# HINet: Half Instance Normalization Network for Image Restoration

Liangyu Chen[1] * Xin Lu[1] * Jie Zhang[1,2] Xiaojie Chu[1,3] Chengpeng Chen[1]
[1] MEGVII Technology [2] Fudan University [3] Peking University
{chenliangyu,luxin,chenchengpeng}@megvii.com
j_zhang19@fudan.edu.cn chuxiaojie@stu.pku.edu.cn

## Abstract

*In this paper, we explore the role of Instance Normalization in low-level vision tasks. Specifically, we present a novel block: Half Instance Normalization Block (HIN Block), to boost the performance of image restoration networks. Based on HIN Block, we design a simple and powerful multi-stage network named HINet, which consists of two subnetworks. With the help of HIN Block, HINet surpasses the state-of-the-art (SOTA) on various image restoration tasks. For image denoising, we exceed it 0.11dB and 0.28 dB in PSNR on SIDD dataset, with only 7.5% and 30% of its multiplier-accumulator operations (MACs),* $6.8\times$ *and* $2.9\times$ *speedup respectively. For image deblurring, we get comparable performance with 22.5% of its MACs and* $3.3\times$ *speedup on REDS and GoPro datasets. For image deraining, we exceed it by 0.3 dB in PSNR on the average result of multiple datasets with* $1.4\times$ *speedup. With HINet, we won the 1st place on the NTIRE 2021 Image Deblurring Challenge - Track2. JPEG Artifacts, with a PSNR of 29.70.*

## 1. Introduction

Normalization is widely used in high-level computer vision tasks: Batch Normalization [18] and IBN [33] in classification [28], Layer Normalization [4] in DETR [6] and GroupNorm [47] in FCOS [40] for detection *etc*. Besides, Instance Normalization [43] is used to style/domain transfer [33, 16] tasks. However, the simple application of normalization to low-level computer vision problems can be suboptimal. For example, Batch Normalization can't improve the performance of the network in super-resolution [26].

In this paper, we carefully integrate Instance Normalization as building blocks to advance the network performance in image restoration tasks. Specifically, we present a Half Instance Normalization Block (HIN Block). Based on HIN Blocks, we further propose a multi-stage network called HINet, which consists of two subnetworks. By stacking HIN Block in each subnetwork's encoder, the receptive field at each scale is expanded, and the robustness of features is also improved. In addition to the architecture of each stage, we adopt cross-stage feature fusion [55] and supervised attention module [55] between two stages to enrich the multi-scale features and facilitate achieving performance gain respectively.

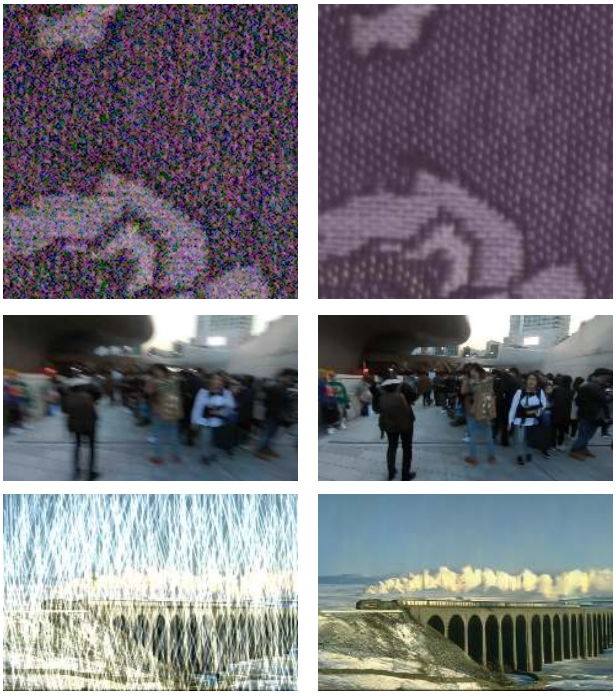

Figure 1: Visualized results of HINet on various image restoration tasks. Left: degraded image. Right: the predicted result of HINet. From top to bottom: image denoising, image deblurring, and image deraining task respectively.

Compared with the state-of-the-art architecture MPRNet [55], HINet surpasses it on various image restoration tasks. For image denoising, we exceed it 0.11 dB and 0.28 dB in PSNR on SIDD [1] dataset, with only 7.5% and 30%

*Equally contribution.

of its multiplier-accumulator operations (MACs), 6.8× and 2.9× speedup respectively. For image deblurring, we get comparable performance with 22.5% of its MACs and 3.3× speedup on REDS [30] and GoPro [31] datasets. For image deraining, we exceed it by 0.3 dB in PSNR on the average result of multiple datasets following [56], with 1.4× speedup. Visualized results of various image restoration tasks are shown in Figure 1. In addition, we apply HIN to various models and various datasets, the results demonstrate the generalization ability of HIN. For example, with the help of HIN, DMPHN [57] increased 0.42 dB in PSNR on GoPro [31] dataset.

Our contributions can be summarized as follows:

- We carefully integrate Instance Normalization as building blocks and proposed a Half Instance Normalization Block. To the best of our knowledge, it is the first model to adopt normalization *directly* with state-of-the-art performance in image restoration tasks.
- Based on HIN Block, we design a multi-stage architecture, HINet, for image restoration tasks, and achieves the state-of-the-art performance with fewer MACs and inference time compares to the SOTA method [55].
- Extensive experiments are conducted to demonstrate the effectiveness of our proposed HIN Block and HINet. With the help of HIN Block and HINet, we won 1st place on the NTIRE 2021 Image Deblurring Challenge - Track2. JPEG Artifacts [32], with a PSNR of 29.70.

## 2. Related Work

### 2.1. Normalization in low-level computer vision tasks:

Normalization has become an essential component in high-level computer vision tasks (especially Batch Normalization) but is rarely used in low-level computer vision tasks. [31] modified the ResBlock [15] by removing batch normalization since they trained the model with a mini-batch of size 2 in deblur. [26] removed batch normalization in super-resolution, which the batch normalization get rid of range flexibility from networks. As the disharmony between image restoration tasks and Batch Normalization (BN) discussed in [51], image restoration tasks commonly use small image patches and small mini-batch size to train the network, which causes the statistics of BN unstable. Moreover, the image restoration task is a per-image dense pixel value prediction task, which is scale sensitivity. While BN is usually helpful in scale insensitive tasks.

