Learning a Non-blind Deblurring Network for Night Blurry Images

Liang Chen[1*†], Jiawei Zhang[2‡†], Jinshan Pan[3], Songnan Lin[2], Faming Fang[1], Jimmy S. Ren[2,4]
[1] Shanghai Key Laboratory of Multidimensional Information Processing,
School of Computer Science and Technology, East China Normal University
[2] SenseTime Research
[3] Nanjing University of Science and Technology
[4] Qing Yuan Research Institute, Shanghai Jiao Tong University, Shanghai, China

Abstract

Deblurring night blurry images is difficult, because the common-used blur model based on the linear convolution operation does not hold in this situation due to the influence of saturated pixels. In this paper, we propose a non-blind deblurring network (NBDN) to restore night blurry images. To mitigate the side effects brought by the pixels that violate the blur model, we develop a confidence estimation unit (CEU) to estimate a map which ensures smaller contributions of these pixels in the deconvolution steps which are optimized by the conjugate gradient (CG) method. Moreover, unlike the existing methods using manually tuned hyper-parameters in their frameworks, we propose a hyper-parameter estimation unit (HPEU) to adaptively estimate hyper-parameters for better image restoration. The experimental results demonstrate that the proposed network performs favorably against state-of-the-art algorithms both quantitatively and qualitatively.

1. Introduction

Non-blind deblurring aims to recover the latent image from the given blurry image and blur kernel. Mathematically, the blurring process can be modeled as

$$B = I \otimes K + \eta, \tag{1}$$

where B, I, K, and η denote the blurry image, latent image, blur kernel, and additive noises, respectively. We use $\otimes$ to represent the convolution operator. Due to the irreversible loss of high-frequency information in the blurring process and the interference of noises, restoring the latent image is fundamentally ill-posed [26]. Thus, an efficient prior term $R(I)$ must be imposed to regularize the solution space, which gives,

$$\min_I \|B - I \otimes K\|^2 + \mu R(I). \tag{2}$$

Then, the solution is the one that minimizes the summation of the fidelity term and the prior term weighted by the parameter μ.

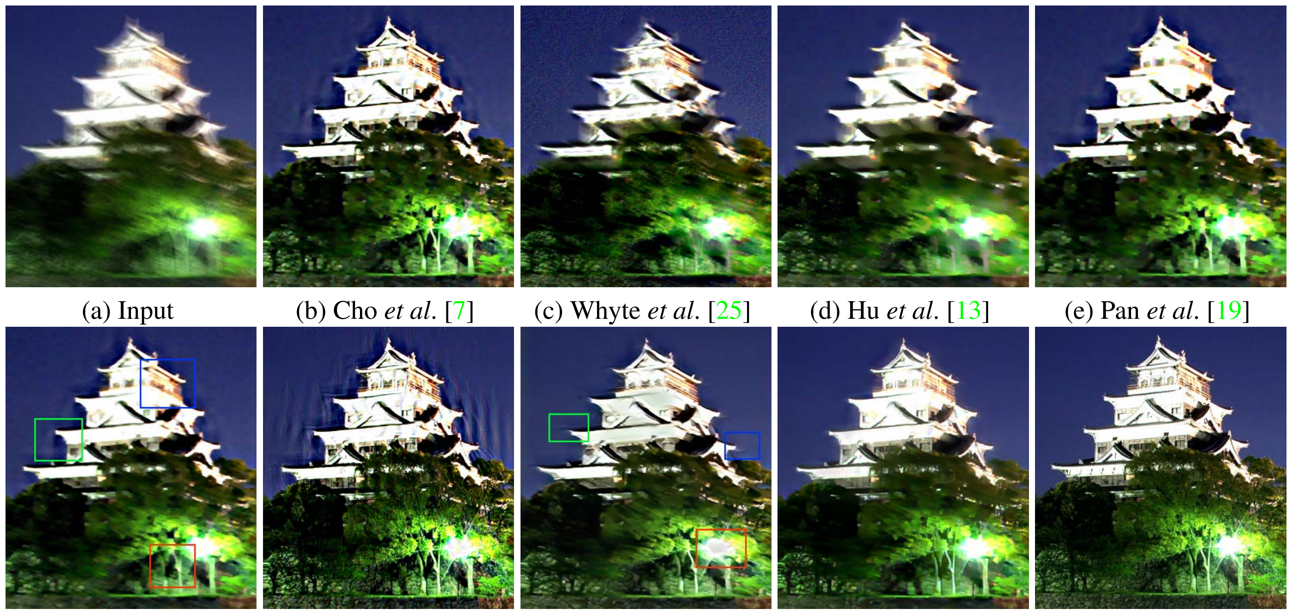
However, the non-blind deblurring task becomes more challenging for night blurry images. The reason is that saturated pixels are often presented under night condition, and these pixels violate the blur model in Eq. (1). As the saturated pixels are usually clipped to the maximum value due to the limited sensor range in the exposure time, a clipping function is adopted in Eq. (1):

$$B = C(I \otimes K) + \eta, \tag{3}$$

where $C(\cdot)$ denotes the clipping function. When $I \otimes K$ is within the dynamic range, $C(I \otimes K) = I \otimes K$; otherwise, $C(I \otimes K)$ returns the maximum intensity of the sensor range. Because of the influence of the non-linear function $C(\cdot)$, the affected pixels can no longer be modeled by the blur model in Eq. (1). Thus, the method based on this model is less effective for the image with saturated pixels (Figure 1 (g) and (h)).

