EfficientDeRain: Learning Pixel-wise Dilation Filtering for High-Efficiency Single-Image Deraining

Qing Guo[1*], **Jingyang Sun**[2] [*], **Felix Juefei-Xu**[3], **Lei Ma**[2†]
Xiaofei Xie[1†], **Wei Feng**[4], **Yang Liu**[1], **Jianjun Zhao**[2]

[1]Nanyang Technological University, Singapore
[2]Kyushu University, Japan   [3]Alibaba Group, USA
[4]College of Intelligence and Computing, Tianjin University, China

Abstract

Single-image deraining is rather challenging due to the unknown rain model. Existing methods often make specific assumptions of the rain model, which can hardly cover many diverse circumstances in the real world, compelling them to employ complex optimization or progressive refinement. This, however, significantly affects these methods' efficiency and effectiveness for many efficiency-critical applications. To fill this gap, in this paper, we regard the single-image deraining as a general image-enhancing problem and originally propose a model-free deraining method, *i.e.*, *Efficient-DeRain*, which is able to process a rainy image within 10 ms (*i.e.*, around 6 ms on average), over 80 times faster than the state-of-the-art method (*i.e.*, RCDNet), while achieving similar de-rain effects. We first propose the novel *pixel-wise dilation filtering*. In particular, a rainy image is filtered with the pixel-wise kernels estimated from a *kernel prediction network*, by which suitable multi-scale kernels for each pixel can be efficiently predicted. Then, to eliminate the gap between synthetic and real data, we further propose an effective data augmentation method (*i.e.*, *RainMix*) that helps to train network for handling real rainy images. We perform comprehensive evaluation on both synthetic and real-world rainy datasets to demonstrate the effectiveness and efficiency of our method. We release the model and code in https://github.com/tsingqguo/efficientderain.git.

1 Introduction

Rain patterns or streaks captured by outdoor vision systems (*e.g.*, stationary image or dynamic video sequence), often lead to sharp intensity fluctuations in images or videos, causing performance degradation for the visual perception systems (Garg and Nayar 2005, 2007) across different tasks, *e.g.*, pedestrian detection (Wang et al. 2018), object tracking (Li et al. 2018a), semantic segmentation (Chen et al. 2020), *etc*. As a common real-world phenomenon, it is almost mandatory that an all-weather vision system is equipped
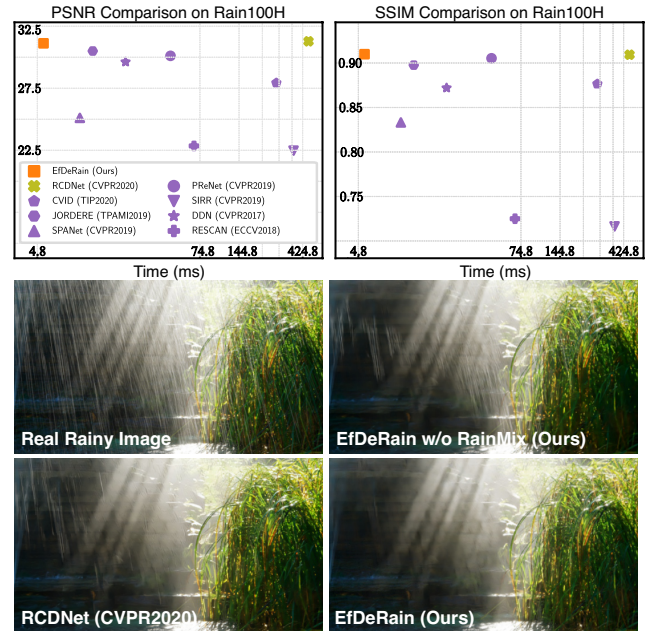

Figure 1: Top: Comparison results (*i.e.*, PSNR vs. Time & SSIM vs. Time) on the Rain100H dataset. Down: an example of using EfDeRain, EfDeRain without RainMix and the state-of-the-art RCDNet (Wang et al. 2020a) to handle a real rainy image. Note that, the time cost of all compared methods are one-by-one evaluated on the same PC.

with the deraining capability for usage. A deraining method processes the rain-corrupted image/video data and removes the rain streaks, with the intention to achieve good image quality for the downstream vision tasks.

In many real-time applications that are efficiency-sensitive and critical, (*e.g.*, vision-based autonomous driving or navigation), being able to perform deraining *efficiently* on-chip is of great importance. A deraining algorithm achieving both high efficiency and high performance (*e.g.*, in terms of PSNR, SSIM), while remaining at low overhead, is of great importance for practical usage.

---

*Qing Guo and Jingyang Sun are co-first authors.

†Lei Ma and Xiaofei Xie are corresponding authors: ma.lei@acm.org, xfxie@ntu.edu.sg.
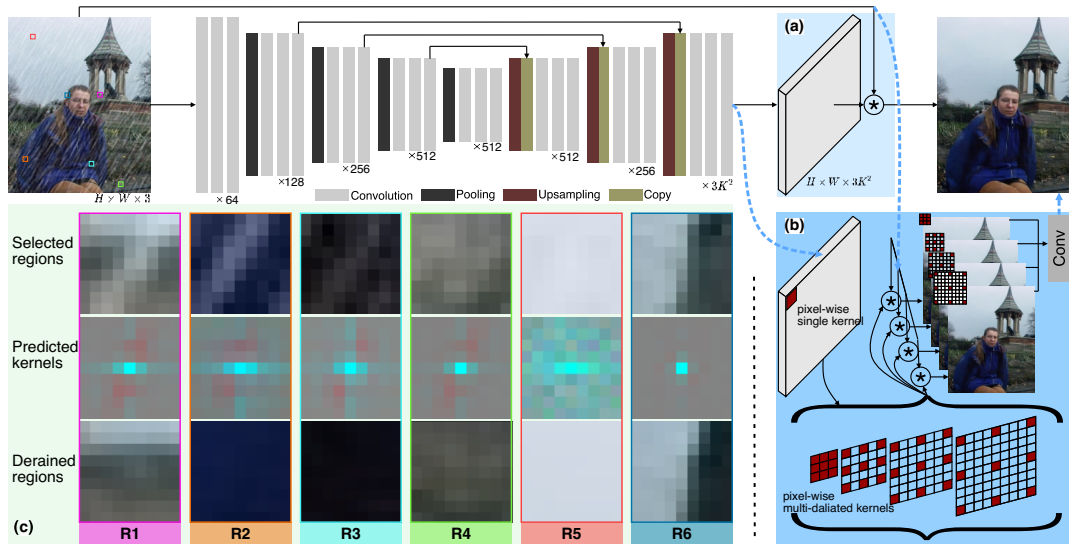
Figure 2: Pipeline of our *EfficientDeRain*. (a) represents the pixel-wise image filtering introduced in Sec. 3.1 where each pixel is processed by an exclusive kernel predicted by a kernel prediction network. (b) we further extend this simple structure and propose the pixel-wise dilation filtering in Sec. 3.2 to handle multi-scale rain streaks. (c) we show that the predicted kernels can be adapted to different image contents while recovering the object boundary in the original image.

