Masked Pre-training Enables Universal Zero-shot Denoiser

Xiaoxiao Ma[1*] **Zhixiang Wei**[1*] **Yi Jin**[1*] **Pengyang Ling**[1,2] **Tianle Liu**[1]
Ben Wang[1] **Junkang Dai**[1] **Huaian Chen**[1†]
[1] University of Science and Technology of China   [2] Shanghai AI Laboratory
{xiao_xiao,zhixiangwei,lpyang27,tleliu,wblzgrsn,junkangdai,anchen}@mail.ustc.edu.cn
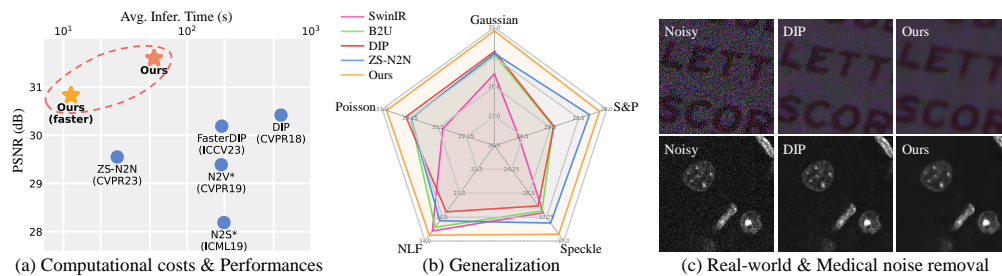{jinyi08}@ustc.edu.cn

Figure 1: (a) Ours surpasses current zero-shot methods with reduced inference time (on CSet with Gaussian σ=25, see Sec. 3.2). (b) It shows better generalization across different noise types than current zero-shot & supervised/unsupervised methods (Sec. 3.3). (c) And can remove spatial correlated real-world noise, results are from SIDD benchmark [1] and FMD [2] (Sec. 3.4, Sec. 3.5).

Abstract

In this work, we observe that model trained on vast general images via masking strategy, has been naturally embedded with their distribution knowledge, thus spontaneously attains the underlying potential for strong image denoising. Based on this observation, we propose a novel zero-shot denoising paradigm, i.e., **M**asked **P**re-train then **I**terative fill (**MPI**). MPI first trains model via masking and then employs pre-trained weight for high-quality zero-shot image denoising on a single noisy image. Concretely, MPI comprises two key procedures: **1) Masked Pre-training** involves training model to reconstruct massive natural images with random masking for generalizable representations, gathering the potential for valid zero-shot denoising on images with varying noise degradation and even in distinct image types. **2) Iterative filling** exploits pre-trained knowledge for effective zero-shot denoising. It iteratively optimizes the image by leveraging pre-trained weights, focusing on alternate reconstruction of different image parts, and gradually assembles fully denoised image within limited number of iterations. Comprehensive experiments across various noisy scenarios underscore the notable advances of MPI over previous approaches with a marked reduction in inference time. Code available at `https://github.com/krennic999/MPI`.

---

[*]Equal contribution.

[†]Corresponding author.

1 Introduction

Image denoising [3, 4], as a branch of image restoration, has been the subject of extensive exploration. The prevalent approach to restore noise-degraded images is learning from multiple noisy instances. Nonetheless, both supervised learning from noisy-clean pairs [5–7] and unsupervised training [8–10] necessitate the collection of additional noisy datasets. Moreover, such methods may foster dependencies on specific patterns or intensities of training noise, hindering their performance in unfamiliar noise situations [11, 12].

As an alternative, zero-shot approaches [13–15] attempt to train network on a single noisy image for denoised output, negating the need for additional noisy data collection. Dedicated to obviating concerns about generalization issues, these techniques include blind-spot networks that reconstruct from corrupted inputs [16, 17], DIPs [13, 18–21] which exploit the characteristics of deep networks to learn the mapping from random noise to noisy images, as well as sub-sample based strategies [22, 23] utilize spatial correlations to generate training pairs from sub-sampled instances.

However, current zero-shot methods train new networks from scratch for each noisy image, which presents two major issues: 1) Despite success in current zero-shot approaches rely on regularization or designed priors such as noise perturbations [13], under-parameterized networks [18, 22], dropout-ensemble [14] and blind-spot networks [16], the limited information from a single image to train network often lead to overly blurred content, noise artifacts or sub-optimal quality. Several methods tend to rely on known noise distribution [3, 20, 24] for more information, but their applicability is limited. 2) Training a new network from scratch for each noisy image is time-consuming. Existing zero-shot methods typically require several minutes [13] or more [14]. And attempts at faster zero-shot denoising [22, 23] often compromise on performance.

Compared to previous zero-shot approaches, learning the feature distribution from vast natural images offers a more intuitive approach. This is grounded in two considerations: Real natural images are both abundant and readily available, and despite variations in noise patterns, many natural images share common characteristics [25]. We seek to enhance zero-shot denoising with minimal reliance on pre-defined priors or regularization, aiming for a better startpoint for various noise patterns instead of from scratch. To this end, we delve into the potential of masked image modelling [26, 27] on natural images with no assumptions about noisy patterns and intensities [28]. Specifically, we make the following observation: combined with a simple ensemble operation, **a masked pre-trained model can naturally denoise images with unseen noise degradation**.

Building upon above observation, we introduce a zero-shot denoising paradigm, *i.e.*, Masked Pre-train then Iterative fill (MPI). MPI first pre-trains a model on ImageNet with pixel-wise masking strategy, then the pre-trained model is optimized on a single image with unseen noise for denoised prediction in zero-shot inference stage. The optimization goal in inference is designed to predict masked regions, and only predictions of masked areas are preserved for denoised prediction, thereby minimizing the gap between pre-training and inference. The pre-trained weights provide more generic knowledge, preventing premature over-fitting during inference and reducing the need for strong regularization. We are able to handle a wider range of noise scenarios with less information about noise patterns or intensities. Remarkably, we find that extracted representation can even generalize to medical images that distinctly different from natural ones [2]. It also offers a better startpoint than scratch training, enabling high-quality denoising around 10 seconds, underscoring the potential of our method in practical application. The main contributions of this paper are as follows:

- We introduce a novel zero-shot denoising paradigm, *i.e.*, Masked Pre-train then Iterative fill (MPI), which introduces masked pre-training in this context for the first time, simultaneously improving both image quality and inference speed on unseen noisy images.

- We develop a pre-training scheme with pixel-wise random masks to capture distribution knowledge of natural images. Based on pre-trained knowledge, we propose iterative filling for zero-shot inference on a specific noisy image. This process is optimized using pre-trained weights, and focuses on alternatively reconstruct different parts of noisy image, predictions in iterations are sequentially assembled for high-quality denoised output with efficiency.

- Extensive experiments demonstrate MPI's superiority, efficiency and robustness in diverse noisy scenarios. In nutshell, MPI achieves significant performance gains across various noise types with reduced inference time, highlighting its potential for practical applications.

2 Methods

In Sec. 2.1, we first investigate the properties of models trained with masking, proving that models trained with masking strategy can learn representations beneficial for denoising. Our observation lead us to propose a zero-shot denoising paradigm that includes pre-training (Sec. 2.2) and iterative optimizing (Sec. 2.3). We further illustrate how to remove spatially correlated real noise in Sec. 2.4.
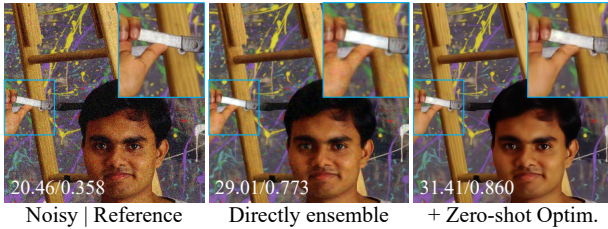

Figure 2: Example of model trained on ImageNet with 70% pixel-wise masking, denoised image is obtained by directly ensemble of predictions from fixed pre-trained weights ("Directly ensemble"), its performance can be further improved with iterative filling ("+Zero-shot Optim.").
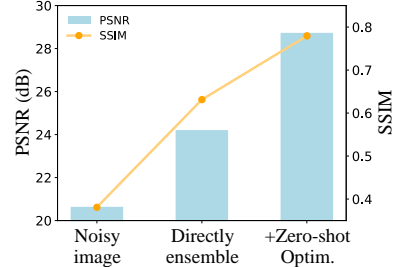

Figure 3: Evaluation on an ImageNet subset shows pre-trained model's inherent denoising ability, but performance limited without optimization.

2.1 Motivation

Masked Image Modeling [26, 27, 29] has significantly advanced computer vision by training on vast natural image sets to grasp their knowledge distributions. It shows great potential applicability under diverse scenarios and have been proven beneficial for high-level downstream tasks [30, 31].

To further explore its capability in denoising, we train a model on natural images with pixel-wise random masks (for details, see Sec. 2.2) and assess its performance against a target image with unseen noise distribution. Surprisingly, we observe that a simple average of predictions from a fixed-state trained model can denoise on unseen noise, as shown in Fig. 3, sometimes achieve remarkably good performance, as an example is presented in Fig. 2. This observation suggests that **a masked pre-trained model can serve as a natural image denoiser.** However, artifacts exist in the results, which can be attributed to lack of knowledge about specific degradation patterns in the target image.

Drawing on prior insight, we develop an efficient zero-shot denoising pipeline, leveraging pre-trained knowledge by incorporating noise characteristics from a single noisy image (Fig. 4), *i.e.*, Masked Pre-train then Iterative fill. The model is firstly pre-trained with random masks M and corresponding element-wise negation $\hat{M}$ to acquire natural image distributions, formulated as:

$$\arg\max_{\theta} p(I \odot \hat{M} | I \odot M; \theta), \tag{1}$$

for I indicates natural image without any degradation priors, typically sourced from extensive datasets (*e.g.* ImageNet [28]). We use element-wise multiplication ($\odot$). For denoising on specific noisy image x, pre-trained parameter θ is loaded and further optimized with known x from t=1 to t=T for T iterations, and predictions are aggregated for final prediction $\overline{y}$:

$$\overline{y} = Ensemble\{\mathcal{D}_{\theta_t}(x)\}_{t=1}^{T}, \tag{2}$$

where $\mathcal{D}_{\theta_t}(\cdot)$ is network parameterized by θ_t, optimized from pre-trained θ. Masked Pre-training process is detailed in Sec. 2.2 and Ensemble in Sec. 2.3.

2.2 Masked Pre-training

Masking strategy. Given the distinct requirements between low-level and high-level tasks in "semantics" [32], we implement specialized masking strategy to achieve finer-grained image representations, *i.e.*, a pixel-wise masking strategy. Specifically, given an input image $I \in \mathbb{R}^{H \times W \times C}$ divided into random patches of size 1, a subset of them are randomly replaced by mask token with probability p (for further discussion of p, see Sec. 4). When the mask token is set to 0, the masked image $M \odot I$ with random mask $M \in \mathbb{R}^{H \times W \times C}$ corresponds to a bernoulli sampling of the input image I. For
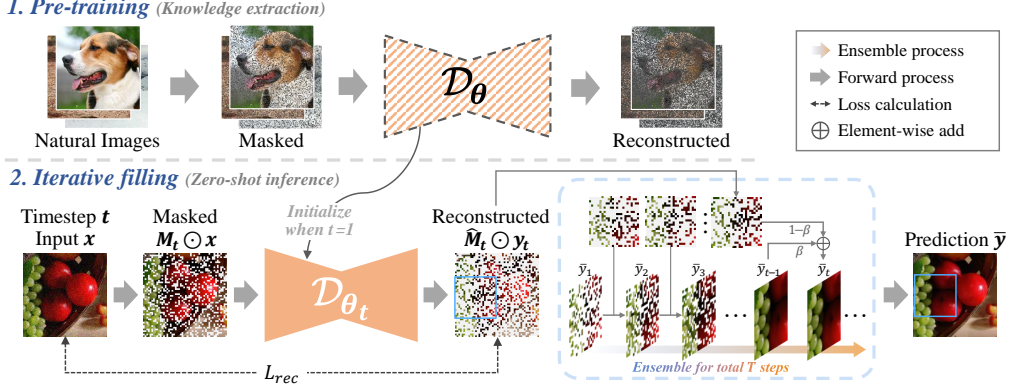
Figure 4: An overview of the proposed MPI paradigm consisting Masked Pre-training and Iterative filling. During pre-training $\mathcal{D}_\theta(\cdot)$ learns to reconstruct masked natural images. And the pre-trained weights θ are saved for zero-shot denoise, *i.e.*, Iterative filling, to denoise a specific noisy image x. During zero-shot inference, network is initialized with pre-trained weights θ, then the weights are further optimized on x for T steps, results from t-th ($t=1, 2, \ldots, T$-1) optimizing steps are gathered to obtain final denoised prediction $\overline{y}$. Compared to current zero-shot methods, just adding one more step to load a pre-trained model enables faster and high-quality zero-shot denoising.

each element $M_{[k]}$ in M, we have:

$$M_{[k]} = \begin{cases} 0, & with\ prob.\ p; \\ 1, & with\ prob.\ 1-p. \end{cases} \tag{3}$$

Pre-training scheme. During pre-training, the network $\mathcal{D}_\theta(\cdot)$ is trained to learn recovering natural image I itself with random mask M:

$$\tilde{I} = \mathcal{D}_\theta(M \odot I). \tag{4}$$

We set the same optimization strategy outlined in [27], focusing loss computation on masked prediction areas $\tilde{I}$. This directs network efforts towards reconstructing these specific regions, with the reconstruction loss denoted as L_{rec}:

$$L_{rec}(\tilde{I}, I) = \left\| \hat{M} \odot \tilde{I} - \hat{M} \odot I \right\|_2. \tag{5}$$

The Mean Squared Error (MSE) loss is adopted to learn relatively smoother representation. For the architecture of network $\mathcal{D}(\cdot)$, we employ the same U-shaped hourglass architecture as in DIP [13], which has been proven a powerful zero-shot denoising architecture [19]. Furthermore, its relatively small parameter configuration enables accelerated training, alleviating potential inference computational costs and rendering it more appropriate for zero-shot denoising tasks.