Significant efforts have been made to solve this problem. Cho *et al.* [7] adopt an Expectation-Maximization-based framework to locate pixels violating the blur model and exclude them during the updating process. However, this strategy requires a predefined density of the saturated pixels and a heuristic estimation of the noise levels for all the images, which may be less effective because the densities and noise levels vary across different situations. Whyte *et al.* [25] extend the Richardson-Lucy algorithm with an approximation function [3] to model the property of saturated pixels. Moreover, [19, 6] develop specially-designed fidelity terms to fit the degrading process in Eq. (3). However, these approaches require empirical parameter tuning

*This work was done when Liang Chen was an intern at SenseTime.
†equal contribution
‡Corresponding author


(a) Input (b) Cho *et al.* [7] (c) Whyte *et al.* [25] (d) Hu *et al.* [13] (e) Pan *et al.* [19]

(f) Dong *et al.* [9] (g) IRCNN [29] (h) RGDN [10] (i) Our result (j) GT

Figure 1: Deblurring results of a night blurry image. Due to the influence of noise and saturated pixel, methods [7, 25, 13, 19, 9] based on the hand-crafted robust functions to model the data fidelity are ineffective while the method [29, 10] based on a linear convolution model generates an image with artifacts in the saturated regions. The part enclosed in the boxes contains artifacts (please zoom-in for a better view).

to fit the specific functions, and the optimization processes are time-consuming. Moreover, the adopted sparse prior [15] can result in visual artifacts for these methods when noise and blur are both present (Figure 1 (b)-(e)). The night blurry image often contains noises, saturation, and blur at the same time, which makes it difficult for these methods to restore a clear image.

As all above mentioned methods manually design sophisticated functions or complex optimizations to solve deblurring problems when saturated pixels are presented, several methods [28, 29, 10, 22] develop deep convolutional neural networks (CNNs) to solve non-blind image deblurring. These algorithms either explicitly or implicitly use deep CNNs as denoisers [28, 29, 10] or learn an end-to-end network to directly estimate latent images [22]. Although decent performance has been achieved, these algorithms do not effectively model the imaging process in Eq. (3). When handling blurred images with saturated pixels, they usually lead to results with ringing artifacts as shown in Figure 1 (g) and (h). As saturated pixels usually exist in low-illumination environments which are likely to cause blurry effect when capturing images in such conditions, it is of great interest to develop an effective algorithm to solve deblurring with saturated pixels.

In this work, we develop a non-blind deblurring network (NBDN) to learn both the fidelity and prior terms to restore saturated blurry images. For the fidelity term, we propose a learning-based confidence estimation unit (CEU) to determine the influence of each pixel on the optimization process. Specifically, pixels that violate Eq. (1) are with small confidences so that they have few influences on the optimization process. Unlike the approach in [7] that uses the residual between the blurry image and the convolution output (i.e. $B - I \otimes K$) to locate these pixels, confidences from CEU are only determined by the blurry image and restored result from the previous iterations, which alleviates the error brought by the inaccurate kernel. For the prior term, we use the prior information derived from a learned regularization unit (LRU) to remove noises as well as artifacts [28]. With the help of these two learned terms, we can obtain results with fine edges while side-effects from saturated pixels being removed.

Moreover, the noise level in image deblurring varies from image to image, which requires exhaustive hyper-parameters tuning in state-of-the-art deblurring methods. To avoid manual hyper-parameter tuning for noise handling, we propose a hyper-parameter estimation unit (HPEU) to adaptively predict corresponding hyper-parameters for every input. In practice, we integrate the information from all previous updating steps and the latent images to help the current hyper-parameter estimation. During each updating step, HPEU can generate different hyper-parameters for images with different levels of noise. After all the variables are obtained via corresponding modules, we propose a CG-based deconvolution step to update the latent image. An overview of NBDN is shown in Figure 2. The contributions

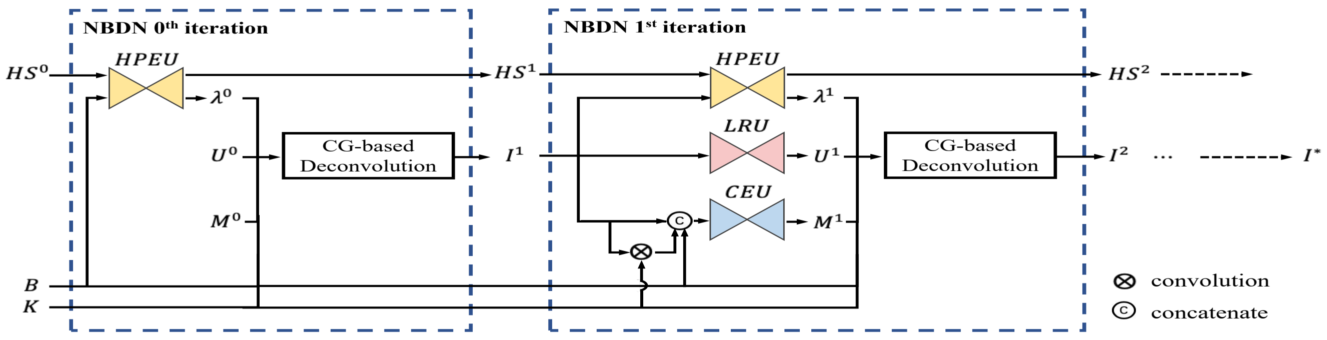


Figure 2: Architecture of NBDN. Given a night blurry image B and the corresponding blur kernel K, we first estimate the hyper-parameter λ^0 with the blurry image and an initial hidden state tensor HS^0 whose values are set to be zeros, and then we conduct an initial CG-based deconvolution step to obtain the updated latent image I^1. Note in this step, the values of the prior U^0 and confidence map M^0 are set to be zeros and ones, respectively. In the following updates, we use the updated HS and the current latent image I to estimate λ through HPEU. Meanwhile, LRU and CEU are used to generate prior information U and confidence maps M. The outputs are further fed into the deconvolution step to obtain a clearer image. We iteratively perform the above updates before the final deblurred image I^* is obtained. Please refer to the manuscript for details.

of this work are summarized as follows.

