From Rain Removal to Rain Generation

Hong Wang,* Zongsheng Yue,* Qi Xie, Qian Zhao, Deyu Meng†
Xi’an Jiaotong University
dymeng@mail.xjtu.edu.cn
August 11, 2020

Abstract

Single image deraining is an important yet challenging issue due to the complex and diverse rain structures in real scenes. Currently, the state-of-the-art performance on this task is achieved by deep learning (DL)-based methods that mainly benefit from abundant pre-collected paired rainy-clean samples either manually synthesized or semi-automatically generated under human supervision. This tends to bring a large labor for data collection and more importantly, such manner neglects to elaborately explore the intrinsic generative mechanism of rain streaks which should be related to the most insightful understanding of the task. Against this issue, we investigate the generative process of rainy image and construct a full Bayesian generative model for generating rains from automatically extracted latent variables that represent physical structural factors for depicting rains, like direction, scale, and thickness. To solve this model, we propose an algorithm where the posteriors of latent variables are parameterized as CNNs and all the involved parameters can be inferred under a concise variational inference framework in a data-driven manner. Especially, the rain layer is modeled as an implicit distribution, parameterized as a generator, which avoids subjective prior assumptions on rains as in traditional model-based methods. More practically, from the learned generator, rain patches can be automatically generated and utilized to simulate diverse training pairs so as to enrich and augment the existing benchmark datasets. Comprehensive experiments substantiate that the proposed model has fine capability of generating plausible samples that not only helps significantly improve the deraining performance of current DL-based single image derainers, but also largely loosens the requirement of large training sample pre-collection for the task.

1 Introduction

Single image rain removal (SIRR), usually regarded as a necessary pre-processing step of outdoor image processing tasks, e.g., autonomous driving [1], scene segmentation [2], and object tracking [3], has attracted increasing attention in recent years. Due to the complex and diverse rain structures in real scenes, SIRR is known as a typical challenging ill-posed issue [4–6].

Against this issue, most conventional model-based methods make efforts on exploring intrinsic prior knowledge of background or rain layers to extract them from rainy images. Specifically, besides regular image prior expressions for rain-free (background) image modeling, like sparse coding and low-rankness [7,11], recently gradually more works put their focus on the specific prior formulation to deliver intrinsic rain characteristics. Typically used rain prior models include dictionary learning [7], Gaussian mixture model (GMM) [12,13], and sparse coding [8,9,14]. Albeit achieving success in certain scenarios, the deraining performance of these traditional model-based methods tend to be degenerated...

*Equal contribution
†Corresponding author
when encountering complicated and diverse practical rain types, due to the difficulty of sufficiently encoding such complex rain shapes, like those with highly variant directions \cite{15}, thicknesses \cite{16}, and scales \cite{14}. Therefore, it is critical to explore more powerful coding manner for fitting general rains in real-world.

Recently, deep learning (DL)-based methods have achieved significant success in SIRR by extracting clean background from a rainy image based on a powerful mapping function, usually parameterized as a convolutional neural network (CNN) \cite{15,17,24}. The remarkable success of these DL-based approaches, however, heavily depends on pre-collected abundant paired rainy-clean training samples, either manually synthesized based on photo-realistic rendering technique \cite{25} or semi-automatically generated by professional photography \cite{26} and human supervision \cite{27} (i.e., non-automated). Evidently, this acquisition process of training samples is time-consuming, labor-cumbersome, and expensive. Besides, the generated rain types are always limited and repetitive, since they have to be synthesized by empirically setting some oscillation parameters of rain \cite{25} or shooting a video under one specific scene to obtain multiple rainy-clean image pairs \cite{26}. The diversity of rain patterns is thus always not sufficiently comprehensive, even for some large-scale rainy-clean dataset.

More importantly, most DL-based methods neglect to elaborately explore the intrinsic generative mechanism underlying rain streaks, which should be essential and helpful for better understanding and handling SIRR task. It is worth mentioning that the work \cite{28} focuses on exploiting CycleGAN to remove rain streaks in an unsupervised manner, which can naturally create new paired dataset as a by-product. Evidently, this work does not specifically explore the intrinsic mechanism of rain and it still aims to accomplish the deraining task.

Our idea is to alternatively investigate the rain generation task instead of traditional rain removal, which can greatly alleviate all aforementioned issues existed in current SIRR research. Any required number of free rainy samples can be easily obtained by this generator, and the generated rains are non-repetitive and with more abundant patterns that are not contained in the original training dataset. Most insightfully, such generator with explicit mapping form tends to provide useful clues for understanding generation intrinsics underlying rains, potentially useful for general processing tasks on the rainy images. Specifically, our contributions are mainly three-fold:

1) To the best of our knowledge, this work is the first to specifically propose a generative model for rain generation. Different from conventional hand-crafted priors for rains in traditional model-based methods, the model can be readily used to deliver implicit rain distribution. All parameters involved in this model can be easily obtained through an elaborately designed variational inference algorithm.

2) A rain generator with explicit mapping form can be obtained, which reflects the manifold projection from latent rain factors, like direction, thickness, and scale, to rain images. Such generator facilitates an easy augmentation of diverse and variant rains through compactly interpolating the rain factors of any two rainy images along the manifold, as shown in Fig. 1. The performance of any current deep derainer is thus expected to be further improved by retraining it on an augmented dataset. Meanwhile, the generalization results on real data with complex rains have favorable visual quality.

3) Instead of directly training on the original large-scale dataset, we have verified that a comparably performable derainer can be efficiently obtained by only using a very small part of this original dataset with some augmented samples generated by our rain generator. The method is thus hopeful to largely save both the labor of collecting training pairs and the computation cost of training deep derainers in SIRR task, which should be very meaningful in practice.

The paper is organized as follows. Section \[2] introduces the related work. Section \[3] presents the generative model underlying rainy image, the deep variational inference algorithm, and network training. Section \[4] demonstrates experimental details and results, and the conclusion is finally made.
2 Related Work

In this section, we briefly review the DL-based derainers and generative models related to this work.