Although we have witnessed the recent progress for deraining, existing methods mostly focus on studying the physical models of rain and background layers, removing rain streaks via solving an optimization problem and employing the power of deep learning and some priors, *e.g.*, the rain having similar local patterns across the image domain (Wang et al. 2020a; Li et al. 2016; Wei et al. 2019; Hu et al. 2019; Yang et al. 2019). An essential problem, *i.e.*, the efficiency of the derain method, however, is largely neglected so far, which brings great limitation for real-time applications.

In particular, existing methods (see Sec. 2) often heavily rely on various rain-generating assumptions and rain-background models, whose goal is to revert such a rain-adding step in the deraining process, involving heavy iterative optimization and subsequent refinement steps. However, some caveats of theirs cannot be overlooked: ❶ the rain model assumptions adopted by many of the algorithms may be limited, which do not well represent and reflect the real-world rain patterns. The models based on these rain model assumptions may not perform as strongly under the real-world scenarios; ❷ many of the existing methods are computationally expensive, either requiring complex iterative optimization to find the optimal solution, or constructing the deraining framework in multiple stages with recurrent or progressive refinement steps involved.

In this work, we approach the single-image deraining problem from a different angle, aiming to propose a efficient while general derain method. First, our proposed method is model-free, which makes no assumptions of how the rain is generated. As we show in the experimental section, rain model assumption is somewhat not mandatory for achieving high-performance deraining, and sometimes such assumptions can even impair the deraining performance. Second,

our proposed method follows a single-stage and does not require either iterative optimization or progressive refinement, thus leading to high-efficiency deraining. Overall, our main contributions are three-fold:

- We propose the *pixel-wise dilation filtering* for efficient deraining. A rainy image is filtered by the pixel-wise kernels estimated from a *kernel prediction network* that can automatically and efficiently predict the suitable multi-scale kernels for each pixel.

- To bridge the gap between synthetic and real data, we further propose the *RainMix* component for simple yet effective data augmentation, which enables us to train networks for handling real rainy images

- We demonstrate the advantage of our method on synthetic and real-world rainy datasets, achieving both high-performance and high-efficiency deraining. As shown in Fig. 1, our method (*i.e.*, EfDeRain) runs around 88 times faster than the state-of-the-art method RCDNet (Wang et al. 2020a) with similar performance in terms of PSNR and SSIM. Furthermore, equipped with the *RainMix*, EfDeRain can achieve much better visualization result on the real rainy images than RCDNet.

2 Related Work

Existing deraining algorithms can be largely categorized into two groups: video-based deraining, *e.g.*, (Kim, Sim, and Kim 2015), and single image deraining. In this section, we discuss several representative DL-based single-image deraining methods, which are most relevant to ours.

In (Wang et al. 2020a), a rain convolutional dictionary network (RCD-Net) is proposed for single image deraining, where the rain shapes are encoded by exploiting the in-
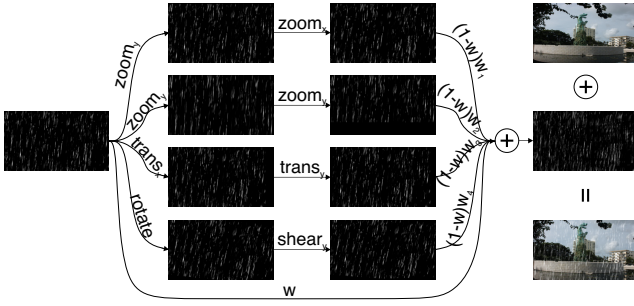
Figure 3: An example of RainMix to generate a rainy image.

trinsic convolutional dictionary learning mechanism, leveraging the proximal gradient technique as the optimization method to seek the optimal solution. RCD-Net is designed as multiple stages to iteratively solve the deraining problem. In (Du et al. 2020), a conditional variational image deraining (CVID) network is proposed for draining, which uses the conditional variational auto-encoder (CVAE) architecture as the backbone. As a variational method, multiple predictions of the derained images from the input rainy image can be generated by the learned decoder, and the final single-image output is obtained by integrating these predictions. In (Yang et al. 2019), a multi-task network to perform joint rain detection and removal (JORDER) is proposed to solve the inverse problem of single-image deraining. Contextual dilated networks are incorporated to extract regional contextual information so that the learned feature can be invariant to rain streaks. To further process rain streaks with various directions and shapes, a recurrent process is adopted to progressively remove the rain streaks in stages. In (Li et al. 2018b), a recurrent squeeze-and-excitation context aggregation net (RESCAN) is designed for single-image deraining. Similarly, the entire procedure is also performed in multiple stages, with a memory enabled RNN architecture to aggregate useful information of earlier stages. In (Wang et al. 2019), a spatial attentive network (SPANet) based on a two-round four-directional RNN architecture is proposed, where three standard residual blocks are used to extract features, and four spatial attentive blocks are used to identify rain streaks progressively in four stages.