In addition to the above, Instance Normalization [42] is proposed to replace Batch Normalization in [41] to improve the performance of the style transfer task. [16] demonstrates Instance Normalization is the normalization of low-level features to some extent. They proposed adaptive instance normalization to the style transfer task by aligning the channel-wise statistics in Instance Normalization of style image to content image. Based on [16], [20] adopts an adaptive instance normalization as a regularizer to build denoiser and transfers knowledge learned from synthetic noise data to the real-noise denoiser. Unlike [20], we extend the Instance Normalization as a method of feature enhancement and apply it to the image restoration tasks *directly* without transfer learning.

### 2.2. Architectures for Image Restoration

The single-stage methods are widely used in image restoration tasks, and these methods generally improve the network capacity through the complex network structure [3, 60]. The multi-stage methods decompose the complex image restoration task into smaller easier sub-tasks, employing a lightweight subnetwork at each stage. [11] introduce the mature Gaussian-Laplacian image pyramid decomposition technology to the neural network, and uses a relatively shallow network to handle the learning problem at each pyramid level. [35] proposes a progressive recurrent network by repeatedly unfolding a shallow ResNet [15], and introduces a recurrent layer to exploit the dependencies of deep features across stages. [58] proposes a deep stacked hierarchical multi-patch network. Each level focus on different scales of the blur and the finer level contributes its residual image to the coarser level. [55] proposes a multi-stage progressive image restoration architecture, where there are two encoder-decoder subnetworks and one original resolution subnetwork. [55] also proposes a supervised attention module (SAM) and a cross-stage feature fusion (CSFF) module between every two stages to enrich the features of the next stage. Our model also uses these two modules to facilitate achieving significant performance gain and uses two simple U-Nets [36] as the subnetworks.

## 3. Approach

In this section, we provide more detailed explanations about HINet and HIN Block in the following subsections. Specifically, we introduce HINet in 3.1 and HIN Block in 3.2.

### 3.1. HINet

The architecture of our proposed Half Instance Normalization Network (HINet) is shown in Figure 2. HINet consists of two subnetworks, each of which is a U-Net [36]. As for U-Net in each stage, we use one $3 \times 3$ convolutional layer to extract the initial features. Then those features are input into an encoder-decoder architecture with four downsamplings and upsamplings. We use convolution with kernel size equal to 4 for downsampling, and use transposed

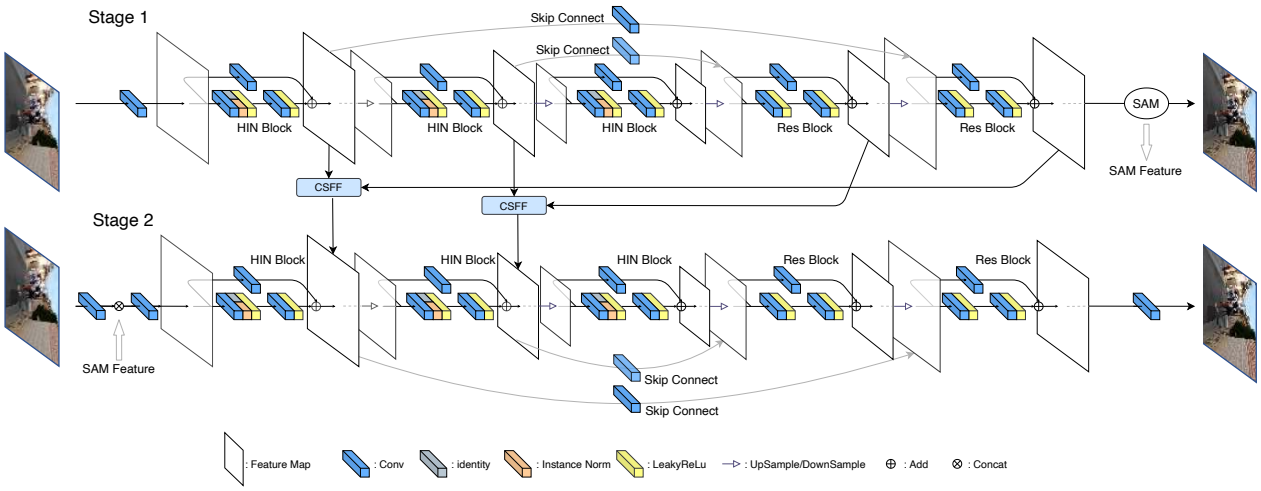


Figure 2: Proposed Half Instance Normalization Network (HINet). The encoder of each subnetwork contains Half Instance Normalization Blocks (HIN Block). For simplicity, we only show 3 layers of HIN Block in the figure, and HINet has a total of 5 layers. We adopt CSFF and SAM modules from MPRNet [56].

convolution with kernel size equal to 2 for upsampling. In the encoder component, we design Half Instance Normalization Blocks to extract features in each scale, and double the channels of features when downsampling. In the decoder component, we use ResBlocks [15] to extract high-level features, and fuse features from the encoder component to compensate for the loss of information caused by resampling. As for ResBlock, we use leaky ReLU [29] with a negative slope equal to 0.2 and remove batch normalization. Finally, we get the residual output of the reconstructed image by using one $3 \times 3$ convolution.

We use cross-stage feature fusion (CSFF) module and supervised attention module (SAM) to connect two subnetworks, where these two modules come from [55]. As for CSFF module, we use $3 \times 3$ convolution to transform the features from one stage to the next stage for aggregation, which helps to enrich the multi-scale features of the next stage. As for SAM, we replace the $1 \times 1$ convolutions in the original module with $3 \times 3$ convolutions and add bias in each convolution. By introducing SAM, the useful features at the current stage can propagate to the next stage and the less informative ones will be suppressed by the attention masks [55].

