Memory Oriented Transfer Learning for Semi-Supervised Image Deraining

A. Network Architecture

Table 1: Network architecture for the MOSS encoder.

Encoder	Settings	Input	Output
Conv	$3 \times 3, 16$	x	$s(x)$
ResBlock	$\begin{bmatrix} 1 \times 1, & 32 \\ 3 \times 3, & 32 \\ 3 \times 3, & 32 \end{bmatrix} \times 1$	$s(x)$	$f_{01}(x)$
AvgPool	2	$f_{01}(x)$	$p_0(x)$
ResBlock	$\begin{bmatrix} 3 \times 3, & 32 \\ 3 \times 3, & 32 \end{bmatrix} \times 3$	$p_0(x)$	$f_0(x)$
ResBlock	$\begin{bmatrix} 1 \times 1, & 48 \\ 3 \times 3, & 48 \\ 3 \times 3, & 48 \end{bmatrix} \times 1$	$f_0(x)$	$f_{11}(x)$
AvgPool	2	$f_{11}(x)$	$p_1(x)$
ResBlock	$\begin{bmatrix} 3 \times 3, & 48 \\ 3 \times 3, & 48 \end{bmatrix} \times 3$	$p_1(x)$	$f_1(x)$
ResBlock	$\begin{bmatrix} 1 \times 1, & 64 \\ 3 \times 3, & 64 \\ 3 \times 3, & 64 \end{bmatrix} \times 1$	$f_1(x)$	$f_{21}(x)$
AvgPool	2	$f_{21}(x)$	$p_2(x)$
ResBlock	$\begin{bmatrix} 3 \times 3, & 64 \\ 3 \times 3, & 64 \end{bmatrix} \times 3$	$p_2(x)$	$f_2(x)$
ResBlock	$\begin{bmatrix} 1 \times 1, & 80 \\ 3 \times 3, & 80 \\ 3 \times 3, & 80 \end{bmatrix} \times 1$	$f_2(x)$	$f_{31}(x)$
ResBlock	$\begin{bmatrix} 3 \times 3, & 80 \\ 3 \times 3, & 80 \end{bmatrix} \times 3$	$f_{31}(x)$	$f_3(x)$
ResBlock	$\begin{bmatrix} 1 \times 1, & 96 \\ 3 \times 3, & 96 \\ 3 \times 3, & 96 \end{bmatrix} \times 1$	$f_3(x)$	$f_{41}(x)$
ResBlock	$\begin{bmatrix} 3 \times 3, & 96 \\ 3 \times 3, & 96 \end{bmatrix} \times 3$	$f_{41}(x)$	$z(x)$

Table 2: Network architecture for the MOSS decoder.

