then dot-product attention, as shown in Equation (1), calculates self-attention feature maps. Multi-head self-attention obtains multiple independent attention feature maps through multiple sets of transformation matrices and Equation (1). Finally, different attention feature maps are concatenated and fused through a fully connected layer to obtain multi-head attention maps.

In Equation (1), Q, K, V are two-dimensional matrices composed of vectors. The dot product of Q and the transpose of K yields a correlation matrix that records correlations between all vectors. Since Q and K are transformed from the same matrix, the correlation matrix describes correlations between input vectors. To avoid gradient vanishing caused by using a single coefficient to equivalently scale the correlation matrix, the activated correlation matrix is multiplied with V to obtain the global self-attention map. Multi-head self-attention is the primary source of Transformer' s global receptive field and long-range feature dependencies.

2 Multi-Branch Window Transformer Deraining Network

Transformer' s global computational approach provides the model with a global receptive field and long-term feature dependencies but creates certain feature redundancies, making it unsuitable for direct application to single-image deraining. This paper proposes a Multi-Branch Window Transformer Network (MBWTNet) that obtains a larger receptive field through window-restricted computation, fully leveraging the advantages of combining Transformer with multi-branch architecture and residual connections to extract features at different levels. As shown in Figure 3, MBWTNet consists of Transformer-based Feature Extraction Blocks (TFEB), Multi-Branch Fusion Modules (MBFM), and residual connections. In CNNs, residual connections address gradient vanishing in deep networks; in this paper, they focus more on the role of shallow features, i.e., completing missing texture information in deep features.

MBWTNet employs three sequentially arranged MBFM modules to extract and fuse features at different levels. The outputs of the first two MBFM modules are transmitted to deep network layers through residual connections, achieving full fusion of shallow and deep features. The output of the third MBFM module is fed into three parallel TFEBs, simultaneously extracting different types of features by increasing network depth and width. The network computation process is shown in Equation (2).

In Equation (2), $MBFM_t$ represents the multi-branch feature fusion module, $TFEB_i$ represents the feature extraction module, x_t are intermediate variables with the same size and channels, where x_0 is the input rainy image and x_{pre} is the predicted image.

2.1 Feature Extraction Module TFEB

Since convolution operations cannot fully capture pixel-wise feature correlations, some CNN-based methods produce unsatisfactory results in removing long, streaky rain patterns, exhibiting either insufficient or excessive deraining. Transformer's global computational approach can fully capture pixel correlations but creates feature redundancy. To address this issue, this paper constructs a feature extraction module based on Swin Transformer [17]. This module adopts Swin Transformer' s window shifting mechanism to limit computational load and enable inter-window information exchange. Swin Transformer causes certain spatial information loss; to address this, this paper proposes a Patch Splicing module to avoid spatial information loss. The feature extraction module is shown in the TFEB part of Figure 3. Feature maps sequentially pass through a patch partition module, linear embedding module, sliding window-based Transformer block (SWT), and patch splicing module to complete feature extraction.

The patch partition module first splits the input feature map of size $H \times W \times C$ into non-overlapping patches, as shown in Equation (3), where H and W are input image dimensions, and W_p and H_p are patch dimensions.

Since Transformer only accepts one-dimensional vectors, Patch Partition then converts patches into 1D vectors along the channel dimension. This vector can be viewed as a "token." Patch size is closely related to position encoding: larger patch sizes result in smaller position encoding dimensions. Other computer vision tasks like image classification focus more on semantic information—for example, the VIT [14] image classification model and image segmentation [16] models use patch sizes of 16×16, yielding position encoding dimensions of $\frac{H}{16} \times \frac{W}{16}$. However, image deraining tasks focus more on pixel and positional information. Therefore, this paper uses a patch size of 3×3, meaning that after Patch Partition, the vector dimension is $3 \times 3 \times C$.

Mapping to higher dimensions provides greater feature expression capability, benefiting the self-attention module in learning rain streak features. This paper uses a fully connected layer in the Linear Embedding module to map vector dimensions to Z, i.e., in Equation (5), the vector dimension is mapped to $(3 \times 3 \times C_2)$.