- We develop an NBDN for deblurring with saturated pixels. Different from other methods, we exclude side-effects from pixels that violate the blur model via a learned confidence map, where the confidences serve to determine the influences of different pixels in the data term.
- We propose an HPEU to adaptively estimate hyper-parameters in the deblurring framework. The integration of previous updating information enables the unit to determine optimal hyper-parameters for the current optimization step.
- We qualitatively and quantitatively evaluate NBDN with numerical and visual experiments against the state-of-the-art methods. The results show that artifacts can be effectively reduced by NBDN especially when the blurry image contains saturated pixels.

2. Proposed Method

As the saturated pixels usually affect the goodness-of-fit to the blur model (1), how to model the saturated pixels is critical for the image restoration. To solve this problem, we develop an effective algorithm to reduce the influence of the saturated pixels and propose a hyper-parameter estimation method for better image restoration in a unified deep CNN model. In the following, we will present the details of our method.

2.1. NBDN for night image deblurring

An intuitive idea is to detect pixels that violate the blur model in Eq. (1) and reduce their effect on the deblurring process. To this end, we develop an effective algorithm to estimate the underlying confidence map M for different pixels. Specifically, pixels that violate the blur model should have smaller confidence values to make sure they have fewer influences on the estimation process, and normal pixels should have higher confidence values so that they contribute more to the latent image restoration. Based on this idea, we use the following objective function to restore the latent image:

$$\min_{I,M} \|M \circ (B - I \otimes K)\|^2 + \mu R(I) + \theta F(M), \quad (4)$$

where $\circ$ represents the Hadamard product, $F(\cdot)$ represents a regularization term enforced on M, and θ is the weight parameter. We can solve Eq. (4) by iteratively minimizing the following sub-problems:

$$\begin{cases} M^t = \arg\min_M \|M \circ (B - I^t \otimes K)\|^2 + \theta F(M), & (5) \\ I^{t+1} = \arg\min_I \|M^t \circ (B - I \otimes K)\|^2 + \mu R(I), & (6) \end{cases}$$

where t denotes the updating step.

Sub-problem w.r.t. M. Eq. (5) suggests that M^t can be obtained based on current variables *i.e.* B, I^t and K. However, it is difficult to find a proper prior for the confidence map, *i.e.* $F(M)$. Instead of using heuristic strategies for the obtaining process as in [7], we develop a learning-based confidence estimation unit (i.e. CEU) to directly approximate the solution of Eq. (5).

Sub-problem w.r.t. I. The half-quadratic splitting method is often adopted to solve Eq. (6). By introducing an auxiliary variable $U \to I$, Eq. (6) can be reformulated into:

$$\min_{I,U} \|M^t \circ (B - I \otimes K)\|^2 + \lambda \|I - U\|^2 + \mu R(U), \quad (7)$$

where λ is a positive hyper-parameter, which is manually tuned in existing literature [27, 18, 16, 5, 4]. We can solve

Algorithm 1 Non-blind Deblurring Network for Night Blurry Images (NBDN)

```
Input: Blurry image $B$ and blur kernel $K$
Input: Iteration numbers $t_{max}$ and $s_{max}$
Output: Restored sharp image $I^*$
1: $I^0 = B$
2: $HS^0 = \mathbf{0}$
3: $\lambda^0, HS^1 = \text{HPEU}(I^0, HS^0)$
4: $I^1 = \text{CG}(\mathbf{B}, \mathbf{I}^0, \mathbf{0}, \mathbf{1}, \mathbf{K}, \lambda^0, s_{max})$
5: for $t = 1$ to $t_{max}$ do
6:     $M^t = \text{CEU}(B, I^t, I^t \otimes K)$
7:     $U^t = \text{LRU}(I^t)$
8:     $\lambda^t, HS^{t+1} = \text{HPEU}(I^t, HS^t)$
9:     $I^{t+1} = \text{CG}(\mathbf{B}, \mathbf{I}^t, \mathbf{U}^t, \mathbf{M}^t, \mathbf{K}, \lambda^t, s_{max})$
10: end for
```

```
CG($\mathbf{B}$, $\mathbf{I}_0$, $\mathbf{U}$, $\mathbf{M}$, $\mathbf{K}$, $\lambda$, $s_{max}$):
Output: $I_{s_{max}}$
1: $\mathbf{b} = \lambda\mathbf{U} + \mathbf{K}^T\mathbf{M}^T\mathbf{M}\mathbf{B}$
2: $\mathbf{A} = \mathbf{K}^T\mathbf{M}^T\mathbf{M}\mathbf{K} + \lambda$
3: $\mathbf{P}_0 = \mathbf{b} - \mathbf{A}\mathbf{I}_0$
4: $\mathbf{r}_0 = \mathbf{P}_0$
5: for $s = 0$ to $s_{max}$ do
6:     $\alpha_s = (\mathbf{r}_s^T\mathbf{r}_s)/(\mathbf{P}_s^T\mathbf{A}\mathbf{P}_s)$
7:     $\mathbf{I}_{s+1} = \mathbf{I}_s + \alpha_s\mathbf{P}_s$
8:     $\mathbf{r}_{s+1} = \mathbf{r}_s - \alpha_s\mathbf{A}\mathbf{P}_s$
9:     $\beta_s = (\mathbf{r}_{s+1}^T\mathbf{r}_{s+1})/(\mathbf{r}_s^T\mathbf{r}_s)$
10:    $\mathbf{P}_{s+1} = \mathbf{r}_{s+1} + \beta_s\mathbf{P}_s$
11: end for
```

$\mathbf{B}$, $\mathbf{I}$, $\mathbf{U}$, $\mathbf{M}$ are B, I, U, M in their vectorized forms; $\mathbf{K}$ is the toeplitz matrix of K w.r.t. I; $\mathbf{0}$ and $\mathbf{1}$ denote the all-zero and all-one matrices.

