"From Rain Generation to Rain Removal": Supplementary Material

Hong Wang[1,2,*], Zongsheng Yue[1,*], Qi Xie[1], Qian Zhao[1], Yefeng Zheng[2], Deyu Meng[3,1,†]
[1]Xi'an Jiaotong University, Xi'an, China
[2]Tencent Jarvis Lab, Shenzhen, China
[3]Macau University of Science and Technology, Macau, China
hongwang01@stu.xjtu.edu.cn zsyzam@gmail.com xie.qi@mail.xjtu.edu.cn
timmy.zhaoqian@gmail.com yefengzheng@tencent.com dymeng@mail.xjtu.edu.cn

Abstract

In this supplementary material, we provide more details on the deduction of the variational objective, and the network structures described in the main submission. Besides, we demonstrate more experimental results in rain generation and rain removal. In the end, we present ablation studies to further analyze our model.

1. More Details of Variational Objective

Here we provide a detailed derivation for variational objective in Section 3.2 of the main text. By introducing the variational approximate posterior $q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o})$, the logarithm of the rainy image distribution $p(\boldsymbol{o})$ can be expressed as:

$$\begin{aligned}
\log p(\boldsymbol{o}) &= \int\int q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o}) \log p(\boldsymbol{o}) \,\mathrm{d}\boldsymbol{z}\mathrm{d}\boldsymbol{b} \\
&= \int\int q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o}) \log \left[\frac{p_\theta(\boldsymbol{o}|\boldsymbol{z},\boldsymbol{b})\, p(\boldsymbol{z})\, p(\boldsymbol{b})}{p(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o})}\right] \mathrm{d}\boldsymbol{z}\mathrm{d}\boldsymbol{b} \\
&= \int\int q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o}) \log \left[\frac{p_\theta(\boldsymbol{o}|\boldsymbol{z},\boldsymbol{b})\, p(\boldsymbol{z})\, p(\boldsymbol{b})}{q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o})} \frac{q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o})}{p(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o})}\right] \mathrm{d}\boldsymbol{z}\mathrm{d}\boldsymbol{b} \\
&= \int\int q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o}) \log \left[\frac{p_\theta(\boldsymbol{o}|\boldsymbol{z},\boldsymbol{b})\, p(\boldsymbol{z})\, p(\boldsymbol{b})}{q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o})}\right] \mathrm{d}\boldsymbol{z}\mathrm{d}\boldsymbol{b} \\
&\quad + \int\int q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o}) \log \left[\frac{q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o})}{p(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o})}\right] \mathrm{d}\boldsymbol{z}\mathrm{d}\boldsymbol{b} \\
&= E_{q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o})}[\log p_\theta(\boldsymbol{o}|\boldsymbol{z},\boldsymbol{b}) p(\boldsymbol{z}) p(\boldsymbol{b}) - \log q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o})] \\
&\quad + D_{KL}\left[q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o}) \,||\, p(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o})\right].
\end{aligned} \tag{1}$$

Obviously, this is just the decomposition form given in Eq. (7) of the main text, as:

$$\log p(\boldsymbol{o}) = \mathcal{L}(\boldsymbol{z},\boldsymbol{b};\boldsymbol{o}) + D_{KL}\left[q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o}) \,||\, p(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o})\right], \tag{2}$$

where

$$\mathcal{L}(\boldsymbol{z},\boldsymbol{b};\boldsymbol{o}) = E_{q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o})}[\log p_\theta(\boldsymbol{o}|\boldsymbol{z},\boldsymbol{b}) p(\boldsymbol{z}) p(\boldsymbol{b}) - \log q(\boldsymbol{z},\boldsymbol{b}|\boldsymbol{o})]. \tag{3}$$

2. More Details on Network Architectures

As shown in the main text, the entire network architecture is constructed as Fig. 1, called variational rain generation network (VRGNet). It is noteworthy that we aim to propose such a variational inference framework toward rain generation without putting more emphasis on the careful design of every sub-network architecture. Specifically, each sub-network adopted in our experiment, is illustrated as:

BNet infers posterior parameters $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ from $\boldsymbol{o}$ and aims to restore the latent clean background $\boldsymbol{b}$. We select the latest baseline network–PReNet [9] due to its simplicity and fast training process. In specific, the adopted PReNet is composed of 6 [*Conv + ReLU +LSTM + ResBlocks+ Conv*] stages. The network parameters are inter-stage sharing. Besides, in each stage, the *ResBlocks* consists of 5 [*Conv+ReLU+Conv+ReLU+Skip connection*] units.

RNet helps infer the posterior parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ for latent variable $\boldsymbol{z}$, and it consists of 5 [*Conv+ReLU*] blocks and a [*Linear layer*] in turn.