In addition to the network architecture, we use Peak Signal-to-Noise Ratio (PSNR) as the metric of the loss function, which is PSNR loss. Let $X_i \in \mathbb{R}^{N \times C \times H \times W}$ denotes the input of subnetwork $i$, where $N$ is the batch size of data, $C$ is the number of channels, $H$ and $W$ are spatial size. $R_i \in \mathbb{R}^{N \times C \times H \times W}$ denotes the final predict of subnetwork $i$, and $Y \in \mathbb{R}^{N \times C \times H \times W}$ is the ground truth in each stage.

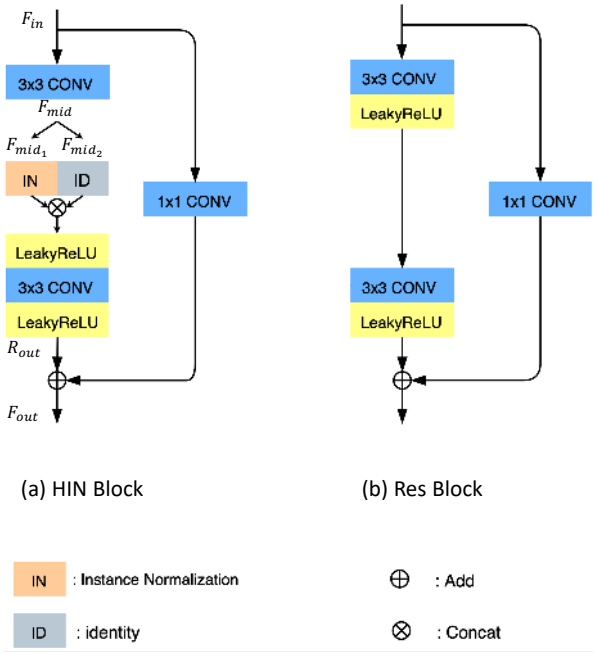


Figure 3: Proposed Half Instance Normalization Block (HIN Block) and ResBlock in details.

Then we optimize HINet end-to-end as follows:

$$Loss = -\sum_{i=1}^{2} PSNR((R_i + X_i), Y) \qquad (1)$$

### 3.2. Half Instance Normalization Block

Because of variance of small image patches differ a lot among mini-batches and the different formulations of training and testing [51], BN[18] is not commonly used in low-level tasks [23, 26]. Instead, Instance Normalization (IN) keeps the same normalization procedure consistent in both training and inference. Further, IN re-calibrates the mean and variance of features without the influence of batch dimension, which can keep more scale information than BN. We use IN to build Half Instance Normalization Block (HIN block). By introducing HIN block, the modeling capacity of HINet is improved (as shown in Figure 4). Moreover, the extra parameters and computational cost introduced by IN can be ignored.

As shown in Figure 3 a. HIN block firstly takes the input features $F_{in} \in \mathbb{R}^{C_{in} \times H \times W}$ and generates intermediate features $F_{mid} \in \mathbb{R}^{C_{out} \times H \times W}$ with $3 \times 3$ convolution, where $C_{in}$/$C_{out}$ is the number of input/output channels for HIN block. Then, the features $F_{mid}$ are divided into two parts ($F_{mid_1}/F_{mid_2} \in \mathbb{R}^{C_{out}/2 \times H \times W}$). The first part $F_{mid_1}$ is normalized by IN with learnable affine parameters and then concatenates with $F_{mid_2}$ in channel dimension. HIN blocks use IN on the half of the channels and keep context information by the other half of the channels. Later experiments will also show that this design is more friendly to features in shallow layers of the network. After the concat operation, the residual features $R_{out} \in \mathbb{R}^{C_{out} \times H \times W}$ are obtained by passing features to one $3 \times 3$ convolution layer and two leaky ReLU layers, which is shown in Figure 3 a. Finally, HIN blocks output $F_{out}$ by add $R_{out}$ with shortcut features (obtained after $1 \times 1$ convolution).

## 4. Experiments

We evaluate our approach on multiple datasets across image restoration tasks. We report the standard metrics in image restoration including PSNR and SSIM. The datasets used for training are described next.

### 4.1. Implementation Details

**Datasets** As in [56], we train our models on SIDD [1] for image denoising, GoPro [31] for image deblurring, and 13,712 clean-rain image pairs (for simplicity, denoted as Rain13k in the following) gathered from [10, 25, 49, 60, 61] for image deraining. In addition, we use REDS [30] dataset for image deblurring with JPEG artifacts, and we denote it as REDS dataset for simplicity. For evaluation, we follow the setting in the NTIRE 2021 Challenge on Image Deblurring [32], *i.e.* use 300 images in the validation set of REDS, denoted as REDS-val-300 next.

**Training** The networks are trained with Adam optimizer. The learning rate is set to $2 \times 10^{-4}$ by default, and decreased to $1 \times 10^{-7}$ with cosine annealing strategy [27]. We train our models on $256 \times 256$ patches with a batch size of 64 for $4 \times 10^5$ iterations. We apply flip and rotation as data augmentation. Following [65], we customize the network to the desired complexity by applying a scale factor $s$ on the number of channels, *e.g.* "HINet $s\times$" denotes scaling the number of channels in basic HINet $s$ times.

| | SIDD [1] | |
|---|---|---|
| Method | PSNR | SSIM |
| DnCNN [64] | 23.66 | 0.583 |
| MLP [5] | 24.71 | 0.641 |
| DM3D [8] | 25.65 | 0.685 |
| CBDNet* [14] | 30.78 | 0.801 |
| RIDNet* [2] | 38.71 | 0.951 |
| AINDNet* [20] | 38.95 | 0.952 |
| VDN [52] | 39.28 | 0.956 |
| SADNet* [7] | 39.46 | 0.957 |
| DANet+* [53] | 39.47 | 0.957 |
| CycleISP* [54] | 39.52 | 0.957 |
| MPRNet [56] | 39.71 | 0.958 |
| HINet 0.5× (**ours**) | <u>39.82</u> | 0.958 |
| HINet (**ours**) | **39.99** | 0.958 |

Table 1: Denoising comparisons on SIDD [1] dataset. * denotes the methods that use additional training data. Best and second best scores are **highlighted** and <u>underlined</u>. Our HINet achieves 0.28 dB absolute improvement in PSNR over the previous best method MPRNet [56].