Eq. (7) by iteratively updating the following equations:

$$\begin{cases} U^t = \arg\min_U \lambda\|I^t - U\|^2 + \mu R(U), & (8) \\ I^{t+1} = \arg\min_I \|M^t \circ (B - I \otimes K)\|^2 + \lambda\|I - U^t\|^2. & (9) \end{cases}$$

Eq. (8) is a typical denoising problem, and the results from [28, 29] show that CNN is a decent solver for this problem. In this paper, we use a learned regularization unit (i.e. LRU) to estimate the auxiliary variable U.

Note that though Eq. (9) is a quadratic problem, but it cannot be solved by fast Fourier transform (FFT) due to involving the Hadamard product operation. We thus use the conjugate-gradient (CG) for the optimization as shown in Algorithm 1. Meanwhile, the whole CG module is differentiable and it can be treated as a layer in our framework. More details are included in the supplementary material.

Meanwhile, as a vital component of Eq. (9), the hyper-parameter λ serves to balance the importance of the fidelity and the penalty terms, which is sensitive to different noise levels and often demands exhaustive search in a large hyper-parameter space. Further, the optimal setting of λ often differs image-by-image. Instead of using the ad-hoc strategy, we propose to use a hyper-parameter estimation unit (i.e. HPEU) to determine λ for every image during the updating stages. To utilize the information from the previous updating steps, we first store it in the hidden state (i.e. HS) tensor, and then we integrate HS with the information from the updated latent image via a convGRU [1] to help the current hyper-parameter estimation.

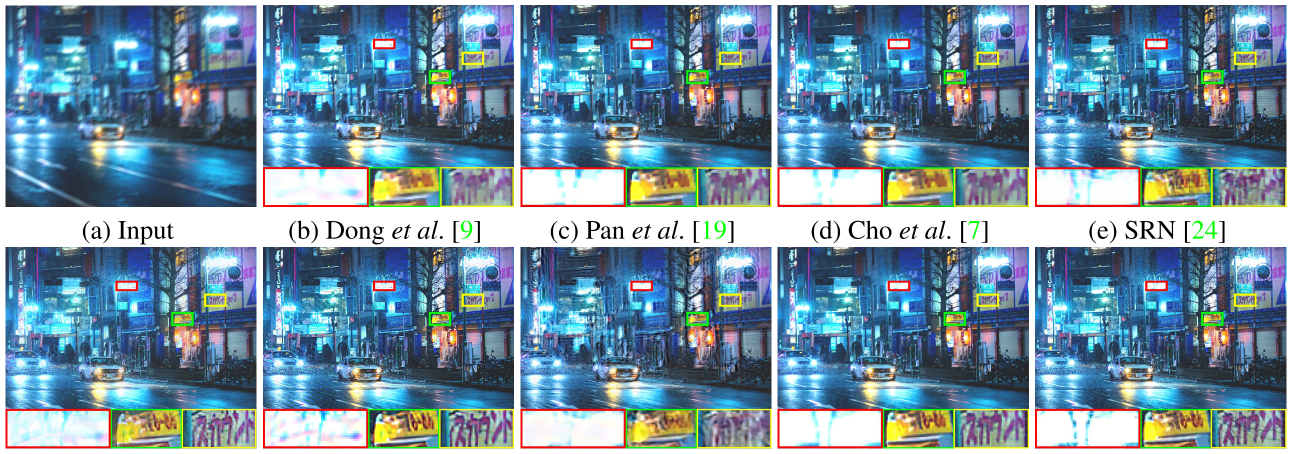
2.2. Network design

The proposed network design is based on Eq. (5), (8) and (9). The inputs of our network include the blurry image and the corresponding blur kernel. We first conduct an initial CG-based deconvolution step with the hyper-parameter obtained from HPEU. In the following updating steps, we perform the hyper-parameter, confidence map and prior estimations via the corresponding networks (i.e. HPEU, CEU and LRU) given the current latent image. The updated latent image can be obtained with the updated λ, M and U by performing the deconvolution step. The detailed algorithm is shown in Algorithm 1.

CEU. Our confidence map estimation module is designed to obtain the confidence map M in Eq. (5). We note that the obtaining process defined in [7] relies heavily on heuristic settings, such as predefined saturated densities and noise levels, which is less effective given images with different situations. We thus propose to use CEU to estimate M. This module uses the blurry image B, the current estimate I^t, and the convolution result $I^t \otimes K$ as inputs, and outputs a map M^t ranging from 0 to 1. The network is constructed with three res-block [12], and each of the block contains two convolution layers to generate 16 features. We add a rectified linear unit (ReLU) after every convolution layer except for the last one for each block, and a sigmoid layer is attached in the end to estimate the final result.

LRU. The auxiliary variable U^t from Eq. (8) can be obtained based on the current estimate I^t. Because there are still artifacts and noises present in I^t, in order to obtain a clearer U^t, we use LRU to remove the artifacts and noises. LRU takes I^t as input and outputs U^t that with fewer ringings and noises, which is implemented by a 3-scale lightweight U-net [23] model. Specifically, each scale in the U-net model is applied with two convolutions, and each convolution layer is attached with a ReLU layer. The features from the first to the last scale are 8, 16 and 32, respectively.