Overall, these existing methods share some commonalities. In particular, they often heavily rely on a presumed rain model to develop their algorithms such as (Yang et al. 2019; Li et al. 2018b; Wang et al. 2020a), or require specific *a priori* domain knowledge of the rain streaks such as (Wang et al. 2019; Du et al. 2020; Fu et al. 2017). Again, many of these methods are built on a recurrent or progressive framework, where later stage deraining can help refine the early stage results such as (Yang et al. 2019; Li et al. 2018b; Wang et al. 2019, 2020a). However, we believe that these strong prerequisites and constraints can potentially hinder both universality and efficiency of the derain method and its real-world deployment. In this work, we originally propose a deraining method from a different perspective, aiming to address both issues.

Algorithm 1: Learning EfficientDeRain via RainMix

Input: KPN($\cdot$), fusion Convolution Conv($\cdot$), Loss function $\mathcal{L}$, Rainy Images $\mathcal{I}^{\mathrm{r}}$, Clean Images $\mathcal{I}$, real rain streak set $\mathcal{R}$, Operation set $\mathcal{O} = \{\mathrm{rot}, \mathrm{shear}_{x/y}, \mathrm{trans}_{x/y}, \mathrm{zoom}_{x/y}\}$.

Output: Pre-trained Network KPN($\cdot$) and Conv($\cdot$).

1 **Function** RainMix($\mathcal{R}$):
2   Sample a rain map $\mathbf{R}^{\mathrm{org}} \sim \mathcal{R}$;
3   Initialize an empty map $\mathbf{R}^{\mathrm{mix}}$;
4   Sample mixing weights $(w_1, w_2, w_3, w_4) \sim$ Dirichlet;
5   **for** $i = 1$ to 4 **do**
6   Sample operations $(\mathrm{o}_1, \mathrm{o}_2, \mathrm{o}_3) \sim \mathcal{O}$;
7   Combine via $\mathrm{o}_{12} = \mathrm{o}_2\mathrm{o}_1$ and $\mathrm{o}_{123} = \mathrm{o}_3\mathrm{o}_2\mathrm{o}_1$;
8   Sample $\mathrm{o} \sim \{\mathrm{o}_1, \mathrm{o}_{12}, \mathrm{o}_{123}\}$;
9        $\mathbf{R}^{\mathrm{mix}} + = w_i \mathrm{o}(\mathbf{R}^{\mathrm{org}})$
10   Sample a blending weight $w \sim$ Beta;
11   **return** $\mathbf{R} = w\mathbf{R}^{\mathrm{org}} + (1 - w)\mathbf{R}^{\mathrm{mix}}$;
12 **End function**;
13 **for** $i = 1$ to $|\mathcal{I}^{\mathrm{r}}|$ **do**
14   Generate rain map via $\mathbf{R} =$ RainMix($\mathcal{R}$);
15   Sample an image pair via $(\mathbf{I}^{\mathrm{r}}, \mathbf{I}) \sim (\mathcal{I}^{\mathrm{r}}, \mathcal{I})$;
16   Sample $\mathbf{X} \sim \{\mathbf{I}^{\mathrm{r}}, \mathbf{I}\}$ and Perform $\mathbf{I}^{\mathrm{rm}} = \mathbf{R} + \mathbf{X}$;
17   Predict pixel-wise kernels via $\mathbf{K} = \mathrm{KPN}(\mathbf{I}^{\mathrm{rm}})$;
18   Derain via Eq. (4) and get $\hat{\mathbf{I}} = \mathrm{Conv}(\{\hat{\mathbf{I}}_l\})$;
19   Calculate Eq. (5) and do back-propagation;
20   Update parameters of KPN($\cdot$) and Conv($\cdot$);

---

3 Methodology

3.1 Pixel-wise Image Filtering for Deraining

In this part, we propose an image filtering-based method for model-free deraining. The rain can be regarded as a kind of degradation that may cause effects such as occlusion, fog, motion blur, *etc*. As a result, it is reasonable to use image filtering methods to handle it, which can be efficient and effective for various degradations. Specifically, we process an input rainy image $\mathbf{I}^{\mathrm{r}} \in \mathbb{R}^{H \times W}$ [1] via pixel-wise filtering

$$\hat{\mathbf{I}} = \mathbf{K} \circledast \mathbf{I}^{\mathrm{r}}, \qquad (1)$$

where $\hat{\mathbf{I}} \in \mathbb{R}^{H \times W}$ is the estimated derained image, and $\circledast$ denotes the pixel-wise filtering operation where each pixel is processed by its exclusive kernel and $\mathbf{K} \in \mathbb{R}^{H \times W \times K^2}$ contains the kernels of all pixels. Specifically, when deraining the $\mathbf{p}$-th pixel of $\mathbf{I}^{\mathrm{r}}$, we have its exclusive kernel, *i.e.*, the vector at $\mathbf{p}$-th position of $\mathbf{K}$, which is reshaped and denoted as $\mathbf{K}_p \in \mathbb{R}^{K \times K}$. We use $\mathbf{p}$ as 2D coordinates for a pixel. Then, we can predict the derained pixel by

$$\hat{\mathbf{I}}(\mathbf{p}) = \sum_{\mathbf{t}, \mathbf{q} = \mathbf{p} + \mathbf{t}} \mathbf{K}_p(\mathbf{t}) \mathbf{I}^{\mathrm{r}}(\mathbf{q}), \qquad (2)$$

where $\mathbf{t}$ ranges from $(-\frac{K-1}{2}, -\frac{K-1}{2})$ to $(\frac{K-1}{2}, \frac{K-1}{2})$.

