FLUID: Few-Shot Self-Supervised Image Deraining

Shyam Nandan Rai, Rohit Saluja, Chetan Arora, Vineeth N Balasubramanian, Anbumani Subramanian, C.V. Jawahar
CVIT - IIIT Hyderabad, India
IIT Hyderabad, India
IIT Delhi, India
shyamnandanraii@gmail.com, rohit.saluja@research.iiit.ac.in, vineethnb@iith.ac.in, chetan@cse.iitd.ac.in, {anbumani, jawahar}@iiit.ac.in

Abstract

Self-supervised methods have shown promising results in denoising and dehazing tasks, where the collection of the paired dataset is challenging and expensive. However, we find that these methods fail to remove the rain streaks when applied for image deraining tasks. The method’s poor performance is due to the explicit assumptions: (i) the distribution of noise or haze is uniform and (ii) the value of a noisy or hazy pixel is independent of its neighbors. The rainy pixels are non-uniformly distributed, and it is not necessarily dependant on its neighboring pixels. Hence, we conclude that the self-supervised method needs to have some prior knowledge about rain distribution to perform the deraining task. To provide this knowledge, we hypothesize a network trained with minimal supervision to estimate the likelihood of rainy pixels. This leads us to our proposed method called FLUID: Few Shot Self-Supervised Image Deraining.

We perform extensive experiments and comparisons with existing image deraining and few-shot image-to-image translation methods on Rain 100L and DDN-SIRR datasets containing real and synthetic rainy images. In addition, we use the Rainy Cityscapes dataset to show that our method trained in a few-shot setting can improve semantic segmentation and object detection in rainy conditions. Our approach obtains a mIoU gain of 51.20 over the current best-performing deraining method. [Project Page]

1. Introduction

Deep learning models require large-scale datasets to learn a computer vision task. Applications such as autonomous navigation systems use many paired images to generalize across different adverse weather conditions. The collection of such a dataset is an expensive and tedious task. Self-supervised methods [16] were introduced, which gave marginal performance behind the supervised methods on various downstream tasks while avoiding the dependency on large-scale labeled datasets.

Figure 1. Image deraining results by the self-supervised method: N2S [47] (col. 2), semi-supervised method: Yasarla et al. [50] (col. 3), and our method: FLUID (col. 4). We can observe N2S [47] fail to remove rainy streaks due to poor prior knowledge about rain distribution, whereas, Yasarla et al. [50] suffer from image artifacts due to its sensitivity to the training sample choice. For a fair comparison, we train all the baselines methods in a few-shot unsupervised setting.

Recently, self-supervised methods have shown good results for image denoising [1, 19, 47] and image dehazing [21]. However, these methods explicitly mention that the following assumptions are made to perform denoising or dehazing tasks: (a) noise or haze is uniformly distributed across an image, and (b) a noisy or hazy pixel value is independent of its neighboring pixels. Applying such self-supervised techniques for image deraining gives poor results (Figure 1: col. 2) because a rainy pixel might/might not depends on nearby pixels, and rain pixels are non-uniformly distributed in the image, unlike haze and noise. This concludes that self-supervised techniques require prior knowledge about rain distribution to
perform image deraining tasks. To provide this knowledge to the self-supervised network, we hypothesize a network trained with minimal supervision to estimate the likelihood of rainy pixels. This leads us to the proposed method: Few-Shot Self-Supervised Image Deraining (FLUID). Figure 1 shows the FLUID deraining results in comparison with self-supervised and semi-supervised methods on real and synthetic images.

The FLUID framework consists of three stages. In the first stage, we train a Probability Estimation Network (PEN) trained in a few-shot setting that predicts the pixel-wise rain likelihood in an image. The trained PEN network helps to provide prior knowledge about rain distribution. In PEN, we predict pixel-wise rain likelihood instead of learning non-rainy pixels. The network can learn to predict rainy pixels independent of textural information present in training images. Recent semi-supervised deraining method [46, 50] performs poorly when trained in a few-shot unsupervised setting. This is because the objective function minimizes the loss between the rainy and clean image pair, enabling to learn the textural image information. This claim becomes evident in Figure 1: col. 3 where we compare our method's performance with the semi-supervised method [50]. This brings us to the conclusion that semi-supervised methods are sensitive to the choice of training samples which is evident from the color shift caused by the choice of the training image.