Standard Transformer offers global attention and long-range feature dependency advantages but suffers from high computational cost and deployment difficulties. Inspired by [17], this paper adopts a sliding window approach to restrict computation. The model structure is shown in Figure 4(a). Each sub-module consists of two LayerNorm (LN) layers, a window-based multi-head self-attention module (W-MSA), and an MLP, with the number of attention heads set to 3. Each patch contains 3×3 pixels, so a 7×7 window yields a receptive field of 21×21. Compared with convolutional layers, window-based Transformer can obtain larger receptive fields and more fully extract features of different sizes within windows. Since window boundaries lack sufficient texture information for feature extraction, inspired by Swin Transformer [17], the second sub-module of this block adopts a shifted multi-head attention module (SW-MSA), where window positions differ from the first, as shown in Figure 4(b). The sliding window mechanism enables boundary pixel information to be within the same window, completing boundary rain streak feature learning and inter-window information exchange. The sliding window-based Transformer computation process is shown in Equation (6).

In Equation (6), $Attention(Q, K, V)$ represents the multi-head attention output, and MLP represents the fully connected output. Unlike the multi-head attention module in standard Transformer, W-MSA uses the attention module shown in Equation (7).

In Equation (7), $Q, K \in \mathbb{R}^{M \times d}$ represent Query and Key, $V \in \mathbb{R}^{M \times d}$ represents Value, $M = 49$ is the number of patches involved in computation (for a 7×7 window), $d = 32$ is the dimension of Query, Key, or Value, and $B \in \mathbb{R}^{M \times M}$ is position encoding. W-MSA reduces computational load and avoids computing redundant features by limiting the number of patches participating in single self-attention computation.

Consistent input and output dimensions are necessary for feature extraction modules to be directly stacked and for fusing features at different levels. The patch splicing module first compresses the high-dimensional vector $\hat{Z}_l$ from Transformer computation to dimension Z through a fully connected layer, considering that the patch splicing module’ s input is a high-dimensional vector that doesn’ t meet patch construction requirements, and that fully connected layers can adaptively preserve important features while suppressing secondary ones. Then, $reshape_b$ converts vectors to $3 \times 3 \times C$ along the channel dimension. Finally, these patches are spliced into a feature map F_{out} with dimensions $H \times W \times C$, i.e., $F_{out} = F_{splic}(F_{rs})$. The above computation process is shown in Equation (8), where F_{linear}, $F_{reshape}$, and F_{splic} represent linear transformation, reshaping, and splicing operations, respectively.

2.2 Multi-Branch Feature Fusion Module

Rainy images contain different types of features such as rain streak size and shape, while background images contain features at different levels. Multi-head self-attention uses different network initializations to learn to extract and fuse different types of features, but this method cannot learn to extract and fuse features at different levels. To better meet the diversity requirements of features in deraining tasks, this paper investigates fusing multiple TFEB modules to improve model performance. Therefore, three multi-branch structures are designed and discussed, as shown in Figure 5.

Figure 5(a) shows a homogeneous multi-branch structure. Since each branch has different initial values and operates independently, they learn toward different feature subspaces during training. Therefore, more branches yield richer features and better deraining performance, but more branches do not necessarily mean a better network, as demonstrated by multi-branch experiments in Section 2.4. Figure 5(b) has the same parameter count as Figure 5(a), with each branch using the same structure, but only half the number of branches. Figure 5(c) has the same number of branches and parameter count as Figure 5(b), but each branch in Figure 5(c) uses a different structure. This structure' s computation process is shown in Equation (9).

In Equation (9), $TFEB_i$ represents the feature extraction module, f represents the multi-branch module input, and x is the output. Since each branch has different initial values and structures, the module can adaptively learn different types and levels of features, enriching the output features. Feature addition alone cannot fully fuse features from different branches; this paper adds a feature extraction module to achieve sufficient feature fusion.

