# Stochastic Window Transformer for Image Restoration

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

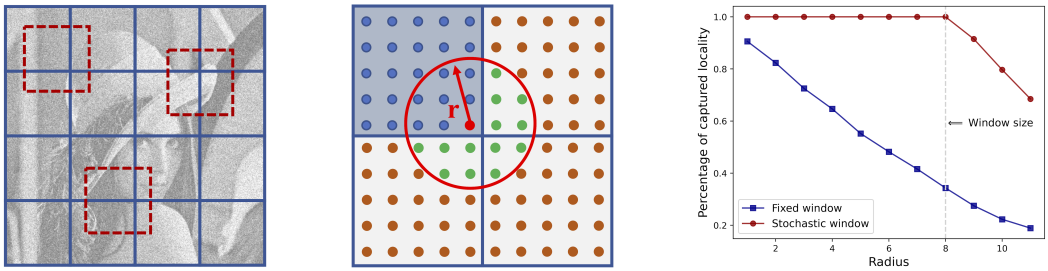
Thanks to the strong representation ability, transformers have attained impressive results for image restoration. However, existing transformers do not carefully take into account the particularities of image restoration. Basically, image restoration requires that the ideal approach should be invariant to translation of degradation, i.e., undesirable degradation should be removed irrespective of its position within the image. Moreover, local relationships play a vital role and should be faithfully exploited for recovering clean images. Nevertheless, most of transformers have resorted to either fixed local window based attention or global attention, which unfortunately breaks the translation invariance and further causes huge loss of local relationships. To address these issues, we propose an elegant stochastic window strategy for transformers. Specifically, we introduce the window partition with stochastic shift to replace the original fixed window partition for training and elaborate the layer expectation propagation algorithm to efficiently approximate the expectation of the induced stochastic transformer for testing. The stochastic window transformer can not only enjoy powerful representation but also maintain the desired property of translation invariance and locality. Experiments validate the stochastic window strategy constantly improves performance on various image restoration tasks (image deraining, denosing, and deblurring) by significant margins.

## 1 Introduction

Image restoration aims to recover the latent clean images from their noise-polluted counterparts, which lays foundation for various computer vision tasks. In general, image restoration methods should satisfy a fundamental constraint: *the ideal approach should remove undesirable degradation irrespective of its position within the image.* In other words, maintaining the translation invariance is the basic requirement for image restoration methods. In this sense, convolutional neural networks (CNNs) are well-suited for various image restoration tasks [52, 21, 9, 69, 66, 57, 62, 54, 68, 6, 14]. In comparison with multi-layer perceptrons (MLPs), the distinct characteristics of CNNs [24, 20, 16] are their built-in locality and translation invariance (weight sharing). These two priors play a vital role in image restoration, i.e., pixels within local regions tend to exhibit strong correlations while translation invariance is the desiderata for any potential image restoration method.

On the other hand, transformer-based methods [2, 51, 30, 58] for image restoration are not compatible with the translation invariance and locality. For now, the patterns that transformers adopted for recovering high-resolution images can be concluded as small patch with global attention (e.g., $48 \times 48$ for IPT [2]) and large patch with local attention (e.g., Uformer [51] and SwinIR [30]). Nevertheless, neither of these two patterns can meet the requirements of translation invariance or locality. Here we detail the latter pattern and the former one is included in the supplemental material.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.




(a) The fixed window partition breaks the translation invariance.

(b) The fixed local window cannot faithfully capture local relationships.

(c) The comparison of percentage of captured locality between the fixed and stochastic window strategy.

Figure 1: Illustration of broken translation invariance and lost locality of the fixed window strategy.

The latter pattern partitions the feature map into fixed non-overlapped windows and self-attention is restricted inside these local windows. However, we argue that the fixed partition strategy will break the translation invariance and further lead to huge loss of local relationships. As shown in Fig. 1(a), for the fixed window partition strategy, translation invariance can only be ensured among specific partitioned windows (blue windows)[1] rather than general windows with arbitrary shifts (red windows), which disagrees with the translation invariance. The reason behind this is that the fixed window partition imposes the artificial prior to windows with specific shifts, which breaks the translation invariance. Fig. 1(b) intuitively illustrates that the fixed window partition cannot capture intact local relationships. Without loss of generality, we consider the Euclidean distance, based on which the neighbor space will be the region encompassed by a circle, in 2D plane to measure locality. For a certain token (central red dot), many tokens (green dots) within its neighbor space (encompassed by the red circle with radius $r$) cannot participate in the calculation of self-attention, which leads to tremendous loss of local relationships. As the neighbor space expands, i.e., increasing radius $r$, the loss will be more severe. Fig. 1(c) quantitatively depicts the proportion of local relationships that are captured by the fixed window partition over the total relationships existing in the neighbor space. Ideally, when the feature map is divided into non-overlapping windows of size $s \times s$, it is expected that the relationships of token pairs with distance less than $s$ can be faithfully captured. However, we can observe that there exists significant loss of local relationships even within the neighbor space whose radius is much smaller than $s$. A seemingly straightforward solution to this problem is to adopt the sliding window strategy to scan the whole feature map using the local window like CNNs. However, this strategy will incur huge overhead in terms of both memory and computing speed.