**DL-based Deraining Approaches:** Very recently, DL has achieved remarkable success in SIRR \([16,17,20,21,26,29–34]\). Instead of pre-setting image priors, these methods design specific network architectures for directly predicting background (or rain layer) from an input rainy image based on a large collection of training rainy-clean image pairs. Fu \([17]\) firstly proposed a 3-layer CNN to remove rains from the HF part of one rainy image. Later, the authors ameliorated the CNN by introducing negative residual structure and deeper hidden layers \([29]\). For better visual quality, Zhang \([19]\) developed a conditional generative adversarial network (GAN) to make the derained image indistinguishable from its corresponding rain-free one. The authors further designed a density-aware multi-stream densely connected CNN for joint rain density estimation and rain removal \([16]\). Recently, recurrent and multi-stage learning are introduced into SIRR task \([15,30,32,35]\). Besides, some works incorporated multi-scale learning to exploit the self-similarity both within the same scale or across different scales \([20,24,36,37]\). Very recently, some researchers tried to embed prior knowledge into deep network to improve the interpretability and generality of network training, such as \([34,38,39]\).

Although performing well in SIRR, these methods require a large number of paired training data. Previous methods attempted to synthesize rainy datasets \([7,12,16,17,30]\) by utilizing some photo-realistic rendering techniques and empirically setting oscillation parameters of rain \([25]\). Later, Wang \([26]\) proposed a semi-automatic method that incorporates temporal priors and human supervision to construct a large-scale dataset covering more rain scenes. Clearly, such non-automated acquisition process is time-consuming and labor-cumbersome. Moreover, the pre-collected samples always contain repetitive rain types, which makes it inefficient to train a deep derainer. Therefore, it is necessary to explore one purely automatic generation mechanism with the capability to simulate rainy-clean image pairs with possibly variant rain types. This constitutes the main goal of this work.

**Generative Models:** As an active research topic in computer vision and machine learning, deep generative model has been widely studied recently, such as variational autoencoder (VAE) \([40,41]\), generative adversarial network (GAN) \([42,43]\), flow-based generative model \([44]\). Especially, as prominent models, VAE and GAN have achieved remarkable success in many image generation tasks, including face modeling \([45,46]\), style transfer \([47]\), image noise generation \([48,49]\) and so on. To the best of our knowledge, there is not any work completely focusing on the rain generation task. Although there is a work \([28]\) which can naturally create new paired dataset as a by-product, it mainly aims to utilize CycleGAN to finish the deraining task, not mentioning any physical mechanism of rain. Therefore, inspired by deep generative model, we take a step forward to explore the intrinsic generative mechanisms of rain streaks.
3 The Proposed Method

For a training set \( D = \{ o_n, x_n \}_{n=1}^N \), where \( o_n \) is the \( n \)-th rainy image and \( x_n \) is the rain-free background, our aim is mainly two-fold: one is to construct a generative model for rainy image, and the other is to design a variational inference algorithm to approximate the posteriors of latent variables. We then construct a rational full Bayesian model for the aforementioned task.

3.1 Constructing Generative Model for SIRR

Given any single rainy image \( o \in \mathbb{R}^d \) with the size \( d \) as height \( \times \) width, the generation process is:

\[
o = b + r,
\]

where \( b \) and \( r \) denote the latent clean background and rain layer underlying \( o \), respectively. Note that in the given training pairs, the rain-free image \( x \) is usually simulated or estimated based on multiple rainy images taken in the same condition like in [26], and thus is not the exact latent clean background \( b \). We thus embed \( x \) into the following Gaussian prior distribution to constrain \( b \) as:

\[
b \sim \mathcal{N}(b|x, \epsilon_0^2 I_d),
\]

where \( I_d \in \mathbb{R}^{d \times d} \) is the unit matrix. \( \epsilon_0^2 \) is a hyper-parameter measuring the similarity between \( x \) and \( b \), and can be easily set as a small value. Note that for synthetic datasets where rainy image is obtained by adding synthesized rain layer on the pre-collected clean image [7,12,17,30], \( x \) can be directly regarded as the true groundtruth \( b \). In this case, Dirac prior on \( b \) is a proper choice which can be also well approximated by Eq. (2) with \( \epsilon_0^2 \) close to 0.

For rain layer \( r \), we model it as the following implicit distribution (‘un’ for ‘unknown’, meaning)

\[
r \sim p_{un}(r|z),
\]

where \( z \) denotes latent variables used to encode the physical structural factors underlying rains, e.g., direction, scale, and thickness [16,30,34]. In practice, we parameterize the implicit rain distribution \( p_{un}(r|z) \) in Eq. (3) as a generator \( G \) with parameters \( W_G \), meaning that every output of \( G \) can be regarded as one sample from \( p_{un}(r|z) \), i.e.,

\[
r \sim p_{un}(r|z) \iff r = G(z; W_G).
\]

As suggested in [50,51], the isotropic Gaussian prior distribution is imposed on \( z \), i.e.,

\[
z \sim \mathcal{N}(z|0, I_t).
\]

From Eqs. (1), (5), it is easy to derive a full Bayesian model for this SIRR task. In the following, we aim to infer the posterior of latent variables \( b \) and \( z \), i.e., \( p(b, z|o) \).

3.2 Variational Posterior of Latent Variables

To approximate the posterior \( p(b, z|o) \), we construct a variational form \( q(b, z|o) \). Like the commonly-used factorized hypothesis in the mean-field variational inference [40], we also introduce the conditional independence assumption as:

\[
q(b, z|o) = q(b|o) q(z|o).
\]

For the latent variable \( b \), Gaussian distribution is employed to model the posterior distribution \( q(b|o) \) since it performs sufficiently well as shown in [52], i.e.,

\[
q(b|o) = \prod_i \mathcal{N}(b_i|\mu_i(o; W_B), \sigma_i^2(o; W_B)) .
\]
where $\mu_i(o; W_B)$ and $\sigma^2_i(o; W_B)$ are mapping functions from $o \in \mathbb{R}^d$ to variational posterior parameters (i.e., mean and variance, respectively) of latent variable $b$. These two functions are jointly parameterized as one network with parameter $W_B$, for restoring clean background image.