---

[1] We exemplify our method via the gray-scale image for better understanding. For color images, we can handle each channel independently.
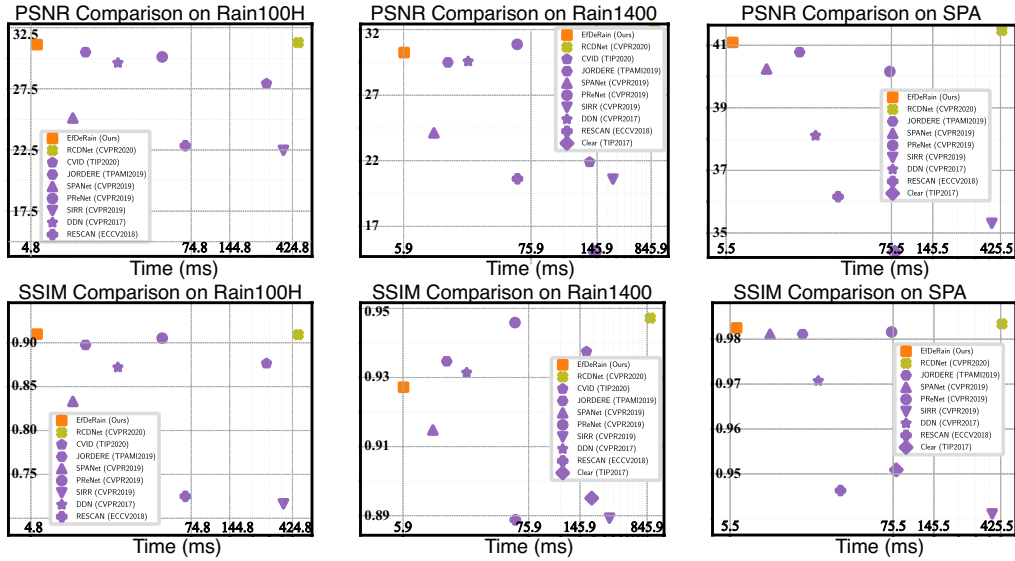
Figure 4: Comparing EfDeRain with nine baseline methods on Rain100H (Yang et al. 2017, 2019), Rain1400 (Fu et al. 2017), and the real SPA dataset (Wang et al. 2019) with PSNR vs. Time and SSIM vs. Time plots.

To realize effective deraining with the simple pixel-wise filtering, we should consider the following *challenges:* ❶ *how to estimate spatial-variant, scale-variant, and semantic-aware kernels effectively and efficiently.* Rain may cause streak occlusion, fog, and blur, at different positions with different appearances. For example, rain streaks could exhibit different scales, directions, and transparency across the image and semantically related to the image contents, *e.g.*, the scene depth (Hu et al. 2019). As a result, the pixel-wise kernels should be adapted to scene information, the spatial and scale variations of rain streaks. Obviously, hand-craft designed kernels can hardly satisfy such requirements. To address this challenge, we propose the *multi-dilated-kernel prediction network* in Sec. 3.2, which takes the rainy image as the input and predict multi-scale kernels for each pixel via the deep neural network (DNN). ❷ *how to train a powerful deraining DNN to bridge the gap to real rain with synthetic data.* Most of the existing deraining DNNs are trained on the synthetic data. However, there is still a gap between real and synthetic rain. Bridging this gap is of great importance for real-world applications. We propose a simple yet effective rain augmentation method, denoted as *RainMix* in Sec. 3.3 to reduce such a gap.

3.2 Learnable Pixel-wise Dilation Filtering

Kernel prediction network. Inspired by recent works on image denoising (Bako et al. 2017; Mildenhall et al. 2018), we propose to estimate the pixel-wise kernels $\mathbf{K}$ for deraining by taking the rainy image as input,

$$\mathbf{K} = \text{KPN}(\mathbf{I}^{\mathbf{r}}), \tag{3}$$

where $\text{KPN}(\cdot)$ denotes a UNet-like network and we show the architecture in Fig. 2. By offline training on rainy-clean image pairs, the kernel prediction network can predict spatial-variant kernels that adapt to the rain streaks with different

thickness and strength while preserving the object boundary. We show a deraining example in Fig. 2, where we validate our method on six representative regions and observe that: ❶ Our method can effectively remove the rain steak while recovering the occluded boundary, as shown in the region one (R1) (see Fig. 2 (c)). ❷ The predicted kernels can adapt to the rain with different strengths. As shown in Fig. 2 (c), from R2 to R5, the rain strength gradually becomes weak and our method can remove all trace effectively. Moreover, according to the visualization of predicted kernels, our network can perceive the positions of rain streak. As a result, the predict kernels assign higher weights to non-rainy pixels and lowers ones to rainy pixels, confirming the effectiveness of our method. ❸ According to the results on R6, our method does not harm the original boundary and makes it even sharper.

Multi-dilated image filtering and fusion. To enable our method to handle multi-scale streaks without harming the efficiency, we extend each predicted kernel to three scales via the idea of dilated convolution (Yu and Koltun 2016).

Intuitively, when the rain streak covers a large region of the image, the large scale kernel can use related pixels far from the rain for effective deraining. A simple solution is to predict multi-scale kernels directly, leading to extra parameters and time costs. Alternatively, we propose to extend the pixel-wise filtering in Eq. (1) to pixel-wise dilation filtering (Yu and Koltun 2016) for convolution layer,

$$\hat{\mathbf{I}}_l(\mathbf{p}) = \sum_{\mathbf{t}, \mathbf{q} = \mathbf{p} + l\mathbf{t}} \mathbf{K}_p(\mathbf{t}) \mathbf{I}^{\mathbf{r}}(\mathbf{q}), \tag{4}$$

where l is the dilation factor to control the applying range of the same filter. In practical, we consider four scales, *i.e.*, $l = 1, 2, 3, 4$. With Eq. (4), we get four derained images, *i.e.*, $\hat{\mathbf{I}}_1, \hat{\mathbf{I}}_2, \hat{\mathbf{I}}_3$, and $\hat{\mathbf{I}}_4$. Then, we use a convolution layer with the size 3×3 to fuse the four images and obtain the final output.
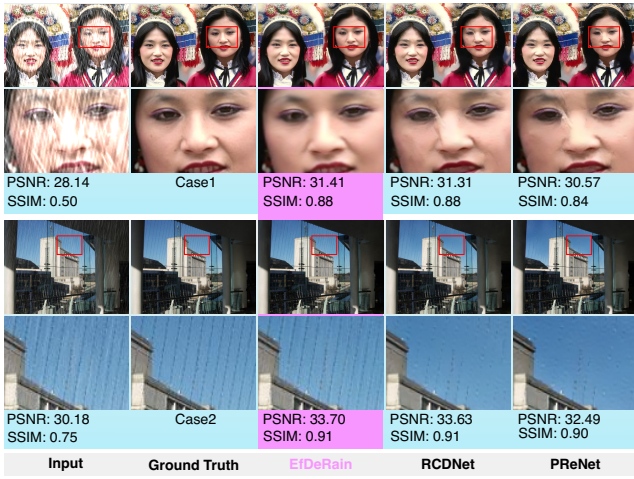
Figure 5: Two visualization results of EfDeRain, RCDNet (Wang et al. 2020a), and PReNet (Ren et al. 2019) on Rain100H (Case1) and Rain1400 (Case2). We magnify the main difference to highlight the advantages of our method.