2.3 Loss Function

Most existing image deraining models use Structural Similarity (SSIM) as the loss function, which has been proven effective by Ren et al. [18]. Although this loss function achieves good structural similarity, the generated images suffer from color distortion to some extent, resulting in lower Peak Signal-to-Noise Ratio (PSNR). In this work, the loss function' s mathematical expression is shown in Equation (10).

In Equation (10), O is the rainy image, B is the corresponding background image, $Loss_{L1}$ is the Sum of Absolute Differences (SAD), $Loss_{SSIM}$ (Structural Similarity) measures structural similarity between two images, with its negative commonly used as a loss function, as shown in Equation (10). Identity loss ($Loss_{ide}$) originates from CycleGAN [19] for constraining color differences in generated images; this paper uses it to constrain color differences in derained images, as shown in Equation (10) for $Loss_{ide}$, using the background image as model input and calculating identity loss between the generated result and label through $Loss_{L1}$. This paper minimizes the sum of three loss values to preserve image structural information while reducing color differences and improving deraining performance.

3.1 Datasets

The existing public datasets Rain100L and Rain100H [20] consist of 1,800 training pairs and 200 test pairs, created by adding rain streaks of different directions to the same background images. Rain100L is a relatively simple deraining dataset, with each image containing one rain direction. Rain100H is a more challenging dataset, with each image containing five rain directions. Rain100L and Rain100H provide datasets of two difficulty levels for network performance evaluation.

However, both datasets suffer from similar backgrounds between training and test sets [18], reducing model credibility. To address this, literature [18] removed 546 similar backgrounds to improve dataset quality, but this reduces sample size and is unfavorable for model generalization. This paper uses complete Rain100H and Rain100L datasets for training and testing to fairly compare with existing mainstream models. Additionally, a new dataset is proposed to improve model credibility. This dataset first randomly selects 100,000 images from the rich ImageNet; then randomly selects 1-4 types of rain streaks from Efficientderain [21] containing 825 rain streak images to add to the selected images; finally selects the first 3,400 synthesized image pairs from the 100,000 pairs as the dataset, with 3,000 training pairs and 400 test pairs. This dataset is named Rain3000, shown in Figure 6. Rain3000 contains both simple and relatively complex rain streaks, beneficial for fitting real rain image feature distributions. Dataset parameters are shown in Table 1.

To verify the effectiveness of the proposed dataset for network training, this paper first trains DCSFN [22], MPRnet [23], and PRENet [24] on datasets Rain3000, Rain100L, and Rain100H separately, then tests on real rainy images, with results shown in Figure 7. Training on Rain3000 enables the DCSFN model to effectively remove rain streaks of different shapes and sizes while preserving background information. MPRNet and PRENet can remove smaller, more natural rain streaks, demonstrating that Rain3000 better fits natural rain streak feature distributions.

3.2 Experimental Settings

The experimental environment uses a Tesla V100 16G GPU, 32GB RAM, and the PyTorch deep learning framework version 1.7.0. Batch size is set to 5, with a total of 500 training epochs. The initial learning rate is 10^{-4}, decaying to 10^{-5} and 10^{-6} at $\frac{3}{5}$ and $\frac{4}{5}$ of total iterations, respectively. This paper compares mainstream algorithms on datasets Rain100L, Rain100H, and Rain3000, and conducts ablation experiments on Rain3000.

This paper uses PSNR and SSIM, widely employed for evaluating deraining performance. SSIM measures content and texture similarity between two images, with a maximum value of 1—closer to 1 indicates higher similarity. PSNR is calculated based on pixel errors between two images; smaller errors yield larger values, indicating better similarity and deraining effect, while larger errors indicate poorer deraining performance.