In the next stage, we use the trained PEN to predict the pixel-wise rain probability that helps identify and mask the rainy regions in the images. We then fill the masked area using image inpainting. The inpainted output acts as a prior to a Self-Supervised Network (SSN). In the last stage, we pass the inpainted output to the SSN. With sufficient prior knowledge about the rain distribution, the SSN can further derain the image and remove image artifacts and blurriness introduced by image inpainting. The efficacy of our proposed model is evaluated on Rain 100L and DDN-SIRR containing natural and synthetic images to show our deraining ability. Our ablation study establishes that our method's performance is consistent irrespective of the choice of the training samples.

- Demonstrates that using derained images from the FLUID framework significantly improves semantic segmentation and object detection compared to existing deraining approaches.

2. Related Works

Single Image Deraining: Single image deraining [49] is the task of generating rain-free images that have been extensively researched over the past few decades. There are also video-based deraining techniques [23, 40, 54], but single image deraining is more challenging due to temporal information's unavailability. We can divide all the single deraining methods into two categories: model-based and deep-learning based methods.

Model-based methods or non-deep learning methods utilize dictionary learning [4, 29], prior-based [56], sparsity-based model [7, 45], and mixture-model based [26] to get the derained images. However, the methods mentioned above struggle to generalize over variations in rainy streaks. Recently, deep-learning models have shown state-of-the-art performance in various computer vision tasks due to efficient feature learning. Leveraging the advantage of deep learning models, Yang et al. [48] proposed a deep network that can detect and remove rain. Later, new approaches were proposed, which were based on Convolutional Neural Network (CNN) [25, 51], generative models [52], and physics-driven models [24].

However, the methods mentioned earlier tend to fail when tested on real rainy images. Wei et al. [46] proposed an efficient semi-supervised approach that used synthetic rainy pair images and unlabeled real rainy images. This approach, without proper initialization, will lead to suboptimal results [50]. Yasarla et al. [50] presented an improved semi-supervised method that used the Gaussian process to leverage the information from unlabeled real rainy images while training. However, these methods perform poorly in few-shot unsupervised settings as they are sensitive to the training image pairs.

Few-Shot Image-to-Image Translation: Few-shot learning for image classification [10, 32] is a widely studied problem. Recently, Liu et al. [28] proposed a method to generate images of unseen classes with only a few samples provided at the testing phase. Later, other few-shot generation methods were proposed for face reenactment [12], interactive video stylization [43], and font style transfer [22]. However, we find that when trained in a few-shot unsupervised setting, the few-shot methods...
struggle to minimize the artifacts by adverse weather conditions.

Self-Supervised Learning: Self-supervised learning [16] refers to the learning of visual features from the unlabeled dataset. This framework trains a network to solve the pretext task using the pseudo-labels generated from a dataset without human supervision. Doersch et al. [8] proposed the first self-supervised learning method that used a pretext task of predicting image patches’ relative position, which improves object detection tasks. Later self-supervised approaches used the pretext tasks such as solving jigsaw puzzles [33], image rotation estimation [18], super-resolution [20], colorization [53], and inpainting [34].

Recent self-supervised denoising methods such as Noise2Void [19], Noise2Self [1], and Noise2Same [47] does not depend on prior noise information for denoising. Although, the availability of noise information further improved the performance. The success of self-supervised models in denoising motivated us to use such frameworks for image deraining.