For $q(z|o)$, it is also assumed as Gaussian distribution to represent rain factors like in [40][51], i.e.,

$$q(z|o) = \prod_j N(z_j|\alpha_j(o; W_R), \beta_j(o; W_R)) , \quad (8)$$

similarly, $\alpha_j(o; W_R)$ and $\beta_j(o; W_R)$ are functions for inferring the posterior parameters of latent variable $z$, and they are integrally parameterized as one rain inference network with parameters $W_R$.

### 3.3 Variational Lower Bound of Marginal Likelihood

In the followings, we aim to design rational objective function to optimize these network parameters $W_B$, $W_R$, and $W_G$ by gradient decent strategy. For notation convenience, we simplify $\mu_i(o; W_B)$, $\sigma^2_i(o; W_B)$, $\alpha_j(o; W_R)$, and $\beta_j(o; W_R)$ as $\mu_i$, $\sigma^2_i$, $\alpha_j$, and $\beta_j$, respectively. For any paired data $(o, x)$ in training set $D$, we can decompose the marginal likelihood of the rainy image $o$ as [53]

$$\log p(o) = \mathcal{L}(b, z; o) + D_{KL}(q(b, z|o)||p(b, z|o)) , \quad (9)$$

where the first term in Eq. (9) is expressed as:

$$\mathcal{L}(b, z; o) = E_{q(b, z|o)}[\log p(o|b, z) p(b) p(z) - \log q(b, z|o)] . \quad (10)$$

Here $E_{p(a)}[f(a)]$ is the expectation of function $f(a)$ about the stochastic variable $a$ with the probability density function $p(a)$. As can be seen, the second term in Eq. (9) is the KL divergence measuring the difference between the variational approximate posterior $q(b, z|o)$ and true posterior $p(b, z|o)$. The non-negative property of KL divergence leads to the following inequality, i.e.,

$$\log p(o) \geq \mathcal{L}(b, z; o) . \quad (11)$$

Clearly, $\mathcal{L}(b, z; o)$ is the variational lower bound on the marginal likelihood $\log p(o)$. Based on Eqs. (6)-(8), the lower bound $\mathcal{L}(b, z; o)$ in Eq. (10) can be equally rewritten as:

$$\mathcal{L}(b, z; o) = E_{q(b, z|o)}[\log p(o|b, z)] - D_{KL}[q(b|o)||p(b)] - D_{KL}[q(z|o)||p(z)] , \quad (12)$$

where

$$D_{KL}(q(b|o)||p(b)) = \sum_{i=1}^d \left\{ \frac{(\mu_i - x_i)^2}{2\sigma_i^2} + \frac{1}{2} \left( \frac{\sigma_i^2}{\sigma_0^2} - \log \frac{\sigma_i^2}{\sigma_0^2} - 1 \right) \right\} , \quad (13)$$

$$D_{KL}(q(z|o)||p(z)) = \sum_{j=1}^2 \left\{ \frac{\alpha_j}{2} + \frac{1}{2} (\beta_j - \log \beta_j - 1) \right\} . \quad (14)$$

Obviously, in the first term of Eq. (12), the conditional rainy image distribution $p(o|b, z)$ is intractable due to the implicit distributions $p_{un}(r|z)$ in Eq. (3). Fortunately, the generator $G$ defined in Eq. (4) makes it possible to sample from $p(o|b, z)$, i.e.,

$$o \sim p(o|b, z) \iff o = b + G(z; W_G) , \quad (15)$$

which motivates us to introduce a discriminator $D$ with parameters $W_D$ to approximate the first term in Eq. (12) by the following two-player game [12]:

$$\min_D \max_G \mathcal{L}_{adv}(b, z) = E_{o \sim p_{data}}[P(o)] - E_{b \sim q(b|o), z \sim q(z|o)}[P(b + G(z; W_G))] . \quad (15)$$

Thus we can reformulate the negative lower bound in Eq. (12) as follows:

$$\hat{\mathcal{L}}(b, z; o) = \gamma \mathcal{L}_{adv}(b, z) + D_{KL}(q(b|o)||p(b)) + D_{KL}(q(z|o)||p(z)) , \quad (16)$$
Figure 2: The flowchart of the proposed variational rain generation network (VRGNet). It contains four sub-networks, naturally constructed based on the variational lower bound in Eq. (12).

Algorithm 1 Variational Inference Algorithm toward Rain Generation

Input: Training data $D=\{o_n, x_n\}_{n=1}^N$, batch size $n_b$, $n_{\text{critic}}$.
Output: Network parameters $W = \{W_B, W_R, W_G, W_D\}$

1: while $W$ is not convergent do
2: for $m = 1$ to $n_{\text{critic}}$ do
3:   $\{o, x\} \leftarrow \text{SampleMiniBatch}(D, n_b)$.
4:   $\{\mu, \sigma^2\} \leftarrow \text{BNet}(o; W_B)$.
5:   $b \leftarrow \text{Reparameterization}(\mu, \sigma^2)$.
6:   $\{\alpha, \beta\} \leftarrow \text{RNet}(o; W_R)$.
7:   $z \leftarrow \text{Reparameterization}(\alpha, \beta)$.
8:   $\hat{o} \leftarrow b + G(z; W_G)$.
9:   Update $D$ with fixed $\text{BNet}$, $\text{RNet}$, and $G$.
10: end for
11: Update $\text{BNet}$ with fixed $\text{RNet}$, $D$, and $G$.
12: Update $\text{RNet}$ and $G$ with fixed $\text{BNet}$ and $D$.
13: end while

where $\gamma$ is a hyper-parameter controlling the importance between the adversarial loss and KL divergence. The value is set empirically and will be explained in experiment section.

Therefore, for optimizing network parameters $W_B, W_R, W_G$, and $W_D$, the total objective function on entire training set, can be formulated as:

$$\sum_{n=1}^N \tilde{L}(b_n, z_n; o_n).$$

Note that during training, the networks parameters $W_B, W_R, W_G$ and $W_D$ are shared across the entire training data, leading to a general statistical inference from $o$ to the latent variables $b$ and $z$.