| | GoPro [31] | |
|---|---|---|
| Method | PSNR | SSIM |
| Xu et al. [48] | 21.00 | 0.741 |
| Hyun et al. [17] | 23.64 | 0.824 |
| Whyte et al. [46] | 24.60 | 0.846 |
| Gong et al. [13] | 26.40 | 0.863 |
| DeblurGAN [21] | 28.70 | 0.858 |
| Nah et al. [31] | 29.08 | 0.914 |
| Zhang et al. [62] | 29.19 | 0.931 |
| DeblurGAN-v2 [22] | 29.55 | 0.934 |
| SRN [39] | 30.26 | 0.934 |
| Gao et al. [12] | 30.90 | 0.935 |
| DBGAN [63] | 31.10 | 0.942 |
| MT-RNN [34] | 31.15 | 0.945 |
| DMPHN [57] | 31.20 | 0.940 |
| Suin et al. [37] | 31.85 | 0.948 |
| MPRNet [56] | <u>32.66</u> | **0.959** |
| HINet (**ours**) | **32.71** | **0.959** |

Table 2: Deblurring comparisons on GoPro [31] dataset. Best and second best scores are **highlighted** and <u>underlined</u>. Our HINet achieves 0.05 dB absolute improvement in PSNR over the previous best method MPRNet [56].

### 4.2. Main Results

We show the effectiveness of HINet on different datasets in Table 1, Table 2 and Table 3. In addition, we compare the MACs (*i.e.* multiplier-accumulator operations) and inference time of MPRNet [56] and HINet in Table 4. MACs is estimated when the input is $1 \times 3 \times 256 \times 256$. Moreover, we conduct quality experiments to show the superiority of our method as shown in Figure 5.

| Method | Test100 [61] | | Rain100H [49] | | Rain100L [49] | | Test2800 [10] | | Test1200 [60] | | Average | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM |
| DerainNet [9] | 22.77 | 0.810 | 14.92 | 0.592 | 27.03 | 0.884 | 24.31 | 0.861 | 23.38 | 0.835 | 22.48 | 0.796 |
| SEMI [45] | 22.35 | 0.788 | 16.56 | 0.486 | 25.03 | 0.842 | 24.43 | 0.782 | 26.05 | 0.822 | 22.88 | 0.744 |
| DIDMDN [60] | 22.56 | 0.818 | 17.35 | 0.524 | 25.23 | 0.741 | 28.13 | 0.867 | 29.65 | 0.901 | 24.58 | 0.770 |
| UMRL [50] | 24.41 | 0.829 | 26.01 | 0.832 | 29.18 | 0.923 | 29.97 | 0.905 | 30.55 | 0.910 | 28.02 | 0.880 |
| RESCAN [24] | 25.00 | 0.835 | 26.36 | 0.786 | 29.80 | 0.881 | 31.29 | 0.904 | 30.51 | 0.882 | 28.59 | 0.857 |
| PreNe [35] | 24.81 | 0.851 | 26.77 | 0.858 | 32.44 | 0.950 | 31.75 | 0.916 | 31.36 | 0.911 | 29.42 | 0.897 |
| MSPFN [19] | 27.50 | 0.876 | 28.66 | 0.860 | 32.40 | 0.933 | 32.82 | 0.930 | 32.39 | 0.916 | 30.75 | 0.903 |
| MPRNet [56] | 30.27 | 0.897 | 30.41 | 0.890 | 36.40 | 0.965 | 33.64 | 0.938 | 32.91 | 0.916 | 32.73 | 0.921 |
| HINet (**ours**) | **30.29** | **0.906** | **30.65** | **0.894** | **37.28** | **0.970** | **33.91** | **0.941** | **33.05** | **0.919** | **33.03** | **0.926** |

Table 3: Deraining comparisons on Test100 [61], Rain100H [49], Rain100L [49], Test2800 [10] and Test1200 [60]. In addition, the average results over these datasets are provided. Best and second best scores are **highlighted** and underlined. Our HINet achieves 0.3 dB absolute improvement in PSNR over the previous best method MPRNet [56].

| Dataset | Method | PSNR | MACs(G) | | Time(ms) | speedup |
|---|---|---|---|---|---|---|
| SIDD [1] | MPRNet [56] | 39.71 | 573.50 | 100% | 78.8 | 1× |
| | HINet 0.5× | 39.82 | 42.88 | 7.5% | 11.6 | 6.8× |
| | HINet | 39.99 | 170.71 | 29.8% | 27.0 | 2.9× |
| REDS [30] | MPRNet [56] | 28.81 | 760.11 | 100% | 90.1 | 1× |
| | HINet | 28.79 | 170.71 | 22.5% | 27.0 | 3.3× |
| GoPro [31] | MPRNet [56] | 32.66 | 760.11 | 100% | 90.1 | 1× |
| | HINet | 32.71 | 170.71 | 22.5% | 27.0 | 3.3× |
| Rain13k | MPRNet [56] | 32.73 | 141.28 | 100% | 37.4 | 1× |
| | HINet | 33.03 | 170.71 | 120.8% | 27.0 | 1.4× |

Table 4: Comparing the PSNR and MACs of MPRNet [56] and ours. For Rain13k, we compare the average PSNR over Test100 [61], Rain100H [49], Rain100L [49], Test2800 [10] and Test1200 [60]. MACs and Time are estimated with the input size of $1 \times 3 \times 256 \times 256$. The proportion of the calculations and the speedup compared to MPRNet [56] are also listed. Runtimes are computed with the Tesla V100 GPU.

**SIDD [1]** For image denoising, we train our model on the 320 high-resolution images of the SIDD dataset, and test on the 1280 patches from SIDD dataset. The results are shown in Table 1 and Table 4. Surprisingly, using only 7.5% of MPRNet [56]'s MACs, our model exceeds it by 0.11 dB in PSNR. Moreover, our model exceeds MPRNet a big margin, 0.28 dB in PSNR under 30% of the MACs, and is 2.9 times faster than MPRNet.