Generator represents the mapping $G(\boldsymbol{z};\theta)$ for generating rain patches from extracted latent variables $\boldsymbol{z}$. Symmetrically, it contains a [*Linear layer*] and 5 [*Transpose Conv + ReLU*] blocks. For back propagation, we adopt the reparameterization trick as proposed in [6].

Discriminator aims to distinguish the training sample $\boldsymbol{o}$ from the generated $\widehat{\boldsymbol{o}}$, which helps the learning of $G(\boldsymbol{z};\theta)$. Similar to the settings of most discriminators [8, 13], the sub-network is composed of 4 [*Conv + LeakyReLU*] blocks and a [Conv layer], and the negative_slope is set as 0.1 in *LeakyReLU* operation. To stabilize the training process, we also introduce the spectral normalization [7] in the sub-network. Besides, motivated by [13], we add the attention mechanism on the last two convolution layers to capture the

†Corresponding author
*Equal contribution

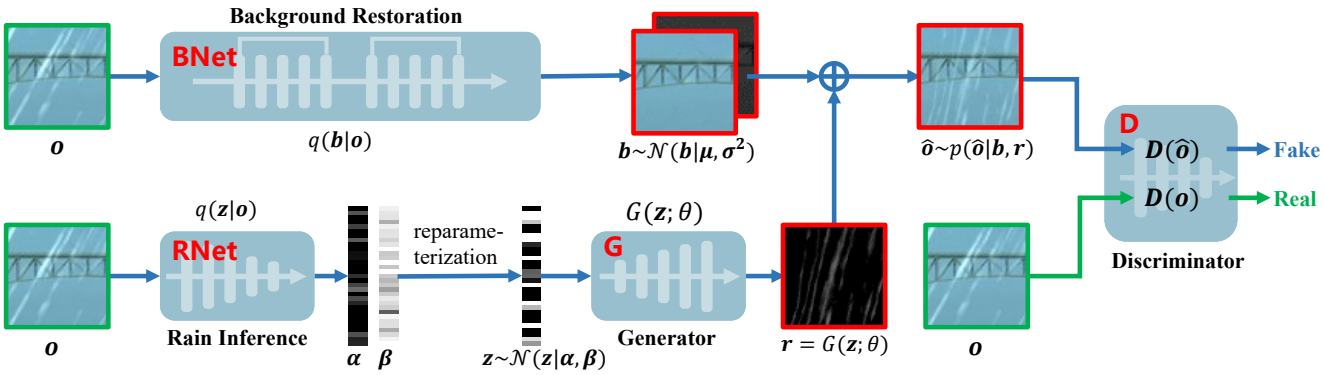

Figure 1. The flowchart of the proposed variational rain generation network (VRGNet).