3.4 Implementation Details

Inference Framework: The overall inference framework for our model is illustrated as Fig. 2, called variational rain generation network (VRGNet). It mainly contains 4 parts as follows:

1) $\text{BNet}$ infers the posterior parameters $\mu$ and $\sigma^2$ in Eq. (7) from the rainy image $o$;
2) $\text{RNet}$ predicts the posterior parameters $\alpha$ and $\beta$ for latent variable $z$ in Eq. (8);
3) $G$ utilizes the extracted latent variables $z$ to generate rain patches that represent examples sampled from the implicit distribution $p_{un}(r|z)$ in Eq. (4). Besides, the reparameterization trick [40] is used in back propagation;
4) $D$ acts as a criterion to optimize the generator $G$ by distinguishing the real data $o$ from the generated $\hat{o}$. To stabilize the training process, we adopt the spectral normalization [54] and self-attention [55] technologies.

More detailed network architectures can be found in the supplementary material.
**Training Strategies:** The entire framework in Fig. 2 is first jointly trained based on the loss function in Eq. (17). The whole training procedure is summarized as Algorithm 1, where we adopt the gradient penalty strategy for $D$ to stabilize the adversarial learning [56].

After obtaining the rain generator $G$, it can be utilized to generate more rainy-clean image pairs freely. Based on the augmented training dataset, including original and generated pairs, we retrain current representative DL-based SOTA derainer to validate the effectiveness of the proposed VRGNet. It is noteworthy that the augmentation operation is implemented on the original dataset, not introducing any extra training pairs beyond it.

## 4 Experimental Results

In this section, based on synthetic and real datasets, we evaluate the superiority of VRGNet in rain generation. Note that due to limited space, some experiments are put in supplementary material.

### 4.1 Experimental Settings

**Training Details:** During the joint training, the entire network in Fig. 2 is optimized by Adam algorithm [57]. The initial learning rates for $BNet$, $RNet$, $G$, and $D$ are $2 \times 10^{-4}$, $1 \times 10^{-4}$, $1 \times 10^{-4}$, and $4 \times 10^{-4}$, respectively, and divided by 2 at epochs [400, 600, 650, 675, 690, 700]. The different initialized learning rate settings for $G$ and $D$ are inspired by TTUR [58]. The prior hyper-parameter $\varepsilon^2_0$ is set as $1 \times 10^{-6}$ as suggested in [52] and the dimension $t$ of latent variable $z$ is 128. In each epoch, the batch size $n_b$ is set as 18, and we randomly crop $18 \times 3000$ patches with size $64 \times 64$ from the rainy image $o$ in $D$ for training. As suggested in [59], the penalty coefficient in WGAN-GP is 10, and $n_{\text{critic}}$ is 5, meaning that we update $D$ 5 times for each updating of $BNet$, $RNet$, and $G$. The coefficient $\gamma$ in Eq. (16) is empirically set as 1 for synthetic datasets and 0.01 for SPA-Data.

As for the augmented training, we augment original benchmark training datasets with a ratio 0.5 by default and utilize them to retrain representative DL-based SOTA derainers based on their default settings. All the training process are realized by PyTorch [59] on one NVIDIA GeForce GTX 1080Ti GPU. Note that the augmentation ratio denotes the proportion of the number of generated samples by VRGNet to that contained in original dataset.

**Performance Metrics:** To objectively evaluate the rain removal performance, two commonly-used metrics are utilized, including peak-signal-to-noise ratio (PSNR) [60] and structure similarity (SSIM) [61]. Considering the sensitivity of the human visual system to the Y channel of a color image in YCbCr space, similar as [30, 32, 34], we also calculate PSNR and SSIM based on this luminance channel.

### 4.2 Evaluation on Synthetic Data

**Representative Methods and Datasets:** We evaluate the effectiveness of VRGNet through augmented training on five latest DL-based methods, including DDN [29], PReNet [32], SPANet [26], JORDER_E [15], and RCDNet [34], based on three widely-used synthetic datasets, including Rain100L, Rain100H [15], and Rain1400 [29]. In the following experiments, 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 [7] and JCAS [9] for comprehensive comparisons.

Table 1 gives the quantitative performance of these deraining methods on different datasets with complex rain patterns. As reported, on each benchmark dataset, all the DL-based methods with augmented training always achieve impressive performance improvement, as compared to their baselines based on original training. These favorable improvements after augmentation validate the effectiveness of VRGNet in rain generation.

Fig. 3 illustrates the visual deraining results on one hard sample from Rain100H. It is easy to observe that due to the powerful fitting capability of deep CNN, DL-based ones obviously outperform model-based DSC and JCAS. Besides, for every DL-based method, when trained on the 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 Rain100H training set.

---

1The code/project links for these comparison methods are listed in supplementary material.

2For RCDNet, we adopt the simplified code framework the authors release.
Table 1: PSNR and SSIM comparisons on synthetic datasets. ‘+’ denotes the one trained on a larger training dataset augmented by the proposed VRGNet, i.e., augmented training.

| Methods     | Input       | DSC         | JCAS        | DDN         | DDN+        | SPANet      | SPANet+     | PReNet      | PReNet+     | JORDER_E    | JORDER_E+   | RCDNet      | RCDNet+     |
|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| Rain100L    | PSNR        | 26.90       | 27.34       | 28.54       | 32.38       | 35.56       | 35.33       | 35.83       | 37.42       | 37.84       | 37.68       | 38.01       |
|             | SSIM        | 0.838       | 0.849       | 0.852       | 0.926       | 0.966       | 0.972       | 0.979       | 0.980       | 0.979       | 0.980       | 0.984       | 0.984       |
| Rain100H    | PSNR        | 13.56       | 13.77       | 14.62       | 22.85       | 26.99       | 25.11       | 27.24       | 30.11       | 30.48       | 30.50       | 32.26       | 31.28       |
|             | SSIM        | 0.371       | 0.312       | 0.451       | 0.725       | 0.797       | 0.833       | 0.883       | 0.905       | 0.910       | 0.905       | 0.920       | 0.909       | 0.921       |
| Rain1400    | PSNR        | 25.24       | 27.88       | 26.20       | 28.45       | 30.27       | 29.85       | 30.24       | 30.50       | 32.26       | 32.00       | 32.40       | 31.28       |
|             | SSIM        | 0.810       | 0.839       | 0.847       | 0.889       | 0.917       | 0.915       | 0.923       | 0.943       | 0.945       | 0.935       | 0.946       | 0.947       | 0.951       |