Algorithm 1: Iterative filling. Pipeline designed to leverage pre-trained representation θ for zero-shot denoising.

Input: Noisy image x, pre-trained parameter θ, network $\mathcal{D}(\cdot)$, exponential weight β, masking ratio p.

Output: denoised ensemble $\overline{y}$ from predictions of iteration $\{y_t\}$.

load pre-trained parameter θ for $\mathcal{D}(\cdot)$ as θ_1

initialize $\overline{y}$

for t *from* 1 *to* T **do**

    generate random mask M_t with mask ratio p

    $y_t = \mathcal{D}_{\theta_t}(M_t \odot x)$

    $\hat{M}_t = \neg M_t$

    $\theta_{t+1} = \theta_t - \nabla_\theta \left\| \hat{M}_t \odot y_t - \hat{M}_t \odot x \right\|_2$

    $\overline{y} \leftarrow \hat{M}_t \odot (\beta \cdot \overline{y} + (1-\beta) \cdot y_t) + M_t \odot \overline{y}$
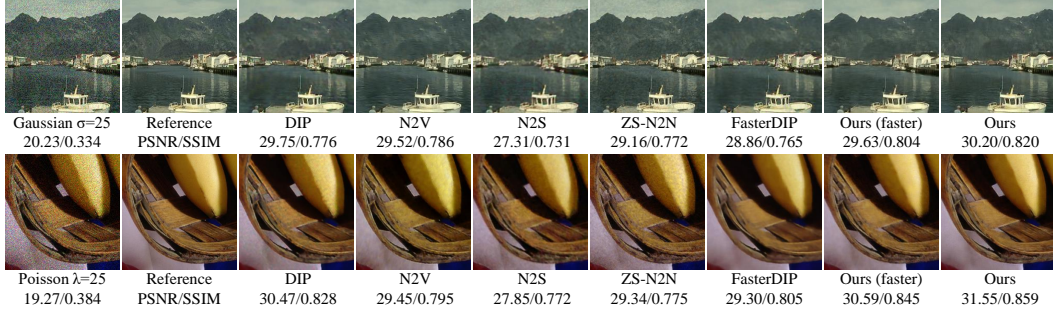
return $\overline{y}$

Figure 5: Qualitative denoising results on Gaussian and Poisson noise. The quantitative PSNR/SSIM results are provided underneath. Noisy patches are from CBSD-44 and McMaster-14, respectively. Best viewed in color (zoom-in for better comparison).

Table 1: Quantitative comparison on CSet, McMaster & CBSD for **Gaussian noise removal**. Best results **highlighted** and second underlined. See Supp. for poisson noise removal.

	σ	DIP [13]	N2V* [16]	N2S* [17]	ZS-N2N [22]	FasterDIP [19]	Ours (faster)	Ours
CSet [3]	10	32.05/0.829	31.55/0.885	28.04/0.819	33.87/0.883	31.59/0.815	33.82/0.889	**34.91/0.909**
	25	30.42/0.795	29.39/0.814	28.19/0.777	29.55/0.765	30.19/0.766	30.83/0.824	**31.61/0.841**
	50	24.73/0.533	27.35/0.694	26.62/0.699	26.10/0.624	26.09/0.669	28.14/**0.715**	**28.26**/0.710
McMaster [33]	10	32.48/0.878	30.98/0.877	28.61/0.839	34.19/0.908	31.48/0.842	34.35/0.921	**35.46/0.937**
	25	31.07/0.856	29.11/0.833	27.59/0.776	29.37/0.786	29.47/0.794	30.99/0.862	**31.90/0.879**
	50	25.72/0.639	24.65/0.676	24.89/0.673	25.82/0.634	24.75/0.663	28.15/**0.779**	**28.37**/0.770
CBSD [34]	10	31.18/0.865	31.18/0.918	28.17/0.853	33.73/0.923	30.89/0.857	34.20/0.935	**35.14/0.947**
	25	29.29/0.828	27.51/0.812	26.93/0.796	29.01/0.815	28.57/0.806	30.00/0.854	**30.58/0.865**
	50	23.06/0.540	25.74/0.700	24.78/0.695	25.37/0.657	24.75/0.669	**27.05/0.712**	26.85/0.703
Avg. Infer. time (s)		451.9	153.9	147.9	16.8	149.2	**10.1**	51.6

2.3 Iterative Filling

Overall design. As observed in Sec. 2.1, an iterative optimization process is designed to leverage pre-trained knowledge for zero-shot denoising. Unlike other MIM approaches [26, 27] that fine-tune with entire images as input, since only one noisy image is accessible, we employ a self-supervised manner to learn the mapping from a noisy image to itself. However, this direct self-mapping approach introduces significant gap between the zero-shot inference stage and pre-training stage and lacks constraints for learning a noise identity mapping.

Considering above challenges, we retain the same masking strategy in Sec. 2.2 for both input and loss computation, *i.e.*, network still learns to reconstruct masked regions, but from single noisy image rather than natural images. This leads to a pixel-based iterative refinement process, which resembles mechanism of blind-spot networks [16]. Specifically, for input noisy image x, random mask M_t and its element-wise negation $\hat{M}_t$ in t-th iteration, prediction y_t and aggregated result $\overline{y}$ can be derived:

$$y_t = \mathcal{D}_{\theta_t}(M_t \odot x); \tag{6a}$$

$$\overline{y} = \sum_t a_t \cdot y_t \odot \hat{M}_t, \tag{6b}$$

where θ_t denotes network parameter at iteration t, a_t is corresponding coefficient where $\sum_t a_t = 1$. The optimization objective at each iteration is as follows:

$$\underset{\theta_t}{\arg\min} \left\| \hat{M}_t \odot y_t - \hat{M}_t \odot x \right\|_2. \tag{7}$$

The optimization task, represented by $L_{rec}(y_t, x)$, learns to reconstruct noisy image cropped by random masks, aligns with pre-training. The alignment minimizes the gap between pre-training and zero-shot inference to avoid over-fitting, and reduces the inference steps required, thus accelerating the denoising process. Thanks to the well-crafted mechanism, we can accomplish high-quality results with preserved details in reduced time **without any other regularization**.

Pixel-based iterative refinement. For a lower mask ratio and reconstruction of more detailed images, we abandon constraints on unmasked regions in previous optimization goals (Eq. 5 and Eq. 7), thus

Figure 6: Qualitative results on unseen noise types. Restormer is trained with Gaussian σ=25. Noisy patches are from kodim07 and kodim12.

Table 2: Quantitative **generalization evaluation** results on Kodak. All supervised/unsupervised methods trained on σ=25 Gaussian, tested on 5 unseen noise types. (Average from all 6 settings.)

Test Noise	Supervised		Unsupervised		Zero-shot			
	SwinIR [7]	Restormer [38]	Nb2Nb [10]	B2U [39]	DIP [13]	ZS-N2N [22]	Ours (faster)	Ours
Gaussian σ=25	32.89/0.895	**33.04/0.897**	32.06/0.880	32.26/0.880	30.05/0.806	29.46/0.775	30.94/0.848	31.78/0.865
Gaussian $\sigma\in$[10,50]	27.29/0.628	30.00/0.729	28.68/0.713	29.24/0.726	29.56/0.783	29.36/0.753	30.89/0.837	**31.66/0.846**
Poisson $\lambda\in$[10,50]	25.06/0.622	26.52/0.683	27.31/0.703	28.22/0.718	28.67/0.758	28.17/0.732	29.94/0.826	**30.57/0.832**
NLF from [40]	32.52/0.862	31.71/0.857	31.88/0.859	31.98/0.859	29.71/0.821	31.02/0.834	32.26/0.886	**33.15/0.901**
Speckle $v\in$[10,50]	31.97/0.841	33.52/0.884	31.31/0.837	31.65/0.847	30.73/0.818	33.78/0.891	34.79/0.924	**35.79/0.933**
S&P $d\in$[0.02,0.05]	23.96/0.614	23.63/0.613	27.04/0.686	29.44/0.796	29.54/0.800	35.25/0.952	35.05/0.953	**36.87/0.964**
Average	28.94/0.744	29.73/0.777	29.71/0.800	30.47/0.804	29.71/0.798	31.17/0.823	32.31/0.879	**33.30/0.890**

making information under these areas unreliable, we preserve only the results corresponding to $\hat{M}$ for final denoised outcome $\overline{y}$. However, as one forward pass provide partial denoising results, an ensemble process is crucial. Specially, we employ an Exponential Moving Average (EMA) strategy to optimize the use predictions during iterations with little increase in inference time (Sec. 4):

$$\overline{y} = \hat{M}_t \odot (\beta \cdot \overline{y} + (1 - \beta) \cdot y_t) + M_t \odot \overline{y}. \tag{8}$$

For an in-depth look at proposed ensemble algorithm, see Alg. 1. During inference, the pre-trained weights not only provide a better startup but also act as regularization for the network, preventing from over-fitting too early and leading to better performance with less inference time (Sec. 4).

2.4 Adaptation to Real-world Noise Removal

Real-world noise exhibit strong spatial correlations, *i.e.* the noise is correlated across adjacent pixels. In such scenarios, employing a straightforward masking mechanism still allows the model to learn information related to noise patterns. To address this problem, we apply larger masking ratios than that used for synthetic noise. Additionally, we integrate a simple Pixel-shuffle Down-sampling (PD) mechanism during zero-shot inference to reduce spatial correlation in noise.

Specifically, instead of directly processing noisy image $x \in \mathbb{R}^{1 \times C \times H \times W}$ in Eq. 6a, we handle its down-sampled versions $Down(x) \in \mathbb{R}^{d^2 \times C \times \frac{H}{d} \times \frac{W}{d}}$ using simple Pixel-shuffle with factor d, and d^2 sub-samples are concatenated along batch dimension for joint denoising. Following the same iterative filling mechanism described above, we apply pixel unshuffle to the denoised result $\overline{y}$ to obtain final denoised outcome $Up(\overline{y})$. We add minimal PD operations to address spatial correlated noise, illustrating the effect of pre-trained weights, performance on real-world noisy dataset can be improved by applying better sub-sampling approaches [35–37] (Sec. 4.2) as they have been intensively studied.

3 Experiments

We assess our method against typical methods including DIP [13], Noise2Void (N2V) [16], Noise2Self (N2S) [17], Zero-Shot Noise2Noise (ZS-N2N) [22], and FasterDIP [19]. We modify N2V and N2S to single-image version (N2V* and N2S*). EMA ensemble result of DIP and FasterDIP are reported with their official code. Refer to supplementary material (Supp.) for EMA results of N2V* and N2S*, and comparison with more DIP-based [20], diffusion-based [41, 42], zero-shot modifications from unsupervised methods [36, 43, 44]. Only non-ensemble ZS-N2N is presented due to negligible performance differences with EMA version. We compare Peak Signal-to-Noise Ratio (PSNR) and
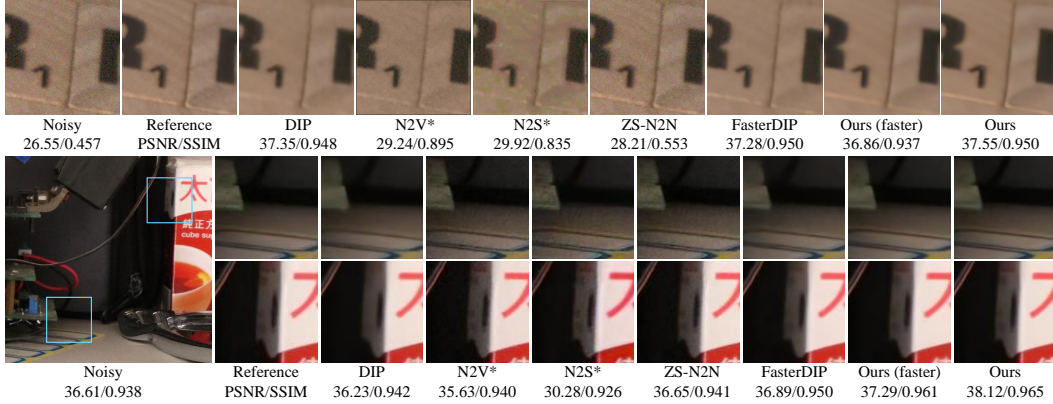
Figure 7: Qualitative results on real noise removal from SIDD and PolyU. Noisy patches are from SIDDval31_1 and Canon80D_8_8_3200_ball_16.