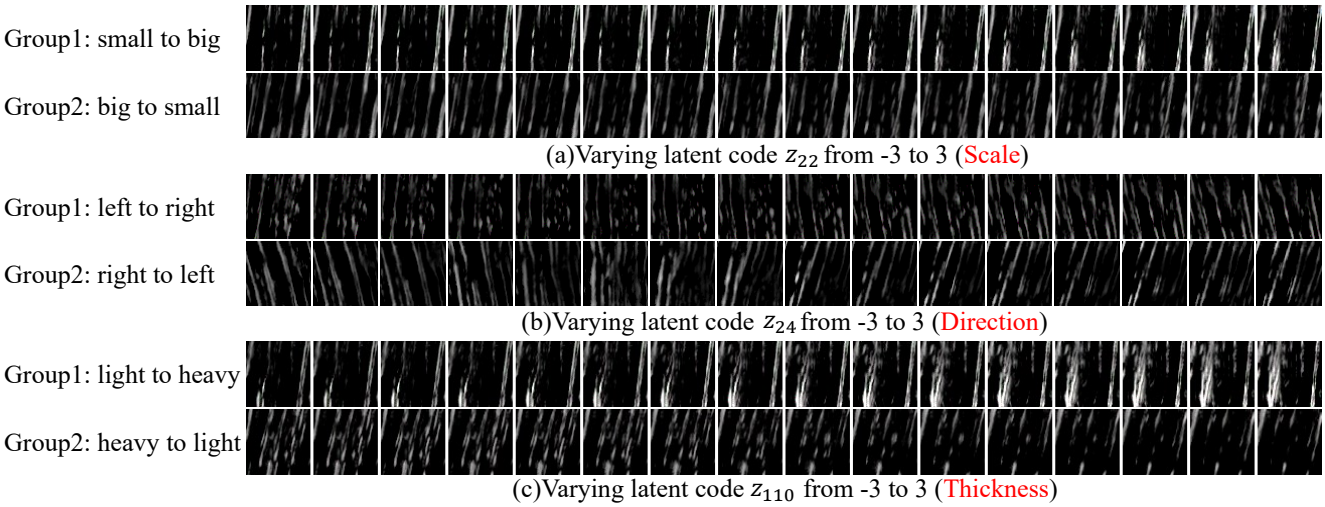

Figure 2. Manipulating latent code $\boldsymbol{z} \in \mathbb{R}^{128}$. Taking subfigure (a) as an example, we sample a random vector (latent code $\boldsymbol{z}$) from the normal distribution, and then only vary the latent element at the 22-th dimension of $\boldsymbol{z}$ from -3 to 3 with the interval as 0.4. Taking each varied vector $\boldsymbol{z}$ as the input of the generator G, the output $\boldsymbol{r}$ corresponds to each rain layer shown in (a), which demonstrates the scale property of rain. When we randomly sample two times from the normal distribution for the latent code $\boldsymbol{z}$ and repeat this experiment, the generated $\boldsymbol{r}$ are correspondingly displayed as two groups. (a)-(c) denote varying different latent elements and the learned latent variables physically represent scale, direction, and thickness, respectively.

global correlation in image.

Note that the number of blocks in these sub-networks, including *RNet*, *Generator*, and *Discriminator*, is set based on the patch size (height $\times$ width of rain patches) during the network training process. In our experiments, the size is set as the commonly-used 64 $\times$ 64 in current SOTAs for this task. If other size settings are required, the number of blocks needs to be correspondingly adjusted.

3. More Rain Generation Experiments

In this section, we provide more disentanglement and latent space interpolation experiments to validate that the proposed rain generator is rational and can finely capture the manifold of rain underlying its implicit distribution.

3.1. Disentanglement Experiments

Fig. 2 shows the resulted rain layers by manipulating the latent code $\boldsymbol{z}$ like the conventional disentanglement operations [1, 2, 5]. The learned generator is obtained by jointly training the proposed VRGNet based on Rain100L. From the figure, we can easily observe that these latent variables well deliver interpretable properties in generating rain layer, including direction, thickness, and scale. That is to say, the proposed VRGNet inclines to discover meaningful latent rain factors, which is finely in accordance with our modeling for rain layer by utilizing latent variables $\boldsymbol{z}$ to encode such physical structural factors.

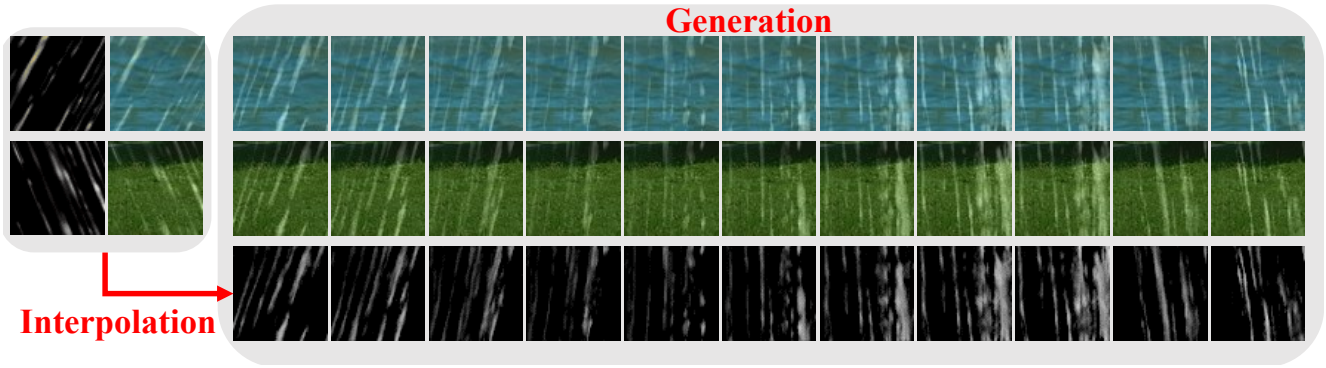

Figure 3. Interpolation. Left: two original rainy images from Rain100L and their rain layers. Right: generated rainy images (the first two rows) and synthetic rains (3rd row) obtained by linearly interpolating the latent codes of the two original rainy images.

Table 1. PSNR and SSIM (mean and standard deviation) of PReNet on the SPA-Data test set. **Baseline** denotes that training samples are all from SPA-Data (∼600K), and **GNet** means the augmented training where training samples consist of 1K real pairs randomly selected from ∼600K and different number of fake pairs. Specifically, the fake samples are generated by our generator that is jointly trained on ∼600K. **In each scene, the training pairs between Baseline and GNet keep the same**, and the result is computed over 5 random attempts. The case that **Baseline** with ∼600K has no randomness about samples.