3.3 Comparison Experiments

To verify MBWTNet' s superiority, this paper compares it with six state-of-the-art deraining methods on datasets Rain100L, Rain100H, and Rain3000:

a) **RESCAN**: Recurrent Squeeze-and-Excitation Context Aggregation Net method [25] (ECCV, 2018), using a recursive structure for multi-stage deraining, with each stage employing multiple context aggregation networks with SE modules and dilated convolutions, plus a memory unit to enhance inter-stage connections.

b) **GCANet**: Gated Context Aggregation Network [26] (WACV, 2019), proposing a context aggregation network using smooth dilated convolutions for dehazing to solve gridding artifacts caused by dilated convolutions, also applicable to image deraining.

c) **NLEDN**: Non-Locally Enhanced Encoder-Decoder Network [27] (ACM MM, 2018), proposing a non-local enhanced autoencoder using region-level non-local enhancement to improve long-range spatial context dependency capture, plus concatenating different-scale regions to enhance inter-region communication.

d) **PREnet**: Progressive Image Deraining Network method [24] (CVPR, 2019), proposing a multi-stage deraining baseline where each stage' s input is the concatenation of the original rainy image and the previous stage' s output, plus using LSTM to mine deep features across stages.

e) **DCSFN**: Deep Cross-Scale Fusion Network for Single Image Rain Removal [22] (ACM MM, 2020), proposing a cross-scale fusion method to learn internal feature relationships across scales, plus using dense connections to enhance long-range spatial dependencies.

f) **MPRnet**: Multi-Stage Progressive Image Restoration [23] (CVPR, 2021), proposing a multi-stage progressive restoration model to balance spatial details and contextual information during image restoration, with each stage supervised by labels, plus cross-stage aggregation of multi-scale features for inter-stage information exchange.

Table 2 shows the comparison results, with optimal values in bold and sub-optimal values underlined. Analysis shows that the proposed algorithm achieves the best performance on all three datasets. On the relatively simple Rain100L dataset, PSNR reaches 38.33 dB. On the challenging Rain100H dataset, PSNR reaches 28.42 dB. On the complex Rain3000 dataset, PSNR reaches 34.51 dB. The algorithm' s advantage is most significant on Rain100H, with PSNR and SSIM improvements of 4.44 dB and 0.1388 over the 2018 RESCAN network, and improvements of 1.87 dB and 0.0516 over the recent MPRNet model, and improvements of 0.66 dB and 0.014 over the second-best DCSFN. This indicates that compared with RESCAN and GCANet' s receptive fields obtained through dilated convolutions, MBWTNet has a broader receptive field and stronger feature representation capability; compared with PREnet, DCSFN, and MPRnet' s feature dependency enhancement methods, MBWTNet has stronger long-range feature dependencies and richer feature expression; compared with NLEDN' s multi-scale approach for inter-region information exchange, MBWTNet's sliding window method is more sufficient and direct.

Figure 8 shows visual deraining effects of each algorithm. RESCAN produces artifacts in derained images. NLEDN, CGAN, DCSFN, and MPRNet achieve good deraining effects but still leave some long rain streaks unremoved. PREnet removes rain streaks but also eliminates some texture details in the background. All six models have certain limitations in restoration quality, while MBWTNet effectively removes rain streaks and satisfactorily recovers texture details, further proving the proposed method' s superiority.

Model parameter count and inference time are important practicality metrics. Figure 9 shows each model' s parameter count and real-time performance. Although the proposed model has a larger parameter count, it achieves the fastest inference speed because the sliding window-based Transformer and fully connected layers use matrix operations, which are more efficient than step-by-step convolution. Figure 9 further demonstrates the proposed algorithm' s practicality.

3.4.1 Impact of Branch Number and Structure on Deraining Performance

To prove the impact of branch number and structure on model deraining performance, two controlled experiments are conducted on Rain3000. The first experiment verifies the effect of branch number on performance with identical branch structures. The model' s other parts remain unchanged, only replacing the multi-branch fusion module MBFM with the structure shown in Figure 5(a) and varying branch numbers to 1, 2, 3, 4, 5, and 6. Results are shown in Figure 10. Performance improves as branch number increases, but improvement becomes limited after exceeding 4 branches. This is because more branches with identical structures produce increasingly similar features, limiting further expression of feature diversity. To balance model scale and performance, this paper adopts a three-branch structure.

The second experiment verifies the effect of identical versus different branch structures on network performance with equal parameter counts. The model' s other parts remain unchanged, only modifying the MBFM structure to those shown in Figures 5(a), 5(b), and 5(c). Results are shown in Table 3, where MBFM-a corresponds to Figure 5(a), and so on.