Table 2: PSNR and SSIM comparisons on SPA-Data testing set.

| Methods     | Input       | DSC         | JCAS        | DDN         | DDN+        | SPANet      | SPANet+     | PReNet      | PReNet+     | JORDER_E    | JORDER_E+   | RCDNet      | RCDNet+     |
|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| SPA-Data    | PSNR        | 34.15       | 34.95       | 34.95       | 36.16       | 39.47       | 38.14       | 38.59       | 40.16       | 40.27       | 40.78       | 41.49       | 41.47       |
|             | SSIM        | 0.927       | 0.942       | 0.945       | 0.946       | 0.974       | 0.973       | 0.974       | 0.981       | 0.984       | 0.980       | 0.985       | 0.983       | 0.985       |

Figure 3: Visual comparison on a test image from Rain100H, including rainy image/groundtruth, derained results from DSC/JCAS, and DL-based SOTAs trained on original (first row) / augmented (second row) Rain100H training set. The images are better observed by zooming in on screen.

Figure 4: Derained results of all competing methods on one typical test image from SPA-Data.

4.3 Evaluation on Real Data

We then evaluate the advantages of VRGNet over real dataset from [26] (called SPA-Data), including ~600k training pairs and 1k testing pairs. Specifically, we select all (a small part) of training pairs from SPA-Data to correspondingly assess the effectiveness (efficiency) of VRGNet, respectively.

Effectiveness: Similar to the experiments on synthesized datasets above, we utilize the entire SPA-Data training set to compare the rain removal effects of all these single image deraining methods in the cases of original training and augmented training. Table 2 provides the quantitative results, which finely confirms the effectiveness of our proposed VRGNet in rain generation.

Fig. 4 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 dataset have the better capability in rain removal and detail recovery.

Efficiency: Instead of utilizing the whole training set above, we randomly select 1k pairs from the original ~600k pairs and augment them with different ratios (i.e., generate different number $N_f$ of fake pairs) for training. Meanwhile, we also randomly choose the same number (i.e., 1k + $N_f$) of real training pairs all from the original SPA-Data and take this case as a comparison baseline. Due to limited space, we simply adopt the PReNet [32] with simplicity and fast training speed to implement this experiment.

Table 3 reports the PSNR/SSIM averaged over 5 repetitions for each different augmentation ratio. From the table, we can easily observe: 1) With the increase of $N_f$ from 0k to 6k, the average

3Note that in all our 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.
Table 3: Average PSNR and SSIM of PReNet on SPA-Data testing set. **Baseline** denotes that training samples are all chosen from SPA-Data training set (∼600k), and **VRGNet** means the augmented training where training samples are composed of 1k real pairs randomly selected from ∼600k and different number of fake pairs generated by our generator. Under each setting, the result is averaged over 5 random repeated attempts.

| # Real samples | Baseline (PSNR/SSIM) | VRGNet (PSNR/SSIM) | DDN / DDN+ SPANet / SPANet+ PReNet / PReNet+ JORDER_E / JORDER_E+ RCDNet / RCDNet+DSC / JCAS |
|----------------|----------------------|---------------------|--------------------------------------------------|
| # Samples      |                      |                     |                                                  |
| (real+fake)    |                      |                     |                                                  |
| 1k+0k          | 39.41/0.9787         | 39.41/0.9787        | DSC / JCAS                                       |
| 1k+0.5k        | 39.70/0.9800         | 39.71/0.9796        | DDN / DDN                                        |
| 1k+1k          | 39.86/0.9809         | 39.83/0.9795        | SPANet / SPANet                                  |
| 1k+2k          | 39.96/0.9813         | 40.25/0.9813        | PReNet / PReNet                                  |
| 1k+3k          | 40.05/0.9814         | 40.24/0.9814        | JORDER_E / JORDER_E                             |
| 1k+4k          | 40.04/0.9816         | 40.53/0.9819        | RCDNet / RCDNet                                  |
| 1k+5k          | 40.00/0.9817         | 40.68/0.9820        |                                                 |
| 1k+6k          | 40.06/0.9819         | 40.70/0.9819        |                                                 |

Figure 5: Generalization performance. From left to right: real rainy image with heavy rain from [39], derained results obtained by DSC/JCAS, and DL-based derainers trained on original (first row) / augmented (second row) SPA-Data. The images are better observed by zooming in on screen.

PSNR/SSIM under augmented training is closer to or even surpasses the performance (40.16/0.9816) under original training based on ∼600k real pairs. 2) Under every $N_f$ setting, the deraining performance in the case of VRGNet is comparable to or even outperforms baseline trained on the same number (1k+$N_f$) of pairs from SPA-Data, especially when $N_f$ is higher. Clearly, the proposed VRGNet indeed has a good ability to generate plausible data with better diversity, which more compactly and sufficiently covers the manifold underlying the high-dimensional rain distribution and more accurately reflects rain patterns in SPA-Data. More interestingly, only with 1k real pairs from SPA-Data and 4k generated samples, the PSNR 40.53dB under augmented training even exceeds the original one 40.16dB trained on ∼600k pairs by about 0.4dB. This strongly substantiates that the learned generator can largely loosen the requirement of large training data sample pre-collection and it has significant superiority in training a better derainer with higher efficiency. This should be very meaningful for real applications.

4.4 Evaluation on Other Unlabeled Data

We verify the generalization performance of all competing methods by utilizing the real unlabeled benchmark dataset from [39]. Fig. 5 displays the derained results on a real rainy image with extremely complicated rain types. As seen, model-based DSC/JCAS suffer from remaining visible ones. However, these DL-based ones remove most rain streaks. Especially, the RCDNet+ finely preserves image details as well as removes more rain streaks even rain marks. Clearly, although the proposed VRGNet is supervised, these DL-based SOTA derainers trained on augmented dataset can still achieve favorable visual quality when they are generalized to other unseen samples. This can be rationally attributed to the diversity of generated rain types.