# Real samples	1K	1.5K	2K	3K	4K	5K	6K	7K	∼600K
Baseline (PSNR), mean±std	39.41±0.24	39.70±0.21	**39.86**±0.20	39.96±0.20	40.05±0.19	40.04±0.18	40.00±0.18	40.06±0.15	40.16
# Samples (real+fake)	1K+0K	1K+0.5K	1K+1K	1K+2K	1K+3K	1K+4K	1K+5K	1K+6K	-
GNet (PSNR), mean±std	39.41±0.24	**39.71**±0.26	39.83±0.20	**40.25**±0.21	**40.24**±0.17	**40.53**±0.20	**40.68**±0.17	**40.70**±0.11	-

# Real samples	1K	1.5K	2K	3K	4K	5K	6K	7K	∼600K
Baseline (SSIM), mean±std	0.9787±8e-4	**0.9800**±7e-4	**0.9809**±6e-4	0.9813±7e-4	**0.9815**±8e-4	0.9814±5e-4	0.9815±6e-4	0.9815±5e-4	0.9816
# Samples (real+fake)	1K+0K	1K+0.5K	1K+1K	1K+2K	1K+3K	1K+4K	1K+5K	1K+6K	-
GNet (SSIM), mean±std	0.9787±8e-4	0.9796±6e-4	0.9795±5e-4	0.9813±8e-4	0.9814±4e-4	**0.9819**±5e-4	**0.9820**±4e-4	**0.9819**±4e-4	-

3.2. Latent Manifold Analysis

We conduct interpolation operations in the latent space to estimate the manifold continuity. Here the VRGNet is jointly trained based on Rain100L. Specifically, for a pair of rainy images selected from Rain100L shown at the left of Fig. 3, we first utilize the inference model *RNet* to obtain their latent codes $\boldsymbol{z}_a$ and $\boldsymbol{z}_b$, and then make linear interpolations between $\boldsymbol{z}_a$ and $\boldsymbol{z}_b$ with different weighting coefficients from 0 to 1. By inputting the weighted latent code $\boldsymbol{z}$ to the rain generator G, we thus synthesize different rain layers shown in the 3rd row at the right of Fig. 3. The first two rows are the generated rainy images by adding these synthetic rains on different backgrounds restored by *BNet*. It is easy to observe that our rain generator has continuity in the latent space in changing the direction of rain streaks and it indeed has a fine capability to generate diverse rain types instead of simply memorizing the patterns in input images.

Note that in order to better observe the variation of rain streaks, in the interpolation experiments as shown in Fig. 1 of the main text, we have not displayed input rainy images that are used to obtain $\boldsymbol{z}$, but provided the corresponding rain layers which are easily obtained by subtracting backgrounds from the rainy images in paired testing dataset.

For better visual effect, we have conducted several groups of interpolation experiments and make each group as a file with the format '.gif' as provided in the submitted supplementary material compressed package. In these experiments, we show the variation of rain streaks in directions, thicknesses, and diversities. In each group experiment, the first and the last frames are the rain layers corresponding to one pair of input rainy images from the existing dataset, and between these two frames are the interpolated results.

3.3. More Small Sample Experimental Results

In this section, we also provide the SSIM results for the small sample experiments in Section 5.3 of the main text, as listed in Table 1. From it, we can observe that with the increase of ratio N_f from 0 to 6, the average PSNR and SSIM under augmented training are superior or at least comparable to the performance (40.16 dB and 0.9816) under original training based on the ∼600K real pairs. Please refer to the main text for more analysis.

4. More Rain Removal Experiments

In this section, we provide more experimental results on several benchmark datasets.

Representative Methods. We evaluate the effectiveness of the augmentation strategy benefitted from