**REDS [30] and GoPro [31]** For image deblurring, we train our model on REDS [30] dataset with jpeg compression artifacts, and evaluate the results on REDS-val-300 as we described above. In addition, we conduct experiments on GoPro [31] dataset for image deblurring following [56, 57] *etc*. It contains 2103 image pairs for training and 1111 pairs for evaluation. In Table 2 and Table 4, we compare our approach to the state-of-the-art methods. We get a comparable performance to MPRNet [56] with only 22.5% MACs and 3.3× speed advantage. It indicates the efficiency of our model.

**Rain13k** For image deraining, we train our model on Rain13k as described above and evaluate results by Y channel in YCbCr color space following [19, 56]. It contains 13712 image pairs in the training set, and we evaluate the results on Test100 [61], Rain100H [49], Rain100L [49], Test2800 [10], Test1200 [60]. We show the results in Table 3. HINet achieves 0.3 dB absolute improvement in PSNR over the previous best method MPRNet [56] and 1.4× faster than it.

### 4.3. Ablation

The core idea of HINet lies in HIN Block. We evaluate it from multiple perspectives in this subsection. It should be noted that these experiments are *not* to achieve the performance of the state-of-the-art, but to illustrate the superiority of HIN on various models and image restoration tasks. Therefore we mainly use the UNet of the first stage of HINet 0.5×, without skip connections between encoder and decoder for ablation experiments for fast feedback, and we denoted the UNet as "HINet Simple" in next. It is trained on $512 \times 512$ patches with a batch size of 32 for $3 \times 10^5$ iterations. For optimizer, we follow the settings in [56] except we set the learning rate to $4 \times 10^{-4}$ instead of $2 \times 10^{-4}$.

**The effectiveness of Half Instance Normalization:** We conduct experiments on various models and datasets, as we shown in Table 5.

On REDS [30] dataset (shown in Table 5a), HIN brings 0.12 dB in PSNR for HINet Simple. This illustrates the effectiveness of HIN on the REDS [30] dataset and HINet Simple model.

On GoPro [31] dataset (shown in Table 5b), we reimplement the DMPHN(1-2-4-8) [57]. We set the learning rate to $2 \times 10^{-4}$ and decreased to $1 \times 10^{-6}$ with cosine annealing strategy [27]. We train the model on $256 \times 256$ patches, with a batch size of 32 for $2 \times 10^5$ iterations. Flip and rotation are applied as data augmentation. HIN brings 0.42 dB boost in PSNR, it demonstrates HIN is effective on different models and different datasets.

We train PRMID [44] and CycleISP [54] on SIDD [1] dataset as shown in Table 5c. For PRMID, we set the initial learning rate to $3 \times 10^{-3}$ with a batch size of 16 for

| Method | HIN? | REDS [30] PSNR | SSIM |
|---|---|---|---|
| HINet Simple | - | 28.11 | 0.847 |
| HINet Simple | ✓ | 28.23 | 0.850 |

(a) Comparison of the models with/without HIN on REDS [30] dataset for deblurring. HIN brings 0.12 dB in PSNR to HINet Simple.

| Method | HIN? | GoPro [31] PSNR | SSIM |
|---|---|---|---|
| DMPHN(1-2-4-8) [57] | - | 30.98 | 0.943 |
| DMPHN(1-2-4-8) [57] | ✓ | 31.40 | 0.948 |

(b) Comparison of the models with/without HIN on GoPro [31] dataset for deblurring. HIN brings 0.42 dB in PSNR to DMPHN(1-2-4-8) [57]. It indicates HIN's effectiveness is robust to datasets and models.

| Method | HIN? | SIDD [1] PSNR |
|---|---|---|
| PRMID [44] | - | 39.30 |
| PRMID [44] | ✓ | 39.39 |
| CycleISP [54] | - | 39.50 |
| CycleISP [54] | ✓ | 39.56 |

(c) Comparison of the models with/without HIN on SIDD [1] dataset for denoising. HIN brings 0.09 dB and 0.06 dB in PSNR to PRMID [44] and CycleISP [54] respectively. It indicates HIN's effectiveness is robust to image restoration tasks and model size.

Table 5: To demonstrate the effectiveness of HIN, we conduct experiments on various datasets and models.

$4 \times 10^5$ iterations. For CycleISP, we set the initial learning rate to $2 \times 10^{-4}$ with a batch size of 16 for $3 \times 10^5$ iterations. Adam optimizer, cosine annealing strategy [27] and flip/rotation as data augmentation are adopted in both cases. HIN brings 0.09 dB and 0.06 dB boost in PSNR on PRMID and CycleISP respectively. Since SIDD [1] and GoPro [31]/REDS [30] are different image restoration task datasets, it demonstrates that HIN is effective in different image restoration tasks.

**Comparison with other Normalizations:** Normalization has not been fully explored in image restoration tasks. We compare HIN with other normalizations to demonstrate the superiority of our proposed HIN. We conduct experiments on REDS [30] dataset on HINet Simple. The results are shown in Table 6. We denote the HINet Simple w/o. Norm as the baseline in the following. BN [18] results a significant performance drop compares to baseline, *i.e.* 0.12 dB in PSNR. We conjure that is because of the inaccurate batch statistics estimation when batch size is small. It is alleviated by SyncBN [59] in some extent, *i.e.* HINet Simple w/. SyncBN brings 0.1 dB gain compares to HINet Simple w/. BN. However, it is still inferior to baseline (28.09 dB vs. 28.11 dB in PSNR). As we can see, with the help of IN [43],

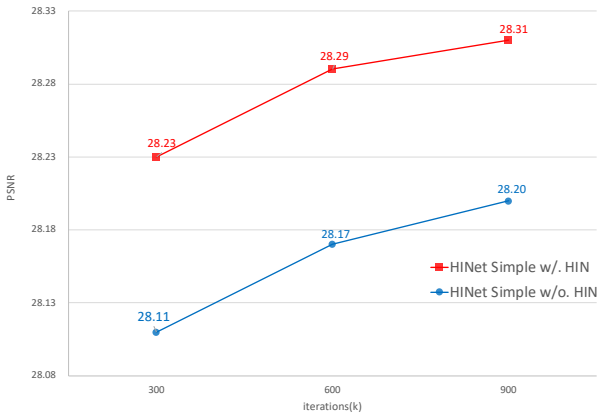


Figure 4: Effectiveness of HIN when training more iterations.