Loss function. We consider two loss functions for training the network, *i.e.*, L_1 and SSIM. Given the derained image, *i.e.*, $\hat{\mathbf{I}}$, and the clean image $\mathbf{I}$ as ground truth, we have

$$\mathcal{L}(\hat{\mathbf{I}}, \mathbf{I}) = \|\hat{\mathbf{I}} - \mathbf{I}\|_1 - \lambda \, \mathrm{SSIM}(\hat{\mathbf{I}}, \mathbf{I}) \tag{5}$$

where we fix $\lambda = 0.2$ for all experiments.

3.3 RainMix: Bridging the Gap to Real Rain

How to reduce the gap of synthetic rainy images to real data is still an open problem. Inspired by recent data augmentation methods (Hendrycks et al. 2020), we explore a novel solution *RainMix* to address this challenge. Garg and Nayar (2006) conducted a detailed study about the appearance of rain streaks and constructed a dataset of real rain streaks by considering different lighting and viewing conditions. Even then, it is still hard to say the collected real rain is exhaustive, which covers all possible situations in the real world, since the rain has diverse appearances due to the influences of various natural factors, such as the wind, the light reflection, and refraction, *etc*. However, it is reasonable to use these real rain streaks to generate more rain streaks through transformations that may occur in the real world, *e.g.*, zooming, shearing, translation, and rotation. *RainMix* is originally designed based on this such intuition.

We show our RainMix-based learning algorithm in Algorithm 1. At each training iteration, we generate a rain map via *RainMix* and add it to the clean or rainy images, to obtain a new rain image for training the kernel prediction network and the fusion convolution layer. Our *RainMix* randomly samples a rain map from the real rain streak dataset (Garg and Nayar 2006) (*i.e.*, line 2 in Algorithm 1) and performs three transformations on the rain map via randomly sampled and combined operations (*i.e.*, line 5-9 in Algorithm 1). Finally, the three transformed rain maps are aggregated via the weights from Dirichlet distribution and further blended with the original sampled rain map via the weight from Beta

Figure 6: Three visualization results of EfDeRain, RCDNet (Wang et al. 2020a) and SPANet (Wang et al. 2019) on the real-world SPA dataset. The red arrow shows the main difference between EfDeRain and other two methods.

distribution (*i.e.*, line 4 and 11 in Algorithm 1). Intuitively, the multiple random processes simulate the diverse rain appearance patterns in the real world. We give an example of RainMix for generating a rainy image in Figure 3.

4 Experiments

4.1 Setups

Datasets. To comprehensively validate and evaluate our method, we conduct the comparison and analysis experiments on 4 popular datasets, including *Rain100H* (Yang et al. 2017, 2019), *Rain1400* (Fu et al. 2017) synthetic dataset, the recent proposed *SPA* real rain dataset (Wang et al. 2019), and the real *Raindrop* dataset (Qian et al. 2018).

Metrics. We employ the commonly used peak signal to noise ratio (PSNR) and structural similarity (SSIM) as the quantitative evaluation metric for all datasets. In general, the larger PSNR and SSIM indicate better deraining results.

Baselines. We perform a large-scale evaluation to compare with a total of 14 (=9+5) state-of-the-art deraining methods, *i.e.*, 9 baselines for the derain streak task (removing rain streak), including rain convolutional dictionary network (RCDNet) (Wang et al. 2020a), conditional variational image deraining (CVID) (Du et al. 2020), joint rain detection and removing (JORDERE) (Yang et al. 2019), spatial attentive deraining method (SPANet) (Wang et al. 2019), progressive image deraining network (PReNet) (Ren et al. 2019), semi-supervised transfer learning for rain removal (SIRR) (Wei et al. 2019), recurrent squeeze-and-excitation context aggregation net (RESCAN) (Li et al. 2018b), deep detail network (Fu et al. 2017), and Clear (Fu et al. 2017). Furthermore, for deraindrop task (*i.e.*, removing train drop) on the Raindrop dataset (Qian et al. 2018), we compare another

PSNR: 27.93 SSIM: 0.23	Case1	PSNR: 30.35 SSIM: 0.78	PSNR: 30.03 SSIM: 0.74	PSNR: 30.11 SSIM: 0.74	PSNR: 29.99 SSIM: 0.72
PSNR: 28.59 SSIM: 0.39	Case2	PSNR: 36.64 SSIM: 0.95	PSNR: 35.52 SSIM: 0.94	PSNR: 35.70 SSIM: 0.94	PSNR: 34.83 SSIM: 0.92
PSNR: 28.29 SSIM: 0.27	Case3	PSNR: 38.25 SSIM: 0.95	PSNR: 36.61 SSIM: 0.93	PSNR: 36.61 SSIM: 0.94	PSNR: 36.19 SSIM: 0.92

Figure 7: Three visualization results of four variants of EfDeRain: v1 denotes the pixel-wise filtering based on KPN without dilation; v2 is the pixel-wise dilation filtering for deraining. v1 and v2 are trained via the L_1 loss. v3 and v4 have the same structure with v2 but are trained via L_1 and SSIM loss functions, *i.e.*, Eq. (5). In addition, the v4 uses RainMix for training.