Decoder	Settings	Input	Output
ResBlock	$\begin{bmatrix} 1 \times 1, & 80 \\ 3 \times 3, & 80 \\ 3 \times 3, & 80 \end{bmatrix} \times 1$	$\hat{z}(x)$	$g_{01}(x)$
ResBlock	$\begin{bmatrix} 3 \times 3, & 80 \\ 3 \times 3, & 80 \end{bmatrix} \times 3$	$g_{01}(x)$	$g_0(x)$
CBN	-	$f_3(x), g_0(x)$	$c_1(x)$
ResBlock	$\begin{bmatrix} 1 \times 1, & 64 \\ 3 \times 3, & 64 \\ 3 \times 3, & 64 \end{bmatrix} \times 1$	$c_1(x)$	$g_{11}(x)$
ResBlock	$\begin{bmatrix} 3 \times 3, & 64 \\ 3 \times 3, & 64 \end{bmatrix} \times 3$	$g_{11}(x)$	$g_1(x)$
CBN	-	$f_2(x), g_1(x)$	$c_2(x)$
Upsample	2	$c_2(x)$	$u_2(x)$
ResBlock	$\begin{bmatrix} 1 \times 1, & 48 \\ 3 \times 3, & 48 \\ 3 \times 3, & 48 \end{bmatrix} \times 1$	$u_2(x)$	$g_{21}(x)$
ResBlock	$\begin{bmatrix} 3 \times 3, & 48 \\ 3 \times 3, & 48 \end{bmatrix} \times 3$	$g_{21}(x)$	$g_2(x)$
CBN	-	$f_1(x), g_2(x)$	$c_3(x)$
Upsample	2	$c_3(x)$	$u_3(x)$
ResBlock	$\begin{bmatrix} 1 \times 1, & 32 \\ 3 \times 3, & 32 \\ 3 \times 3, & 32 \end{bmatrix} \times 1$	$u_3(x)$	$g_{31}(x)$
ResBlock	$\begin{bmatrix} 3 \times 3, & 32 \\ 3 \times 3, & 32 \end{bmatrix} \times 3$	$g_{31}(x)$	$g_3(x)$
CBN	-	$f_0(x), g_3(x)$	$c_4(x)$
Upsample	2	$c_4(x)$	$u_4(x)$
ResBlock	$\begin{bmatrix} 1 \times 1, & 16 \\ 3 \times 3, & 16 \\ 3 \times 3, & 16 \end{bmatrix} \times 1$	$u_4(x)$	$g_{41}(x)$
ResBlock	$\begin{bmatrix} 3 \times 3, & 16 \\ 3 \times 3, & 16 \end{bmatrix} \times 3$	$g_{41}(x)$	$g(x)$
Sub	-	$s(x), g(x)$	$g(x)$
Conv	$3 \times 3, 3$	$g(x)$	$g(x)$
Tanh	-	$g(x)$	y

Table 1 and Table 2 are the network architectures of the encoder and decoder of the proposed network, respectively. We use residual blocks as the basic component of our network and illustrate its design details in Fig. 1. We employ average pooling to reduce the size of features in the encoder, and up-sampling by bilinear interpolation to recover the size of features in the decoder. Besides, following BigGAN [1], we utilize Conditional Batch Normalization (CBN) to fuse the features $g_j(x)$ of the decoder with the corresponding features $f_{i_j}(x)$ of the encoder, where $f_{i_j}(x)$ and $g_j(x)$ are considered as the input and condition, respectively.

For the memory M, we set the number of memory items

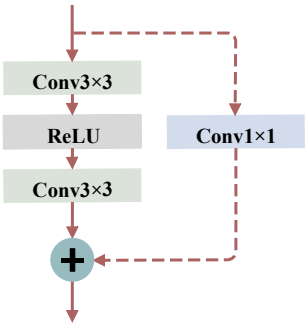


Figure 1: Illustration of the basic residual block. The dash line denotes that the convolution layer of 1×1 kernel-size is used only when the channel number of the input does not equal to that of the output.

as 512, and the dimension of each memory item as 96, just the same with the encoding $z(x)$ and the memory-based representation $\hat{z}(x)$.

B. Training Algorithm

We firstly pre-train the online network f_θ on labeled data using Algorithm 1 for the first 10 epochs. Then we train the proposed MOSS on both labeled and unlabeled data for semi-supervised deraining using Algorithm 2.

Algorithm 1 Pre-training MOSS for supervised deraining

$\theta \leftarrow$ Randomly initialize online network f_θ
while not converged **do**
  $(X_L, Y_L) \leftarrow$ Random batch from labeled dataset
  $Z_L \leftarrow \text{Enc}(X_L)$
  $M \leftarrow$ Self-supervised updating via Eq. (3)
  $\hat{Z}_L \leftarrow$ Soft-attentive reading via Eq. (5)
  $\tilde{Y}_L \leftarrow \text{Dec}(\hat{Z}_L, X_L)$
  $L_{SU} \leftarrow L_{SU}(\tilde{Y}_L, Y_L)$
  $\theta \leftarrow \theta - \eta \nabla_\theta L_{SU}$ ▷ Perform Adam updates for θ
end while

C. User Study

We conduct a user study to compare the performance of the proposed method with PreNet [3], DRD [2], SIRR [4] and Syn2Real [5]. All the methods are trained on the DDN-SIRR train set [4]. In the following, we first describe the real-world test set and evaluation details, and then report the results with an analysis.