3. FLUID: Few-Shot Self-Supervised Image Deraining

3.1. Overview

We begin the formulation of the framework by a set of rainy images: \( I^L = \{I^L_i : i = 1, 2, \ldots, n\} \) and the corresponding clean images: \( I = \{I_i : i = 1, 2, \ldots, n\} \). The value of \( n \) in our framework is 1, 3, and 5. The unpaired rainy image set without the clean image is denoted by: \( I^{NL} = \{I^{NL}_i : i = 1, 2, \ldots, m\} \), where \( m \gg n \). Firstly, we train a Probability Estimation Network (PEN) on \( I^L \) and \( I \) to get the pixel-wise rain probability of an image. We then use the trained PEN network to get pixel-wise

\[ f_P(x) = P_r(x/I^L_i) \]

(1)

We learn the function \( f_P(x) \) by training a UNet [38] to estimate pixel-wise rain probability. We train the UNet on

![Figure 2. Overview of FLUID framework: The FLUID framework consists of three stages. A) Rain Probability Estimation: We train a Probability Estimation Network (PEN) that predicts the pixel-wise rain probability of an image. The trained PEN network helps in the generation of prior knowledge for the Self-Supervised Network (SSN). B) Prior Generation: We pass unpaired rainy images \( I^{NL} \) to estimate pixel-wise rain probability in this stage. The predicted pixel-wise rain probability map values are thresholded \( Th \) by giving 0 to rainy pixel and 1 to non-rainy. Then, we perform element-wise multiplication between the \( I^{NL} \) and its corresponding thresholded probability map \( P^{NL} \). As a result, the rainy regions are masked out. We then fill the masked area through image inpainting \( IN \). The inpainted output \( I^p \) acts as a prior for SSN. C) Self-Supervised Learning: Finally, the generated prior \( I^p \) trains the SSN that refines the results further by minimizing the image artifacts introduced by image inpainting and the tiny rain streaks that are undetected by PEN.](image)

![Figure 3. Visualization of output at various stages: Left to Right: col. 1: input rainy image, col. 2: PEN output, col. 3: masking of rain by taking the dot product between rainy image and PEN output, col. 4: output obtained from image inpainting, col. 5: refined output from Self-Supervised Network.](image)
binary cross-entropy loss which is given by:

\[
l_{\text{PEN}} = \frac{1}{N} \sum_{x=1}^{N} P_i^{L}(x) \log(f_P(x)) + (1 - P_i^{L}(x)) \log(1 - f_P(x))
\]

(2)

\(P_i^{L}(x)\) represents the ground truth rain probability of \(I_i^{L}(x)\) at location \(x\) and \(N\) is the total number of pixels.

In PEN, we predict pixel-wise rain likelihood instead of learning non-rainy pixels since rain streaks are mostly textureless. Hence, the trained PEN will predict rainy pixels independent of textural information present in training images. Data augmentation plays a significant role in improving the rain detection capabilities of PEN across various rain patterns shown in Supp. Sec. 1. Figure 3: col. 2 shows the rain streaks predicted by PEN.

### 3.3. Prior Generation

We now use the trained PEN to generate the priors \(I^p\) for SSN. We pass the unpaired rainy images \(I^{NL}\) through trained PEN to generate pixel-wise rain probability. The output probability map inferred from PEN is thresholded \(th\) to 0 for rainy pixels and 1 for rainy pixels, which results in \(P^{NL}\). We then perform element-wise multiplication between \(I^{NL}\) and \(P^{NL}\) that masks the rainy regions. Figure 3: col. 3 shows the masked image. Now, we fill the masked areas by image inpainting \(f_{IN}\) that gives the prior for SSN shown in Figure 3: col. 4. The entire process can be formulated as:

\[
I^p = f_{IN}(P^{NL} \odot I^{NL})
\]

(3)

We used a statistical inpainting method by Damelin et al. [6] for the image inpainting task. We did not use pre-trained inpainting network for inpainting as they can give biased results based on the training dataset.