HINet Simple w/. IN exceed baseline by 0.03 dB in PSNR. It indicates IN facilitates the training of image restoration tasks. As shown in Table 6, HIN exceeds its counterparts, and it brings 0.12 dB gain in PSNR compares to baseline. It demonstrates the superiority of our proposed HIN.

| Method | PSNR |
|---|---|
| HINet Simple w/o. Norm | 28.11 |
| HINet Simple w/. BN [18] | 27.99 |
| HINet Simple w/. SyncBN [59] | 28.09 |
| HINet Simple w/. LN [4] | 28.09 |
| HINet Simple w/. IN [43] | 28.14 |
| HINet Simple w/. IBN [33] | 28.18 |
| HINet Simple w/. GN [47] | 28.19 |
| HINet Simple w/. HIN(**ours**) | **28.23** |

Table 6: Comparison of different normalization approaches on the image restoration task. Experiments are conducted on REDS [30] dataset. HINet Simple w/. BN means a fully normalized model with BN in the encoder.

**More training iterations:** We analyze the impact of increasing the number of training iterations based on HINet Simple. We train the model on REDS [30] dataset for 300k, 600k, and 900k iterations with/without HIN respectively. The results are shown in Figure 4. The gap between HINet Simple w/. HIN and HINet Simple w/o. HIN does not decrease as the number of iterations increased. We conjure that this is because HIN increases the upper limit of the model, not just speeds up the convergence.

**Guideline of add HIN layer in an existing network:** HINet Simple consists of 5 encoder blocks and 5 decoder blocks. We further explore the appropriate add location of HIN Block. The results are shown in Table 7. It indicates that adding HIN to all encoder blocks gets the highest score. In addition, adding HIN to the encoder and decoder causes performance drop *i.e.* 28.23 dB to 28.21 dB. It demonstrates

that adding more HIN does not necessarily lead to better performance. In practice, add one HIN layer to each encoder block might be a good choice.

| Method | Encoder 1 | 2 | 3 | 4 | 5 | Decoder | PSNR |
|---|---|---|---|---|---|---|---|
| HINet Simple | - | - | - | - | - | - | 28.11 |
| | ✓ | ✓ | - | - | - | - | 28.15 |
| | - | ✓ | ✓ | - | - | - | 28.19 |
| | - | - | ✓ | ✓ | - | - | 28.19 |
| | - | - | - | ✓ | ✓ | - | 28.21 |
| | ✓ | ✓ | ✓ | - | - | - | 28.19 |
| | - | ✓ | ✓ | ✓ | - | - | 28.20 |
| | - | - | ✓ | ✓ | ✓ | - | 28.21 |
| | ✓ | ✓ | ✓ | ✓ | - | - | 28.21 |
| | - | ✓ | ✓ | ✓ | ✓ | - | 28.22 |
| | ✓ | ✓ | ✓ | ✓ | ✓ | - | **28.23** |
| | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | 28.21 |

Table 7: Guideline of add HIN layer in an existing network (*e.g*. HINet Simple): adding HIN to all encoder blocks gets highest score, while more HIN does not necessarily lead to better performance.

### 4.4. Extension to HINet:

In order to achieve better performance on NTIRE 2021 Image Deblurring Challenge Track2. JPEG artifacts [32], we extend HINet, and adopt test time augmentation strategy. To further enhance the performance, we ensemble 3 similar models. In this subsection, we discuss the impact of these three methods on the results. And at the end, the results of the development phase and the test phase are provided. The results are evaluated on REDS-val-300, except the test phase result.

**Wider, Deeper:** It has been demonstrated that scaling up the model from width and depth improves the model capacity [38, 51]. For width, we simply use HINet $2\times$. For depth, we add two residual blocks at the end of each encoder block and decoder block. It achieves a PSNR of 29.05 dB on REDS-val-300.

**Test Time Augmentation and Ensemble:** We adopt flip and rotation as test time augmentation. It brings about 0.14 dB in PSNR. In addition, we randomly crop hundreds of patches, randomly adopt flip and rotation augmentation on them. It brings about 0.05 dB in PSNR. We simply average the predictions of 3 models as a model ensemble. It brings about 0.01 dB in PSNR. With these strategies, our model boost PSNR from 29.05 dB to 29.25 dB.

**Development phase result:** For the development phase, we randomly crop 720 patches. The results are shown in Table 8.

**Test phase result:** For the test phase, we randomly crop 1000 patches. The results are shown in 9.

| Participants | PSNR | SSIM | rank |
|---|---|---|---|
| **ours** | **29.25** | **0.8190** | 1 |
| participant A | 29.17 | 0.8183 | 2 |
| participant B | 29.14 | 0.8170 | 3 |
| participant C | 29.11 | 0.8171 | 4 |
| participant D | 29.10 | 0.8165 | 5 |
| participant E | 29.01 | 0.8141 | 6 |
| participant F | 28.94 | 0.8145 | 7 |
| participant G | 28.75 | 0.8093 | 8 |
| participant H | 28.68 | 0.8103 | 9 |
| participant I | 28.66 | 0.8082 | 10 |

Table 8: Development phase result of NTIRE 2021 Image Deblurring Challenge Track 2. JPEG artifacts [32]. Best and second best scores are **highlighted** and underlined. Our proposed method outperform others by 0.08 dB in PSNR.

| Participants | PSNR | SSIM | rank |
|---|---|---|---|
| **ours** | **29.70** | 0.8403 | 1 |
| Noah_CVLab | 29.62 | 0.8397 | 2 |
| CAPP_OB | 29.60 | 0.8398 | 3 |
| Baidu | 29.59 | 0.8381 | 4 |
| SRC-B | 29.56 | 0.8385 | 5 |
| Mier | 29.34 | 0.8355 | 6 |
| VIDAR | 29.33 | **0.8565** | 7 |
| DuLang | 29.17 | 0.8325 | 8 |
| TeamInception | 29.11 | 0.8292 | 9 |
| Giantpandacv | 29.07 | 0.8286 | 10 |
| Maradona | 28.96 | 0.8264 | 11 |
| LAB FHD | 28.92 | 0.8259 | 12 |
| SYJ | 28.81 | 0.8222 | 13 |
| Dseny | 28.26 | 0.8081 | 14 |
| IPCV IITM | 27.91 | 0.8028 | 15 |
| DMLAB | 27.84 | 0.8013 | 16 |
| Blur Attack | 27.41 | 0.7887 | 17 |

Table 9: NTIRE 2021 Image Deblurring Challenge Track 2. JPEG artifacts result [32]. Best and second best scores are **highlighted** and underlined. We exceed other participants over 0.08 dB in PSNR.