Dataset and evaluation details. We collect 48 real-world rainy images searched from the internet as the real-world test set, ensuring that they have not been contained in the train set. A user-study database is built by performing rain removal on the test images using the above methods, leading to 48 groups of 6 images, each of which contains

Algorithm 2 Training MOSS for semi-supervised deraining

$\theta \leftarrow$ Pre-trained weights of the online network
$\xi \leftarrow \theta$ ▷ Perform deepcopy for the target network
while not converged **do**
  $(X_L, Y_L) \leftarrow$ Random batch from labeled dataset
  $X_U \leftarrow$ Random batch from unlabeled dataset
  $Y_P \leftarrow f_\xi(X_U)$ ▷ Produce pseudo-labels
  $(X_N, \hat{Y}) \leftarrow Augment(X_L, Y_L, X_U, Y_P)$ ▷ Produce noisy data
  $Z_L \leftarrow \text{Enc}(X_L)$
  $M \leftarrow$ Self-supervised updating via Eq. (3)
  $\hat{Z}_L \leftarrow$ Soft-attentive reading via Eq. (5)
  $\tilde{Y}_L \leftarrow \text{Dec}(\hat{Z}_L, X_L)$
  $Z_N \leftarrow \text{Enc}(X_N)$
  $M \leftarrow$ Self-supervised updates via Eq. (3)
  $\hat{Z}_N \leftarrow$ Soft-attentive reading via Eq. (5)
  $\tilde{Y}_N \leftarrow \text{Dec}(\hat{Z}_N, X_N)$
  $L_{total} \leftarrow \lambda_1 L_{SU}(\tilde{Y}_L, Y_L) + \lambda_2 L_{UN}(\tilde{Y}_N, \hat{Y}) + \lambda_3 L_{TV}(\tilde{Y}_N)$
  $\theta \leftarrow \theta - \eta \nabla_\theta L_{total}$ ▷ Perform Adam updates for θ
  $\xi \leftarrow \upsilon\, \xi + (1 - \upsilon)\, \theta$ ▷ Perform updates for ξ
end while

an original rainy image and 5 results of different methods. Given a group of images, we let the users select the best one in deraining performance. Specifically, we randomly sample 10 groups from the database and show them to an individual user. For fairness, we shuffle the orders of images in each group and make the methods anonymous, except for the original one fixed as reference. Note that we do not collect the privacy of the evaluators and all questionnaires are anonymous. We distribute the questionnaire to a wide range of online users without constraints, and finally obtain answers from a total of 404 human evaluators. The database and the original data of the questionnaire will be released.

Averaged selection percentage. Fig. 2 shows the averaged selection percentage for each method. As can be observed, our method achieves the best performance for real-world rain removal. Besides, our method and Syn2Real outperforms other methods with a large margin, which is consistent with the comparison results in the main text.

The visual results, including the selection percentages for individual images, are illustrated in Fig. 4, which further demonstrate that our method can obtain promising deraining results for real-world rainy images. Additionally, we provide one failure case in the rightmost column in Fig. 4. The rainy image of a running man with a dog is heavily damaged by rain of complex patterns. Even though, our method can recover cleaner background scene than other methods, according to most of human evaluators.

Rank-n scores. Furthermore, we compute the rank-n scores to gain an insight of deraining robustness. For each

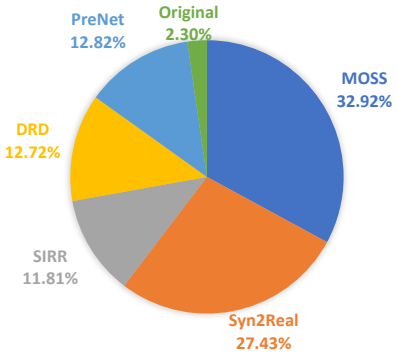
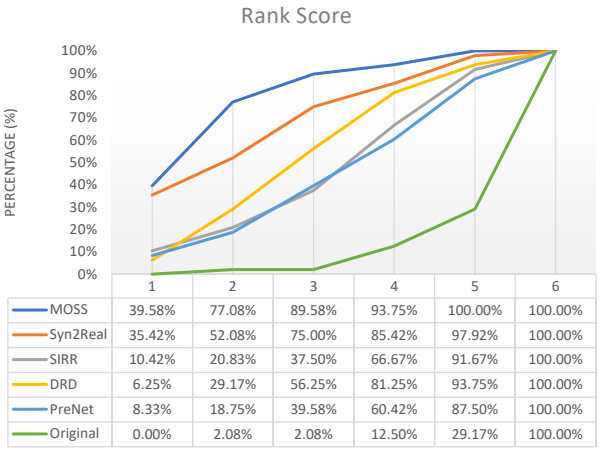


Figure 2: Averaged selection percentage of user study.