In this work, we propose a novel stochastic window strategy to impart translation invariance to the transformer and make it faithfully model local relationships. Specifically, instead of partitioning the feature map into fixed non-overlapped windows, we choose to cover the whole feature map with stochastically shifted window partition (see Fig. 2). By introducing stochastic shift, all windows are treated equally so that the translation invariance and locality can be ensured. As shown in Fig. 1(c), our stochastic window strategy can capture complete local relationships until the radius of neighbor space reaches the window size. It is noteworthy that unlike the sliding window strategy, the time and memory overhead consumed for training stochastic window transformer is comparable to the fixed window strategy. For testing, we further design the layer expectation propagation algorithm to efficiently approximate the effect of taking expectation with respect to the stochastic shift, during which the translation invariance and locality can also be maintained. The contributions are threefold. 1. We analyze that there exists the translation invariance breaking and loss of local relationships in existing transformer-based image restoration approaches. To our knowledge, this is the first work to analyze and point out the importance of translation invariance for image restoration. 2. We propose a new stochastic window strategy, comprising the stochastic window for training and layer expectation propagation for testing, to compensate for the broken translation invariance of transformers and

[1]These windows are processed using shared weights.

enable them to faithfully model local relationships. 3. We conduct extensive experiments to validate that by equipping our stochastic window strategy, the performance on various image restoration tasks (e.g., image deraining, denoising and deblurrring) can be constantly improved.

## 2 Related works

**Image Restoration.** Image Restoration [63, 11, 5, 33, 31, 35, 59, 37, 27, 26] aims to restore the clean image from its degraded version. In recent years, remarkable progress against traditional model-driven methods [4, 50, 15] has been achieved due to the development of deep learning technologies [25], especially CNNs [24, 20, 16]. Instead of relying on preset image priors, learning based methods directly learn to project from noisy to clean ones from a large collection of noisy-clean image pairs. Numerous representative CNNs have sprung up across various classical image restoration tasks, including image denoising [23, 56, 64, 62, 7, 70], image super-resolution[68, 6, 14, 18, 71], image deblurring [9, 21, 36, 21, 44, 61, 40, 39], image deraining [54, 47, 13, 60, 48, 38, 28], etc. In general, the translation invariance (weight sharing) and locality have been hard-coded into the inherent structure of CNNs so that it seems CNNs are well-suited for image restoration tasks. However, compared with Transformer, CNNs are restricted by their limited flexibility.

**Vision Transformer.** Recently, with great success of transformer [46, 8] in the NLP field, Vision Transformers [10, 45, 49, 32, 53, 55, 3] have also been prevalent in vision community. ViT [10] treated image patches as token sequence and applied the vanilla transformer on it for image classification. With the goal of bringing in reasonable priors into transformer to improve efficiency, Swin [32] introduced local window based attention and established a hierarchical architecture. Inspired by the key insights from high-level vision, a few transformers [2, 30, 51, 58] for low-level vision have arose. But most of them directly transfer high-level designs (e.g., local attention) into low-level vision without careful consideration of its particularities. As we discuss above, unlike CNNs, transformers do not possess the translation invariance and intact locality. In general, low-level tasks usually require to accomplish pixel-level regression, where more strict locality and translation invariance are expected. Besides, the CNNs-style sliding window strategy will incur huge burden in terms of memory and computing speed. Therefore, an efficient mechanism to remedy the broken translation invariance and locality of transformer is essential for low-level vision.

## 3 Stochastic Window Transformer for Image Restoration

Transformers have attained impressive performance for image restoration due to their strong representation ability. Given the quadratic complexity in both computation and memory usage, transformers for image restoration tend to employ the local attention. Except for high-efficiency, local attention should have been designed to model local relationships. However, as we analyzed before, local attention breaks the ideal translation invariance and further leads to huge loss of local relationships. As shown in Fig. 2, for a feature map, the window partition with arbitrary shift $(\xi_h, \xi_w)$ contains a comparable quantity of local relationships and implies the translation invariance among divided windows. In other words, all the window partitions are equally informative in terms of translation invariance and locality and should be treated equally. Therefore, it is unreasonable for the fixed window strategy to express infinite favoritism towards the certain partition (e.g. $(0, 0)$ or $(\frac{s}{2}, \frac{s}{2})$) while simply discarding others. In this work, we propose the stochastic window strategy, by which all the window partitions are treated fairly. The network can be trained based on local attention but with stochastic rather than fixed shift. Therefore, transformer with the stochastic window can be trained as efficiently as the fixed window. At test time, we propose the layer expectation propagation algorithm to approximate the expectation of the introduced stochastic shift, which also helps to maintain the desired property of translation invariance and locality.

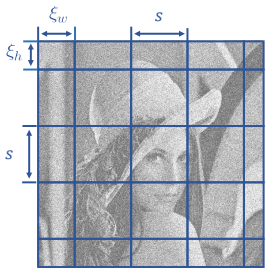



Figure 2: Window partition with shift $(\xi_h, \xi_w)$. $s \times s$: spatial size of the local window.

### 3.1 Stochastic Window Strategy

Stochastic window strategy aims to impart translation invariance to transformer and make it faithfully exploit local relationships. Canonical transformer consists of alternate layers of self-attention (SA) and MLP. In order to promote efficiency, transformer tends to employ local window based attention and shifted window strategy is utilized to allow inter-window connections. Specifically, the feature map is partitioned into non-overlapped windows and then SA is computed within local windows. Suppose the window partition is denoted by $\mathrm{Par}(\cdot; s, \xi_h, \xi_w)$, where size of the local window is $s$ and shift of the whole window partition is $(\xi_h, \xi_w)$ (see Fig. 2). With these notations, the shifted window transformer can be formulated as:

$$\begin{aligned} z_l &= \mathrm{SA}(\mathrm{Par}(x_{l-1}; s, 0, 0)) + x_{l-1}, \\ x_l &= \mathrm{MLP}(z_l) + z_l, \\ z_{l+1} &= \mathrm{SA}(\mathrm{Par}(x_l; s, \frac{s}{2}, \frac{s}{2})) + x_l, \\ x_{l+1} &= \mathrm{MLP}(z_{l+1}) + z_{l+1}, \end{aligned} \tag{1}$$