HPEU. Considering the high discrepancy of noise conditions in different images and the cumbersome work of man-


(a) Input (b) Dong *et al.* [9] (c) Pan *et al.* [19] (d) Cho *et al.* [7] (e) SRN [24]

(f) FCNN [28] (g) IRCNN [29] (h) RGDN [10] (i) Our result (j) GT

Figure 3: Deblurring results of a night blurry image. Some details are erased in the results from the optimization-based methods [7, 19, 9] and the end-to-end learning model SRN [24] (shown in the green and yellow boxes). The proposed method generates a result with finer details and fewer ringing artifacts compared to existing learning-based non-blind deblurring methods FCNN [28] and IRCNN [29] which are ineffective in dealing with saturated regions (shown in red boxes). Please zoom-in for a better view.

ual parameter tuning, we adopt an HPEU to adaptively estimate the hyper-parameter λ in Eq. (9). The information from previous updatings is stored via a $32 \times 128 \times 128$ hidden state (i.e. HS) tensor. During the processing phase, HPEU takes the current image I^t and HS as inputs and estimates the updated HS and the hyper-parameter λ. Specifically, HPEU first convolves I^t to generate information which is further integrated with the information from HS by a convGRU [1]. The updated HS can be obtained during the integration process, and it is further used to acquire the hyper-parameter. In practice, we use 8 convolution layers and an adaptive pooling layer to generate features, and we use a fully connected layer to obtain the final estimation.

Please refer to the supplementary material for detailed network configurations.

2.3. Training loss

Inspired by the settings in [28], we progressively train the weights of CEU and LRU for multiple iterations, and then we train HPEU by fixing CEU and LRU. The procedure is achieved by minimizing the loss function $\mathcal{L}$:

$$\mathcal{L} = \frac{1}{N}\sum_{i}^{N}\left(\sum_{t=0}^{t_{max}}(\|U_i^t - I_i^{gt}\|_1 + \|I_i^{t+1} - I_i^{gt}\|_1) + \|I_i^* - I_i^{gt}\|_1\right), \tag{10}$$

where N is the number of training samples in every batch; U^t and I^{t+1} are the output of LRU and the updated latent image in the t-th iteration; I^* and I^{gt} are the final output of NBDN and the ground truth image, respectively. Specifically, for each updating step t, we firstly train LRU using the first term of $\mathcal{L}$ with M fixed as **1**. Then, CEU is trained using the second term of $\mathcal{L}$ while the weights of LRU are fixed. Note in these steps, the hyper-parameters λ are fixed as their initial settings using the same strategy from [28] (keeping λ^{t+1} larger than λ^t). After LRU and CEU are both converged, HPEU is trained with both the weights of LRU and CEU fixed, and the training procedure is fulfilled by minimizing the last term of $\mathcal{L}$. All these networks are fine-tuned in the end using the last term of $\mathcal{L}$ to obtain the final optimal results.

3. Experimental Results

3.1. Training details

Our implementation is based on PyTorch [21]. The patch size is set to be 256×256 in the proposed network. We use ADAM optimizer [14] by setting $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$ with the learning rate 0.0001. We use the weight initialization method in [11] as the proposed network initialization.

The number of iterations for NBDN (i.e. t_{max} in Algorithm 1) is empirically set to be 4 as a trade-off between accuracy and efficiency, and the maximum iteration numbers for the CG loops (i.e. s_{max} in Algorithm 1) are empirically set to be 15 across different updates.

Datasets. To generate enough blurry images for training, we download 500 night images from Flickr. For every image, we randomly crop 10 patches of size 256×256. The motion kernels are generated according to [2], and their sizes range from 11 to 33 pixels. Similar to [9], we convolve every image patch with 5 generated blur kernels, which gives a total of 25,000 samples in our training dataset. To synthesize saturated regions, we first enlarge the range of both blurry and sharp images by a factor of 1.2 same as the

	Cho [7]	Hu [13]	Pan [19]	Dong [9]	SRN [24]	Nan [17]	FCNN [28]	IRCNN [29]	RGDN [10]	Ours
Results with GT blur kernels										
PSNR	28.64	24.96	28.45	25.97	25.11	28.59	29.46	29.43	28.47	**30.06**
SSIM	0.8846	0.7851	0.8837	0.8272	0.8064	0.8692	0.9051	0.9010	0.8427	**0.9065**
Results with blur kernels from [20]										
PSNR	27.18	24.58	26.94	25.13	25.11	27.62	26.91	28.29	27.72	**28.45**
SSIM	0.8646	0.7830	0.8621	0.8069	0.8064	0.8544	0.8699	0.8860	0.8458	**0.8901**

Table 1: Evaluations on the given night blurry images.