	1	2	3	4	5	6
MOSS	39.58%	77.08%	89.58%	93.75%	100.00%	100.00%
Syn2Real	35.42%	52.08%	75.00%	85.42%	97.92%	100.00%
SIRR	10.42%	20.83%	37.50%	66.67%	91.67%	100.00%
DRD	6.25%	29.17%	56.25%	81.25%	93.75%	100.00%
PreNet	8.33%	18.75%	39.58%	60.42%	87.50%	100.00%
Original	0.00%	2.08%	2.08%	12.50%	29.17%	100.00%

Figure 3: Rank-n scores of user study.

method, we calculate the percentage of images whose top-k selections contain it. The rank-n score R_i^n for the i-th method is defined as

$$R_i^n = \frac{1}{N}\sum_{j=1}^{N} \mathbb{1}(i \in F_{topk}(x_j, n)), \tag{1}$$

where x_j is the j-th image from the test database of N images (here $N = 48$), F_{topk} is the operation of top-k.

The rank-n scores are reported in Fig. 3. It can observed that our method consistently surpasses other methods. For example, 77.08% and 89.58% percentages of the recovered images by our method are considered as the top-2 and top-3 pleasing results, respectively. This demonstrates that the proposed method can achieve stronger robustness toward real-world image deraining compared with other state-of-the-art methods.

D. Ablation Study

As aforementioned in the main text, an ablation study is conducted and prove that each component of the proposed method is essential for semi-supervised deraining. In this section, we give a more detailed discussion about the roles of these components. For convenience, we report again the results of the ablation study in Table 3. Furthermore, we give a detailed discussion about the decay rates τ in Eq. (3) and υ in Eq. (7).

Table 3: Results of ablation study on DDN-SIRR.

Dataset	Metrics	Basic	w/o Memory	w/o Self-Training	w/o TV	w/o Skip-Connect	Ours
Dense	PSNR	19.99	22.21	20.99	22.68	22.00	22.91
	SSIM	0.835	0.870	0.860	0.876	0.858	0.883
Sparse	PSNR	25.74	26.82	25.83	27.58	26.80	27.78
	SSIM	0.881	0.906	0.890	0.908	0.900	0.912

Memory. The proposed memory aims to record prototypical patterns of rain degradations. As shown in Table 3, the performance decreases both for images of dense rain streaks and those of sparse rain streaks when removing the memory module from the proposed method. It proves that the memory module is helpful to improve deraining performance for various rain appearances.

Self-Training. The proposed self-training not only provides a way to transfer knowledge from supervised to unsupervised deraining, but also augments training data with more types of rain degradations. Table 3 shows that removing self-training results in heavy degradations in deraining performance. This implies that the proposed self-training is essential for our method, since it offers a strong supervision for unlabeled data and improves the deraining robustness through data augmentation.

Total Variation. The Total Variation regularizer term helps slightly smooth the deraining results of augmented noisy data. When removing it, there is a slight decrease in performance of deraining. The TV term is helpful for deraining, since it offers an additional unsupervised prior-based constraint on the deraining process of unlabeled data.

Skip Connection. As outlined in the network architectures of MOEDN, we employ a skip-connection between the first and the last convolution layers. It ensures that the major parts of MOEDN should focus on learning rain degradations rather than irrelevant background details. Therefore, the memory module can record only the rain-relevant features, which eases the memory task since rain degradations may have much less diversities than the background information. Table 3 empirically demonstrates that it is plausible to design such a skip connection.