where $z_l$ and $x_l$ denote the output feature of SA and MLP for layer $l$, respectively. To keep notation uncluttered, we overlook layer normalization in Eq. 1. We figure out that all the window partitions are equally informative and preserving only one configuration each layer will lead to the broken translation invariance and tremendous loss of locality. The thought of informative equality among all the window partitions leads us to the stochastic window transformer, which is computed as:

$$\begin{aligned} z_l &= \mathrm{SA}(\mathrm{Par}(x_{l-1}; s, \xi_h^l, \xi_w^l)) + x_{l-1}, \quad (\xi_h^l, \xi_w^l) \in \Re(s), \\ x_l &= \mathrm{MLP}(z_l) + z_l, \end{aligned} \tag{2}$$

where the set $\Re(s)$ contains all possible shifts. Given the periodicity of the window partition, $\Re(s)$ can be simplified as

$$\Re(s) := [0, \ldots, s-1] \times [0, \ldots, s-1], \tag{3}$$

where $\times$ means the Cartesian product. During training, $(\xi_h^l, \xi_w^l)$ are treated as random variables and sampled from $\Re(s)$ *independently and uniformly*. Suppose the total number of SA layer is $N$, stochastic shifts from different layers $\{(\xi_h^l, \xi_w^l)\}_{l=0}^{N-1}$ are deliberately designed to be independent so that faithful locality and translation invariance can be ensured on layer level. By this treatment, despite all possible shifts are taken into account for each individual layer, single forward propagation requires only a set of *sampled* shifts $\{(\xi_h^l, \xi_w^l)\}_{l=0}^{N-1}$ so that the stochastic window transformer can be trained efficiently. For testing, from Bayesian perspective, stochastic shifts should be averaged according to their posterior distribution. Hence, the exact inference procedure is:

$$\begin{aligned} F(x)^{\mathrm{test}} &= \sum_{\xi_h^0, \xi_w^0, \ldots, \xi_h^{N-1}, \xi_w^{N-1}} F(x; \xi_h^0, \xi_w^0, \ldots, \xi_h^{N-1}, \xi_w^{N-1}) \mathbb{P}(\xi_h^0, \xi_w^0, \ldots, \xi_h^{N-1}, \xi_w^{N-1}) \\ &= \sum_{(\xi_h^{N-1}, \xi_w^{N-1})} \cdots \sum_{(\xi_h^0, \xi_w^0)} F(x; \xi_h^0, \xi_w^0, \ldots, \xi_h^{N-1}, \xi_w^{N-1}) \mathbb{P}(\xi_h^0, \xi_w^0) \cdots \mathbb{P}(\xi_h^{N-1}, \xi_w^{N-1}), \end{aligned} \tag{4}$$

where $F(\cdot)$ denotes the function of the overall transformer and $\mathbb{P}$ is the uniform distribution. The derivation of Eq. 4 follows $\{(\xi_h^l, \xi_w^l)\}_{l=0}^{N-1}$ are independent. Obviously, according to Eq. 4, the single exact inference requires forward propagation exponential times so that the computational overhead of exact inference grows exponentially with the depth $N$, which is prohibitively expensive. Consequently, we turn to seek an approximate inference, which should also guarantee the translation invariance and intact locality, to replace the costly exact inference. Inspired by the approximation in Dropout [43], we propose the general layer expectation propagation algorithm to approach the original exact inference process. The layer expectation propagation is formulated as:

$$\begin{aligned} z_l^{\mathrm{test}} &= \sum_{(\xi_h^l, \xi_w^l)} \mathrm{SA}(\mathrm{Par}(x_{l-1}^{\mathrm{test}}; s, \xi_h^l, \xi_w^l)) \mathbb{P}(\xi_h^l, \xi_w^l) + x_{l-1}^{\mathrm{test}} \\ &= \mathop{\mathbb{E}}_{(\xi_h^l, \xi_w^l) \sim \mathbb{P}} \left[ \mathrm{SA}(\mathrm{Par}(x_{l-1}^{\mathrm{test}}; s, \xi_h^l, \xi_w^l)) \right] + x_{l-1}^{\mathrm{test}}, \\ x_l^{\mathrm{test}} &= \mathrm{MLP}(z_l^{\mathrm{test}}) + z_l^{\mathrm{test}}. \end{aligned} \tag{5}$$

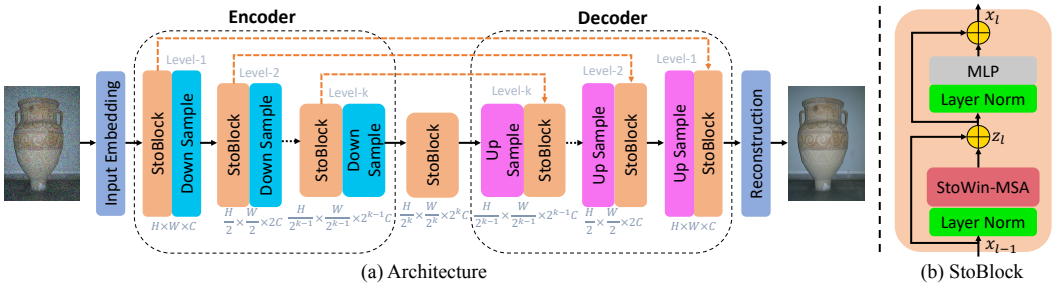



Figure 3: (a) The overall architecture of Stoformer††; (b) the structure of Stoformer†† block. StoWin-MSA is multi-head self-attention with stochastic window partition.

According to Eq. 4, the single approximate inference merely requires one forward propagation of the expected signal, which accelerates inference considerably. We then examine whether the stochastic window strategy is able to keep the translation invariance and model intact local relationships.