### 3.4. Self-Supervised Learning

The generated prior have blurry regions introduced by image inpainting and have tiny rainy streaks undetected by PEN. Now, we use SSN to improve the quality of prior and further derained the image to get the final derained image \(I\). Firstly, we pass \(I^p\) and \(I^{NL}\) through the SSN as an input. Next, we calculate mean square loss \(l_{mse}\) between \(I^p\) and \(I\) to retrieve the average prior content. Now, we use total variation loss [2] denoted by \(l_{tv}\) on \(I\) to minimize the tiny streaks which are undetected by the PEN. \(l_{tv}\) and \(l_{mse}\) smoothens the output image which reduces the high frequency detail. Hence, we use VGG loss [20] denoted by \(l_{vgg}\) that adds high-frequency details into \(I\). We calculate the \(l_{vgg}\) using features map denoted by \(\phi_{vgg}\) obtained from ReLU activation layers of the pretrained VGG16 [42] network. Figure 4 illustrates the flow of input and output to the SSN along with various training losses. The final objective is given by:

\[
l_{SSN}((P_i^{NL}; I_i^{NL}), \hat{I}_i) = l_{mse} + \lambda_1 l_{tv} + \lambda_2 l_{vgg}
\]

(4)

\[
l_{SSN}((P_i^{NL}; I_i^{NL}), \hat{I}_i) = \frac{1}{N} \sum_{x=1}^{N} ||I_i^{NL}(x) - \hat{I}_i(x)||_2 +
\]

\[
\frac{\lambda_1}{N} \sum_{x=1}^{N} ||I_i^{NL}(x)||_1 + \frac{\lambda_2}{M} \sum_{x=1}^{M} ||\phi_{vgg}(I_i^{NL}(x)) - \phi_{vgg}(\hat{I}_i(x))||_2
\]

(5)

\(I^p\) and \(\hat{I}_i\) represent the value of prior and its corresponding SSN output at pixel \(x\), respectively. \(M\) denotes \(\phi_{vgg}\) dimension. \(\lambda_1\) and \(\lambda_2\) are the hyperparameter empirically estimated during the network’s training. Figure 3: col. 5 shows the SSN output.

### 4. Experimentation

#### 4.1. Experimental Settings

##### 4.1.1 Dataset and Evaluation

**Rain 100L:** Yang et al. [48] synthesized the dataset using the rain streak rendering method by Garg et al. [11] on the clean images of BSD200 [30]. It consists of 200 pairs of training images and 100 pairs of test images. We divide the training image pairs into two parts for our experiments: 5 image pairs for training and 195 image pairs for validation.

**DDN-SIRR:** This dataset consists of synthetic rainy and rain-free image pairs and unpaired natural rainy images created by Wei et al. [46]. The rain-free images were taken from the UCID [41] dataset. We use the synthetic dataset in our few-shot experiments by randomly choosing five image pairs for training and 400 image pairs for validation and testing. Further, we test the trained model on a set of 100 real rainy images having dense and sparse rain streak.

**Rainy Cityscapes:** Halder et al. [13] proposed a physics-based rain rendering method to inject rain into the clean images realistically. Using this method, Halder et al. [13] creates a rainy cityscapes dataset consisting of rain and rain-free images of Cityscapes [5]. We use this dataset to show improvement in semantic segmentation. We randomly
choose five training image pairs, 300 validation image pairs, and 200 image pairs for the test.

**Evaluation Metrics:** We use Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) to evaluate the performance of deraining methods for the synthetic datasets as the ground-truth is available. For natural rainy images, we use Blind/Referenceless Image Spatial Quality Evaluator (BRISQUE) [31].

### 4.1.2 Training Details

We train the PEN on twenty thousand epochs with batch size 1. The initial learning rate is $1e^{-4}$, which is reduced to $1e^{-5}$ after ten thousand epochs. We train the SSN for 500 epochs with a learning rate of $1e^{-3}$ and a batch size of 16. While training both the networks, the input is given by randomly cropping $128 \times 128$ image patch, which is randomly rotated between $(-180^\circ, 180^\circ)$. The values of $\lambda_1$ and $\lambda_2$ in Eq 5 were empirically found best to be $1e^{-3}$ and 0.04. We use the value of $Th$ to be 0.95.