## 5. Conclusion

In this work, we reuse Normalization in image restoration tasks. Specifically, we introduce Instance Normalization into a residual block and design an effective and efficient block: Half Instance Normalization Block (HIN Block). In HIN Block, we apply Instance Normalization for half of the intermediate features and keep the content information at the same time. Based on HIN Block, we further propose a multi-stage network called HINet. Between each stage, we use feature fusion and attention-guided map [55] across stages to ease the flow of information and enhance the multi-scale feature expression. Our proposed HINet surpasses the SOTA on various image restoration tasks. In addition, by using HINet, we won 1st place on the NTIRE 2021 Image De-blurring Challenge - Track2. JPEG Artifacts [32].

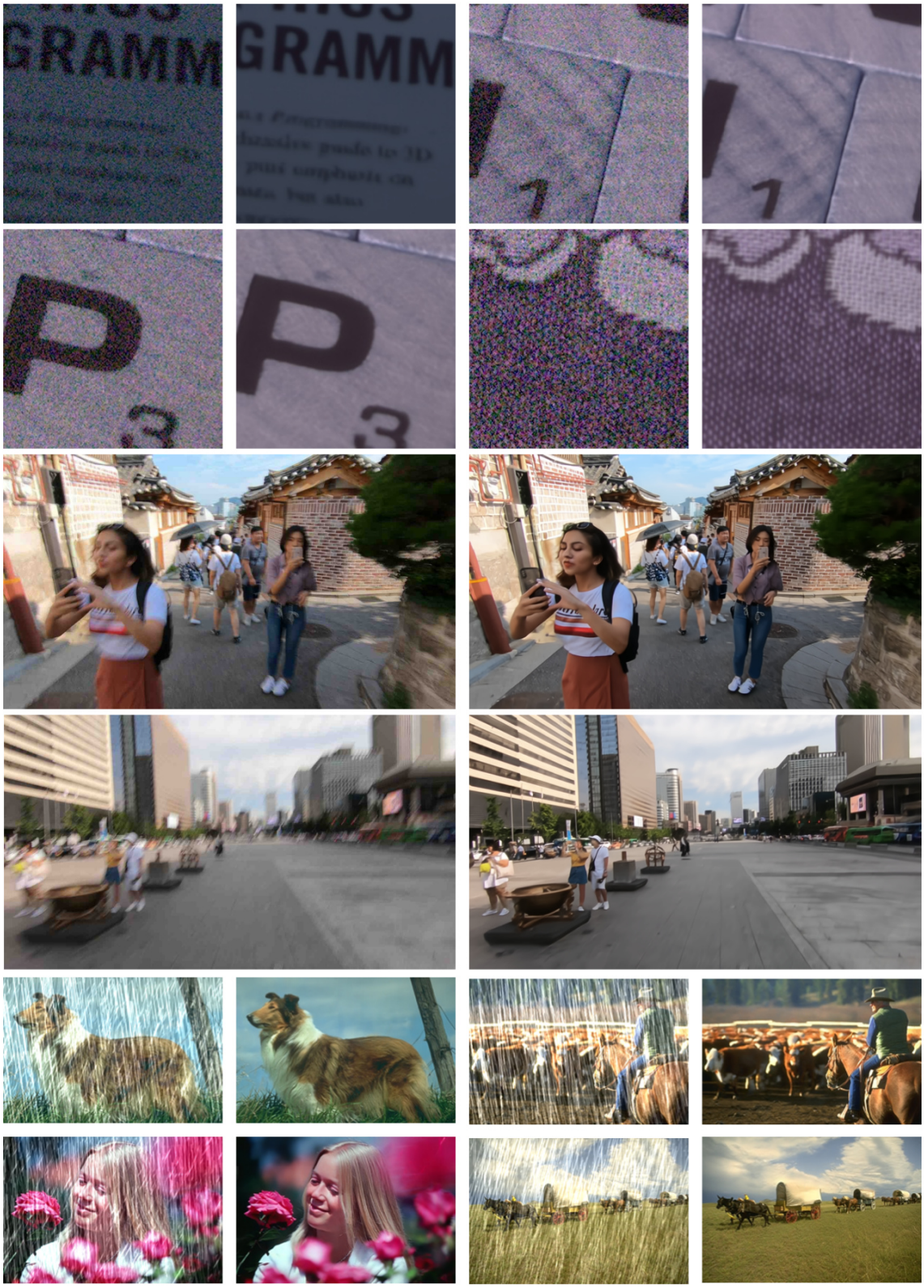

Figure 5: More visualized results of HINet on various image restoration tasks. For each image pair, the left one is degraded and the right one is predicted by HINet.

## References


[1] Abdelrahman Abdelhamed, Stephen Lin, and Michael S Brown. A high-quality denoising dataset for smartphone cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1692–1700, 2018.

[2] Saeed Anwar and Nick Barnes. Real image denoising with feature attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3155–3164, 2019.

[3] Saeed Anwar and Nick Barnes. Densely residual laplacian super-resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[4] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[5] Harold C Burger, Christian J Schuler, and Stefan Harmeling. Image denoising: Can plain neural networks compete with bm3d? In *2012 IEEE conference on computer vision and pattern recognition*, pages 2392–2399. IEEE, 2012.

[6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020.

[7] Meng Chang, Qi Li, Huajun Feng, and Zhihai Xu. Spatial-adaptive network for single image denoising. In *European Conference on Computer Vision*, pages 171–187. Springer, 2020.