Table 3: Quantitative comparison on SIDD, PolyU and FMD for **real noise removal**.

| Methods | SIDD [1] | | PolyU [45] | FMD [2] | Avg. Infer. time (s) |
	validation	benchmark			
DIP [13]	33.68/0.802	33.67/0.863	37.91/0.952	32.85/0.840	333.2
N2V* [16]	26.74/0.627	25.34/0.595	35.04/0.921	29.79/0.817	98.1
N2S* [17]	26.78/0.573	26.93/0.658	32.82/0.930	31.61/0.759	114.4
ZS-N2N [22]	25.59/0.422	25.61/0.559	36.04/0.915	31.65/0.768	15.1
FasterDIP [19]	33.55/0.795	33.55/0.859	37.99/0.957	32.07/0.821	138.2
Ours (faster)	33.68/0.828	33.60/0.896	37.62/0.957	32.68/0.846	**7.9**
Ours	**34.43/0.844**	**34.32/0.903**	**38.11/0.962**	**32.97/0.847**	37.2

Structure Similarity Index Measure (SSIM) on synthetic (Sec. 3.2, Sec. 3.3) and real noise (Sec. 3.4). Additional tests on medical images (Sec. 3.5) show our method's adaptability beyond natural images.

3.1 Experimental Setup

Pre-training. Pre-training is performed on two Nvidia RTX 3090 GPUs using Adam optimizer with β_1=0.9 and β_2=0.9. Initial learning rate is $2e^{-3}$ and decays to $1e^{-5}$ with cosine annealing strategy over 80K iterations with a batch size of 64. We initiate pre-training on randomly cropped 256×256 patches from subset of ImageNet [28] with around 48,000 images. Two sets of pre-trained weights with masking probability p (p=0.3 for synthetic noise and a higher ratio of 0.8∼0.95 for spatially correlated noise) are trained. Further discussion of p is in Sec. 4.

Zero-shot inference. We set learning rate during inference to $2e^{-3}$, and same masking ratio p as pre-training (0.3 for synthetic, 0.8∼0.95 for real noise) is set. EMA weight β=0.99 for 1000 iterations (specially, 800 iterations for SIDD). Additionally, with β=0.9, we achieve performance surpassing most zero-shot methods within 200 iterations, denoted as **"faster"**. See Supp. for detailed setting.

3.2 Gaussian & Poisson Noise

We investigate Gaussian Noise with $\sigma\in$[10,25,50] and Poisson noise with $\lambda\in$[10, 25, 50] separately on three datasets: CSet [3], McMaster [33] and CBSD [34], with 9, 18 and 68 high-quality images, respectively. Results are shown in Table 1. The model is tested across various noise types with same experimental setups, without prior knowledge of noise distribution or intensity.

Analysis. DIP tends to produce over-blurry results and struggles especially with intense noise. While ZS-N2N manages to remove weak noise, its simple down-sample approach falters with stronger noise and cause artifacts. As Fig. 5 illustrates, under Gaussian noise σ=25 and Poisson noise λ=25, our method excels in both noise reduction and detail preservation. In some cases, we see an improvement of over 1dB, highlighting the effectiveness of our zero-shot paradigm.

Average inference time is listed in Table 1. Our "faster" version achieve the fastest inference speed while surpassing comparing methods in most cases. Even with $\beta = 0.99$, our method exhibits competitive inference time and significantly better performance. Params and FLOPs are in Supp.
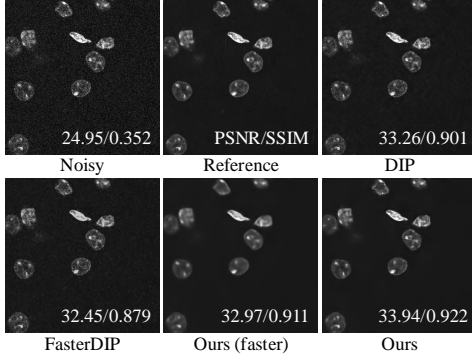
Figure 8: Validation of pre-trained representations on image content differs from natural images. Comparing between "Baseline" (w/o pre-train) and "Ours" (w pre-train). Noisy patch is from TwoPhoton_MICE_3. See quantitative comparison at Table 4.
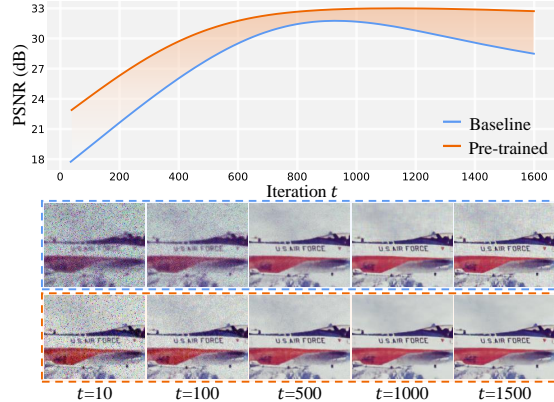

Figure 9: Effect of pre-trained model. Examples using Gauss σ=25 removal on F_16 with β=0.99. Pre-trained results are labeled in orange, while default initialized results are labeled in blue.

3.3 Generalization on Unseen Noise

We believe zero-shot denoising with natural image knowledge offers new perspectives on improving generalizability of denoising methods. We select several recent supervised (SwinIR [7], Restormer [38]) and unsupervised (Neighbor2Neighbor [10], Blind2Unblind [39]) methods trained on Gaussian with σ=25 for demostration. Testing them on 5 unknown noise types on Kodak [46].

Analysis. As illustrated in Table 2 and Fig. 6, although methods trained on multiple noisy images achieve better results on noisy cases with the same distribution, they exhibit poor generalization performance. In contrast, zero-shot methods often perform better generalization capabilities, especially our method which achieves the best performance across all types of generalization noise.

3.4 Real Noisy Datasets

We assess denoising capability of MPI on synthetic noise in previous experiments. However, real-world noise is more complicated and challenging. We test on SIDD [1] and PolyU [45] datasets, including 1280 patches from the SIDD validation and 1280 from SIDD benchmark, and all 100 official patches from PolyU to show our paradigm on real images. Due to the differences between synthetic and real noise, we report results from comparison methods from their optimal iteration.

Analysis. As shown in Table 3, our method excels over other zero-shot approaches on both datasets. This underlines our method's effectiveness on real-world noise removal. Fig. 7 show our method's capability to balance noise removal and detail retention. In essence, our method is adept at real-world denoising, offering a robust solution for image quality enhancement in challenging situations.

3.5 Generalization to Medical Images

The pre-trained model, which has learned the feature distributions of natural images, raises a question: Can this knowledge be applied to other image types? To answer this question, we select a fluorescence microscopy dataset (FMD) [2] characterized by colors and textures distinctly different from natural images, using all released 48 images in testset for evaluation. See Supp. for more image types.

Analysis. Our method still excels in denoising performance, as seen in Table 3. Despite such monochromatic microscopic images are not included in pre-training dataset and exhibits large differences between natural images, pre-trained knowledge still enhances zero-shot denoising performance, as evidenced in Fig. 8 and Table 4, demonstrating the generalizability of pre-trained weights.
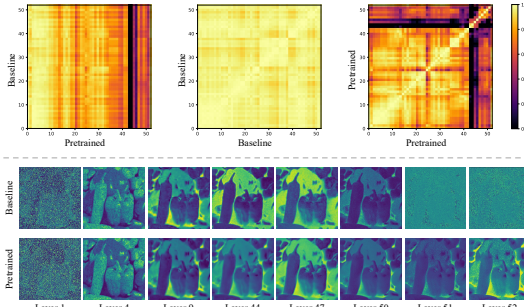
8

Figure 10: CKA (above) [47] and PCA (below) visualization of features extracted from the final timestep of model. Distribution of pre-trained model ("Pretrained") and from scratch ("Baseline") during inference is significantly different in last layers. Pre-trained model tends to restore the complete image, while the baseline model primarily focusing on restoring masked regions only.
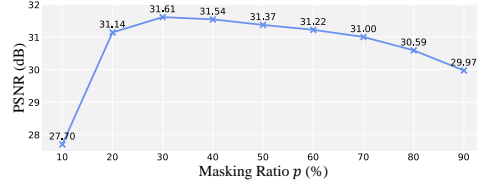

Figure 11: Effect of masking ratios. 30% balances noise removal and prevents over-smoothing for synthetic noise.

Table 4: Ablation of pre-training, with default settings noted in `gray`. "SIDD" denotes SIDD validation.

β	Pretrain	CSet [3]	SIDD [1]	FMD [2]
0.99	✓	31.61/0.841	34.43/0.844	32.97/0.847
	✗	30.90/0.811	32.31/0.746	31.44/0.786
0.90	✓	30.83/0.824	33.68/0.828	32.68/0.846
	✗	30.10/0.806	33.42/0.824	32.31/0.833

4 Ablation Study & Discussion

4.1 Ablation

Pre-trained weights. Building on Sec. 2.1, we question the role of pre-trained weights in zero-shot inference by comparing inference with pre-trained weights and optimizing from scratch, resembling a standard blind-spot network [16]. As depicted in Fig. 9 and Table 4, the latter quickly peaks and risks over-fitting due to the simple task of content recovery from masked images, making it challenging to specify an optimal iteration for all images. Conversely, the pre-trained model achieves better initial performance and maintains close-to-optimal performance for more extended period of time.

Unlike other zero-shot techniques train models from scratch for a single image to learn noise-resistant image content, we offer a new perspective by showing that a pre-trained model can aid in zero-shot tasks. The pre-trained weights, encapsulating views from multiple natural images, making it more robust to iteration count and provides better options for faster zero-shot denoising.

Moreover, We investigated the impact of pre-training on inference at the hidden layer level (see Fig. 10). Features extracted with pre-trained weights exhibit significant divergence from those produced by the baseline, *i.e.*, the usual zero-shot denoising approach. Specifically, pre-trained model restores the complete image, with more distinct features between layers, whereas the baseline model's features are less differentiated between layers, tending to only restore the masked parts, which may result in sub-optimal convergence towards local minima.

Masking ratios. Fig. 11 shows impact of different masking ratios on denoising with Gaussian σ=25. Lower masking ratios fails to completely remove noise, while higher masking ratios can cause overly smoothed results. A 30% masking ratio balances detail preservation and noise reduction for synthetic noise. However, a higher p of 0.8~0.95 is required for real-world noise. See Supp. for more details.

Ensemble strategy. We explore ensemble strategies, including EMA-based ("EMA"), straightforward averaging during iterations ("Average") and average after specific optimization step ("Avg after 500e", where 500 is optimal). Due to the inability to obtain predictions for all pixels in a single forward pass (Sec. 2.3), the "w/o Ensemble" result comes from the final prediction for each pixel. And "Last" provided for results of final forward prediction, a significant performance drop is caused by unreliable pixels (For details, see Supp.). Additionally, to validate our mask-based ensemble strategy, we remove masks from Eq. 7 and Eq. 8, with full-pixel loss and ensemble ("EMA w/o Mask"). See results in Table 5. Proposed "EMA" achieve significant better performance, aiding denoising with efficiency.

4.2 Discussion

Our method uses masking and minimal PD during inference to highlight pre-training's role without explicit regularization. We now explore further enhancements during inference with more strategies.

Table 5: Ablation of ensemble strategy. "Time (s)" denotes Infer. time (s)

Ensemble strategy	PSNR/SSIM	Time (s)
Avg after 500e	30.88/0.793	48.3
Average	31.28/0.835	49.7
EMA w/o mask	23.48/0.441	52.1
w/o Ensemble	30.23/0.797	51.7
Last	13.73/0.154	49.0
EMA	31.61/0.841	53.5

Table 6: Discussion on over-fitting.

Methods/ PSNR	Iteration			Avg. Infer. time (s)
	1,000	1,100	1,500	
Ours	31.61	31.58	31.35	62.6
Ours+ES [21]	31.66	31.65	31.66	51.7

Table 7: Discussion of downsampling on SIDD validation with 256×256.