Decay Rate τ. The proposed memory M is updated in a self-supervised way via Eq. (3). The key hyper-parameter in Eq. (3) is the decay rate τ that controls the updating speed of M. When $\tau = 0$, M is instantaneously updated to the second term in Eq. (3) at each step. When $\tau = 1$, M is constant without updating. We conduct experiments on DDN-SIRR with different values of the decay rate τ and report averaged PSNR and SSIM on the DDN-SIRR synthetic test set in table 4. As can be seen, there is a trade-off between

updating M quickly and updating M slowly. All values between 0 and 0.9999 lead to better performance than the ablation version without the memory module, which further proves that the proposed memory is beneficial to image deraining. Besides, utilizing a memory module without updating ($\tau = 1$), we obtain worse deraining results even than the version without it. It is because the random-initialized memory cannot reflect various statistics of rain degradations. We set τ to be 0.999 in our full version of MOSS since it achieves the best results of deraining.

Table 4: Results of different decay rates τ on DDN-SIRR.

Decay rate τ	PSNR	SSIM
0	25.06	0.876
0.9	25.24	0.896
0.99	25.01	0.894
0.999	25.35	0.898
0.9999	25.19	0.894
1	24.13	0.885
w/o	24.52	0.888

Decay Rate υ. As aforementioned, the self-training mechanism plays a significant role in the proposed method for semi-supervised deraining. The target network is updated with an exponential moving average of the online network by a decay rate υ. Similar to τ, when $\upsilon = 0$, the target network is instantaneously updated to the online network; when $\upsilon = 1$, the target network is fixed during training (note that the target network is initialized using the pre-trained weights by Algorithm 1). We also conduct experiments on DDN-SIRR with different values of the decay rate υ and reports the results in Table 5. It can be observed all values of υ can help improve deraining accuracy compared to the ablation version without self-training. This further implies that self-training is essential for the proposed semi-supervised deraining method. Besides, directly using the online network ($\upsilon = 0$) to produce pseudo-labels can also boost image deraining with a large margin. However, using a pre-trained network ($\upsilon = 1$) can only attain a marginal increase compared to the ablation one. This demonstrates that updating the target network according to the online network is important for the proposed self-training. We set υ to be 0.999 to achieve a better trade-off between updating the target network quickly and updating it slowly.

E. Additional Visual Results

In this section, we provide additional visual results on real-world rainy images and limited labeled data, as illustrated in Fig. 4, Fig. 5, Fig. 6, and Fig. 7. In addition, we provide visual results of the noised data via Eq.(9) in the main text in Fig. 8. It can be observed that the augmentation can yield various yet valid rain appearances.

Table 5: Results of different decay rates υ on DDN-SIRR.

Decay rate υ	PSNR	SSIM
0	25.17	0.887
0.9	24.93	0.890
0.99	25.28	0.892
0.999	25.35	0.898
0.9999	25.04	0.885
1	24.15	0.8884
w/o	23.41	0.875

References

[1] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2018. 1

[2] Sen Deng, Mingqiang Wei, Jun Wang, Yidan Feng, Luming Liang, Haoran Xie, Fu Lee Wang, and Meng Wang. Detail-recovery image deraining via context aggregation networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14560–14569, 2020. 2, 5, 6

[3] Dongwei Ren, Wangmeng Zuo, Qinghua Hu, Pengfei Zhu, and Deyu Meng. Progressive image deraining networks: a better and simpler baseline. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3937–3946, 2019. 2, 5, 6

[4] Wei Wei, Deyu Meng, Qian Zhao, Zongben Xu, and Ying Wu. Semi-supervised transfer learning for image rain removal. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3877–3886, 2019. 2, 5, 6

[5] Rajeev Yasarla, Vishwanath A Sindagi, and Vishal M Patel. Syn2real transfer learning for image deraining using gaussian processes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2726–2736, 2020. 2, 5, 6, 7, 8