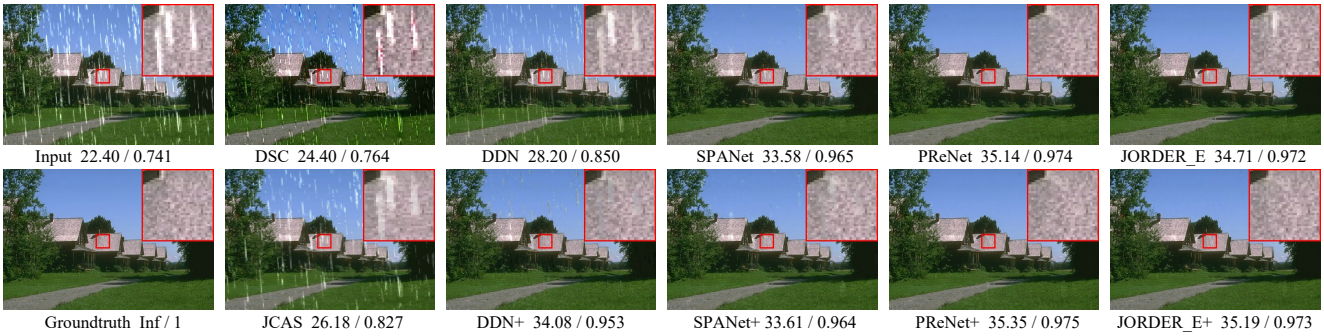

Figure 4. **Vertical contrast**. Performance comparison on a test image from Rain100L, including rainy image/groundtruth, derained results from DSC/JCAS, and deep derainers trained on the original (1st row) / augmented (2nd row) Rain100L training set. PSNR/SSIM is listed behind each result for easy reference. The images are better observed by zooming in on screen.

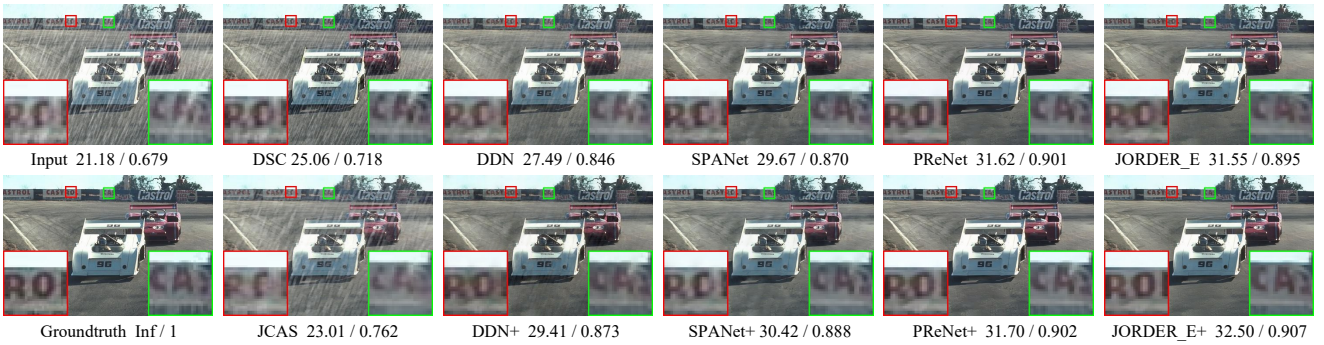

Figure 5. **Vertical contrast**. Performance comparison on a test image from Rain1400, including rainy image/groundtruth, derained results from DSC/JCAS, and deep derainers trained on the original (1st row) / augmented (2nd row) Rain1400 training set.

Table 2. PSNR and SSIM comparisons on SPA-Data testing set. "+" denotes the augmented training. △↑ represents the performance gain brought by the augmented rains generated by our rain generator that is jointly trained on the SPA-Data training set. **Note that the baseline of one method "A+" is "A"**.

Methods		Input	DSC	JCAS	DDN	DDN+	△↑	SPANet	SPANet+	△↑	PReNet	PReNet+	△↑	JORDER_E	JORDER_E+	△↑
SPA-Data	PSNR	34.15	34.95	34.95	36.16	39.47	3.31	38.14	38.59	0.45	40.16	40.27	0.11	40.78	41.49	0.71
	SSIM	0.927	0.942	0.945	0.946	0.974	0.028	0.973	0.974	0.001	0.981	0.984	0.003	0.980	0.985	0.005

VRGNet through latest DL-based SIRR methods, including DDN [3], PReNet [9], SPANet [10], and JORDER_E [11]. In the followings, we use notation 'A+' to denote the results of the method A after being retrained on the augmented dataset. Note that although our proposed VRGNet aims to help better train these DL-based SOTA derainers via data augmentation, we also list the performance of two representative model-based methods DSC [12] and JCAS [4] for more comprehensive comparisons.

4.1. More Results on Synthetic Data

Fig. 4 and Fig. 5 illustrate the deraining results on two typical hard samples, from Rain100L and Rain1400, respectively. From the two figures, it is easy to observe that for every DL-based method, when trained on augmented dataset generated by VRGNet, its reconstructed background (2nd row) has better visual quality, especially in texture preservation, than the corresponding one (1st row) trained on original training set. Clearly, the VRGNet has the potential to generate rains with better diversity. Note that the performance gain varies among different deep derainers, which is mainly caused by their different model capacities.

Note that Fig. 4 and Fig. 5 are the performance comparisons on one test image. More quantitative comparisons on the entire testing set are listed in Table 2 of the main text.