	PSNR/SSIM	Time (s)
+PD	34.42/0.843	29.6
+RSG [37]	34.75/0.852	38.3

Over-fitting. Although pre-training mitigates over-fitting for synthetic and real noise, the over-parameterized network may still learn noise patterns over time due to lack of explicit mechanisms to avoid identity mapping. This is common challenge in many zero-shot models, we suggest early-stopping [21] (Table 6,"ES" for early-stopping) to avoid over-fitting and reduce inference time.

Additionally, we compare with other over-fitting prevention methods, *e.g.*, TV regularization of output and augmentation to input image. These approaches either resulted in suboptimal performance or longer inference times. More details on over-fitting and prevention strategies can be found in Supp.

Sub-sampling. Shown in Sec. 2.4, minimal pixel-shuffle is used to reduce spatial correlation in real-world noise, but may cause chessboard artifact and reduce performance due to its regular down-sampling strategy. Better down-sampling strategies have been widely studied, and here we choose RSG [37] to illustrate, see results at Table 7. For more comparison and visual comparison, see Supp.

5 Acknowledgements

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] Yide Zhang, Yinhao Zhu, Evan Nichols, Qingfei Wang, Siyuan Zhang, Cody Smith, and Scott Howard. A poisson-gaussian denoising dataset with real fluorescence microscopy images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11710–11718, 2019.

[3] 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.

[4] Matteo Maggioni, Vladimir Katkovnik, Karen Egiazarian, and Alessandro Foi. Nonlocal transform-domain filter for volumetric data denoising and reconstruction. *IEEE transactions on image processing*, 22(1):119–133, 2012.

[5] 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, jul 2017.

[6] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Ffdnet: Toward a fast and flexible solution for CNN based image denoising. *IEEE Transactions on Image Processing*, 2018.

[7] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1833–1844, 2021.

[8] Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. Noise2noise: Learning image restoration without clean data. In *International Conference on Machine Learning*, pages 2965–2974. PMLR, 2018.

[9] Wenchao Du, Hu Chen, and Hongyu Yang. Learning invariant representation for unsupervised image restoration. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 14483–14492, 2020.

[10] Tao Huang, Songjiang Li, Xu Jia, Huchuan Lu, and Jianzhuang Liu. Neighbor2neighbor: Self-supervised denoising from single noisy images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14781–14790, 2021.

[11] Hao Chen, Chenyuan Qu, Yu Zhang, Chen Chen, and Jianbo Jiao. Multi-view self-supervised disentanglement for general image denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12281–12291, 2023.

[12] Haoyu Chen, Jinjin Gu, Yihao Liu, Salma Abdel Magid, Chao Dong, Qiong Wang, Hanspeter Pfister, and Lei Zhu. Masked image training for generalizable deep image denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1692–1703, 2023.

[13] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[14] Yuhui Quan, Mingqin Chen, Tongyao Pang, and Hui Ji. Self2self with dropout: Learning self-supervised denoising from single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1890–1898, 2020.

[15] Jun Cheng, Tao Liu, and Shan Tan. Score priors guided deep variational inference for unsupervised real-world single image denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12937–12948, 2023.

[16] Alexander Krull, Tim-Oliver Buchholz, and Florian Jug. Noise2void-learning denoising from single noisy images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2129–2137, 2019.

[17] Joshua Batson and Loic Royer. Noise2self: Blind denoising by self-supervision. In *International Conference on Machine Learning*, pages 524–533. PMLR, 2019.

[18] Reinhard Heckel and Paul Hand. Deep decoder: Concise image representations from untrained non-convolutional networks. In *International Conference on Learning Representations*, 2018.

[19] Yilin Liu, Jiang Li, Yunkui Pang, Dong Nie, and Pew-Thian Yap. The devil is in the upsampling: Architectural decisions made simpler for denoising with deep image prior. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12408–12417, 2023.

[20] Yeonsik Jo, Se Young Chun, and Jonghyun Choi. Rethinking deep image prior for denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5087–5096, 2021.

[21] Zenglin Shi, Pascal Mettes, Subhransu Maji, and Cees GM Snoek. On measuring and controlling the spectral bias of the deep image prior. *International Journal of Computer Vision*, 130(4):885–908, 2022.

[22] Youssef Mansour and Reinhard Heckel. Zero-shot noise2noise: Efficient image denoising without any data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14018–14027, June 2023.

[23] Jason Lequyer, Reuben Philip, Amit Sharma, Wen-Hsin Hsu, and Laurence Pelletier. A fast blind zero-shot denoiser. *Nature Machine Intelligence*, 4(11):953–963, Nov 2022.

[24] Markku Makitalo and Alessandro Foi. Optimal inversion of the anscombe transformation in low-count poisson image denoising. *IEEE Transactions on Image Processing*, 20(1):99–109, 2011.

[25] Anish Mittal, Rajiv Soundararajan, and Alan C Bovik. Making a "completely blind" image quality analyzer. *IEEE Signal processing letters*, 20(3):209–212, 2012.

[26] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

[27] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9653–9663, 2022.

[28] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[29] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. In *International Conference on Learning Representations*, 2021.

[30] Yunyao Mao, Jiajun Deng, Wengang Zhou, Yao Fang, Wanli Ouyang, and Houqiang Li. Masked motion predictors are strong 3d action representation learners. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10181–10191, 2023.

[31] Jiang-Tian Zhai, Xialei Liu, Andrew D Bagdanov, Ke Li, and Ming-Ming Cheng. Masked autoencoders are efficient class incremental learners. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19104–19113, 2023.

[32] Yihao Liu, Anran Liu, Jinjin Gu, Zhipeng Zhang, Wenhao Wu, Yu Qiao, and Chao Dong. Discovering distinctive" semantics" in super-resolution networks. *arXiv preprint arXiv:2108.00406*, 2021.

[33] Lei Zhang, Xiaolin Wu, Antoni Buades, and Xin Li. Color demosaicking by local directional interpolation and nonlocal adaptive thresholding. *Journal of Electronic imaging*, 20(2):023016–023016, 2011.

[34] David Martin, Charless Fowlkes, Doron Tal, and Jitendra 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. ICCV 2001*, volume 2, pages 416–423. IEEE, 2001.

[35] Yuqian Zhou, Jianbo Jiao, Haibin Huang, Yang Wang, Jue Wang, Honghui Shi, and Thomas Huang. When awgn-based denoiser meets real noises. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13074–13081, 2020.

[36] Wooseok Lee, Sanghyun Son, and Kyoung Mu Lee. Ap-bsn: Self-supervised denoising for real-world images via asymmetric pd and blind-spot network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17725–17734, 2022.

[37] Yizhong Pan, Xiao Liu, Xiangyu Liao, Yuanzhouhan Cao, and Chao Ren. Random sub-samples generation for self-supervised real image denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12150–12159, 2023.

[38] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5728–5739, 2022.

[39] Zejin Wang, Jiazheng Liu, Guoqing Li, and Hua Han. Blind2unblind: Self-supervised image denoising with visible blind spots. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2027–2036, 2022.

[40] Tobias Plotz and Stefan Roth. Benchmarking denoising algorithms with real photographs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1586–1595, 2017.

[41] Yinhuai Wang, Jiwen Yu, and Jian Zhang. Zero-shot image restoration using denoising diffusion null-space model. *arXiv preprint arXiv:2212.00490*, 2022.

[42] Tomer Garber and Tom Tirer. Image restoration by denoising diffusion models with iteratively preconditioned guidance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 25245–25254, 2024.

[43] Dan Zhang, Fangfang Zhou, Yuwen Jiang, and Zhengming Fu. Mm-bsn: Self-supervised image denoising for real-world with multi-mask based on blind-spot network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4189–4198, 2023.

[44] Hyemi Jang, Junsung Park, Dahuin Jung, Jaihyun Lew, Ho Bae, and Sungroh Yoon. Puca: patch-unshuffle and channel attention for enhanced self-supervised image denoising. *Advances in Neural Information Processing Systems*, 36, 2024.

[45] Jun Xu, Hui Li, Zhetong Liang, David Zhang, and Lei Zhang. Real-world noisy image denoising: A new benchmark. *arXiv preprint arXiv:1804.02603*, 2018.

[46] Rich Franzen. Kodak lossless true color image suite. *source: http://r0k. us/graphics/kodak*, 4(2):9, 1999.

[47] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pages 3519–3529. PMLR, 2019.

[48] Jun Xu, Yuan Huang, Ming-Ming Cheng, Li Liu, Fan Zhu, Zhou Xu, and Ling Shao. Noisy-as-clean: Learning self-supervised denoising from corrupted image. *IEEE Transactions on Image Processing*, 29:9316–9329, 2020.

[49] Nick Moran, Dan Schmidt, Yu Zhong, and Patrick Coady. Noisier2noise: Learning to denoise from unpaired noisy data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12064–12072, 2020.

[50] Tongyao Pang, Huan Zheng, Yuhui Quan, and Hui Ji. Recorrupted-to-recorrupted: Unsupervised deep learning for image denoising. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2043–2052, 2021.

[51] Yi Zhang, Dasong Li, Ka Lung Law, Xiaogang Wang, Hongwei Qin, and Hongsheng Li. Idr: Self-supervised image denoising via iterative data refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2098–2107, 2022.

[52] Reyhaneh Neshatavar, Mohsen Yavartanoo, Sanghyun Son, and Kyoung Mu Lee. Cvf-sid: Cyclic multivariate function for self-supervised image denoising by disentangling noise from image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17583–17591, 2022.

[53] Xin Lin, Chao Ren, Xiao Liu, Jie Huang, and Yinjie Lei. Unsupervised image denoising in real-world scenarios via self-collaboration parallel generative adversarial branches. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12642–12652, 2023.

[54] Xiaohe Wu, Ming Liu, Yue Cao, Dongwei Ren, and Wangmeng Zuo. Unpaired learning of deep image denoising. In *European conference on computer vision*, pages 352–368. Springer, 2020.

[55] Samuli Laine, Tero Karras, Jaakko Lehtinen, and Timo Aila. High-quality self-supervised deep image denoising. *Advances in Neural Information Processing Systems*, 32, 2019.

[56] Zichun Wang, Ying Fu, Ji Liu, and Yulun Zhang. Lg-bpn: Local and global blind-patch network for self-supervised real-world denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18156–18165, 2023.

[57] Yaochen Xie, Zhengyang Wang, and Shuiwang Ji. Noise2same: Optimizing a self-supervised bound for image denoising. *Advances in neural information processing systems*, 33:20320–20330, 2020.

[58] Yeong Il Jang, Keuntek Lee, Gu Yong Park, Seyun Kim, and Nam Ik Cho. Self-supervised image denoising with downsampled invariance loss and conditional blind-spot network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12196–12205, October 2023.

[59] Junyi Li, Zhilu Zhang, Xiaoyu Liu, Chaoyu Feng, Xiaotao Wang, Lei Lei, and Wangmeng Zuo. Spatially adaptive self-supervised learning for real-world image denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9914–9924, 2023.

[60] Kwanyoung Kim and Jong Chul Ye. Noise2score: tweedie's approach to self-supervised image denoising without clean images. *Advances in Neural Information Processing Systems*, 34:864–874, 2021.

[61] Kwanyoung Kim, Taesung Kwon, and Jong Chul Ye. Noise distribution adaptive self-supervised image denoising using tweedie distribution and score matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2008–2016, 2022.

[62] Zongsheng Yue, Hongwei Yong, Qian Zhao, Deyu Meng, and Lei Zhang. Variational denoising network: Toward blind noise modeling and removal. *Advances in neural information processing systems*, 32, 2019.

[63] Mohammad Zalbagi Darestani and Reinhard Heckel. Accelerated mri with un-trained neural networks. *IEEE Transactions on Computational Imaging*, 7:724–733, 2021.

[64] Metin Ersin Arican, Ozgur Kara, Gustav Bredell, and Ender Konukoglu. Isnas-dip: Image-specific neural architecture search for deep image prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1960–1968, 2022.

[65] Chao Wang, Zhedong Zheng, Ruijie Quan, Yifan Sun, and Yi Yang. Context-aware pretraining for efficient blind image decomposition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18186–18195, 2023.

[66] Naishan Zheng, Man Zhou, Yanmeng Dong, Xiangyu Rui, Jie Huang, Chongyi Li, and Feng Zhao. Empowering low-light image enhancer through customized learnable priors. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12559–12569, 2023.

[67] Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *Proceedings of the European conference on computer vision (ECCV)*, pages 85–100, 2018.

[68] 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.

[69] Abdelrahman Abdelhamed, Marcus A Brubaker, and Michael S Brown. Noise flow: Noise modeling with conditional normalizing flows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3165–3173, 2019.

[70] Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T Barron. Unprocessing images for learned raw denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11036–11045, 2019.

[71] Chang Qiao, Di Li, Yuting Guo, Chong Liu, Tao Jiang, Qionghai Dai, and Dong Li. Evaluation and development of deep neural networks for image super-resolution in optical microscopy. *Nature Methods*, 18(2):194–202, 2021.

[72] Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun. Learning to see in the dark. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3291–3300, 2018.