**Translation invariance.** During training, the stochastic window strategy treats arbitrarily shifted window equally and eliminates the particularity of certain windows introduced by the fixed window partition (blue windows in Fig. 1(b)). Therefore, the stochastic window strategy can maintain the translation invariance during training. For testing, by the layer expectation propagation, each individual token is able to aggregate information from its neighbor space with shared weights like CNNs. Consequently, the translation invariance can also be well maintained. We also provide experimental evidence to support the translation invariance in Sec. 4.2.

**Locality.** We have argued that the fixed window partition will lead to huge loss of local relationships since the majority of equally informative window partitions are simply discarded. As shown in Fig. 1(b), there exists some token pairs whose distance is small enough but they are not present in the same local window, which results in tremendous loss of locality. By contrast, with the stochastic window strategy, the window partition is randomly shifted, which will guarantee that any token pair will attend in the same window as long as their distance is smaller than the window size $s$. As shown in Fig. 1(c), the stochastic window strategy can capture intact local relationships when the radius of neighbor space is not larger than window size while the fixed window partition suffers from severe loss of local relationships. For testing, the proposed layer expectation propagation also considers all the shifted window partitions so that intact locality can be ensured as well.

### 3.2 Network Architecture

The proposed stochastic window transformer provides an effective and efficient way to integrate the well-acknowledged priors, i.e., locality and translation invariance, into transformer. For image restoration tasks which require pixel-level regression, faithfully exploiting these priors is especially important. To validate the effectiveness of the proposed stochastic window strategy, we integrate it with the widely-used U-shaped architecture [41, 19]. The resulting transformer, named Stoformer††, not only enjoys the intrinsically strong representation ability of Transformer but also maintains the ideal translation invariance and intact local relationships. Fig. 3 illustrates the overall architecture.

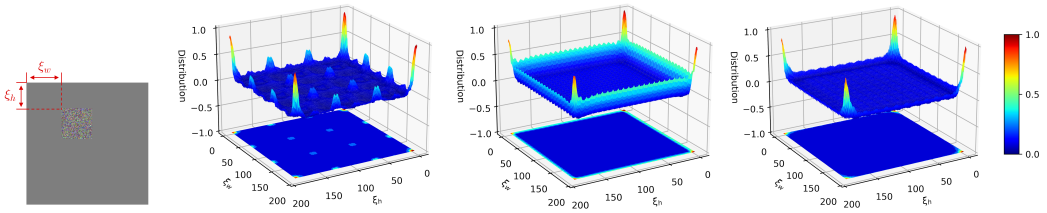
### 3.3 Implementation Details

**Loss function.** The loss function adopted for training is the Charbonnier loss [1], whose mathematical expression is:

$$L(I', I) = \sqrt{||I' - I||^2 + \epsilon^2}, \tag{6}$$

where $I'$ and $I$ are the restored and ground-truth image respectively. The constant $\epsilon$ is empirically set to $10^{-3}$.

**Training details.** Stoformer†† employs a four-level encoder-decoder structure. The numbers of StoBlock are $\{1, 2, 8, 8\}$ for level-1 to level-4 of Encoder and the blocks for Decoder are mirrored.




(a) Input (b) Fixed window strategy (c) VDN (d) Stochastic window strategy

Figure 4: Toy Experiment. The stochastic window strategy (d) helps to remedy the broken translation invariance caused by the fixed window partition (b).

The number of channel is set to 32 and the window size is $8 \times 8$. We train the network with Adam optimizer ($\beta_1 = 0.9, \beta_2 = 0.999$) with the initial learning rate $3 \times 10^{-4}$ gradually reduced to $1 \times 10^{-6}$ with the cosine annealing. The training samples are augmented by the horizontal flipping and rotation of 90°, 180°, or 270°. Please refer to the supplemental material for task-specific settings.

# 4 Experiments

## 4.1 Experimental Setup

In this section, we validate the effectiveness of the stochastic window strategy. Except the previous state-of-the-art methods are included for comparisons, we also elaborate four Stoformer variants: Stoformer$\Diamond\Diamond$, Stoformer$\dagger\Diamond$, Stoformer$\Diamond\dagger$, Stoformer$\dagger\dagger$. The first symbol aims to indicate whether the stochastic window is adopted for training($\dagger$: the stochastic window; $\Diamond$: the fixed window) and the second symbol represents whether the layer expectation propagation is adopted for testing($\dagger$: layer expectation propagation; $\Diamond$: the fixed window). In all experiments, we use the shifted window to specify the fixed window (Eq. 1). These variants are identical except for the aforementioned window strategy. In particular, Stoformer$\Diamond\Diamond$ is reduced to the traditional fixed window transformer and Stoformer$\dagger\dagger$ is the proposed stochastic window transformer.

## 4.2 Toy Experiment

To illustrate the broken translation invariance of the fixed window strategy and the remedy of our proposed stochastic window strategy to this effect, a toy experiment is performed: we add a Gaussian noise patch with $\sigma = 50$ whose spatial size is $64 \times 64$ to an 8-bit pure color image with spatial size $256 \times 256$, the pixel value of which is constantly set to the medium value 127. We move the Gaussian noise patch with stride 1 and obtain total $192 \times 192 = 36864$ noisy images. Then we denoise these noisy images using the fixed and stochastic window transformer, respectively and the PSNR distributions with respect to $(\xi_h, \xi_w)$ are normalized to the fixed range $[0, 1]$ and plotted in Fig. 4. To further highlight the translation invariance of stochastic window transformer, we also include a typical CNNs-based denoiser VDN [56], which can ensure the translation invariance well due to inherent structure of CNNs, as the inference. Fig. 4 indicates clearly that there are great variations, especially for the interior, in the PSNR distribution of fixed window transformer while stochastic window transformer flattens the distribution significantly. Both stochastic window transformer and VDN contain the relatively uniform distribution for the interior, which indicates that the stochastic window strategy helps to maintain the desired translation invariance quite well. It is worth noting that all these distributions have variations on the image boundary, which can be attributed to the difference of the contexts of the image boundary.