4.2. More Results on Real SPA-Data

We then evaluate the effectiveness of the proposed generator on the real SPA-Data [10], including ∼600K training pairs and 1K testing pairs. During the augmented training phase, the exploited rain generator is trained on the entire SPA-Data training set (∼600K). Note that this section represents the same domain test experiments, instead of the generalization case as shown in Section 6.2 of the main text.

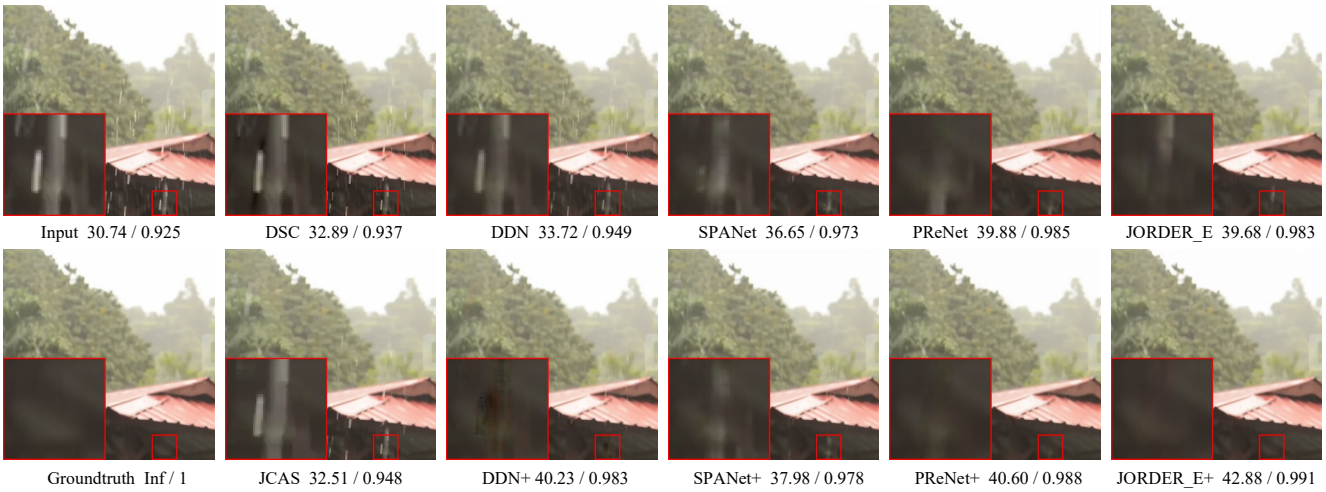

Figure 6. **Vertical contrast**. Performance comparison on a test image from SPA-Data, including rainy image/groundtruth, derained results from DSC/JCAS, and deep SOTAs trained on the original (1st row) / augmented (2nd row) SPA-Data training set.

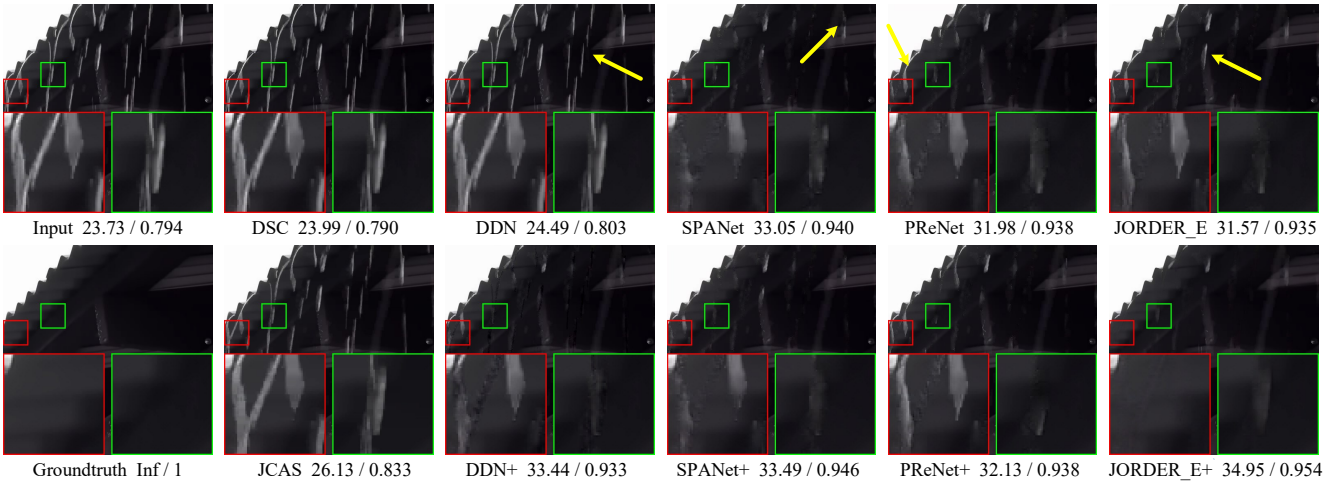

Figure 7. **Vertical contrast**. Generalization comparison on a test image from SPA-Data, including rainy image/groundtruth, derained results from DSC/JCAS, and deep derainers trained on the original (1st row) / augmented (2nd row) Rain100L training set.