A Introduction

This document provides supplementary materials for the main paper. Specifically, a brief conclusion of works related to ours is in Sec. B Sec. C presents more details and demonstrations about proposed iterative filling (Sec. C.1, Sec. C.2) and different strategies to adapt to real-world spatial-correlated noise (Sec. C.3). Sec. D presents more discussion about masking ratio (Sec. D.1), over-fitting (Sec. D.2) and different downsampling strategies (Sec. D.3). Additionally, we conduct further analysis of pre-training (Sec. E) and expanded our framework to other network structures (Sec. F). More experimental details and qualitative comparison results can be found in Sec. G and Sec. I.

B Related Works

B.1 Unsupervised Image Denoising

Unlike supervised approaches [5–7, 38], unsupervised denoising focuses on situations when paired data is unavailable. Methods in this category include:

Paired noisy-noisy images. To learn consistent representations from varied noise observations of the same scene, Lehtinen *et al.* [8] train on mapping from two noisy observations of the same scene. Additional approaches utilize synthetic noise to generate noisy pairs, as seen in [48–52], as well as [11] learns shared latent from multiple noise observations.

Unpaired noisy-clean images. Du *et al.* [9] propose to learn decoupled representations of contents and noise from images. Lin *et al.* [53] extend this by using separated noise representations to guide noise synthesis, thereby enhancing the denoising process. Additionally, Wu *et al.* [54] employ a distillation loss from both real and synthetic noisy images.

Noisy images only. Techniques like blind-spot [16, 17, 39, 55–57], substitution followed by image reconstruction [16, 17], multiple sub-sampled images from a single noisy scene [10] or above approaches combined [36, 37, 58] are developed when only one observation is available from a noise scene. Li *et al.* [59] integrate blind spot strategies and structural insights for adaptive denoising. Score matching and posterior inference are also utilized in [60, 61].

In the context of zero-shot denoising tasks, only a single noisy image is visible during training, presenting greater challenges than methods described above.

B.2 Zero-shot Image Denoising

Compared to unsupervised methods, zero-shot denoising is more challenging as it aims to train a network for denoising when a single noisy image available. Typical strategies involve utilizing spatial correlations [3, 4], variation based priors [15, 62] or low-frequency characteristics of images, corrupting and reconstructing part of images [14, 16, 17], or constructing paired training sets from sub-sampled noisy images [22, 23]. Among which Noise2Void [16] is initially designed for learning from multiple noisy images, shows promise in its zero-shot version. Noise2Self [17]reconstructs cyclically masked regions of input noisy image. Although there is a gap compared to supervised or unsupervised methods. While dropout-ensemble [14] is adapted on a noisy image for better performance, it leads to over-smoothing and incurs large computational costs. Noise2Fast [23] and Zero-Shot Noise2Noise [22] are fast but struggle to completely remove noise from images, especially spatially-correlated real noise, resulting in suboptimal visual results. DIP [13] and its variants [18, 63] exploits the features of deep networks to learn mappings from random noise to images. Early stopping [21] or other approaches [20, 64] are used to prevent over-fitting, FasterDIP [19] further discusses the influence of network structure on its performance. However, current zero-shot methods often takes a long time, and parameter settings are carefully selected for various image contents and noise degradation.

B.3 Masked Image Modeling

Masked Image Modeling (MIM) helps in learning pre-trained representations for downstream tasks by masking a portion of input images [26, 27, 29] and training models to predict the masked contents. Due to its impressive effects in high-level tasks [30, 31], Masked Image Modeling has also found

applications in low-level visual tasks. For instance, Wang *et al.* [65] applies random patch masks during the pre-training of image deraining and desnowing models in handling adverse weather conditions. Zheng *et al.* [66] integrates Masked Autoencoder (MAE) to learn illumination-related structural information in a supervised low-light enhancement framework. Notably, despite the successful applications of Masked Image Modeling in several low-level vision tasks, its application in the pre-training scheme for denoising models has not yet been explored.

C   More Details about MPI

C.1   More Details about Iterative Filling

In the main paper, we mention that Iterative Filling is optimization steps based on pre-trained weights for zero-shot denoising. And as depicted in Sec. 3.3 in main paper, to fully leverage the results of each optimization step and preserve more image detail, only the masked regions $\hat{M}_t \odot y_t$ are constrained by the loss at each optimization timestep and considered reliable, others are labeled as "unreliable pixels" and discarded (see Sec. C.2 to know why we need "unreliable pixels"). During the iterative optimization, we maintain an ensemble version $\overline{y}$, assembling reliable parts $\hat{M}_t \odot y_t$ of each prediction y_t via EMA while keeping the rest unchanged, as shown in Fig. 12.

With sufficient iterations, ensemble from each pixel is ensemble from hundreds of predictions, ensuring a high-quality ensemble outcome.
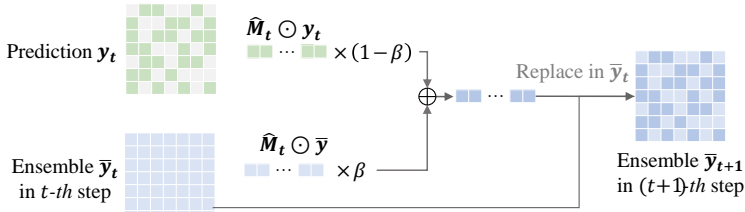

Figure 12: Details of EMA process in Iterative filling. Masked regions of predictions from each optimization step t is assembled to a ensemble $\overline{y}$.

C.2   Why Discard "Unreliable Pixels"

During the zero-shot denoising process, unmasked parts of each optimization result $M_t \odot y_t$ needed be discarded, referred to as "unreliable pixels", primarily due to the following reasons:

1) The pre-training task is set to reconstruct masked regions, that is only masked areas are constrained for reconstruction, while pixels in unmasked areas significantly differ from the actual pixels in the image. This discrepancy might result from the skip connections in the network architecture of DIP [13]. To maximize the utilize of pre-trained weight and avoid conflicting, pixels corresponding to these unmasked regions should not be considered during optimizing.

2) For spatially uncorrelated noise, we employ an extremely low mask ratio and distinct mask settings for each color channel to preserve as much image information as possible. In this scenario, it is impractical to expect the network to retain all pixel values in its output, as this can easily lead to identity mapping.

3) While partial convolution [67] or others designed for image inpainting can mitigate this problem, they often lead to sub-optimal performance with risks of over-fitting to noise, and require specialized network architectures, limiting the adaptability of proposed framework to other network structures.

For our zero-shot denoising framework, which obtains denoised images via iterative optimization, using a portion of noisy image as cues and training network to complete this "fill-in-the-blank" task proves most effective. Moreover, the optimization generates reconstructions with various masks in iterations, assembling these results to achieve final denoised prediction requires only maintaining an ensemble result additionally, incurring negligible time and space resources.
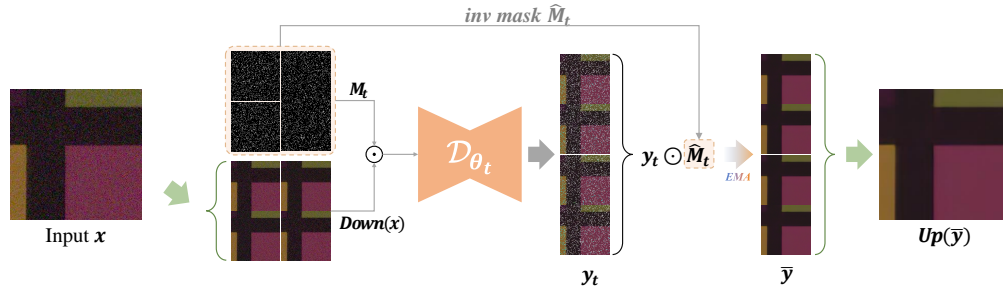
Figure 13: An overview of the zero-shot denoising stage with adaptation to real-world noise. We adapt downsample "$Down(\cdot)$" and upsample "$Up(\cdot)$" to achieve noisy subsamples with less spatial correlation in noise, labeled in green arrows, and larger masking ratio is used to further deal with remaining spatial correlations. Actually, not all real noisy images needed to be sub-sampled, we only adapt "$Down(\cdot)$" and "$Up(\cdot)$" to SIDD dataset.

C.3 Different Strategies in Dealing with Synthetic & Real Noise

We adapt downsample "$Down(\cdot)$" and upsample "$Up(\cdot)$" to achieve noisy subsamples with less spatial correlation in noise, and larger masking ratio 80%~95% (90% for SIDD and 85% for others) with a unified mask for all channels is used to further deal with remaining spatial correlations. Actually, not all real noisy images needed to be sub-sampled, we only adapt "$Down(\cdot)$" and "$Up(\cdot)$" to SIDD dataset. See Fig. 13 for a simple framework, which specifically designed for real-world noise is labeled in green arrows.

D Additional Discussion

D.1 Masking Ratio

In the main paper, we discussed the optimal masking ratio for removing spatially uncorrelated synthetic noise and notes that significantly larger mask ratios are used for real noise. For real spatially correlated noise, the situation becomes more complex, with no single optimal mask ratio. Based on experience, the most effective mask range is between 80%~95%, which is influenced by the noise's spatial correlation, noise intensity, and the information in image.

For the SIDD dataset, we investigated the impact of the masking ratio on SIDD validation, as shown in Fig. 14, finding that a 90% masking ratio is optimal. This is attributed to SIDD images containing limited content information and the noise exhibiting strong spatial correlation.

However, the optimal masking ratio for SIDD differs from that for synthetic noise, primarily due to the spatial correlation of the noise within the image. Synthetic noise is spatially uncorrelated, meaning noise signals at neighboring positions do not influence each other. In contrast, real noise, after undergoing a series of ISP processes, exhibits a more complex distribution, resembling blurred spots rather than independent points (see Fig. 15). For synthetic noise, selecting a small masking ratio allows for quicker recovery of image details. Conversely, for real noise, a small masking ratio
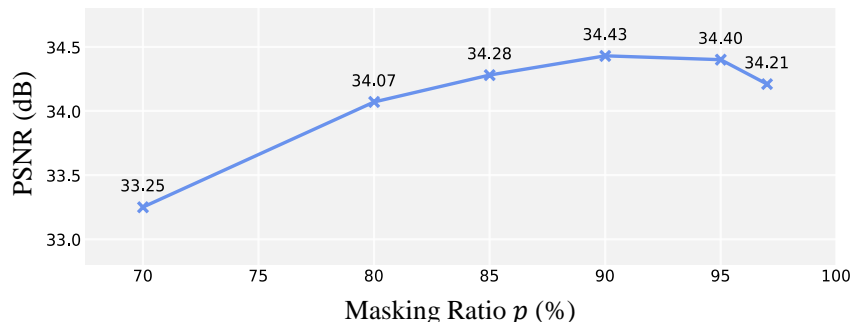

Figure 14: Effect of masking ratios on real-world noise (SIDD validation). 90% for removing strong spatial correlated noise.
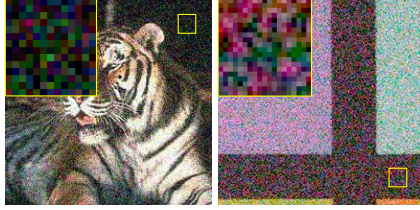
Figure 15: Illustration of spatial-correlated real-world noise (right) and synthetic noise (left).

Table 8: Extension of proposed pre-training strategy into other network architectures. A performance improvement can be observed for both settings of beta=0.9 and 0.99 in the experiment across various network architectures.

Methods/ PSNR	Iteration			Avg. Infer. time (s)
	800	900	1,200	
Ours	34.43	34.34	33.48	39.9
Ours+ES [21]	34.64	34.63	34.60	34.5

may lead to the model fitting the noise distribution by relying on neighboring pixel values. In such cases, a larger masking ratio helps mitigate the influence of noise.

D.2 Over-fitting & Regularization

In the main text, we highlighted that our proposed zero-shot denoising framework still faces over-fitting issues with increased iteration counts, with Table 8 and Fig. 16 showing this problem's impact across more datasets and more regularization. To address this issue, after validation and comparison, we recommend adopting a simple early-stopping strategy to prevent over-fitting—a straightforward, effective approach without additional computational costs. We also compared other strategies:

- Employing TV regularization helps against overfitting but still leads to performance drop and lower peak PSNR as iterations increase.
- Adding random transformations to the input image, including flips and random translations, lead to steady performance improvement and higher peak PSNR over more iterations, but increase inference time.
- Early stopping stops close to peak performance with minimal calculation, providing stable, high-quality results with no added time.
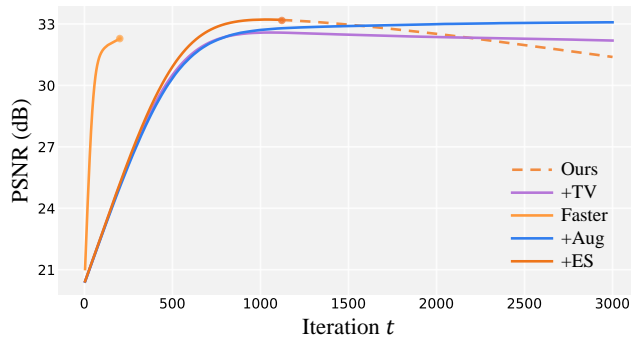

Figure 16: Influence of different regularization strategies during iterations, including Total Variation ("+TV"), random image augmentation ("+Aug"), and early stopping ("+ES"). "Ours" and "Faster" are the methods evaluated in mainpaper. Example is tested on F16_512rgb with Gaussian σ=25.