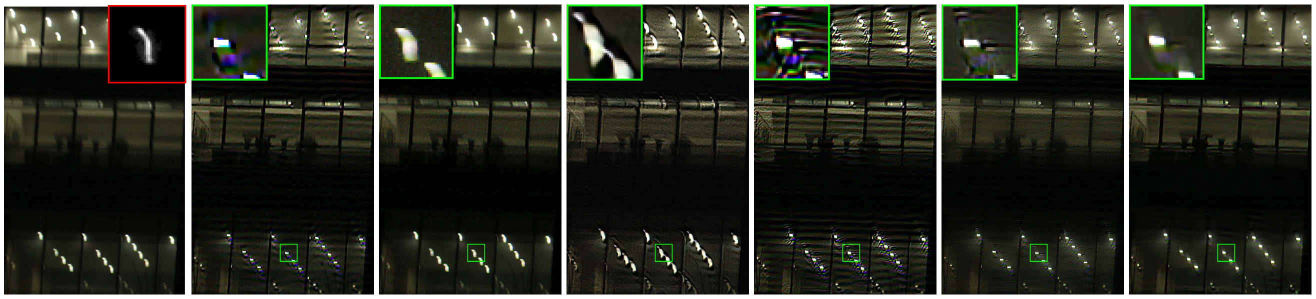


Figure 4: Deblurring results of a real-world night blurry image. The kernel in the red rectangle in (a) is from the robust kernel estimation method [8]. The proposed method performs favorably compared with existing non-blind deblurring methods. Please zoom-in for a better view.

setting in [22], and then clip the images into the dynamic range of 0 to 1. Additionally, random Gaussian noises with standard deviations ranging from 0 to 0.01 are added to the blurry images.

For the test dataset, we further download 100 night images from Flickr, and then generate the blurry images in the same way as the training dataset, where the training data and test data do not overlap. We also use real night blurry images to evaluate the effectiveness of NBDN.

3.2. Comparisons with state-of-the-arts

In this paper, we compare our method with the robust optimization-based models [7, 19, 9] and some recent learning-based arts [28, 29, 10, 17]. Additionally, an end-to-end deep neural network SRN [24] is also compared to further validate the effectiveness of NBDN. For a fair comparison, we use the original implementations of these methods and fine-tune the networks of [28, 24] in our training dataset.

Synthetic data. We first evaluate all the methods in the given testing dataset in terms of average PSNR and SSIM. As shown in Table 1, NBDN outperforms exiting arts in both terms of PSNR and SSIM whether the images are restored with the ground truth kernels or estimated using the existing blind deblurring method [20]. The example given in Figure 3 demonstrates the differences between the compared approaches.

The end-to-end deblurring approach SRN [24] is fine-tuned using the proposed training data, but it achieves less effective results compared to others because of the lack of the blur kernel information and ignoring the degrading process. Note that the optimization-based methods [7, 19] explicitly exclude saturated pixels in their models. Thus, saturated regions can be recovered with few artifacts in their results. But these approaches use the sparse image prior which often leads to the loss of details in practice (as depicted in Figure 3 (c) and (d)). A similar result is reported for [9] in Figure 3 (b). In comparison, the learning-based arts [28, 29, 10] use more efficient learned priors to recover fine edges in most situations, and they achieve relatively better performance in terms of average PSNR and SSIM. However, saturated pixels are not specifically considered in their frameworks. As a result, their results are somehow degraded with ringing artifacts in saturated regions (as shown in Figure 3 (f), (g), (h)). By considering the saturated regions and the learned image prior, the results from NBDN are hardly affected by the saturated pixels, where the sharp edges are also restored.

Methods	Total parameters (M)	Running time (s)
Cho *et al.* [7]	-	5.25 (CPU)
Pan *et al.* [19]	-	15.41 (CPU)
SRN [24]	21	0.37
FCNN [28]	0.45	0.13
IRCNN [29]	0.15	1.11
RGDN [10]	1.26	5.34
Ours	0.39	0.25

Table 2: Model sizes and running time comparisons.

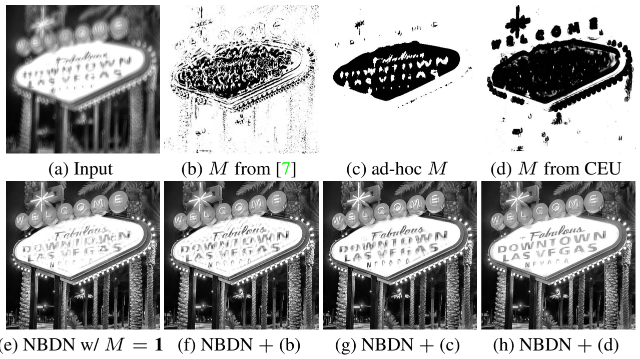


Figure 5: Comparisons of the results by different confidence map estimation methods. Note in (b) - (d), dark pixels indicate small confidence values, (b) and (d) are from the last iteration of corresponding methods. The results demonstrate that CEU is able to generate a properer confidence map, which leads to fewer artifacts in the saturated regions.

GT blur kernels				
	M fixed as $\mathbf{1}$	M from [7]	ad-hoc M	M from CEU
PSNR	29.36	29.47	29.45	30.06
SSIM	0.9037	0.9037	0.8930	0.9065
Estimated kernels from [20]				
	M fixed as $\mathbf{1}$	M from [7]	ad-hoc M	M from CEU
PSNR	28.20	28.25	27.92	28.45
SSIM	0.8878	0.8877	0.8775	0.8901

Table 3: Comparison on the test dataset w.r.t. different confidence map estimation methods.

Real data. We further show a challenging real-world example with saturation and noises and demonstrate the results in Figure 4. We use the robust kernel estimation method from [8] to obtain the blur kernel. Owing to the effectiveness of the learned prior, methods from [28, 29] can ease the blur to some extent, but their results still contain artifacts because saturated pixels are not properly handled in their models. Moreover, the optimization-based method [7] contains severe ringings in their final results. It is mainly because that the adopted sparse regularization term is ineffective when the noises are presented in the natural data, and artifacts around saturated regions cannot be properly handled because of the mislocation of the saturated regions. The end-to-end approach SRN [24] can hardly restore the blurry images when the blurring process is ignored. In contrast, NBDN can generate sharper images with fewer artifacts in the saturated regions.