5 state-of-the-art methods (Li et al. 2019) as baselines, including GMM (Li et al. 2016), JORDER (Yang et al. 2017), DDN (Fu et al. 2017), CGAN (Zhang, Sindagi, and Patel 2019), DID-MDN (Zhang and Patel 2018), and DeRaindrop (Qian et al. 2018). Note that, the time cost of all methods are one-by-one evaluated on the same PC with the Intel Xeon CPU (E5-1650) and NVIDIA Quadro P6000 GPU.

4.2 Comparison on Rain100H&1400 Dataset

We compare our method with the 9 baseline methods in Fig. 4 on Rain100H and Rain1400 datasets. In general, our method, *i.e.*, EfDeRain, achieves the lowest time cost while obtaining comparable PSNR or SSIM with the top methods on both datasets. More specifically, on the challenging Rain100H dataset where the rain streaks cover a large portion of the image, we observe that EfDeRain achieves almost the same PSNR and SSIM with the rank1 method, *i.e.*, RCD-Net (Wang et al. 2020a), while being 88 times faster. Compared with other methods, *e.g.*, JORDERE (Yang et al. 2019) and CVID (Du et al. 2020), our method exhibits clear advantages on both recovery quality and efficiency. For example, EfDeRain achieves 11. 4% relative PSNR improvement over CVID while running over 50 times faster. Similarly, in terms of the Rain1400 dataset, our method still achieves the lowest time cost while maintaining the competitive PSNR and SSIM in line with the state-of-the-art methods, *e.g.*, PReNet and RCDNet. The main reason is likely to be that the rain streak of Rain1400 is slighter than Rain100H and a lot of the regions are not covered by the rain. Meanwhile, as shown in the Fig. 2, our method can not only remove rain streaks but enhance the object boundary, leading to the negative score of PSNR and SSIM that evaluate the recovery quality instead of the enhancement capability.

We further compare the visualization results of EfDeRain with the state-of-the-art baseline methods, *i.e.*, RCDNet and PReNet, in Fig. 5 and observe that: ❶ In Case1, EfDeRain can remove the rain streaks more effectively than the other two methods since there are clear rain traces on derained images, *i.e.*, the white streaks near the nose. ❷ Compared with

the baseline methods, EfDeRain removes the rain streak while recovering the original details well. In Case2, RCD-Net and PReNet remove or destroy the original streak-like details, *e.g.*, the fine lines. In contrast, our method preserves these details while removing the rain effectively, demonstrating that our method could understand the scene better and predict kernels for different pixels.

4.3 Comparison on Real-world SPA Rain Dataset

We further compare our method with 8 baseline methods on the SPA dataset (Wang et al. 2019), where the rainy image is real and its ground truth is obtained by human labeling and multi-frame fusion. As shown in Fig. 4, our method achieves almost the same PSNR and SSIM with the top method, *i.e.*, RCDNet, and outperforms all other baselines while running over 71 times more efficient than RCDNet.

We also visually compare our method with RCDNet and SPANet in Fig. 6. Obviously, all the results demonstrate that our method can handle various rain traces with different patterns and achieves better visualization results than RCDNet and SPANet. In particular, both RCDNet and SPANet fail to remove the wider rain streak in Case2 while our method successfully handles all rain streaks and the obtained derained image is almost the same with the ground truth.

4.4 Comparison on Real-world Raindrop Dataset

Besides the rainy streak image dataset, we also compare our method on the deraindrop task to show the generalization capability of our method. We train our network on the Raindrop dataset (Qian et al. 2018) and compare it with 6 state-of-the-art baseline methods. In particular, the method DeRaindrop (Qian et al. 2018) is specifically designed for this problem, where the region of the raindrop is perceived by an attentive-recurrent network. As shown in Table 1, our method without changing any architectures or hyperparameters, achieves the second-best results with competitive SSIM to DeRaindrop, and outperforms all other deraining methods, demonstrating both the effectiveness and generality of our method.

	GMM	JORDER	DDN	CGAN	DID-MDN	DeRaindrop	EfDeRaiin
PSNR	24.58	**27.52**	25.23	21.35	24.76	**31.57**	**28.48**
SSIM	0.7808	**0.8239**	0.8366	0.7306	0.7930	**0.9023**	**0.8971**

Table 1: Results on Raindrop dataset (Qian et al. 2018). The baseline results are reported by (Li et al. 2019) and (Qian et al. 2018). We highlight the top three results.

EfDeRain	w/o dilation(v1)	+dilation(v2)	+ssim loss(v3)	+RainMix(v4)
PSNR	30.12	30.27	30.35	31.02
SSIM	0.8834	0.8918	0.8970	0.9079
Time (ms)	4.41	6.05	5.93	5.97

Table 2: Ablation study on Rain100H. We consider 4 variants of EfDeRain: v1 is the pixel-wise filtering based on KPN without dilation structure; v2 is the pixel-wise dilation filtering for deraining. v1 and v2 are trained based on the L_1 loss. v3 and v4 have the same structure with v2 but are trained via L_1 and SSIM loss functions, *i.e.*, Eq. (5). The v4 uses RainMix for training.

4.5 Ablation Study

We validate the effectiveness of our main contributions, *i.e.*, pixel-wise dilation filtering (Sec. 3.2) and RainMix (Sec. 3.3), on Rain100H dataset. We also discuss the effectiveness of the loss function in Eq. (5). More specifically, we develop four variants of our method: the first version (*v1*) is the pixel-wise filtering based on KPN without the dilation structure; the second version (*v2*, *i.e.*, Sec. 3.2) denotes the pixel-wise dilation filtering for deraining. *v1* and *v2* are trained based on the L_1 loss. *v3* and *v4* share the same structure with *v2* but are trained via L_1 and SSIM loss functions, *i.e.*, Eq. (5). In addition, the final version (*v4*) uses RainMix in Sec. 3.3 for training. As shown in Table 2, the PSNR and SSIM of the 4 versions gradually increase and reach the highest performance on our final version with the pixel-wise dilation filtering trained via the RainMix as well as L_1 and SSIM loss functions. This demonstrates that all of our main contributions are beneficial for effective deraining. Moreover, we also analyze the time cost of the four variants and observe that: the proposed dilation filtering only leads to around 1.5 ms additional costs.