### 4.1.3 Baselines

Since there is no previous few-shot image deraining work, we baseline FLUID performance with a) few-shot/unsupervised/supervised image-to-image translation methods and b) semi/fully supervised deraining methods. We train all the methods in a few-shot unsupervised setting for a fair comparison, i.e., only a few rainy/clean image pairs were given, rest were unpaired during training.
Table 1. **Quantitative comparison (PSNR/SSIM):** Results in blue background shows the performance of our method. Orange and green background shows the results of supervised and semi-supervised methods in few-shot setting. Gray background shows the results when we train the network with all the training samples.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>PSNR / SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rain 100L [1Shot]</td>
<td>ID [17] (TIP’12)</td>
<td>23.13 / 0.70</td>
</tr>
<tr>
<td></td>
<td>CNN [9] (ICCV’13)</td>
<td>23.70 / 0.81</td>
</tr>
<tr>
<td></td>
<td>DSC [29] (ICCV’15)</td>
<td>24.16 / 0.87</td>
</tr>
<tr>
<td></td>
<td>LP [26] (CVPR’16)</td>
<td>25.91 / 0.89</td>
</tr>
<tr>
<td></td>
<td>DerainDrop [35] (CVPR’18)</td>
<td>15.69 / 0.53</td>
</tr>
<tr>
<td></td>
<td>RESCAN [25] (ECCV’18)</td>
<td>17.44 / 0.59</td>
</tr>
<tr>
<td></td>
<td>SPANet [44] (CVPR’19)</td>
<td>18.46 / 0.65</td>
</tr>
<tr>
<td></td>
<td>ID-CGAN [52] (TCSVT’19)</td>
<td>18.66 / 0.68</td>
</tr>
<tr>
<td></td>
<td><strong>FLUID (Ours)</strong></td>
<td><strong>26.87 / 0.86</strong></td>
</tr>
</tbody>
</table>

| DDN-SIRR [1Shot] | ID [17] (TIP’12) | 23.13 / 0.70 |
| | CNN [9] (ICCV’13) | 23.70 / 0.81 |
| | DSC [29] (ICCV’15) | 24.16 / 0.87 |
| | LP [26] (CVPR’16) | 25.91 / 0.89 |
| | DerainDrop [35] (CVPR’18) | 15.69 / 0.53 |
| | RESCAN [25] (ECCV’18) | 17.44 / 0.59 |
| | SPANet [44] (CVPR’19) | 18.46 / 0.65 |
| | ID-CGAN [52] (TCSVT’19) | 18.66 / 0.68 |
| | **FLUID (Ours)** | **26.87 / 0.86** |

Table 2. Result comparison of deraining method with FLUID on Rain 100L dataset. The methods above the dotted line are trained on full training dataset and the below are trained in 5-shot unsupervised setting. Supv. denotes supervision.

We compare our framework performance with i) supervised: Pix2Pix [15], ii) unsupervised: UNIT [27], CycleGAN [55], and MUNIT [14], and iii) few-shot: UNIT [27], CycleGAN [55], and MUNIT [14] in 1-shot unsupervised setting. Qualitative results in Figure 6 show that our method can minimize the rain streaks, whereas the baseline methods suffer from image artifacts. Table 2 shows the performance comparison of our method with the supervised methods: ID [17], CNN [9], DSC [29], LP [26], DerainDrop [35], SPANet [44], RESCAN [25], and ID-CGAN [52] trained on Rain 100L dataset. We can observe in Table 2 (row: 1-4, 9), our method trained only on 5-shot setting achieves better PSNR compared to initial deraining methods: ID [17], CNN [9], DSC [29], LP [26], DerainDrop [35], SPANet [44], RESCAN [25], and ID-CGAN [52] trained on Rain 100L dataset. We also observe in Table 2 (row: 5-9), our model significantly outperforms recent deraining methods: DerainDrop [35], SPANet [44], RESCAN [25], and ID-CGAN [52] in 5-shot setting. Quantitatively, we get 8.21 / 0.18 PSNR/SSIM gain over the best-supervised method.