D.3 Down-sampling in Real-world Denoising

As illustrated in Fig. 13, specialized downsampling is employed to reduce the spatial correlation of real noise, with different downsampling strategies yielding varying outcomes. Simple pixelshuffle ("+PD") can easily lead to checkerboard artifacts, whereas more randomized sub-sampling strategies [37]

Table 9: Discussion of noise in pre-trained dataset. Additional assumptions of noise in pre-trained dataset result in lower performance ("+Gauss (σ=25)(N2N)").

Pre-train Mode	PSNR	SSIM
Ours	31.61	0.841
+Gauss (σ=25)(N2N)	31.24	0.827

("+RSG") can more effectively disrupt noise spatial correlations and, due to their randomness, avoid checkerboard artifacts, as depicted in Fig. 17. A potential issue with this approach is the over-smoothed denoising predictions. Therefore, downsampling is only applied in strong spatially-correlated real noise.
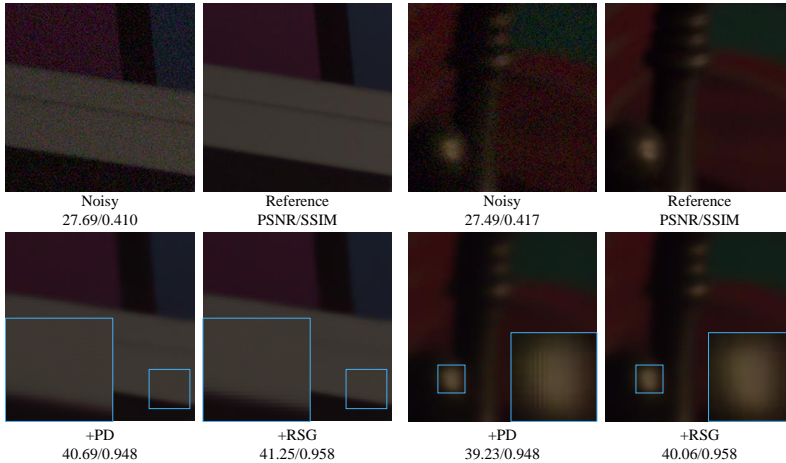

Figure 17: Validation of different down-sampling strategies in real-world denoising. Better down-sampling strategies can further enhance denoising performance of our pipeline. Noisy patches are from SIDDval_12_2 and SIDDval_20_3.

E   Additional Analysis on Pre-training

E.1   Noise Intensity

In the main text, we have already mentioned that pre-training aids in the removal of various types of noise. We validated the relationship between pre-trained weights and different input noise intensities on the CSet dataset, as shown in Fig. 18. Pre-training enhances denoising performance across different noise levels, particularly in the case of strong noise, where the knowledge provided by pre-training effectively avoids over-fitting to the noise.

E.2   Masking Ratio

We analyze the impact of pre-training on different paradigms under various masking ratios, as shown in Fig. 19. Our study reveals that pre-training plays a significant role in enhancing denoising performance across various masking ratios (especially in cases of $20 \leq p \leq 80$).

E.3   Discuss of Noise in Pre-training

In the main text, we use the well-known natural image dataset ImageNet without making any assumptions about the presence or type of noise in each image, hoping to learn the statistical distribution rules from a large number of natural images. Here, we add synthetic noise of specified distribution and intensity during pre-training, and perform pre-training from noise to itself (denoted as "+Gauss (σ=25) (N2N)"), and adopt the same iterative denoising strategy, proving that additional assumptions about specific noise type or noise level in pre-training leads to a decline in effectiveness, as shown in Table. 9. Networks that are too small fail to learn sufficient denoising information, falling short of the corresponding zero-shot approaches.
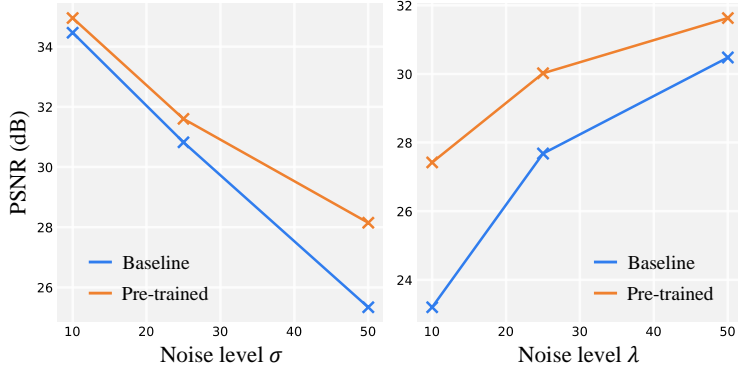
Figure 18: Effect of pre-training on different noise levels from Gaussian (left) and Poisson (right) on CSet. Pre-training is beneficial for all 6 noise levels, especially in cases of intense noise.
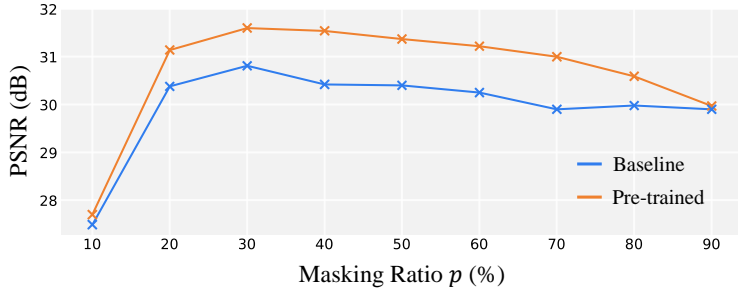

Figure 19: Effect of pre-training on different masking ratios p from Gaussian noise on CSet. Pre-training is beneficial for all masking ratios, especially in cases of $20 \leq p \leq 80$.

Table 10: Extension of proposed pre-training strategy into other network architectures. A performance improvement can be observed for both settings of beta=0.9 and 0.99 in the experiment across various network architectures.

	Params (M)	β	Pre-trained	Baseline	Infer.time (s)
DnCNN [5]	0.56	0.9	30.49/0.812	26.76/0.720	15.1
		0.99	31.69/0.845	30.68/0.825	75.0
ResNet [68]	0.26	0.9	30.46/0.812	29.20/0.778	8.0
		0.99	31.43/0.838	31.16/0.836	39.4

F   Extension to other network structures

In the main text, we discuss the effect of pre-training on the proposed model using the same network architecture as DIP [13]. We are curious whether this prior knowledge could be applied to other model architectures. Here, we compare the impact of pre-training under different model settings. Specifically, we evaluate the removal of Gaussian noise with sigma=25 on CSet using three additional network architectures (DnCNN [5], ResNet [68]), as shown in Table 10. The pre-training approach consistently brings performance gains across various network architectures.

G   More Experimental Settings & Results

G.1   Quantitative analysis of Poisson noise removal

Due to space limitations in the text, there is no quantitative comparison of Poisson noise, which is listed in this section, see Table 11.

G.2   Quantitative comparison with more methods

Here we compare more recent methods including additional DIP-based zero-shot method (DIP-SURE [20]), diffusion-based methods (DDNM [41], DDPG [42]), and zero-shot modifications from unsupervised methods (AP-BSN [36], MM-BSN [43], PUCA [44]), refer to Table 12 for results.

Table 11: Quantitative comparison on CSet, McMaster and CBSD dataset for **Poisson noise removal** ($\lambda \in [10,25,50]$). For best results **highlighted** and second <u>underlined</u>.

	λ	DIP [13]	N2V* [16]	N2S* [17]	ZS-N2N [22]	FasterDIP [19]	Ours (faster)	Ours
CSet [3]	10	22.88/0.495	26.50/0.650	25.34/0.661	25.70/0.618	25.02/0.633	**27.79/0.714**	<u>27.55/0.696</u>
	25	27.57/0.681	27.16/0.755	27.16/0.750	28.06/0.711	27.73/0.710	<u>29.70/0.784</u>	**30.02/0.785**
	50	30.03/0.775	29.88/0.818	27.68/0.780	29.79/0.780	28.86/0.749	<u>31.00/0.830</u>	**31.68/0.841**
McMaster [33]	10	24.45/0.644	25.97/0.696	25.68/0.735	26.09/0.689	26.14/0.730	**28.26/0.793**	<u>28.15/0.770</u>
	25	29.23/0.801	28.84/0.807	27.28/0.782	28.49/0.775	28.03/0.783	<u>30.34/0.856</u>	**30.92/0.862**
	50	31.13/0.856	30.60/0.871	27.78/0.803	30.34/0.834	28.70/0.792	<u>31.74/0.886</u>	**32.64/0.900**
CBSD [34]	10	21.81/0.544	23.17/0.686	24.49/0.681	25.25/0.662	23.53/0.643	**26.27/0.716**	<u>26.05/0.709</u>
	25	26.83/0.741	26.96/0.798	26.20/0.775	27.66/0.776	26.64/0.757	<u>28.91/0.823</u>	**29.00/0.824**
	50	29.35/0.828	28.25/0.835	26.95/0.808	29.55/0.833	27.91/0.781	<u>30.45/0.871</u>	**31.00/0.881**

Table 12: Quantitative comparison with other DIP-based method (DIP-SURE [20]), Diffusion methods (DDNM, DDPG), zero-shot methods modified from other Self-supervised methods (AP-BSN, MM-BSN, PUCA). DIP-SURE, DDNM and DDPG requires additional noise variance as input, and DIP-SURE applies different iterations for each image.

	Method	CSet+Gaussian			SIDD validation	PolyU	FMD	Avg. Infer. time (s)
		$\sigma=10$	$\sigma=25$	$\sigma=50$				
DIP	DIP-SURE(peak)	35.37/0.916	31.88/0.855	28.81/0.775	30.45/0.727	35.87/0.944	32.04/0.798	-
	DIP-SURE(last)	34.98/0.908	31.50/0.840	28.76/0.762	26.63/0.649	35.78/0.942	32.07/0.793	367.3
Diff.	DDNM	36.22/0.927	32.4/0.859	29.99/0.793	28.11/0.597	37.15/0.935	28.99/0.685	26.7
	DDPG	32.43/0.826	27.07/0.606	15.95/0.183	29.84/0.612	35.79/0.887	30.41/0.735	24.3
Self-supervised	AP-BSN*	-	25.04/0.671	-	33.34/0.847	32.64/0.928	29.27/0.799	351.4
	MM-BSN*	-	25.27/0.676	-	33.36/0.843	33.07/0.930	29.73/0.810	505.3
	PUCA*	-	24.74/0.640	-	33.52/0.816	33.31/0.927	30.22/0.808	450.0
	Ours	34.91/0.909	31.61/0.841	28.26/0.710	34.43/0.844	38.11/0.962	32.97/0.847	45.8

Specifically, for DIP-SURE, we report both peak performance and the performance at the final iteration. Since DIP-SURE is specifically designed for Gaussian and Poisson noise, and requires the input of Gaussian noise variance, for real-world denoising tasks we provide estimated variance from paired data, to report the best performance. For diffusion-based methods, which are trained exclusively on Gaussian noise and also require variance as prior, we use the same variance estimation approach to report their best results. For self-supervised methods, which can be easily adapted to a single image with minimal changes, we follow their original settings. In each iteration, we crop eight same-size patches from the noisy image to form a batch, and perform inference on the full image every 10 iterations, and combine denoised images using the same ensemble strategy as our method for fairness.

We observe that DIP-SURE, due to its priors on noise type and variance, performs slightly better than our method under Gaussian noise settings. However, its performance significantly drops when dealing with real noise, especially when reporting the last performance. Since diffusion models are inherently Gaussian denoisers, they perform well on Gaussian noise when the variance is known, but also face challenges with real noise. The modified blind-spot network-based methods can handle severe real noise relatively well, but they may suffer from potential image detail loss and require long inference times.