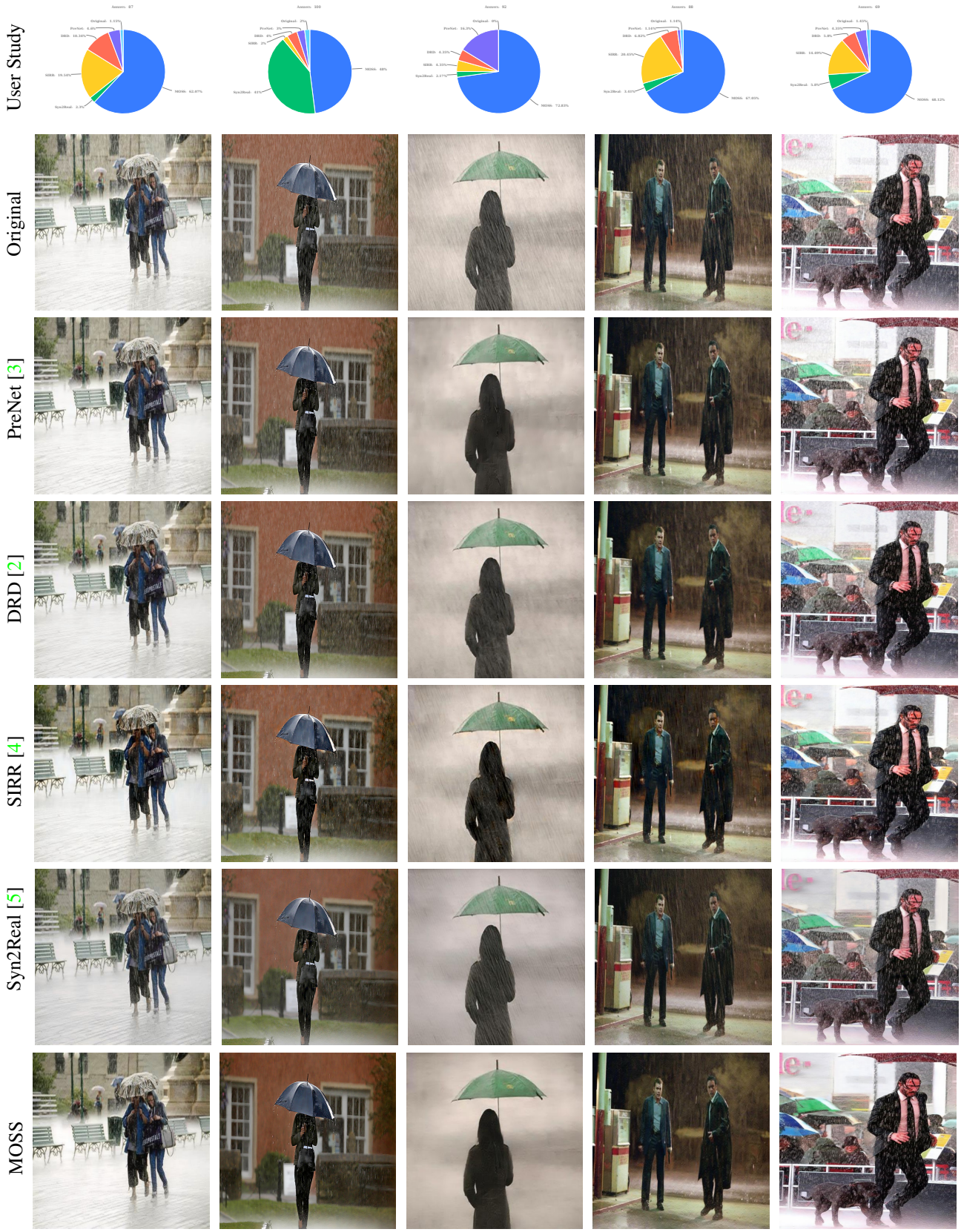


Figure 4: Visual results of the real-world rainy images for user study. The rightmost column shows a failure case for every method. Though, our method still achieves the best deraining performance according to the user study.