Model sizes and running time comparisons. The proposed method does not require lots of parameters and performs favorably against other state-of-the-art methods in the term of running time. We compare with the existing approaches [7, 19, 28, 29, 10, 24] on the same PC with an Intel(R) Xeon(R) CPU and an Nvidia Tesla 1080 GPU. Table 2 summarizes total parameters from different models and the average running time in an image with a size of 300×300. NBDN conducts one of the fastest running time among the compared methods.

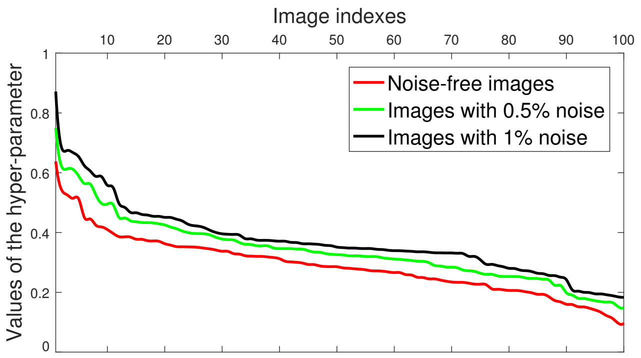


Figure 6: Sorted estimated hyper-parameters from the last iteration of NBDN w.r.t. different levels of noise.

		PSNR	SSIM
GT blur kernels	learned static λ	28.84	0.8881
	λ from HPEU	30.06	0.9065
kernels from [20]	learned static λ	27.79	0.8723
	λ from HPEU	28.45	0.8901

Table 4: Comparison on the given test dataset with different hyper-parameter settings.

4. Analysis and Discussions

4.1. Effectiveness of CEU

The confidence map M plays an important role in our deblurring pipeline. As shown in Figure 5 (e), when NBDN is trained with all the confidence values in M fixed as one, the saturated pixels will disturb the deblurring process, resulting in artifacts in the restored image. The work in [7] suggests a predefined function to compute M, which requires a heuristic guess of the density of saturated pixels, and the computed results rely heavily on the residual information (i.e. $B - I \otimes K$). However, Figure 5 (b) shows that this function often detects image edges, and the artifacts in the restored result can not be fully removed when we use this method to compute M while training NBDN (Figure 5 (f)). Meanwhile, we show that if M is defined in an ad-hoc strategy (pixels with values larger than a predefined threshold [1] in the blurry image disobey the blur model, and the corresponding confidence values are set to be zeros as shown in Figure 5 (c)), NBDN trained with this scheme also results in an image with artifacts in the saturated area (Figure 5 (g)). In comparison, CEU uses the blurry image

[1] We use 0.85 for the threshold value in this setting.

GT blur kernels				
iterations	2	3	4 (proposed)	5
PSNR	26.22	29.29	30.06	30.10
SSIM	0.8303	0.8869	0.9065	0.9063
Estimated kernels from [20]				
iterations	2	3	4 (proposed)	5
PSNR	25.63	28.02	28.45	28.50
SSIM	0.8207	0.8789	0.8901	0.8907

Table 5: Comparison on the given test dataset when NBDN is trained with different iterations.

and the updated latent image to guide the estimation. As shown in Figure 5 (d), pixels violate the blur model correspond to small values in the confidence map. Figure 5 (h) shows that NBDN restores a visually more pleasant result when M is estimated from CEU. Quantitative evaluations of the four schemes on the test dataset are shown in Table 3, where M from CEU leads to the best performance among the compared approaches when integrated into NBDN.

4.2. Effectiveness of HPEU

The proposed HPEU considers the varying noise levels and contexts in different images to generate dynamic hyper-parameters. Compared to [28] who learns static hyper-parameters, our method enables dynamic control of the strength of the regularization term when encountered different noise levels. We verify the effectiveness of HPEU by adding 0, 0.5% and 1% random noises to the clean images in the testing dataset and compare the estimated hyper-parameters. As shown in Figure 6, hyper-parameters are larger, which enlarges the denoising effect derived from the penalty term, when the blurry images are added with severer noises. This observation validates the significance of HPEU when the blurry images are with varying noises.

We further verify the effectiveness of HPEU by conducting an ablation study of the deblurring results on the given test dataset. Following the instruction in [28], we learn optimal hyper-parameter settings by replacing HPEU with learnable hyper-parameters λ and retrain the weights of NBDN. Results from these two schemes are shown in Table 4, where results generated by the dynamic parameter setting (i.e. HPEU) outperform that by the learned static one.

4.3. Convergence analysis

We use 4 iterations for NBDN and 15 iterations for the CG loop in our setting. We quantitatively evaluate the convergence properties of our method on the given test dataset and show the results in Table 5 and 6. All the compared models are trained in the same way as we train the original implementation of NBDN.

GT blur kernels				
iterations	5	10	15 (proposed)	20
PSNR	29.17	29.84	30.06	30.04
SSIM	0.8958	0.9014	0.9065	0.9070
Estimated kernels from [20]				
iterations	5	10	15 (proposed)	20
PSNR	27.74	28.18	28.45	28.42
SSIM	0.8787	0.8852	0.8901	0.8923

Table 6: Comparisons of the results on the test dataset when NBDN is trained with different iterations in the CG-loop.

Table 5 shows NBDN converges well after 4 iterations. The results do not change significantly after 3 iterations. Note that the 2-iteration network is different from the 2 iteration stage of the proposed four-iteration network.

Table 6 reports that NBDN does not generate better results when 20 iterations are used in the CG loop compared to the method using 15 iterations in the CG. Thus, we use 15 iterations for the CG loop as the trade-off between accuracy and speed.