4.5 Latent Manifold Analysis

We conduct interpolations of real images in the latent space to estimate the manifold continuity. For a pair of rainy images, we first utilize the inference model, i.e., $RNet$, to map them to latent factors $z$ and then make linear interpolations with different weighting coefficients between their latent factor representations. Fig. 5(b) shows that our rain generator function has continuity in the latent space in changing the direction (1st and 2nd rows) and thickness (3rd row) of rain streaks. This manifold continuity substantiates that the proposed model indeed has a fine capability to generate diverse rain types instead of simply memorizing the patterns in input image. More vivid animation effects can be seen in supplementary material.
5 Conclusion

In this paper, we have explored the rain generative mechanism and constructed a full Bayesian model for generating rains from latent variables. To solve this model, we have proposed a variational rain generation network (VRGNet), which learns an approximate posterior to true posterior of latent variables conditioned on input rainy image, and implicitly extracts the complex distribution of rains via a GAN in a data-driven manner. From the learned generator, rain patches can be automatically generated to simulate diverse training samples, which facilitate a beneficial augmentation and enrichment of the existing benchmark dataset. Comprehensive experiments have validated the superiority of VRGNet in generating plausible samples and thus helping significantly improve the deraining performance of current DL-based SOTAs. Especially, the small sample experiment validates that VRGNet has great potential to largely save the cost of training sample pre-collection and train a better derainer with higher efficiency.

References

[1] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. pages 3354–3361, 2012.

[2] Chang Cheng, Andreas Koschan, Chunghao Chen, D L Page, and Mongi A Abidi. Outdoor scene image segmentation based on background recognition and perceptual organization. IEEE Transactions on Image Processing, 21(3):1007–1019, 2012.

[3] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Kernel-based object tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence, 25(5):564–575, 2003.

[4] Hong Wang, Minghan Li, Yichen Wu, Qian Zhao, and Deyu Meng. A survey on rain removal from video and single image. arXiv: Image and Video Processing, 2019.

[5] Siyuan Li, Xiaochun Cao, Iago Breno Araujo, Wenqi Ren, Zhangyang Wang, Eric K Tokuda, Roberto Hirata Junior, Roberto Cesajarjunior, Jiawan Zhang, and Xiaojie Guo. Single image deraining: A comprehensive benchmark analysis. pages 3838–3847, 2019.

[6] Wenhan Yang, Robby T Tan, Shiqi Wang, Yuming Fang, and Jiaying Liu. Single image deraining: From model-based to data-driven and beyond. arXiv: Image and Video Processing, 2019.

[7] 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.

[8] He Zhang and Vishal M Patel. Convolutional sparse and low-rank coding-based rain streak removal. In IEEE Winter Conference on Applications of Computer Vision, pages 1259–1267, 2017.

[9] 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.

[10] Lei Zhu, Chi Wing Fu, Dani Lischinski, and Pheng Ann Heng. Joint bi-layer optimization for single-image rain streak removal. In Proceedings of the IEEE international conference on computer vision, pages 2526–2534, 2017.

[11] Yi Chang, Luxin Yan, and Sheng Zhong. Transformed low-rank model for line pattern noise removal. In Proceedings of the IEEE International Conference on Computer Vision, pages 1726–1734, 2017.

[12] Yu Li. Rain streak removal using layer priors. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2736–2744, 2016.

[13] Wei Wei, Lixuan Yi, Qi Xie, Qian Zhao, Deyu Meng, and Zongben Xu. Should we encode rain streaks in video as deterministic or stochastic? In Proceedings of the IEEE International Conference on Computer Vision, pages 2516–2525, 2017.

[14] Minghan Li, Qi Xie, Qian Zhao, Wei Wei, Shuhang Gu, Jing Tao, and Deyu Meng. Video rain streak removal by multiscale convolutional sparse coding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6644–6653, 2018.
[15] 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.

[16] He Zhang and Vishal M Patel. Density-aware single image de-raining using a multi-stream dense network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 695–704, 2018.

[17] Xueyang Fu, Jiabin Huang, Xinghao Ding, Yinghao Liao, and John Paisley. Clearing the skies: A deep network architecture for single-image rain removal. *IEEE Transactions on Image Processing*, 26(6):2944–2956, 2017.

[18] Ruoteng Li, Loong-Fah Cheong, and Robby T Tan. Single image deraining using scale-aware multi-stage recurrent network. *arXiv preprint arXiv:1712.06830*, 2017.

[19] He Zhang, Vishwanath Sindagi, and Vishal M Patel. Image de-raining using a conditional generative adversarial network. *IEEE Transactions on Circuits and Systems for Video Technology*, 2019.

[20] Rajeev Yasarla and Vishal M Patel. Uncertainty guided multi-scale residual learning-using a cycle spinning cnn for single image de-raining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8405–8414, 2019.

[21] Youzhao Yang and Hong Lu. Single image deraining via recurrent hierarchy enhancement network. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 1814–1822, 2019.

[22] Rajeev Yasarla and Vishal M Patel. Confidence measure guided single image de-raining. *IEEE Transactions on Image Processing*, 29:4544–4555, 2020.

[23] Guoqing Wang, Changming Sun, and Arcot Sowmya. Eri-net: Entangled representation learning for single image de-raining. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5644–5652, 2019.

[24] Kui Jiang, Zhongyuan Wang, Peng Yi, Chen Chen, Baojin Huang, Yimin Luo, Jiayi Ma, and Junjun Jiang. Multi-scale progressive fusion network for single image deraining. *arXiv preprint arXiv:2003.10985*, 2020.

[25] Kshitiz Garg and Shree K Nayar. Photorealistic rendering of rain streaks. *ACM Transactions on Graphics (TOG)*, 25(3):996–1002, 2006.

[26] 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.

[27] Xiaowei Hu, Chi-Wing Fu, Lei Zhu, and Pheng-Ann Heng. Depth-attentional features for single-image rain removal. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8022–8031, 2019.