G.3 Ensemble results of N2V and N2S

In the main text, we present versions of DIP [13] and FasterDIP [19] with EMA (Exponential Moving Average) ensembles, as these processes are included in their source codes. To provide additional information for comparison, we also adapted N2V* [16] and N2S* [17] to their corresponding EMA ensemble versions, as shown in Table 13, 14, 15. Generally, the ensemble versions of these methods can improve the PSNR by 1~2 dB. However, even though the enhanced N2V may outperform our Faster version in some cases, it does not affect the performance comparison with our β=0.99 version, which remains the best. Moreover, our β=0.99 version achieves this with less than half the inference time required by these methods.
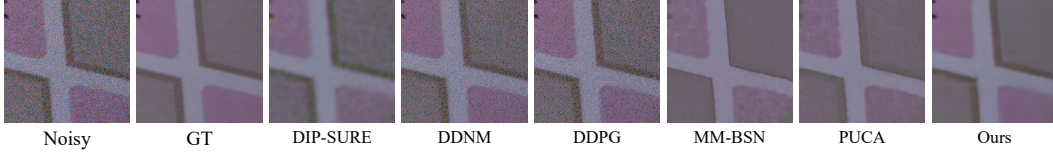
Figure 20: Comparison of different methods under SIDD validation (SIDDval_34_22).

Table 13: Quantitative comparison of ensemble version of N2V* and N2S* on CSet, McMaster and CBSD dataset for Gaussian and Poisson noise removal.

		Gaussian			Poisson		
		β=10	β=25	β=50	λ=10	λ=25	λ=50
	CSet [3]	34.05/0.895	30.99/0.825	27.70/0.703	27.02/0.675	29.55/0.790	30.95/0.830
N2V* [16]	McMaster [33]	34.31/0.920	30.97/0.862	27.94/0.766	26.93/0.751	30.06/0.812	31.78/0.884
	CBSD [34]	33.15/0.918	30.05/0.850	26.27/0.702	25.42/0.714	28.55/0.811	30.40/0.869
	CSet [3]	29.92/0.843	28.76/0.787	26.67/0.704	26.83/0.670	27.94/0.765	28.91/0.797
N2S* [17]	McMaster [33]	29.85/0.867	28.42/0.785	25.07/0.678	26.35/0.760	28.24/0.814	28.99/0.836
	CBSD [34]	28.50/0.854	27.51/0.803	25.07/0.705	24.83/0.714	26.59/0.785	27.53/0.817

G.4   Details of Unseen Noise

Gaussian Noise. Gaussian noise follows a normal distribution and is commonly encountered in digital imaging, especially during sensor data acquisition and transmission. It represents random variations in intensity and color information in images, making it a fundamental noise model in image processing. For each element in clean image $I_{[k]}$, is represented by:

$$\hat{I}_{[k]} = I_{[k]} + \sigma \cdot N_{[k]}, \tag{9}$$

where $N_{[k]}$ represents random variable sampled from a standard normal distribution, which is characterized by its standard deviation (σ).

Poisson Noise. Poisson noise is prevalent in scenarios with low-light conditions, such as astronomical imaging or medical imaging, where the photon count is inherently random and follows a Poisson distribution. Poisson noise models the variation of intensity based on a Poisson distribution, is generally expressed as:

$$\hat{I}_{[k]} = P(I_{[k]} \cdot \lambda)/\lambda, \tag{10}$$

where λ indicates the event rate, and $P(\cdot)$ denotes random variable generated from Poisson distribution.

Noise Level Function (NLF). Noise level function, also referred to as heteroscedastic gaussian model [69], is commonly described by a varying standard deviation across the image. This type of noise is widely used to express the read-shot noise in camera imaging pipeline, where different parts of the image exhibit different noise levels. It is typically modeled as:

$$\hat{I}_{[k]} \sim \mathcal{N}(\mu = I_{[k]}, \sigma^2 = \sigma_r + \sigma_s \cdot I_{[k]}) \tag{11}$$

where σ_r and σ_r represent different standard deviations in distinct regions of the image. Noise parameter calibrated for [40] in work [70] obeys a log-linear rule:

$$log(\sigma_r) = 2.18 \cdot log(\sigma_s) + 1.2 \tag{12}$$

We choose $\sigma_s \in [0.01, 0.012]$ to better illustrate the generality.

Speckle Noise. Speckle noise is an interference pattern produced by the coherent processing of a signal, especially in active radar and ultrasound imaging. This noise is particularly common in radar and ultrasound images, where it can significantly degrade the quality of the image. Its mathematical representation is:

$$\hat{I}_{[k]} = I_{[k]} + I_{[k]} \cdot U_{[k]}, \tag{13}$$

where $U_{[k]}$ is sampled from uniform distribution with mean 0 and v representing the standard deviation of the noise.

Table 14: Quantitative comparison of ensemble version of N2V* and N2S* on Kodak with 5 noise types for generalization evaluation.

| | Gaussian | | Poisson | NLF | Speckle | S&P | Average |
	$\sigma=25$	$\sigma\in[10,50]$	$\lambda\in[10,50]$		$v\in[10,50]$	$d\in[0.02,0.05]$	
N2V*	30.95/0.850	30.90/0.836	29.55/0.825	32.26/0.888	33.82/0.917	34.26/0.948	31.95/0.877
N2S*	28.34/0.804	27.91/0.790	27.21/0.779	28.82/0.840	28.84/0.846	28.88/0.828	28.33/0.816

Table 15: Quantitative comparison of ensemble version of N2V* and N2S* on SIDD, PolyU and FMD dataset for real-world noise removal.

| Methods | SIDD [1] | | PolyU [45] | FMD [2] |
	validation	benchmark		
N2V* [16]	28.51/0.670	27.19/0.645	36.11/0.921	30.85/0.754
N2S* [17]	27.41/0.584	27.59/0.684	35.39/0.939	31.72/0.762

Salt-and-Pepper Noise (S&P). Salt-and-Pepper noise, also known as impulse noise, is characterized by sharp and sudden disturbances in an image signal. It's typically represented as sparse white and black pixels, hence the name. This noise can be caused by sharp and sudden disturbances in the image signal, often due to transmission errors, faulty memory locations, or timing errors in digital image sensors. Its mathematical representation is:

$$\hat{I}_{[k]} = I_{[k]} + S_{[k]} - S_{[k]},\tag{14}$$

where $S_{[k]}$ and $P_{[k]}$ represents salt and pepper noise, respectively. For each of them are Bernoulli sample with probability d of I_{max}/I_{min} and probability $1-d$ of 0, which makes the probability total affected is $2 \cdot d$.

G.5   Additional Computational Costs

In analyzing the performance of deep learning models, it's crucial to consider both the Floating Point Operations (FLOPs) and the model parameters. FLOPs give us an insight into the computational complexity of the model, which affects inference time and resource utilization, while the number of parameters indicates its capacity to learn and adapt to complex data patterns. A balance between these two aspects is essential for efficient and effective model performance. Our analysis, as reflected in the comparison between Table 16, demonstrates that our method successfully achieves this balance. It maintains computational efficiency without compromising the model's ability to accurately process and analyze data, an essential factor for practical application in varied computational environments.

G.6   Zero-shot Denoising on More Image Types

In the main text, we demonstrate the ability of proposed MPI to generalize to other types of images through a medical imaging dataset. Further here, we explore new types of images, including a microscopy imaging dataset BioSR [71] and extremely low-light dataset SID [72]. See Fig. 21 and Fig. 22 for qualitative examples.

Table 16: Efficiency comparisons of deep learning-based methods on Params and FLOPs under input size 256×256 with a single forward step. Iterations used for synthetic noise is provided for reference.

Method	Params (M)	FLOPs (G)	Iters
DIP [13]	2.3	19.66	3,000
N2V* [16]	1.2	80.50	1,500
N2S* [17]	0.07	1.57	1,800
ZS-N2N [22]	0.02	1.45	2,000
FasterDIP [19]	0.05~0.92	0.5~8.8	3,000
Ours(faster)	0.73	8.11	1,000
Ours	0.73	8.11	200

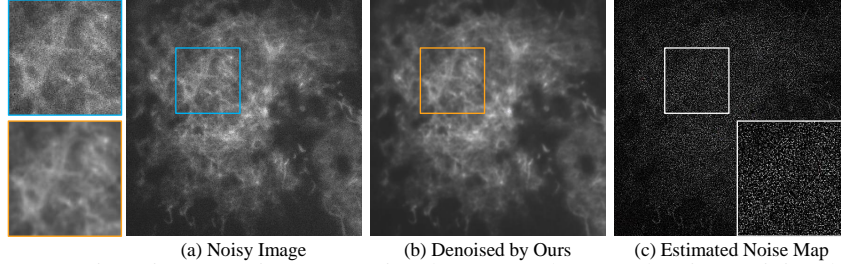(a) Noisy Image   (b) Denoised by Ours   (c) Estimated Noise Map

Figure 21: On a noisy microscopy image (a) using the proposed MPI to denoise retaining the structural information in the image as much as possible (b), see the noise map (c).
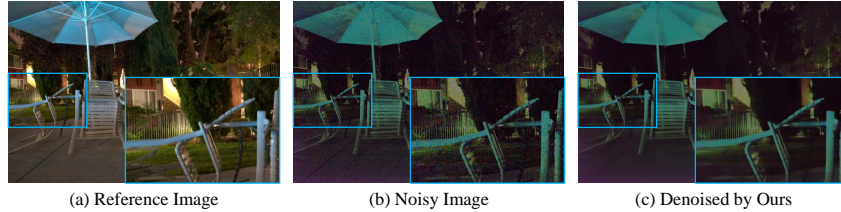

(a) Reference Image   (b) Noisy Image   (c) Denoised by Ours

Figure 22: For extremely low-light images, there is serious color bias in the expected denoising result (a) and captured noisy image (b). This color bias is still retained in the denoising result, but the noise is basically removed. This is discussed in Sec. H.2.

H   Concluding Remarks

In this study, we introduce Masked Pre-train then Iterative fill (MPI), a zero-shot denoising paradigm utilizing pre-trained model with random masks on natural images. The pre-trained weights is optimized on a specific noisy image through Iterative filling process, and predictions with corresponding masks during inference is combined for enhanced quality and faster inference.

H.1   Broader impacts

From the perspective of our work, we have pioneered the use of generalizable knowledge from natural images without any assumptions about noise degradation, offering an efficient framework for handling diverse synthetic and real noises with significantly reduced inference time, which is a critical issue in zero-shot denoising and makes their practical applications feasible. Notably, our zero-shot method excels in generalization compared to current supervised and unsupervised methods, offering new insights into denoising.

H.2   Limitations

Although proposed MPI has shown effects in removal of various types of noises, the mask-based noise-supervised denoising setting does not seem to allow the removal of non-zero mean noise. So when dealing with extremely low-light images with severe color bias, the color bias still remains after denoising; this is a common problem in zero-shot denoising, because there is no prior regarding noise-clean image pair in specific domain, but it may limit several practical applications, and we are currently trying to solve this problem in other ways.

I   Additional Qualitative Results

The following figures show the denoising comparison on both synthetic noise removal (see Fig. 23 - Fig. 31) and denoising real noise data (Fig. 32 - Fig. 35).

Gaussian σ=10	Reference	DIP
28.13/0.690	PSNR/SSIM	34.03/0.923
N2V*	N2S*	ZS-N2N
34.57/0.940	30.57/0.893	34.33/0.90
FasterDIP	Ours (faster)	Ours
34.17/0.920	35.81/0.946	36.84/0.957

Figure 23: Qualitative comparison of results on CBSD [34] with Gaussian σ=10. Noisy patch is from CBSD-11.