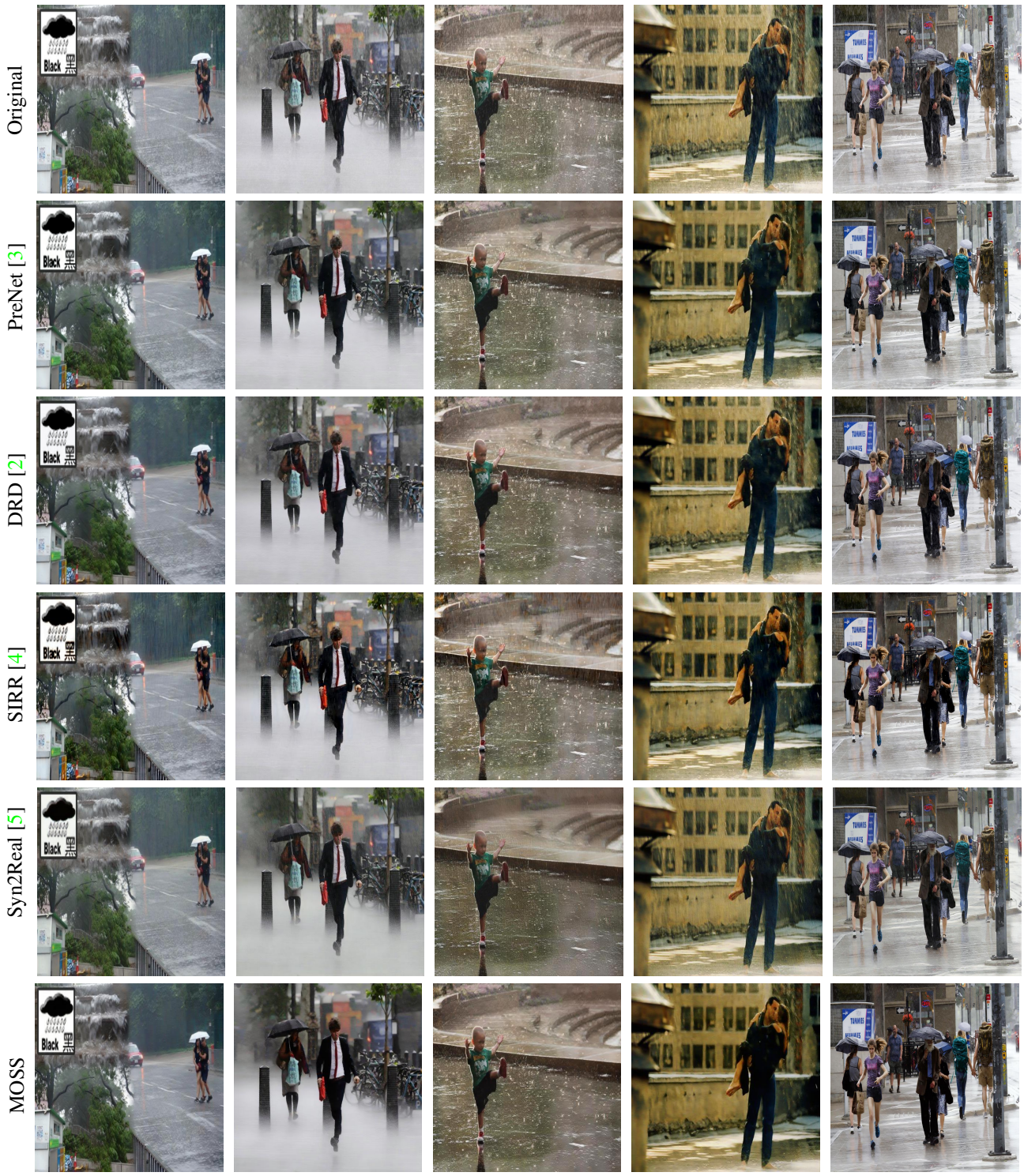


Figure 5: Visual results on the DDN-SIRR real-world test set.

Input Rainy Images

Syn2Real [5] using only labeled data

Syn2Real [5] using both labeled and unlabeled data

MOSS using only labeled data

MOSS using both labeled and unlabeled data

Ground-Truth images

Figure 6: Visual results on the Rain200H dataset with 10% labeled data.