Analysis of Table 3 shows that modules with different branch structures have richer feature expression capability when branch number and parameter count are equal. Combined with Figure 10 and Table 3, increasing identical branch numbers improves deraining performance, but this improvement is limited. Different branch structures can greatly adaptively capture correlations between different types and levels of features. Therefore, avoiding identical branch structures better facilitates learning the mapping from rainy to rain-free images.

3.4.2 Loss Function Properties

To verify the impact of different loss functions on model deraining performance, experiments are conducted on each loss function component. The model uses the same three-branch structure, evaluated by SSIM and PSNR. Table 4 shows network performance trained with different loss functions. Analysis shows that using $Loss_{ssim}$ alone yields the best SSIM metric. Using $Loss_{ssim}$ and $Loss_{ide}$ together achieves the same SSIM as $Loss_{ssim}$ alone while improving PSNR by 0.03 dB. Adding $Loss_{ide}$ degrades $Loss_{L1}$ performance because both loss functions are based on pixel differences, with $Loss_{ide}$ suppressing $Loss_{L1}$ performance. Additionally, $Loss_{L1}$ limits performance when only $Loss_{ssim}$ is used. Only when all three loss functions are included does the network achieve optimal performance, improving PSNR by 0.07 dB compared with $Loss_{ssim}$ alone. Since $Loss_{ssim}$ is calculated based on image content structure and lacks pixel texture information, $Loss_{L1}$ and $Loss_{ide}$ effectively supplement pixel information from different perspectives. This also demonstrates that identity loss, used to constrain color differences in image style transfer tasks, can similarly constrain color differences in image deraining tasks.

4 Conclusion

For image deraining, this paper proposes a Multi-Branch Window Transformer Deraining Network (MBWTNet). The network first combines Transformer with a window mechanism to build a feature extraction module with direct local pixel correlation, large receptive field, and no spatial information loss. Based on this module, a multi-branch module is constructed to extract and fuse different types and levels of features. Finally, feed-forward networks and skip connections build an end-to-end deraining network. Additionally, this paper proposes a new deraining dataset Rain3000 based on ImageNet, consisting of 3,000 training pairs and 400 test pairs, with rich background textures and diverse rain streak types. The proposed model is compared with several deep learning methods on public datasets Rain100L, Rain100H, and the private dataset Rain3000, achieving the best results in both visual quality and quantitative metrics. However, there are limitations, such as the lack of description of channel correlations in the algorithm. Future research will consider combining global channel attention and window channel attention to enhance the model' s ability to capture channel correlations.

References

[1] Zhang Yulong, Wang Qiang, Chen Mingkang, et al. Survey of intelligent rain removal algorithms for cloud-iot systems [J]. Computer Science, 2021, 48(12): 231-242.

[2] Chen Shuman, Chen Wei, Yin Zhong. Research status and prospect of single image rain removal algorithm [J]. Application Research of Computers, 2022, 39(1): 9-17.

[3] Chen Yilei, Hsu Chiouting. A generalized low-rank appearance model for spatio-temporally correlated rain streaks [C]// Proc of the IEEE International Conference on Computer Vision. 2013: 1968-1975.

[4] Li Yu, Tan R T, Guo Xiaojie, et al. Rain streak removal using layer priors [C]// Proc of the IEEE conference on computer vision and pattern recognition. 2016: 2736-2744.

[5] Li Siyuan, Ren Wenqi, Zhang Jiawan, et al. Single image rain removal via a deep decomposition–composition network [J]. Computer Vision and Image Understanding, 2019, 186: 48-57.

[6] Kang Liwei, Lin Chiawen, Fu Yuhsiang. Automatic single-image-based rain streaks removal via image decomposition [J]. IEEE Trans on image processing, 2011, 21(4): 1742-1755.

[7] Fu Xueyang, Huang Jiabin, Ding Xinghao, et al. Clearing the skies: A deep network architecture for single-image rain removal [J]. IEEE Trans on Image Processing, 2017, 26(6): 2944-2956.