Table 2 provides the quantitative results, which finely confirms the effectiveness of our proposed VRGNet in real rain generation[1]. Fig. 6 displays the visual comparisons on a test rainy image with complicated rain types from SPA-Data, and shows that all the DL-based derainers trained on the augmented SPA-Data have better capability in rain removal and detail recovery. Note that due to the dual influence of network structure and the quality of training set, the improvement room $\triangle\uparrow$ for every method is different.

[1]Note that in our all experiments, the used patch size is different from the default setting in SPANet. Under this training setting, the retrained SPANet has lower performance on SPA-Data than the original one released.

4.3. More Generalization Results on Real SPA-Data

Fig. 7 shows the derained results on a test rainy image from SPA-Data. From it, we can observe that as compared to original training, all DL-based methods with the augmented training have achieved better visual quality and higher PSNR/SSIM. Note that due to the dual influence of network structure and the quality of training set, the improvement room $\triangle\uparrow$ for every method is different.

5. More Analysis about VRGNet

5.1. Derained Results of VRGNet

When jointly training the VRGNet as shown in Fig. 1, we adopt the latest PReNet [9] as the *BNet* due to its simplicity

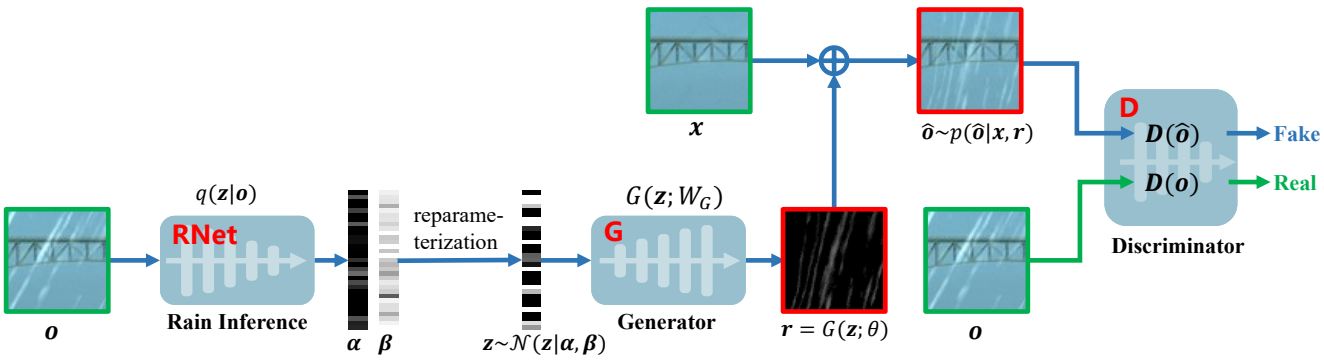

Figure 8. The flowchart of VRGNet- that directly regards $\boldsymbol{x}$ as $\boldsymbol{b}$.

Table 3. PSNR and SSIM on benchmark datasts under different cases, including jointly training the VRGNet (PReNet-) and only training BNet (PReNet).

Datasets	Rain100L		Rain100H		Rain1400		SPA-Data	
Metrics	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Input	26.90	0.838	13.56	0.371	25.24	0.810	34.15	0.927
PReNet-	36.94	0.975	30.08	0.887	32.19	0.941	39.70	0.978
PReNet	37.42	0.979	30.11	0.905	32.24	0.944	40.16	0.981

Table 4. Average PSNR and SSIM of PReNet on SPA-Data testing set. VRGNet- denotes the simplified VRGNet by removing *BNet*, as shown in Fig. 8. Under each setting, the result is averaged over 5 random repeated attempts.

# Samples (real+fake)	1K+0.5K	1K+1K	1K+2K
VRGNet- (PSNR / SSIM)	39.41 / 0.9790	39.39 / 0.9787	39.35 / 0.9784
VRGNet (PSNR / SSIM)	**39.71 / 0.9796**	**39.83 / 0.9795**	**40.25 / 0.9813**

and fast training speed. After the joint training, the derained results (denoted as PReNet-) on benchmark datasets are reported in Table 3. Naturally, we find that due to the regularization effect of adversarial loss in Eq. (17) of the main text, the performance of PReNet- is a little lower than (but comparable to) that only training BNet (PReNet) based on the negative SSIM loss. This trend is consistent with that in most GAN based methods for low-level tasks.