[28] Yanyan Wei, Zhao Zhang, Jicong Fan, Yang Wang, Shuicheng Yan, and Meng Wang. De-raincyclegan: An attention-guided unsupervised benchmark for single image deraining and rainmaking. *arXiv: Computer Vision and Pattern Recognition*, 2019.

[29] 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.

[30] Wenhan Yang, Robby T Tan, Jiashi Feng, Jiaying Liu, Zongming Guo, and Shuicheng Yan. Deep joint rain detection and removal from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1357–1366, 2017.

[31] Jinshan Pan, Sifei Liu, Deqing Sun, Jiawei Zhang, and Ming-Hsuan Yang. Learning dual convolutional neural networks for low-level vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3070–3079, 2018.

[32] 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.

[33] Ruoteng Li, Loongfah Cheong, and Robby T Tan. Heavy rain image restoration: Integrating physics model and conditional adversarial learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1633–1642, 2019.
[34] Hong Wang, Qi Xie, Qian Zhao, and Deyu Meng. A model-driven deep neural network for single image rain removal. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020.

[35] Xia Li, Jianlong Wu, Zhouchen Lin, Hong Liu, and Hongbin Zha. Recurrent squeeze-and-excitation context aggregation net for single image deraining. In Proceedings of the European Conference on Computer Vision, pages 254–269, 2018.

[36] Xueyang Fu, Borong Liang, Yue Huang, Xinghao Ding, and John Paisley. Lightweight pyramid networks for image deraining. IEEE Transactions on Neural Networks and Learning Systems, 2019.

[37] Yupei Zheng, Xin Yu, Miaomiao Liu, and Shunli Zhang. Residual multiscale based single image deraining. In Conference on BMVC, 2019.

[38] Pan Mu, Jian Chen, Risheng Liu, Xin Fan, and Zhongxuan Luo. Learning bilevel layer priors for single image rain streaks removal. IEEE Signal Processing Letters, 26(2):307–311, 2019.

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

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

[41] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. arXiv: Machine Learning, 2014.

[42] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Advances in neural information processing systems, pages 2672–2680, 2014.

[43] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv: Learning, 2015.

[44] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. arXiv: Learning, 2014.

[45] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiri Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 8183–8192, 2018.

[46] Xiaoming Li, Ming Liu, Yuting Ye, Wangmeng Zuo, Liang Lin, and Ruigang Yang. Learning warped guidance for blind face restoration. pages 278–296, 2018.

[47] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE international conference on computer vision, pages 2223–2232, 2017.

[48] Jingwen Chen, Jiawei Chen, Hongyang Chao, and Ming Yang. Image blind denoising with generative adversarial network based noise modeling. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3155–3164, 2018.

[49] Dong-Wook Kim, Jae Ryun Chung, and Seung-Won Jung. Grdn: Grouped residual dense network for real image denoising and gan-based real-world noise modeling. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pages 0–0, 2019.

[50] 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.

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

[52] Zongsheng Yue, Hongwei Yong, Qian Zhao, Deyu Meng, and Lei Zhang. Variational denoising network: Toward blind noise modeling and removal. In Advances in Neural Information Processing Systems, pages 1688–1699, 2019.

[53] David M Blei and Michael I Jordan. Variational inference for dirichlet process mixtures. Bayesian Analysis, 1(1):121–143, 2006.

[54] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. arXiv preprint arXiv:1802.05957, 2018.
[55] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. arXiv preprint arXiv:1805.08318, 2018.

[56] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In Advances in neural information processing systems, pages 5767–5777, 2017.

[57] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[58] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In Advances in neural information processing systems, pages 6626–6637, 2017.

[59] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[60] Q. Huynh-Thu and M. Ghanbari. Scope of validity of psnr in image/video quality assessment. Electronics Letters, 44(13):800–801, 2008.

[61] Wang Zhou, Bovik Alan Conrad, Sheikh Hamid Rahim, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process, 13(4):600–612, 2004.

[62] 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.
Supplementary Materials

Figure 6: Manipulating latent code $z \in \mathbb{R}^{128}$. In all figures of latent code manipulation, we make each one latent element in $z$ vary from -3 to 3 with the interval as 0.4 while fixing the other latent elements in $z$ which are sampled from normal distribution. The different rows correspond to different sampled samples of fixed latent elements. For example, in (a), we sample a random vector (latent code $z$ ) from the normal distribution with the dimension as 128, and only vary the latent element at the 24-th dimension from -3 to 3. Taking this varied vector $z$ as the input of the Generator $G(z; W_z)$ trained on Rain100L, the corresponding output $r$ is thus each rain layer shown in (a). When we randomly sample two times from the normal distribution for the latent code $z$ and repeat this experiments, the generated $r$ are correspondingly displayed as two rows. (a)-(c): learned latent variables (rain factors depicting rain) physically representing direction, thickness, and scale, respectively.

A More Experiments for Model Verification

In this section, we provide the disentanglement experiments and latent space interpolation experiments. These experiments validate that our proposed generative model is indeed rational and it is able to finely capture the manifold of rain underlying the implicit distribution, which makes it possible to sample more diverse rain types through the generator.

A.1 Disentanglement Experiments

Fig. 6 shows the resulted rain layers by manipulating the latent code $z$ like the conventional disentanglement operations [50,51,62]. From the figure, we can easily observe that these latent variables well depict interpretable properties in generating rain layer, including direction, thickness, and scale. That is to say, the proposed VRGNet inclines to discover meaningful rain factors, which is finely in accordance with our modeling for rain layer by utilizing latent variables $z$ to encode such physical structural factors underlying rains as shown in Eq. (3) of the main text.

A.2 Latent Space Interpolation Experiments

We conduct interpolations of real images in the latent space to estimate the manifold continuity. For a pair of rainy RGB images as shown in the left subfigure of Fig. 7, we first utilize the inference model, i.e., RNet in Fig. 2 to map them to latent factors $z$ and then make linear interpolations with different weighting coefficients between their latent factor representations. The resulted rain layers are shown in the 3rd row of the right subfigure of Fig. 7. It is easy to observe that our rain generator function has continuity in the latent space in changing the direction of rain streaks and it has a fine capability to generate diverse rain types. Besides, the first two rows on the right of
Figure 7: Interpolation experiments. Left: a pair of rainy image used to obtain the corresponding latent code \( z \) through \( RNet \) for interpolating. The corresponding rain layers for the two rainy images are also depicted alongside for better observing the change of generated rain layer. Right: the interpolated results. The first two rows are the generated rainy image by adding the simulated rain layers as shown in the 3rd row on different backgrounds restored by the \( BNet \) in Fig. 2.