[8] Du Yingjun, Xu Jun, Zhen Xiantong, et al. Conditional variational image deraining [J]. IEEE Trans on Image Processing, 2020, 29: 6288-6301.

[9] Zhang He, Patel V M. Density-aware single image de-raining using a multi-stream dense network [C]// Proc of the IEEE conference on computer vision and pattern recognition. 2018: 695-704.

[10] He Jingwei, Lei Yu, Xia Guisong, et al. Single image deraining with continuous rain density estimation [J]. IEEE Trans on Multimedia, 2021.

[11] Jiang Kui, Wang Zhongyuan, Yi Peng, et al. Multi-scale progressive fusion network for single image deraining [C]// Proc of the IEEE/CVF conference on computer vision and pattern recognition. 2020: 8346-8355.

[12] Wang Hong, Xie Qi, Zhao Qian, et al. A model-driven deep neural network for single image rain removal [C]// Proc of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 3103-3112.

[13] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need [C]// Advances in neural information processing systems. 2017: 5998-6008.

[14] Dosovitskiy A, Beyer L, Kolesnikov A, et al. An image is worth 16x16 words: Transformers for image recognition at scale [EB/OL]. (2021-06-03). https://arxiv.org/abs/2010.11929.

[15] Chen Jieneng, Lu Yongyi, Yu Qihang, et al. Transunet: Transformers make strong encoders for medical image segmentation [EB/OL]. (2021-02-08). https://arxiv.org/abs/2102.04306.

[16] Ranftl R, Bochkovskiy A, Koltun V. Vision Transformers for dense prediction [C]// Proc of the IEEE/CVF International Conference on Computer Vision. 2021: 12179-12188.

[17] Liu Ze, Lin Yutong, Cao Yue, et al. Swin Transformer: Hierarchical vision Transformer using shifted windows [EB/OL]. (2021-08-17). https://arxiv.org/abs/2103.14030.

[18] Ren Dongwei, Zuo Wangmeng, Hu Qinghua, et al. Progressive image deraining networks: A better and simpler baseline [C]// Proc of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019: 3937-3946.

[19] Zhu Junyan, Park Taesung, Isola Phillip, et al. Unpaired image-to-image translation using cycle-consistent adversarial networks [C]// Proc of the IEEE international conference on computer vision. 2017: 2223-2232.

[20] Yang Wenhan, Tan Robby T, Feng Jiashi, et al. Deep joint rain detection and removal from a single image [C]// Proc of the IEEE conference on computer vision and pattern recognition. 2017: 1357-1366.

[21] Guo Qing, Sun Jingyang, Felix Juefei-Xu, et al. Efficientderain: Learning pixel-wise dilation filtering for high-efficiency single-image deraining [EB/OL]. (2020-09-19). https://arxiv.org/abs/2009.09238.

[22] Wang Cong, Xing Xiaoying, Su Zhixun, et al. DCSFN: deep cross-scale fusion network for single image rain removal [C]// Proc of the 28th ACM international conference on multimedia. 2020: 1643-1651.

[23] Zamir S W, Arora A, Khan S, et al. Multi-stage progressive image restoration [C]// Proc of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 14821-14831.

[24] Ren Dongwei, Zuo Wangmeng, Hu Qinghua, et al. Progressive image deraining networks: A better and simpler baseline [C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019: 3937-3946.

[25] Li Xia, Wu Jianlong, Lin Zhouchen, et al. Recurrent squeeze-and-excitation context aggregation net for single image deraining [C]// Proc of the European Conference on Computer Vision (ECCV). 2018: 254-269.

[26] Chen Dongdong, He Mingming, Fan Qingnan, et al. Gated context aggregation network for image dehazing and deraining [C]// 2019 IEEE winter conference on applications of computer vision (WACV). IEEE, 2019: 1375-1383.

[27] Li Guanbin, He Xiang, Zhang Wei, et al. Non-locally enhanced encoder-decoder network for single image de-raining [C]// Proc of the 26th ACM international conference on Multimedia. 2018: 1056-1064.