We further validate the advantages of contributions via the Rain100H visualization results in Fig. 7 and observe that: ❶ In general, our final version can not only remove the heavy rain streak but recovering the original details, thus achieving the highest PSNR and SSIM scores. ❷ When comparing the EfDeRain-*v1* with *v2* (*i.e.*, dilation-enhanced *v1*), we can conclude that the dilation structure clearly facilitates to remove more rain streaks. For example, in Case 1, the rain traces in *v1* have been obviously suppressed in *v2*. ❸ The SSIM loss function helps recover the details but enhances the rain streak. For example, in Case 1, with the SSIM loss function(*i.e.*, EfDeRain-*v3*), the boundary of the sun in *v3* becomes much sharper than that in *v2* and *v1*. However, the rain streak boundary becomes obvious as well. We observe similar results in other cases. ❹ By combing dilation structure, SSIM loss function, and RainMix, our final version, *i.e.*, EfDeRain-*v4*, is able to remove the heavy rain effectively while recovering the details very well.

5 Conclusions

In this paper, we propose a novel model-free deraining method denoted as *EfficientDeRain*. Our method can not only achieve the significantly high performance but runs over 80 times more efficient than the state-of-the-art method. Two major contributions benefit to the results: *First*, we proposed and designed the novel *pixel-wise dilation filtering* where each pixel is filtered by multi-scale kernels estimated from a pre-trained kernel prediction network. *Second*, we propose a simple yet effective data augmentation method for training the network, *i.e.*, *RainMix*, bridging the gap between synthesis data and real data. Finally, we perform a large-scale evaluation to comprehensively validate our method on popular and challenging synthesis datasets, *i.e.*, Rain100H and Rain1400, and real-world datasets, *i.e.*, SPA and Raindrop, all of which demonstrate the advantage of our method in terms of both efficiency and deraining quality.

In future, we will study the effects of deraining to other computer vision tasks, *e.g.*, object segmentation (Guo et al. 2018, 2017b) and object tracking (Guo et al. 2020c,a, 2017a), with the state-of-the-art DNN testing works (Xie et al. 2019a; Du et al. 2019; Xie et al. 2019b; Ma et al. 2018a,b, 2019). We also would like to study the single-image deraining from the view of adversarial attack methods, *e.g.*, (Guo et al. 2020b; Wang et al. 2020b; Cheng et al. 2020a; Gao et al. 2020; Cheng et al. 2020b; Tian et al. 2020).

References

Bako, S.; Vogels, T.; McWilliams, B.; Meyer, M.; Novák, J.; Harvill, A.; Sen, P.; DeRose, T.; and Rousselle, F. 2017. Kernel-predicting convolutional networks for denoising Monte Carlo renderings. *ACM Trans. Graph.* 36(4): 97:1–97:14.

Chen, Y.-C.; Lin, Y.-Y.; Yang, M.-H.; and Huang, J.-B. 2020. Show, Match and Segment: Joint Weakly Supervised Learning of Semantic Matching and Object Co-segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* .

Cheng, Y.; Guo, Q.; Juefei-Xu, F.; Xie, X.; Lin, S.-W.; Lin, W.; Feng, W.; and Liu, Y. 2020a. Pasadena: Perceptually Aware and Stealthy Adversarial Denoise Attack. *arXiv preprint arXiv:2007.07097* .

Cheng, Y.; Juefei-Xu, F.; Guo, Q.; Fu, H.; Xie, X.; Lin, S.-W.; Lin, W.; and Liu, Y. 2020b. Adversarial Exposure Attack on Diabetic Retinopathy Imagery. *arXiv preprint arXiv:2009.09231* .

Du, X.; Xie, X.; Li, Y.; Ma, L.; Liu, Y.; and Zhao, J. 2019. Deepstellar: Model-based quantitative analysis of stateful deep learning systems. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 477–487.

Du, Y.; Xu, J.; Zhen, X.; Cheng, M.-M.; and Shao, L. 2020. Conditional Variational Image Deraining. *IEEE Transactions on Image Processing* .

Fu, X.; Huang, J.; Ding, X.; Liao, Y.; and Paisley, J. 2017. Clearing the Skies: A Deep Network Architecture for Single-Image Rain Removal. *IEEE Transactions on Image Processing* 26(6): 2944–2956.

Fu, X.; Huang, J.; Zeng, D.; Huang, Y.; Ding, X.; and Paisley, J. 2017. Removing rain from single images via a deep detail network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3855–3863.

Gao, R.; ; Guo, Q.; Juefei-Xu, F.; Yu, H.; Ren, X.; Feng, W.; and Wang, S. 2020. Making Images Undiscoverable from Co-Saliency Detection. *arXiv preprint arXiv:2009.09258* .

Garg, K.; and Nayar, S. K. 2005. When does a camera see rain? In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, 1067–1074. IEEE.

Garg, K.; and Nayar, S. K. 2006. Photorealistic Rendering of Rain Streaks. In *ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, 996–1002.

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

Guo, Q.; Feng, W.; Zhou, C.; Pun, C.-M.; and Wu, B. 2017a. Structure-regularized compressive tracking with online data-driven sampling. *IEEE Transactions on Image Processing* 26(12): 5692–5705.

Guo, Q.; Han, R.; Feng, W.; Chen, Z.; and Wan, L. 2020a. Selective spatial regularization by reinforcement learned decision making for object tracking. *IEEE Transactions on Image Processing* 29: 2999–3013.

Guo, Q.; Juefei-Xu, F.; Xie, X.; Ma, L.; Wang, J.; Yu, B.; Feng, W.; and Liu, Y. 2020b. Watch out! Motion is Blurring the Vision of Your Deep Neural Networks. *Advances in Neural Information Processing Systems* 33.

Guo, Q.; Sun, S.; Dong, F.; Feng, W.; Gao, B. Z.; and Ma, S. 2017b. Frequency-tuned ACM for biomedical image segmentation. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 821–825. IEEE.

Guo, Q.; Sun, S.; Ren, X.; Dong, F.; Gao, B. Z.; and Feng, W. 2018. Frequency-tuned active contour model. *Neurocomputing* 275: 2307–2316.