Next, we compare our proposed method with Wei et al. [46], Yasarla et al. [50], RESCAN [25], ID-CGAN [52], and Rainy2Clean on the test set of Rain 100L and DDN-
### Table 4. Performance evaluation on real rainy images (DDN-SIRR). (↓) indicates lower the score better the performance.

<table>
<thead>
<tr>
<th>Method</th>
<th>BRISQUE Score ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rainy Image</td>
<td>32.28</td>
</tr>
<tr>
<td>Yasarla et al. [50]</td>
<td>31.93</td>
</tr>
<tr>
<td>Ours</td>
<td>30.67</td>
</tr>
<tr>
<td>Rainy2Clean</td>
<td>27.89</td>
</tr>
</tbody>
</table>

**Figure 7. Real rainy (DDN-SIRR) results in 5-shot setting.** (a) Input rainy image. (b) Results from semi-supervised method: Yasarla et al. [50]. (c) Derained output from our proposed method. (d) Results from the method trained on all training samples: Rainy2Clean. We observe that our approach shows performance close to Rainy2Clean. Notably, it works better in removing rainy streaks, as observed in the bottom row results.

**Figure 8. Deraining performance with and without self-supervised network:** We observe SSN can minimize the image artifacts by inpainting and the rain streaks left undetected by PEN. SIRR dataset in 1-shot, 3-shot, and 5-shot setting. Figure 5 and Table 1 shows the qualitative and quantitative results. We observe that our model outperforms the other deraining methods in the few-shot settings. We find that semi-supervised methods [46, 50] struggle to remove the rain and cannot retain the input image statistics. This is because of the poor latent representation learned by their supervised networks. The visual results of fully supervised methods [25, 52] are lower than semi-supervised methods as they do not have the choice to improve their latent representation of their model using real rainy images. Figure 7 and Table 4 show our method’s performance on real rainy images of the DDN-SIRR dataset trained in the 5-shot setting. We observe in Figure 7, our method acts more effectively in removing the rain streaks than Rainy2Clean.

### 4.3. Effectiveness of SSN

We investigate the effectiveness of using SSN in our deraining framework by defining various methods with different losses (Section 3). w/o SSN: Train without SSN. \( M^1 \): Train with SSN on \( l_{mse} \). \( M^2 \): Train with SSN on \( l_{mse} \) and \( l_{tv} \). w/ SSN: Train with SSN on \( l_{mse} \), \( l_{tv} \), and \( l_{vgg} \). We trained all the methods on the Rain 100L dataset and presented the results in Table 6. We can see w/ SSN shows the best performance demonstrating the effectiveness of the combination of loss used in SSN. In Figure 8, we observe that w/ SSN can minimize the image artifacts due to image inpainting and the rain streaks that are undetected by PEN.

### 4.4. Generalization

We demonstrate our proposed method’s performance consistency irrespective of the training pair in a 1-shot set-
5. Applications

In this section, we employ FLUID in improving semantic segmentation and object detection under rainy conditions.
References

[22] Chenhao Li, Yuta Taniguchi, Min Lu, and Shin’ichi Konomi. Few-Shot Font Style Transfer Between Different Languages. In WACV, 2021. 2
[23] Minghu Li, Xi Qie, Qian Zhao, Wei Wei, Shuhang Gu, Jing Tao, and Deyu Meng. Video Rain Streak Removal By Multiscale Convolutional Sparse Coding. In CVPR, 2018. 2


[38] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In International Conference on Medical image computing and computer-assisted intervention, 2015. 3


[40] Varun Santhaseelan and Vijayan K Asari. Utilizing Local Phase Information to Remove Rain from Video. IJCV, 2015. 2