[8] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on image processing*, 16(8):2080–2095, 2007.

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

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

[11] Xueyang Fu, Borong Liang, Yue Huang, Xinghao Ding, and John Paisley. Lightweight pyramid networks for image deraining. *IEEE transactions on neural networks and learning systems*, 31(6):1794–1807, 2019.

[12] Hongyun Gao, Xin Tao, Xiaoyong Shen, and Jiaya Jia. Dynamic scene deblurring with parameter selective sharing and nested skip connections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3848–3856, 2019.

[13] Dong Gong, Jie Yang, Lingqiao Liu, Yanning Zhang, Ian Reid, Chunhua Shen, Anton Van Den Hengel, and Qinfeng Shi. From motion blur to motion flow: A deep learning solution for removing heterogeneous motion blur. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2319–2328, 2017.

[14] Shi Guo, Zifei Yan, Kai Zhang, Wangmeng Zuo, and Lei Zhang. Toward convolutional blind denoising of real photographs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1712–1722, 2019.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[16] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017.

[17] Tae Hyun Kim, Byeongjoo Ahn, and Kyoung Mu Lee. Dynamic scene deblurring. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3160–3167, 2013.

[18] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[19] Kui Jiang, Zhongyuan Wang, Peng Yi, Chen Chen, Baojin Huang, Yimin Luo, Jiayi Ma, and Junjun Jiang. Multi-scale progressive fusion network for single image deraining. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8346–8355, 2020.

[20] Yoonsik Kim, Jae Woong Soh, Gu Yong Park, and Nam Ik Cho. Transfer learning from synthetic to real-noise denoising with adaptive instance normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3482–3492, 2020.

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

[22] Orest Kupyn, Tetiana Martyniuk, Junru Wu, and Zhangyang Wang. Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8878–8887, 2019.

[23] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

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

[25] Yu Li, Robby T Tan, Xiaojie Guo, Jiangbo Lu, and Michael S Brown. Rain streak removal using layer priors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2736–2744, 2016.

[26] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.

[27] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[28] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, pages 116–131, 2018.

[29] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Citeseer, 2013.

[30] Seungjun Nah, Sungyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Radu Timofte, and Kyoung Mu Lee. Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[31] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3883–3891, 2017.

[32] Seungjun Nah, Sanghyun Son, Suyoung Lee, Radu Timofte, Kyoung Mu Lee, et al. NTIRE 2021 challenge on image deblurring. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2021.

[33] Xingang Pan, Ping Luo, Jianping Shi, and Xiaoou Tang. Two at once: Enhancing learning and generalization capacities via ibn-net. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 464–479, 2018.

[34] Dongwon Park, Dong Un Kang, Jisoo Kim, and Se Young Chun. Multi-temporal recurrent neural networks for progressive non-uniform single image deblurring with incremental temporal training. In *European Conference on Computer Vision*, pages 327–343. Springer, 2020.

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

[36] 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*, pages 234–241. Springer, 2015.

[37] Maitreya Suin, Kuldeep Purohit, and AN Rajagopalan. Spatially-attentive patch-hierarchical network for adaptive motion deblurring. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3606–3615, 2020.

[38] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019.

[39] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8174–8182, 2018.

[40] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9627–9636, 2019.

[41] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor S Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *ICML*, volume 1, page 4, 2016.

[42] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[43] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6924–6932, 2017.

[44] Yuzhi Wang, Haibin Huang, Qin Xu, Jiaming Liu, Yiqun Liu, and Jue Wang. Practical deep raw image denoising on mobile devices. In *European Conference on Computer Vision*, pages 1–16. Springer, 2020.

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

[46] Oliver Whyte, Josef Sivic, Andrew Zisserman, and Jean Ponce. Non-uniform deblurring for shaken images. *International journal of computer vision*, 98(2):168–186, 2012.

[47] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[48] Li Xu, Shicheng Zheng, and Jiaya Jia. Unnatural l0 sparse representation for natural image deblurring. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1107–1114, 2013.

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

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

[51] Jiahui Yu, Yuchen Fan, Jianchao Yang, Ning Xu, Zhaowen Wang, Xinchao Wang, and Thomas Huang. Wide activation for efficient and accurate image super-resolution. *arXiv preprint arXiv:1808.08718*, 2018.

[52] Zongsheng Yue, Hongwei Yong, Qian Zhao, Lei Zhang, and Deyu Meng. Variational denoising network: Toward blind noise modeling and removal. *arXiv preprint arXiv:1908.11314*, 2019.

[53] Zongsheng Yue, Qian Zhao, Lei Zhang, and Deyu Meng. Dual adversarial network: Toward real-world noise removal and noise generation. In *European Conference on Computer Vision*, pages 41–58. Springer, 2020.

[54] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Cycleisp: Real image restoration via improved data synthesis. In *CVPR*, 2020.

[55] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. *arXiv preprint arXiv:2102.02808*, 2021.

[56] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. In *CVPR*, 2021.

[57] Hongguang Zhang, Yuchao Dai, Hongdong Li, and Piotr Koniusz. Deep stacked hierarchical multi-patch network for image deblurring. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[58] Hongguang Zhang, Yuchao Dai, Hongdong Li, and Piotr Koniusz. Deep stacked hierarchical multi-patch network for image deblurring. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5978–5986, 2019.

[59] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7151–7160, 2018.

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

[61] He Zhang, Vishwanath Sindagi, and Vishal M Patel. Image de-raining using a conditional generative adversarial network. *IEEE transactions on circuits and systems for video technology*, 30(11):3943–3956, 2019.

[62] Jiawei Zhang, Jinshan Pan, Jimmy Ren, Yibing Song, Linchao Bao, Rynson WH Lau, and Ming-Hsuan Yang. Dynamic scene deblurring using spatially variant recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2521–2529, 2018.

[63] Kaihao Zhang, Wenhan Luo, Yiran Zhong, Lin Ma, Bjorn Stenger, Wei Liu, and Hongdong Li. Deblurring by realistic blurring. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2737–2746, 2020.

[64] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, 26(7):3142–3155, 2017.

[65] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6848–6856, 2018.