5. Conclusion

In this paper, we propose a non-blind deblurring network for night blurry images. Our algorithm can incorporate the existing scheme that treat the deblurring process as iteratively denoising and deconvolution. To explicitly detect pixels that violate the convolution model during the deblurring process, we learn a confidence map that can determine influences to the optimization from every pixel. We have developed a hyper-parameter estimation unit to dynamically handle different noise levels, which does not require the ad-hoc strategies to determine the hyper-parameter value when handling each image. We have proposed a CG-based method which can be embedded in the deep CNNs for better image restoration. The experimental results demonstrate that the proposed approach achieves favorable performance against state-of-the-art deblurring methods.

Acknowledgements

The work of Jinshan Pan was supported by the National Natural Science Foundation of China (Nos. 61872421, 61922043), and the National Key Research and Development Program of China under Grant 2018AAA0102001; The work of Faming Fang was supported by the Key Project of the National Natural Science Foundation of China under Grant 61731009, the NSFC-RGC under Grant 61961160734, the National Natural Science Foundation of China under Grant 61871185, and the Science Foundation of Shanghai under Grant 20ZR1416200.

References

[1] Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. *arXiv preprint arXiv:1511.06432*, 2015. 4, 5

[2] Ayan Chakrabarti. A neural approach to blind motion deblurring. In *ECCV*, 2016. 5

[3] Chunhui Chen and Olvi L Mangasarian. A class of smoothing functions for nonlinear and mixed complementarity problems. *Computational Optimization and Applications*, 1996. 1

[4] Liang Chen, Faming Fang, Shen Lei, Fang Li, and Guixu Zhang. Enhanced sparse model for blind deblurring. In *ECCV*, 2020. 3

[5] Liang Chen, Faming Fang, Tingting Wang, and Guixu Zhang. Blind image deblurring with local maximum gradient prior. In *CVPR*, 2019. 3

[6] Liang Chen, Faming Fang, Jiawei Zhang, Jun Liu, and Guixu Zhang. Oid: Outlier identifying and discarding in blind image deblurring. In *ECCV*, 2020. 1

[7] Sunghyun Cho, Jue Wang, and Seungyong Lee. Handling outliers in non-blind image deconvolution. In *ICCV*, 2011. 1, 2, 3, 4, 5, 6, 7

[8] Jiangxin Dong, Jinshan Pan, Zhixun Su, and Ming-Hsuan Yang. Blind image deblurring with outlier handling. In *ICCV*, 2017. 6, 7

[9] Jiangxin Dong, Jinshan Pan, Deqing Sun, Zhixun Su, and Ming-Hsuan Yang. Learning data terms for non-blind deblurring. In *ECCV*, 2018. 2, 5, 6

[10] Dong Gong, Zhen Zhang, Qinfeng Shi, Anton van den Hengel, Chunhua Shen, and Yanning Zhang. Learning an optimizer for image deconvolution. *IEEE TNNLS*, 2020. 2, 5, 6, 7

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *CVPR*, 2015. 5

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4

[13] Zhe Hu, Sunghyun Cho, Jue Wang, and Ming-Hsuan Yang. Deblurring low-light images with light streaks. In *CVPR*, 2014. 2, 6

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[15] Dilip Krishnan and Rob Fergus. Fast image deconvolution using hyper-laplacian priors. In *NIPS*, 2009. 2

[16] Jun Liu, Ming Yan, and Tieyong Zeng. Surface-aware blind image deblurring. *IEEE TPAMI*, 43(3):1041–1055, 2021. 3

[17] Yuesong Nan and Hui Ji. Deep learning for handling kernel/model uncertainty in image deconvolution. In *CVPR*, 2020. 6

[18] Jinshan Pan, Zhe Hu, Zhixun Su, and Ming-Hsuan Yang. l_0-regularized intensity and gradient prior for deblurring text images and beyond. *IEEE TPAMI*, 39(2):342–355, 2017. 3

[19] Jinshan Pan, Zhouchen Lin, Zhixun Su, and Ming-Hsuan Yang. Robust kernel estimation with outliers handling for image deblurring. In *CVPR*, 2016. 1, 2, 5, 6, 7

[20] Jinshan Pan, Deqing Sun, Hanspeter Pfister, and Ming-Hsuan Yang. Blind image deblurring using dark channel prior. In *CVPR*, 2016. 6, 7, 8

[21] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Neurips*, 2019. 5

[22] Wenqi Ren, Jiawei Zhang, Lin Ma, Jinshan Pan, Xiaochun Cao, Wangmeng Zuo, Wei Liu, and Ming-Hsuan Yang. Deep non-blind deconvolution via generalized low-rank approximation. In *NIPS*, 2018. 2, 6

[23] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 4

[24] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. In *CVPR*, 2018. 5, 6, 7

[25] Oliver Whyte, Josef Sivic, and Andrew Zisserman. Deblurring shaken and partially saturated images. *IJCV*, 2014. 1, 2

[26] David P. Wipf and Haichao Zhang. Revisiting bayesian blind deconvolution. *JMLR*, 2014. 1

[27] Li Xu, Shicheng Zheng, and Jiaya Jia. Unnatural l_0 sparse representation for natural image deblurring. In *CVPR*, 2013. 3

[28] Jiawei Zhang, Jinshan Pan, Wei-Sheng Lai, Rynson W. H. Lau, and Ming-Hsuan Yang. Learning fully convolutional networks for iterative non-blind deconvolution. In *CVPR*, 2017. 2, 4, 5, 6, 7, 8

[29] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep cnn denoiser prior for image restoration. In *CVPR*, 2017. 2, 4, 5, 6, 7