## 4.3 Experiments on Image Deraining

We validate the stochastic window strategy on image deraining task. Specifically, except the elaborate four Stoformer variants, existing seven deraining methods are included: GMM [29], DDN [13],

Figure 5: Visual comparison of image deblurring on the GoPro.

Table 1: Quantitative results of image deraining on the SPA-Data.

| Method | SPA-Data | |
|---|---|---|
| | PSNR | SSIM |
| GMM [29] | 34.30 | 0.9428 |
| DDN [13] | 36.97 | 0.9604 |
| SPANet [48] | 40.24 | 0.9811 |
| JORDER-E [54] | 40.78 | 0.9811 |
| RCDNet [47] | 41.47 | 0.9834 |
| SPAR [37] | 44.10 | 0.9872 |
| Uformer [51] | 47.84 | 0.9925 |
| Stoformer◊◊ | 47.43 | 0.9923 |
| Stoformer◊† | 46.95 | 0.9917 |
| Stoformer†◊ | 48.85 | 0.9935 |
| Stoformer†† | 48.97 | 0.9938 |

Table 2: Quantitative results of image deblurring. Stoformers are trained only on GoPro dataset.

| Method | GoPro | | HIDE | |
|---|---|---|---|---|
| | PSNR | SSIM | PSNR | SSIM |
| Nah *et al.* [36] | 29.08 | 0.914 | 25.73 | 0.874 |
| DeblurGAN [21] | 28.70 | 0.858 | 24.51 | 0.871 |
| DeblurGAN-v2 [22] | 29.55 | 0.934 | 26.61 | 0.875 |
| DBGAN [65] | 31.10 | 0.942 | 28.94 | 0.915 |
| IPT [2] | 32.52 | - | - | - |
| MPRNet [59] | 32.66 | 0.959 | 30.96 | 0.939 |
| SPAIR [37] | 32.06 | 0.953 | 30.29 | 0.931 |
| Stoformer◊◊ | 32.80 | 0.959 | 30.73 | 0.937 |
| Stoformer◊† | 31.62 | 0.950 | 29.94 | 0.928 |
| Stoformer†◊ | 33.17 | 0.963 | 30.91 | 0.940 |
| Stoformer†† | 33.24 | 0.964 | 30.99 | 0.941 |

SPANet [48], JORDER-E [54], RCDNet [47], SPAIR [37], Uformer [51]. All these methods are evaluated on SPA-Data [48] and the performance comparison is shown in Tab. 1. Due to the limited space, visualization of image deraining is included in the supplement material.

### 4.4 Experiments on Image Denoising

We conduct denoising experiments on the additive white Gaussian Noise benchmark datasets, which include Set12 [62], BSD68 [34], Urban100 [17], Kodak24 [12] and McMaster [67]. Following previous works [62, 56, 63], a single model is trained to tackle with various noise levels. Tabs. 3 and 4 report PSNR scores of existing methods as well as Stoformer variants for color and grayscale image denoising, respectively. The supplemental material contains visualization of image denoising.

Table 3: Gaussian color image denoising. A single model is learned for various noise levels.

| Method | CBSD68 | | | Kodak24 | | | McMaster | | | Urban100 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\sigma = 15$ | $\sigma = 25$ | $\sigma = 50$ | $\sigma = 15$ | $\sigma = 25$ | $\sigma = 50$ | $\sigma = 15$ | $\sigma = 25$ | $\sigma = 50$ | $\sigma = 15$ | $\sigma = 25$ | $\sigma = 50$ |
| IRCNN [63] | 33.86 | 31.16 | 27.86 | 34.69 | 32.18 | 28.93 | 34.58 | 32.18 | 28.91 | 33.78 | 31.20 | 27.70 |
| FFDNet [64] | 33.87 | 31.21 | 27.96 | 34.63 | 32.13 | 28.98 | 34.66 | 32.35 | 29.18 | 33.83 | 31.40 | 28.05 |
| DnCNN [62] | 33.90 | 31.24 | 27.95 | 34.60 | 32.14 | 28.95 | 33.45 | 31.52 | 28.62 | 32.98 | 30.81 | 27.59 |
| VDN [56] | 33.90 | 31.35 | 28.19 | - | - | - | - | - | - | - | - | - |
| FuncNet [33] | 34.28 | - | - | 35.25 | - | - | - | - | - | - | - | - |
| DRUNet [66] | 34.30 | 31.69 | 28.51 | 35.31 | 32.89 | 29.86 | 35.40 | 33.14 | 30.08 | 34.81 | 32.60 | 29.61 |
| Restormer [58] | 34.39 | 31.78 | 28.59 | 35.44 | 33.02 | 30.00 | 35.55 | 33.31 | 30.29 | 35.06 | 32.91 | 30.02 |
| Stoformer◊◊ | 34.34 | 31.73 | 28.52 | 35.32 | 32.91 | 29.83 | 35.53 | 33.35 | 30.34 | 35.04 | 32.83 | 29.66 |
| Stoformer◊† | 34.30 | 31.73 | 28.50 | 35.22 | 32.90 | 29.80 | 35.40 | 33.22 | 30.18 | 35.00 | 32.78 | 29.61 |
| Stoformer†◊ | 35.10 | 32.40 | 29.13 | 35.50 | 33.08 | 30.00 | 36.00 | 33.83 | 30.80 | 35.37 | 33.14 | 30.00 |
| Stoformer†† | 35.13 | 32.47 | 29.16 | 35.53 | 33.12 | 30.03 | 36.03 | 33.86 | 30.84 | 35.42 | 33.19 | 30.06 |