Fig. 7 are the generated rainy image by adding the simulated rain layers in the 3rd row on different backgrounds restored by the \( BNet \) in Fig. 2 which are quite different from the original input rainy images on the left of Fig. 7.

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 \( z \), but provided the corresponding rain layers which are easily obtained by subtracting backgrounds from the corresponding 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 of experiments, the first and the last frames are the rain layers corresponding to the pair of real rainy images, and between these two frames are the interpolated results.

B More Experimental Results

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

Representative Methods: We evaluate the effectiveness of VRGNet through augmented training on five latest DL-based methods, including DDN [29], PReNet [32], SPANet [26], JORDER_E [15], and RCDNet [34]. In the following experiments, 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 [7] and JCAS [9] for comprehensive comparisons.
B.1 More Results on Synthetic Data

**Synthetic Datasets:** Besides Rain100H, other two commonly-used benchmark datasets are also used, including Rain100L [15] and Rain1400 [29]. Specifically, Rain100L contains 200 rainy-clean image pairs for training and 100 ones for testing. Rain1400 includes 14 kinds of different rain streak orientations and magnitudes, and consists of 12600 image pairs for training and 1400 ones for testing.

Fig. 8 and Fig. 9 illustrate the intuitive 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) always has better visual quality, especially in the sense of better texture preservation and rain removal, than the corresponding one (1st row) trained on original training set.

B.2 More Results on Other Unlabeled Data

To further validate the generalization performance, we additionally introduce an real unlabeled benchmark dataset from [26], including 147 rainy images without groundtruth.

Fig. 10 displays the derained results on a real rainy image with dense and short rain streaks from [26]. As observed, traditional model-based DSC/JCAS leave obvious rain streaks in the derained results. However, these DL-based SOTA derainers trained on augmented Rain100L training set can achieve favorable visual quality when they are generalized to other unseen samples from [26]. This should be rationally attributed to the diversity of rain types generated by VRGNet.

C More Details of Network Architectures

As displayed in main text, the entire network architecture is constructed as Fig. 11, 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 of our focus on the careful design of every sub-network. Specifically, each sub-network adopted in our experiment, is illustrated as follows:

- **BNet** infers posterior parameters $\mu$ and $\sigma^2$ from $o$ and aims to restore the latent clean background $b$. We select the latest baseline network-PReNet [32] 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 $\alpha$ and $\beta$ for latent variable $z$, and it consists of 5 [$Conv + ReLU$] blocks and a [$Linear layer$] in turn.
**Figure 11:** The flowchart of the proposed variational rain generation network (VRGNet).

*Generator* represents the mapping $G(z; W_G)$ for generating rain patches from extracted latent variables $z$, which represents the sampling process from the implicit distribution $p_{un}(r|z)$. Symmetrically, it contains a [Linear layer] and 5 [Transpose Conv + ReLU] blocks. For back propagation, here is a reparameterization trick [10].

*Discriminator* aims to distinguish the training sample $o$ from the generated $\hat{o}$, which helps the learning of implicit distributions $p_{un}(r|z)$. Similar to most discriminators [43, 55], the sub-network is composed of 4 [Conv + LeakyReLU] blocks + 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 [54]. Besides, motivated by [55], we add the attention mechanism on the last two convolution layers to capture the global correlation in image.

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

## D More Analysis on the Role of BNet

As shown in Fig. [11] after the joint training, the *BNet* does not play any role in rain layer generation for data augmentation. However, this subnetwork is indeed necessary as analyzed in the following.

For convenience, we briefly denote VRGNet- as the model discarding *BNet* and directly regarding the rain-free image $x$ as the latent background $b$, as shown in Fig. [12]. In this setting, the posterior assumption $q(b|o) = \prod_i \mathcal{N}(b_i|\mu_i(o; W_B), \sigma^2_i(o; W_B))$ as Eq. (7) of the main text can be simply set as a Dirac distribution without any parameters, i.e.,

$$q(b|o) = \text{Dirac}_x(b),$$

where Dirac$_x(\cdot)$ means the Dirac distribution centered at point $x$. This hard assumption will lead to the degraded network framework displayed as Fig. [12]. As a special case, it indeed simplifies our proposed inference framework to some extent, but has stricter requirements for the accuracy of the estimated rain-free image $x$. However, if the pre-collected “rain-free” image $x$ is not sufficiently accurate, it will naturally degrade the training performance of *RNet* and 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. (7) 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 dataset–SPA-Data [28] in which the rain-free image $x$ is estimated based on multiple rainy images taken in the same condition, and thus is not the exact latent clean background $b$. We randomly select 1k pairs from the original ~600k pairs and augment them with different ratios (i.e., generate different number $N_f$ of fake pairs) for training. Table [4] reports the PSNR/SSIM averaged over 5 repetitions for each different augmentation ratio. From the table, we can easily observe that 1) Under each augmentation ratio setting, VRGNet significantly surpasses VRGNet- in terms of both PSNR and SSIM in average; 2) With the increase of $N_f$ from 0.5k to 2k, the average PSNR/SSIM results of VRGNet get better while that of VRGNet- becomes worse, which
Figure 12: The flowchart of VRGNet- that directly regards \( x \) as \( b \).

Table 4: Average PSNR and SSIM of PReNet on SPA-Data testing set. Training samples are composed of 1k real pairs randomly selected from \(~600k\) in SPA-Data and different number of fake pairs generated by VRGNet- and VRGNet, respectively. VRGNet- denotes the simplified VRGNet by removing BNet, as shown in Fig. 12. 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)     | \textbf{39.71}/0.9796 | \textbf{39.83}/0.9795 | \textbf{40.25}/0.9813 |

is mainly because VRGNet- does not capture the essential rain distribution without the guidance of BNet.