5.2. More Analysis on the Role of BNet

From Fig. 1, after the joint training, the *BNet* does not play roles in new rain layer augmentation. However, this subnetwork is indeed necessary as analyzed below.

For convenience, we briefly denote VRGNet- as the model discarding *BNet* and directly regarding the rain-free image $\boldsymbol{x}$ as the latent background $\boldsymbol{b}$, as shown in Fig. 8. In this setting, the posterior assumption $q(\boldsymbol{b}|\boldsymbol{o}) = \prod_{j=1}^{d}\mathcal{N}\left(b_j|\mu_j(\boldsymbol{o};W_B),\sigma_j^2(\boldsymbol{o};W_B)\right)$ as Eq. (13) of the main text can be simply set as a Dirac distribution without any parameters, i.e.,

$$q(\boldsymbol{b}|\boldsymbol{o}) = \text{Dirac}_{\boldsymbol{x}}(\boldsymbol{b}), \tag{4}$$

where $\text{Dirac}_{\boldsymbol{x}}(\cdot)$ means the Dirac distribution centered at point $\boldsymbol{x}$. This hard assumption will lead to the degraded network framework displayed as Fig. 8. As a special case, it indeed simplifies our proposed inference framework (Fig. 1) to some extent, but has stricter requirements for the accuracy of the estimated rain-free image $\boldsymbol{x}$. If the pre-collected "rain-free" image $\boldsymbol{x}$ is not sufficiently accurate, it will naturally degrade the training performance of *RNet* and the generator G. In contrast, the introduction of *BNet* is able to alleviate this issue by providing a better predicted background, and then helps G generate more plausible rain layer to fool discriminator D. Therefore, we propose to adopt the more general posterior assumption as Eq. (13) of the main text and retain *BNet* in this paper.

To further substantiate the analysis above, we compare VRGNet- and VRGNet based on the semi-automatically generated real SPA-Data [10], including ∼600K training pairs and 1K testing pairs. Specifically, in SPA-Data, the rain-free image $\boldsymbol{x}$ is estimated based on multiple rainy images taken in the same condition, and thus is not the exact latent clean background $\boldsymbol{b}$. First, we execute the joint training on the VRGNet- (Fig. 8) and VRGNet (Fig. 1), respectively, based on the ∼600K training pairs, and obtain the corresponding different generator G. Then we randomly select 1K pairs from the original ∼600K pairs and separately augment them with ratio N_f (i.e., generate N_fK fake pairs) by utilizing the learned two different generators.

Table 4 reports the PSNR/SSIM averaged over 5 repetitions for each different augmentation ratio N_f. From the table, we can easily observe that 1) Under each N_f setting, the performance of VRGNet significantly surpasses VRGNet-. 2) With the increase of N_f from 0.5 to 2, the average PSNR/SSIM results of VRGNet get better while that of VRGNet- becomes worse. This is mainly because VRGNet- does not finely capture the essential rain distribution without the guidance of *BNet*.

References

[1] Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β-VAE. *arXiv preprint arXiv:1804.03599*, 2018. 2

[2] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2016. 2

[3] Xueyang Fu, Jiabin Huang, Delu Zeng, Huang Yue, Xinghao Ding, and John Paisley. Removing rain from single images via a deep detail network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3855–3863, 2017. 4

[4] Shuhang Gu, Deyu Meng, Wangmeng Zuo, and Zhang Lei. Joint convolutional analysis and synthesis sparse representation for single image layer separation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1708–1716, 2017. 4

[5] Hyunjik Kim and Andriy Mnih. Disentangling by factorising. *arXiv preprint arXiv:1802.05983*, 2018. 2

[6] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 1

[7] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018. 1

[8] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 1

[9] Dongwei Ren, Wangmeng Zuo, Qinghua Hu, Pengfei Zhu, and Deyu Meng. Progressive image deraining networks: a better and simpler baseline. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3937–3946, 2019. 1, 4, 5

[10] Tianyu Wang, Xin Yang, Ke Xu, Shaozhe Chen, Qiang Zhang, and Rynson WH Lau. Spatial attentive single-image deraining with a high quality real rain dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12270–12279, 2019. 4, 6

[11] Wenhan Yang, Robby T. Tan, Jiashi Feng, Jiaying Liu, Shuicheng Yan, and Zongming Guo. Joint rain detection and removal from a single image with contextualized deep networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP(99):1–1, 2019. 4

[12] Luo Yu, Xu Yong, and Ji Hui. Removing rain from a single image via discriminative sparse coding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3397–3405, 2015. 4

[13] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018. 1