Table 4: Gaussian grayscale image denoising. A single model is learned for various noise levels.

| Method | BSD68 | | | Urban100 | | | Set12 | | |
|---|---|---|---|---|---|---|---|---|---|
| | $\sigma = 15$ | $\sigma = 25$ | $\sigma = 50$ | $\sigma = 15$ | $\sigma = 25$ | $\sigma = 50$ | $\sigma = 15$ | $\sigma = 25$ | $\sigma = 50$ |
| DnCNN [62] | 31.62 | 29.16 | 26.23 | 32.28 | 29.80 | 26.35 | 32.67 | 30.35 | 27.18 |
| FFDNet [64] | 31.63 | 29.19 | 26.29 | 32.40 | 29.90 | 26.50 | 32.75 | 30.43 | 27.32 |
| IRCNN [63] | 31.63 | 29.15 | 26.19 | 32.46 | 29.80 | 26.22 | 32.76 | 30.37 | 27.12 |
| DRUNet [66] | 31.91 | 29.48 | 26.59 | 33.44 | 31.11 | 27.96 | 33.25 | 30.94 | 27.90 |
| Restormer [58] | 31.95 | 29.51 | 26.62 | 33.67 | 31.39 | 28.33 | 33.35 | 31.04 | 28.01 |
| Stoformer$\Diamond\Diamond$ | 31.94 | 29.51 | 26.62 | 33.60 | 31.38 | 27.90 | 33.26 | 31.02 | 27.96 |
| Stoformer$\Diamond\dagger$ | 31.89 | 29.46 | 26.59 | 33.48 | 31.09 | 27.85 | 33.18 | 30.96 | 27.92 |
| Stoformer$\dagger\Diamond$ | 32.57 | 30.06 | 27.03 | 34.19 | 31.84 | 28.58 | 33.83 | 31.51 | 28.40 |
| Stoformer$\dagger\dagger$ | 32.57 | 30.06 | 27.07 | 34.24 | 31.92 | 28.72 | 33.85 | 31.53 | 28.46 |

### 4.5 Experiments on Image Deblurring

We also perform deblurring experiments on the benchmark datasets (GoPro [36] and HIDE [42]). The model is trained only on the GoPro dataset and directly evaluated on the HIDE. Tab. 2 presents PSNR and SSIM scores of different deblurring methods and Stoformers. Fig. 5 presents an image deblurring example from GoPro [36]. More visual results are provided in the supplemental material.

**Remark.** From above experiments on various image restoration tasks, we can make the following observations and analyses:

- Stoformer$\Diamond\Diamond$ vs. Stoformer$\Diamond\dagger$: Without the stochastic window strategy for training, directly applying the layer expectation propagation will degrade performance dramatically, which reveals that the performance gain cannot simply attribute to feature ensemble at test time and the stochastic window for training matters.
- Stoformer$\Diamond\Diamond$ vs. Stoformer$\dagger\Diamond$: Even without the layer expectation propagation algorithm for testing, the stochastic window partition for training alone is also conductive to boost performance. This is reasonable since a large amount of local information lost by Stoformer$\Diamond\Diamond$ is re-exploited by Stoformer$\dagger\Diamond$.
- Stoformer$\dagger\dagger$ vs. Others: Equipped with the stochastic window for training and layer expectation propagation for testing (Stoformer$\dagger\dagger$), the model achieves the highest performance.

### 4.6 Analytic Experiments

**Eliminating Blocking Artifacts.** Since the fixed window strategy implies infinite preference towards specific window partition, the learned feature map will consequently contain annoying blocking artifacts. By contrast, the proposed stochastic window strategy treats all window partitions equally, facilitating to keep translation invariance and model locality. Therefore, the blocking artifacts should be significantly eliminated by the stochastic window strategy. Fig. 6 provides experimental evidence to support our inference. Specifically, the learned feature map by the fixed window strategy (Fig. 6(b)) contains obvious blocking artifacts while the stochastic window strategy (Fig. 6(c)) can remove these side effects as expected. It is surprising that with the stochastic window strategy, the feature is divided into non-overlapping local windows for further processing but in turn nearly no blocking artifacts are preserved. Please refer to the supplemental material for more visualization.

**Trade-off in Layer Expectation Propagation.** Although the layer expectation propagation accelerates inference significantly, the strict conduction still requires to take average of $s \times s$ features yielded by self-attention. Here, we investigate the influence of the number of averaged features to model performance. Fig. 7 shows the trend of model performance and *FLOPs of testing* [2] on a $256 \times 256$ image with respect to the number of averaged features each layer. Fig. 7 reveals that PSNR first rises rapidly and then tends to saturate with the increasing of the averaged feature number while FLOPs increases linearly with that number, which provides a promising trade-off between performance and

[2]Note that the transformer with the stochastic window strategy can be trained as efficiently as the fixed window strategy.

(a) Input (b) Fixed window strategy (c) Stochastic window strategy

Figure 6: The stochastic window strategy can eliminate the blocking artifacts.

FLOPs. For instance, we can decrease the averaged feature number from 64 to 8 to keep relatively high performance and efficient inference as well.

**More Powerful Representation.** Fig. 8 presents the loss curve on SPA-Data during training and the PSNR curve on the test set is plotted in Fig. 9. Compared with the fixed window strategy, our proposed stochastic window strategy attains not only higher PSNR on the test set but also lower training error, which demonstrates that the stochastic window strategy endows model more powerful representation. This can be anticipated since the stochastic window strategy can exploit complete local relationships, which are discarded by the fixed window strategy.