Input Rainy Images

Syn2Real [5] using only labeled data

Syn2Real [5] using both labeled and unlabeled data

MOSS using only labeled data

MOSS using both labeled and unlabeled data

Ground-Truth images

Figure 7: Visual results on the Rain200H dataset with 40% labeled data.

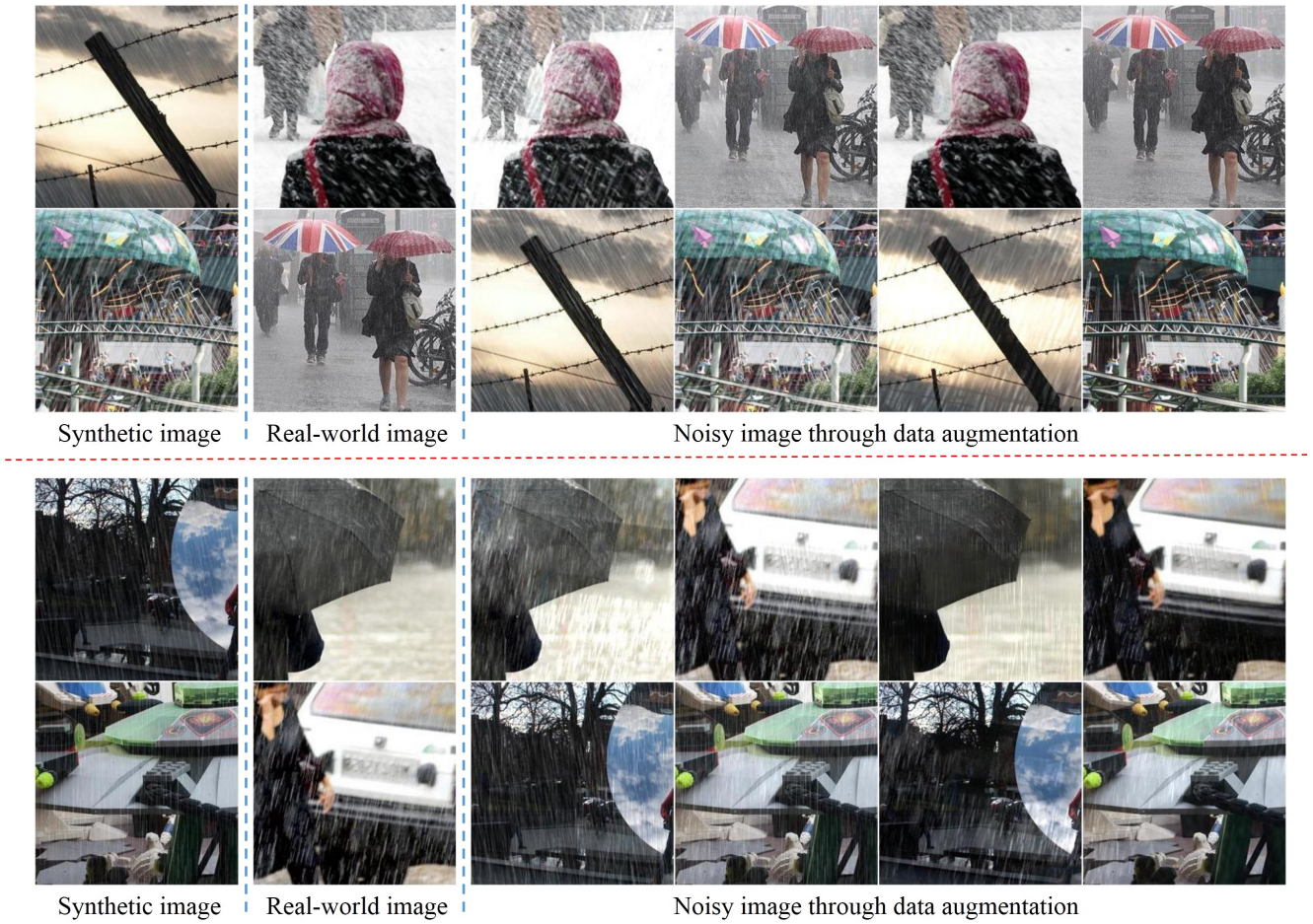


Figure 8: Example images of data augmentation.