Guo, Q.; Xie, X.; Juefei-Xu, F.; Ma, L.; Li, Z.; Xue, W.; Feng, W.; and Liu, Y. 2020c. SPARK: Spatial-aware online incremental attack against visual tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Hendrycks, D.; Mu, N.; Cubuk, E. D.; Zoph, B.; Gilmer, J.; and Lakshminarayanan, B. 2020. AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty. *International Conference on Learning Representations* .

Hu, X.; Fu, C.-W.; Zhu, L.; and Heng, P.-A. 2019. Depth-attentional features for single-image rain removal. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 8022–8031.

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

Li, F.; Tian, C.; Zuo, W.; Zhang, L.; and Yang, M.-H. 2018a. Learning Spatial-Temporal Regularized Correlation Filters for Visual Tracking. In *CVPR*, 4904–4913.

Li, S.; Araujo, I. B.; Ren, W.; Wang, Z.; Tokuda, E. K.; Junior, R. H.; Cesar-Junior, R.; Zhang, J.; Guo, X.; and Cao, X. 2019. Single Image Deraining: A Comprehensive Benchmark Analysis. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* .

Li, X.; Wu, J.; Lin, Z.; Liu, H.; and Zha, H. 2018b. Recurrent squeeze-and-excitation context aggregation net for single image deraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 254–269.

Li, Y.; Tan, R. T.; Guo, X.; Lu, J.; and Brown, M. S. 2016. Rain Streak Removal Using Layer Priors. In *Proceedings of the IEEE International Conference on Computer Vision*, 2736–2744.

Ma, L.; Juefei-Xu, F.; Sun, J.; Chen, C.; Su, T.; Zhang, F.; Xue, M.; Li, B.; Li, L.; Liu, Y.; Zhao, J.; and Wang, Y. 2018a. DeepGauge: Multi-Granularity Testing Criteria for Deep Learning Systems. In *The 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE)*.

Ma, L.; Juefei-Xu, F.; Xue, M.; Li, B.; Li, L.; Liu, Y.; and Zhao, J. 2019. DeepCT: Tomographic Combinatorial Testing for Deep Learning Systems. *Proceedings of the IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)* .

Ma, L.; Zhang, F.; Sun, J.; Xue, M.; Li, B.; Juefei-Xu, F.; Xie, C.; Li, L.; Liu, Y.; Zhao, J.; and Wang, Y. 2018b. DeepMutation: Mutation Testing of Deep Learning Systems. In *The 29th IEEE International Symposium on Software Reliability Engineering (ISSRE)*.

Mildenhall, B.; Barron, J. T.; Chen, J.; Sharlet, D.; Ng, R.; and Carroll, R. 2018. Burst Denoising with Kernel Prediction Networks. In *CVPR*, 2502–2510.

Qian, R.; Tan, R. T.; Yang, W.; Su, J.; and Liu, J. 2018. Attentive Generative Adversarial Network for Raindrop Removal From a Single Image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Ren, D.; Zuo, W.; Hu, Q.; Zhu, P.; and Meng, D. 2019. Progressive Image Deraining Networks: A Better and Simpler Baseline. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Tian, B.; Guo, Q.; Juefei-Xu, F.; Chan, W.; Cheng, Y.; Li, X.; Xie, X.; and Qin, S. 2020. Bias Field Poses a Threat to DNN-based X-Ray Recognition. *arXiv preprint arXiv:2009.09247* .

Wang, H.; Xie, Q.; Zhao, Q.; and Meng, D. 2020a. A Model-driven Deep Neural Network for Single Image Rain Removal. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3103–3112.

Wang, R.; Juefei-Xu, F.; Xie, X.; Ma, L.; Huang, Y.; and Liu, Y. 2020b. Amora: Black-box adversarial morphing attack. In *ACM Multimedia Conference (ACMMM)*.

Wang, T.; Yang, X.; Xu, K.; Chen, S.; Zhang, Q.; and Lau, R. W. 2019. Spatial attentive single-image deraining with a high quality real rain dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 12270–12279.

Wang, X.; Xiao, T.; Jiang, Y.; Shao, S.; Sun, J.; and Shen, C. 2018. Repulsion Loss: Detecting Pedestrians in a Crowd. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7774–7783.

Wei, W.; Meng, D.; Zhao, Q.; Xu, Z.; and Wu, Y. 2019. Semi-Supervised Transfer Learning for Image Rain Removal. In *Proceedings of the IEEE International Conference on Computer Vision*.

Xie, X.; Ma, L.; Juefei-Xu, F.; Xue, M.; Chen, H.; Liu, Y.; Zhao, J.; Li, B.; Yin, J.; and See, S. 2019a. DeepHunter: A Coverage-Guided Fuzz Testing Framework for Deep Neural Networks. In *ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA)*.

Xie, X.; Ma, L.; Wang, H.; Li, Y.; Liu, Y.; and Li, X. 2019b. DiffChaser: Detecting Disagreements for Deep Neural Networks. In *IJCAI*, 5772–5778.

Yang, W.; Tan, R. T.; Feng, J.; Guo, Z.; Yan, S.; and Liu, J. 2019. Joint rain detection and removal from a single image with contextualized deep networks. *IEEE transactions on pattern analysis and machine intelligence* 42(6): 1377–1393.

Yang, W.; Tan, R. T.; Feng, J.; Liu, J.; Guo, Z.; and Yan, S. 2017. Deep Joint Rain Detection and Removal from a Single Image. In *CVPR*, 1685–1694.

Yu, F.; and Koltun, V. 2016. Multi-Scale Context Aggregation by Dilated Convolutions. In *International Conference on Learning Representations*.

Zhang, H.; and Patel, V. M. 2018. Density-Aware Single Image Deraining Using a Multi-stream Dense Network. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 695–704.

Zhang, H.; Sindagi, V.; and Patel, V. M. 2019. Image De-raining Using a Conditional Generative Adversarial Network. *IEEE Transactions on Circuits and Systems for Video Technology* 1–1.