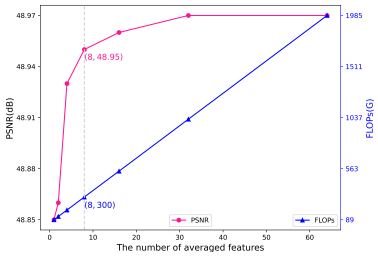



Figure 7: Trade-off between performance and FLOPs of testing.

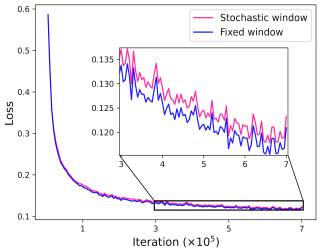



Figure 8: Training loss on the training set of SPA-Data.

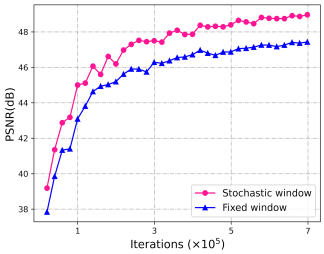



Figure 9: PSNR score on the test set of SPA-Data.

## 5 Conclusions

In this paper, we analyze the deficiencies of existing transformers for image restoration. We figure out that the fixed window strategy will inevitably lead to the broken translation invariance and loss of locality. To tackle with these issues, the novel stochastic window strategy is proposed. Specifically, we utilize the window partition with stochastic shift to replace the fixed window partition for training, which can ensure the translation invariance and intact locality. For testing, we propose the efficient layer expectation propagation to approximately take expectation of the introduced stochastic shift. Based on this strategy, we conduct extensive comparison experiments on various image restoration tasks to validate the effectiveness of the proposed stochastic window strategy.

## Broad Impact

Nowadays, image acquisition system inevitably suffers from various degradation, ranging from inherent noise of capturing instruments, the shaking during shooting, to unpredictable weather condition. Hence, image restoration itself has important value of research and application. Our proposed stochastic window strategy can make transformer more sophisticated in restoring degraded images. However, from a societal point of view, negative consequences may also come along. For example, the deviation from the actual image textures caused by image restoration technology may affect fair judgment in medical and criminal situations. In these cases, it is necessary to combine expert knowledge to make rational decisions.

# References


[1] P. Charbonnier, L. Blanc-Feraud, G. Aubert, and M. Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *Proceedings of 1st International Conference on Image Processing*, 1994.

[2] H. Chen, Y. Wang, T. Guo, C. Xu, Y. Deng, Z. Liu, S. Ma, C. Xu, C. Xu, and W. Gao. Pre-trained image processing transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[3] X. Chu, Z. Tian, Y. Wang, B. Zhang, H. Ren, X. Wei, H. Xia, and C. Shen. Twins: Revisiting the design of spatial attention in vision transformers. *Advances in Neural Information Processing Systems*, 2021.

[4] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, 2007.

[5] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, 2007.

[6] T. Dai, J. Cai, Y. Zhang, S.-T. Xia, and L. Zhang. Second-order attention network for single image super-resolution. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, 2019.

[7] Z. Dai, H. Liu, Q. V. Le, and M. Tan. Coatnet: Marrying convolution and attention for all data sizes. *Advances in Neural Information Processing Systems*, 2021.

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[9] J. Dong, S. Roth, and B. Schiele. Deep wiener deconvolution: Wiener meets deep learning for image deblurring. *Advances in Neural Information Processing Systems*, 2020.

[10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[11] Y. Fan, J. Yu, Y. Mei, Y. Zhang, Y. Fu, D. Liu, and T. S. Huang. Neural sparse representation for image restoration. *Advances in Neural Information Processing Systems*, 2020.

[12] R. Franzen. Kodak lossless true color image suite, 1999. URL `http://r0k.us/graphics/kodak/`.

[13] X. Fu, J. Huang, X. Ding, Y. Liao, and J. Paisley. Clearing the skies: A deep network architecture for single-image rain removal. *IEEE Transactions on Image Processing*, 26(6):2944–2956, 2017.

[14] M. Haris, G. Shakhnarovich, and N. Ukita. Deep back-projection networks for super-resolution. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018.

[15] K. He, J. Sun, and X. Tang. Single image haze removal using dark channel prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(12):2341–2353, 2011.

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2016.

[17] J.-B. Huang, A. Singh, and N. Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2015.

[18] Y. Huang, S. Li, L. Wang, T. Tan, et al. Unfolding the alternating optimization for blind super resolution. *Advances in Neural Information Processing Systems*, 2020.

[19] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2012.

[21] O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018.

[22] O. Kupyn, T. Martyniuk, J. Wu, and Z. Wang. Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[23] S. Laine, T. Karras, J. Lehtinen, and T. Aila. High-quality self-supervised deep image denoising. *Advances in Neural Information Processing Systems*, 2019.

[24] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4): 541–551, 1989.

[25] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[26] C. Li, C. Guo, L.-H. Han, J. Jiang, M.-M. Cheng, J. Gu, and C. C. Loy. Low-light image and video enhancement using deep learning: a survey. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 2021.

[27] C. Li, C. Guo, and C. C. Loy. Learning to enhance low-light image via zero-reference deep curve estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[28] X. Li, J. Wu, Z. Lin, H. Liu, and H. Zha. Recurrent squeeze-and-excitation context aggregation net for single image deraining. In *Proceedings of the European Conference on Computer Vision*, 2018.

[29] Y. Li, R. T. Tan, X. Guo, J. Lu, and M. S. Brown. Rain streak removal using layer priors. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2016.

[30] J. Liang, J. Cao, G. Sun, K. Zhang, L. Van Gool, and R. Timofte. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[31] D. Liu, B. Wen, Y. Fan, C. C. Loy, and T. S. Huang. Non-local recurrent network for image restoration. *Advances in Neural Information Processing Systems*, 2018.

[32] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[33] F. Luo, X. Wu, and Y. Guo. Functional neural networks for parametric image restoration problems. *Advances in Neural Information Processing Systems*, 2021.

[34] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision*, 2001.

[35] C. Mou, J. Zhang, and Z. Wu. Dynamic attentive graph learning for image restoration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[36] S. Nah, T. Hyun Kim, and K. Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017.

[37] K. Purohit, M. Suin, A. Rajagopalan, and V. N. Boddeti. Spatially-adaptive image restoration using distortion-guided networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[38] D. Ren, W. Zuo, Q. Hu, P. Zhu, and D. Meng. Progressive image deraining networks: A better and simpler baseline. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[39] W. Ren, J. Zhang, L. Ma, J. Pan, X. Cao, W. Zuo, W. Liu, and M.-H. Yang. Deep non-blind deconvolution via generalized low-rank approximation. *Advances in Neural Information Processing Systems*, 2018.

[40] W. Ren, J. Zhang, J. Pan, S. Liu, J. Ren, J. Du, X. Cao, and M.-H. Yang. Deblurring dynamic scenes via spatially varying recurrent neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[41] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, 2015.

[42] Z. Shen, W. Wang, X. Lu, J. Shen, H. Ling, T. Xu, and L. Shao. Human-aware motion deblurring. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[43] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15 (56):1929–1958, 2014.

[44] X. Tao, H. Gao, X. Shen, J. Wang, and J. Jia. Scale-recurrent network for deep image deblurring. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018.

[45] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, 2021.

[46] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[47] H. Wang, Q. Xie, Q. Zhao, and D. Meng. A model-driven deep neural network for single image rain removal. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

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

[49] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[50] Y. Wang, J. Yang, W. Yin, and Y. Zhang. A new alternating minimization algorithm for total variation image reconstruction. *SIAM Journal on Imaging Sciences*, 1(3):248–272, 2008.

[51] Z. Wang, X. Cun, J. Bao, and J. Liu. Uformer: A general u-shaped transformer for image restoration. *arXiv preprint arXiv:2106.03106*, 2021.

[52] L. Xu, J. S. Ren, C. Liu, and J. Jia. Deep convolutional neural network for image deconvolution. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, 2014.

[53] J. Yang, C. Li, P. Zhang, X. Dai, B. Xiao, L. Yuan, and J. Gao. Focal attention for long-range interactions in vision transformers. *Advances in Neural Information Processing Systems*, 2021.

[54] W. Yang, R. T. Tan, J. Feng, Z. Guo, S. Yan, and J. Liu. Joint rain detection and removal from a single image with contextualized deep networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(6):1377–1393, 2020.

[55] Q. Yu, Y. Xia, Y. Bai, Y. Lu, A. L. Yuille, and W. Shen. Glance-and-gaze vision transformer. *Advances in Neural Information Processing Systems*, 2021.

[56] Z. Yue, H. Yong, Q. Zhao, D. Meng, and L. Zhang. Variational denoising network: Toward blind noise modeling and removal. *Advances in neural information processing systems*, 2019.

[57] S. W. Zamir, A. Arora, S. Khan, M. Hayat, F. S. Khan, M.-H. Yang, and L. Shao. Learning enriched features for real image restoration and enhancement. In *European Conference on Computer Vision*, 2020.

[58] S. W. Zamir, A. Arora, S. Khan, M. Hayat, F. S. Khan, and M.-H. Yang. Restormer: Efficient transformer for high-resolution image restoration. *arXiv preprint arXiv:2111.09881*, 2021.

[59] S. W. Zamir, A. Arora, S. Khan, M. Hayat, F. S. Khan, M.-H. Yang, and L. Shao. Multi-stage progressive image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[60] H. Zhang, V. Sindagi, and V. 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.

[61] J. Zhang, J. Pan, J. Ren, Y. Song, L. Bao, R. W. Lau, and M.-H. Yang. Dynamic scene deblurring using spatially variant recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[62] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7): 3142–3155, 2017.

[63] K. Zhang, W. Zuo, S. Gu, and L. Zhang. Learning deep cnn denoiser prior for image restoration. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[64] K. Zhang, W. Zuo, and L. Zhang. Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, 2018.

[65] K. Zhang, W. Luo, Y. Zhong, L. Ma, B. Stenger, W. Liu, and H. Li. Deblurring by realistic blurring. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[66] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte. Plug-and-play image restoration with deep denoiser prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[67] L. Zhang, X. Wu, A. Buades, and X. Li. Color demosaicking by local directional interpolation and nonlocal adaptive thresholding. *Journal of Electronic imaging*, 20(2):023016, 2011.

[68] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European Conference on Computer Vision*, 2018.

[69] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu. Residual dense network for image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(7):2480–2495, 2020.

[70] Y. Zhang, K. Li, K. Li, G. Sun, Y. Kong, and Y. Fu. Accurate and fast image denoising via attention guided scaling. *IEEE Transactions on Image Processing*, 30:6255–6265, 2021.

[71] Y. Zhang, H. Wang, C. Qin, and Y. Fu. Aligned structured sparsity learning for efficient image super-resolution. *Advances in Neural Information Processing Systems*, 2021.

# Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See supplemental material.

(c) Did you discuss any potential negative societal impacts of your work? [Yes] See Sec. 5.

(d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...

(a) Did you state the full set of assumptions of all theoretical results? [Yes]

(b) Did you include complete proofs of all theoretical results? [Yes]

3. If you ran experiments...

(a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Sec. 3.3 and supplemental material.

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] The random seed is fixed for reproducibility.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See supplemental material.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes]

(b) Did you mention the license of the assets? [Yes]

(c) Did you include any new assets either in the supplemental material or as a URL? [No]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] The data used for experiments are all publicly available.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] The data we used contain no personally identifiable information or offensive content.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]