Figure 24: Qualitative comparison of results on CBSD [34] with Gaussian σ=25. Noisy patch is from CBSD-31.
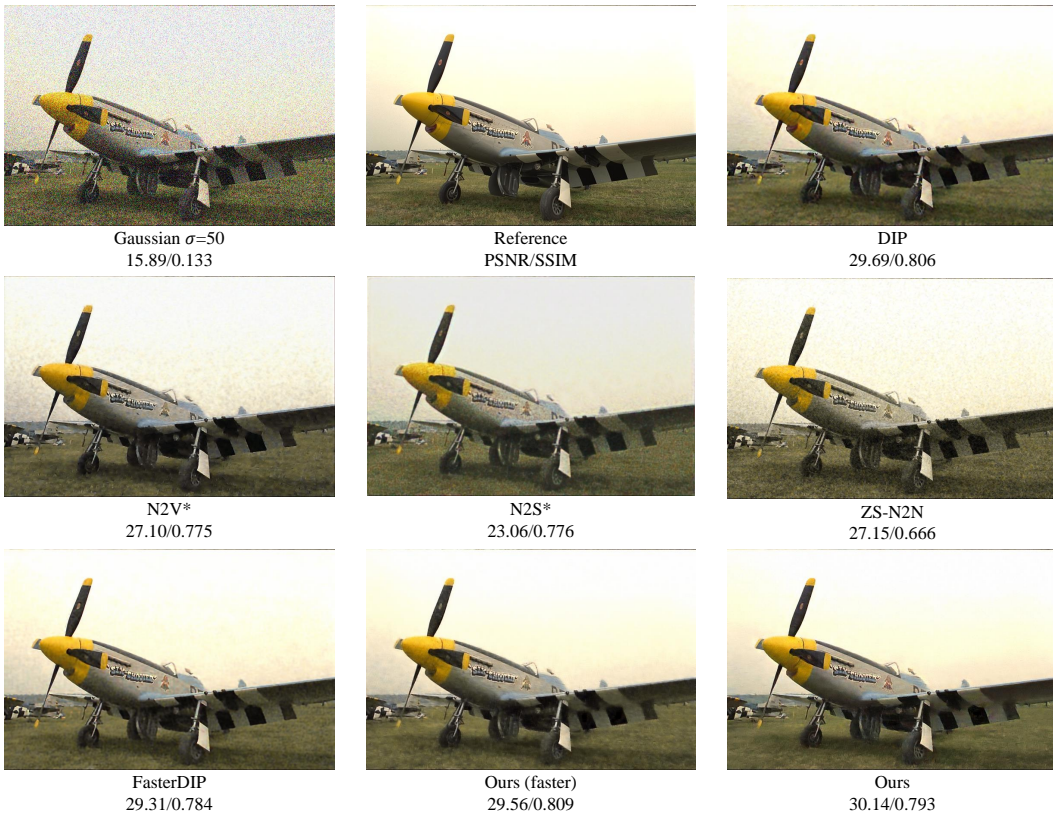
Figure 25: Qualitative comparison of results on Kodak [46] with Gaussian σ=50. Noisy patch is from kodim20.
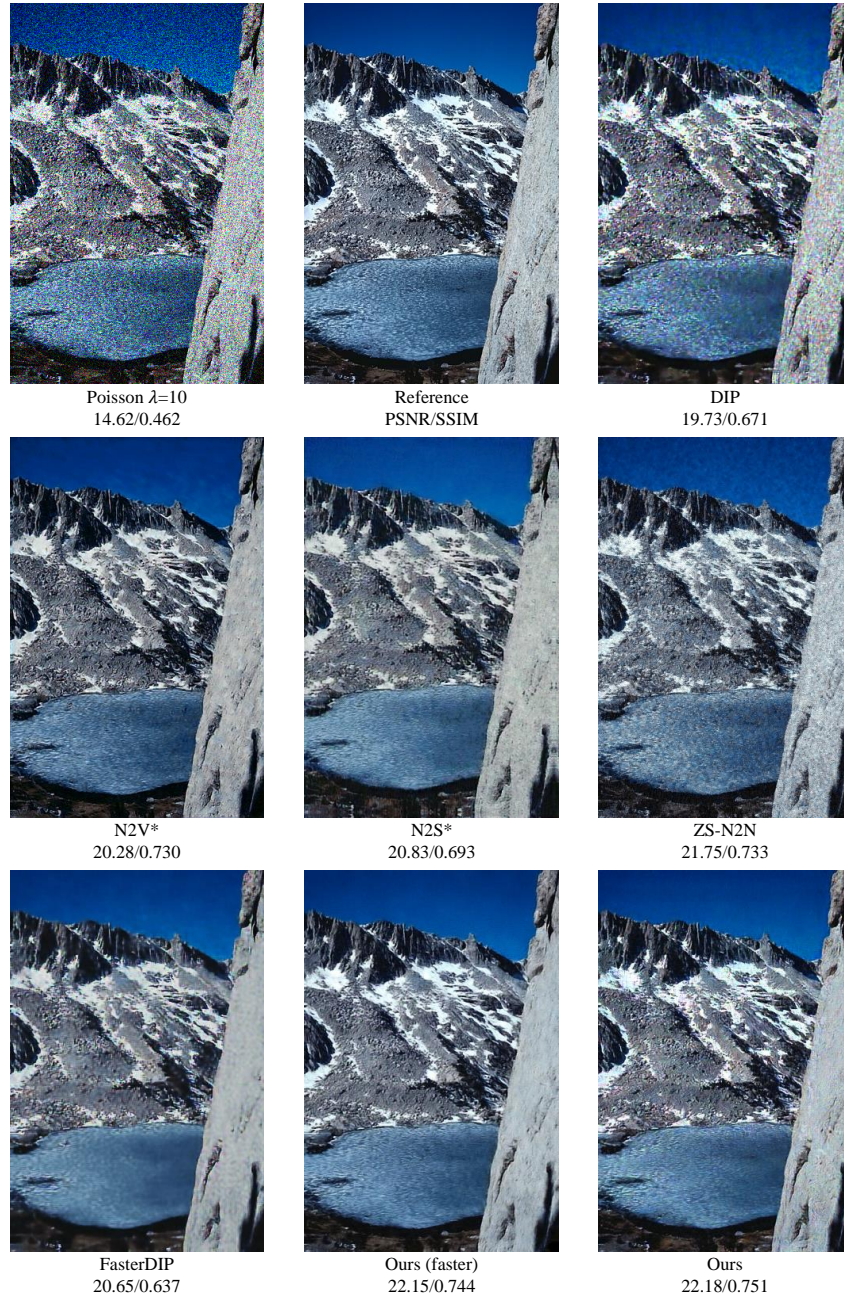
Poisson λ=10	Reference	DIP
14.62/0.462	PSNR/SSIM	19.73/0.671

N2V*	N2S*	ZS-N2N
20.28/0.730	20.83/0.693	21.75/0.733

FasterDIP	Ours (faster)	Ours
20.65/0.637	22.15/0.744	22.18/0.751

Figure 26: Qualitative comparison of results on CBSD [34] with Poisson λ=10. Noisy patch is from CBSD-33.

Figure 27: Qualitative comparison of results on CBSD [34] with Poisson λ=25. Noisy patch is from CBSD-56.
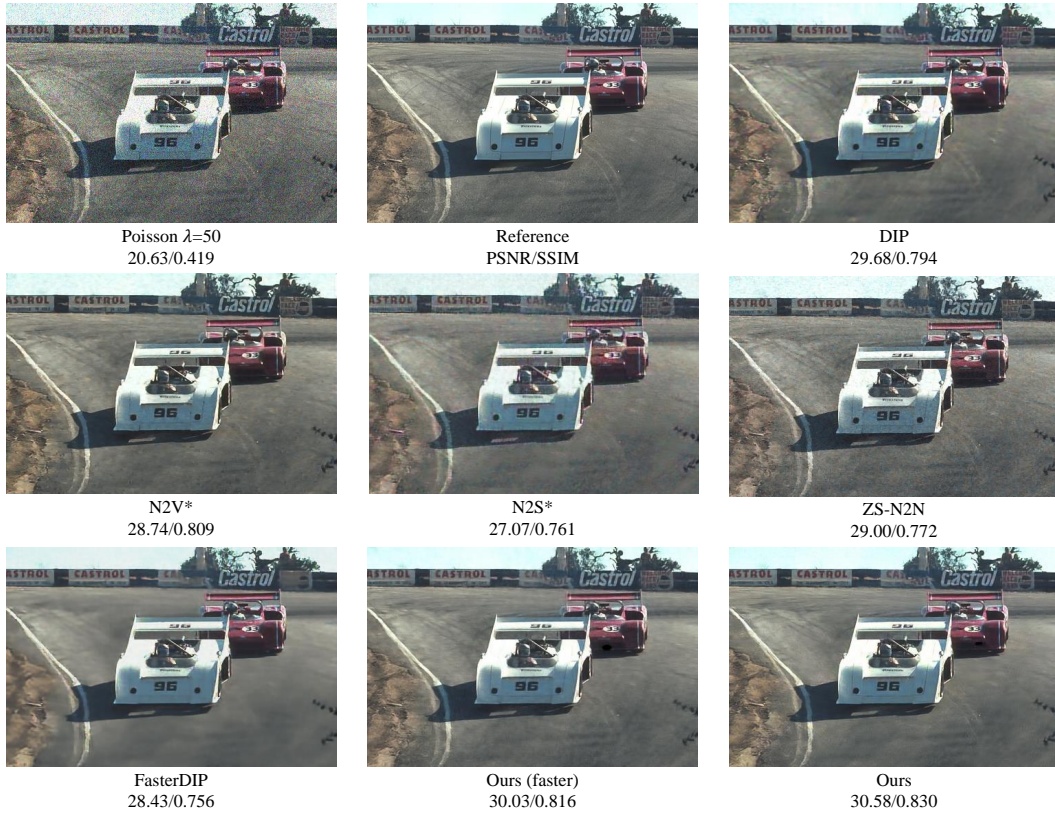
Figure 28: Qualitative comparison of results on CBSD [34] with Poisson λ=50. Noisy patch is from CBSD-05.
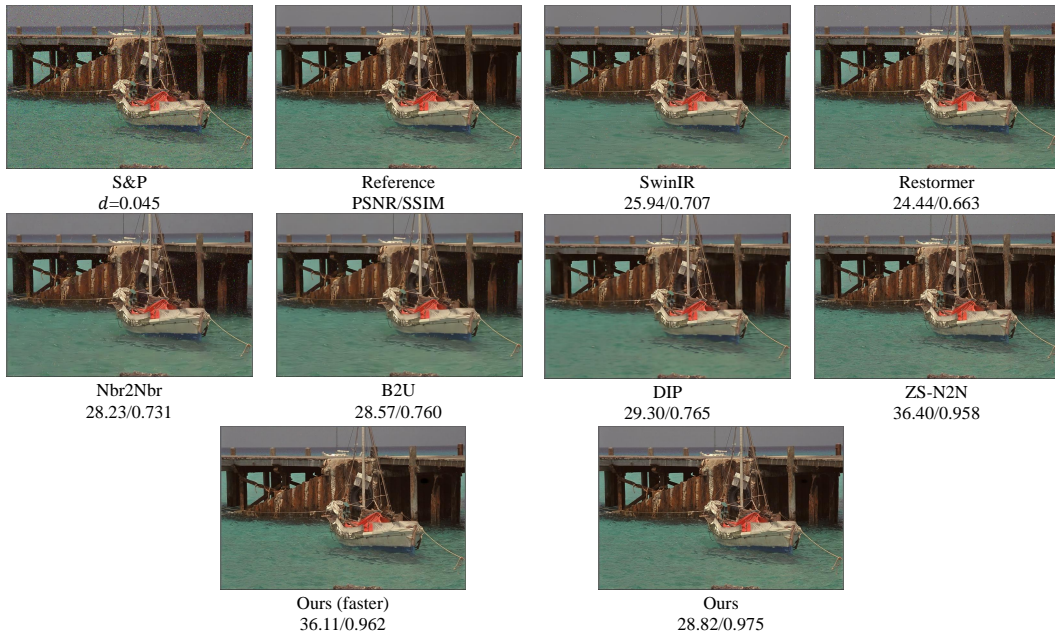

Figure 29: Qualitative comparison of generalization on Kodak [46] with S&P d=0.045. Noisy patch is from kodim11.
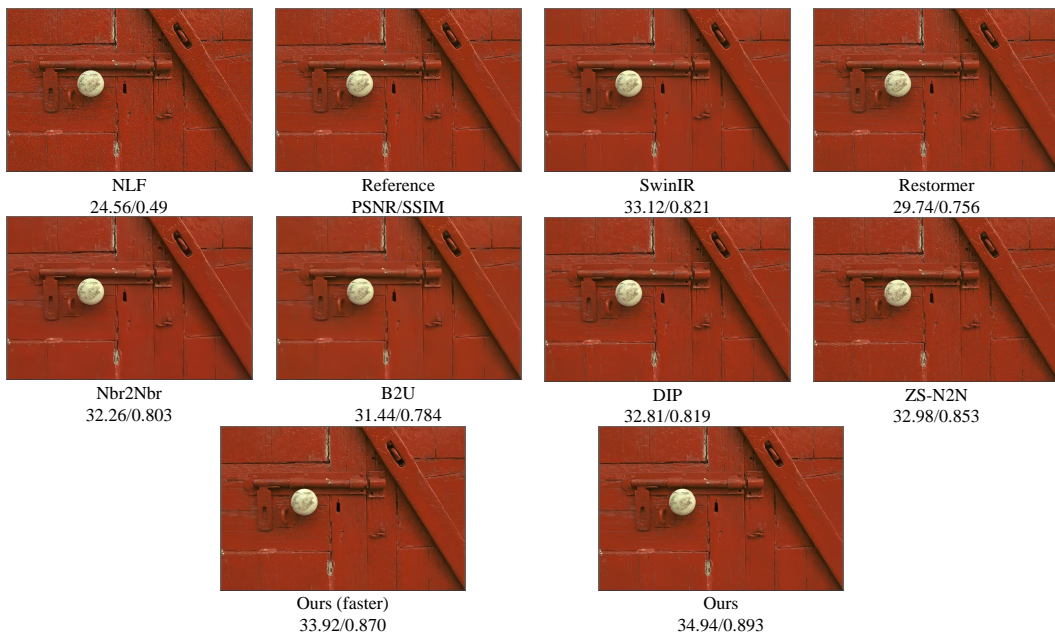
16

Figure 30: Qualitative comparison of generalization on Kodak [46] with NLF [40]. Noisy patch is from kodim02.

Figure 31: Qualitative comparison of generalization on Kodak [46] with Poisson λ=40. Noisy patch is from kodim04.
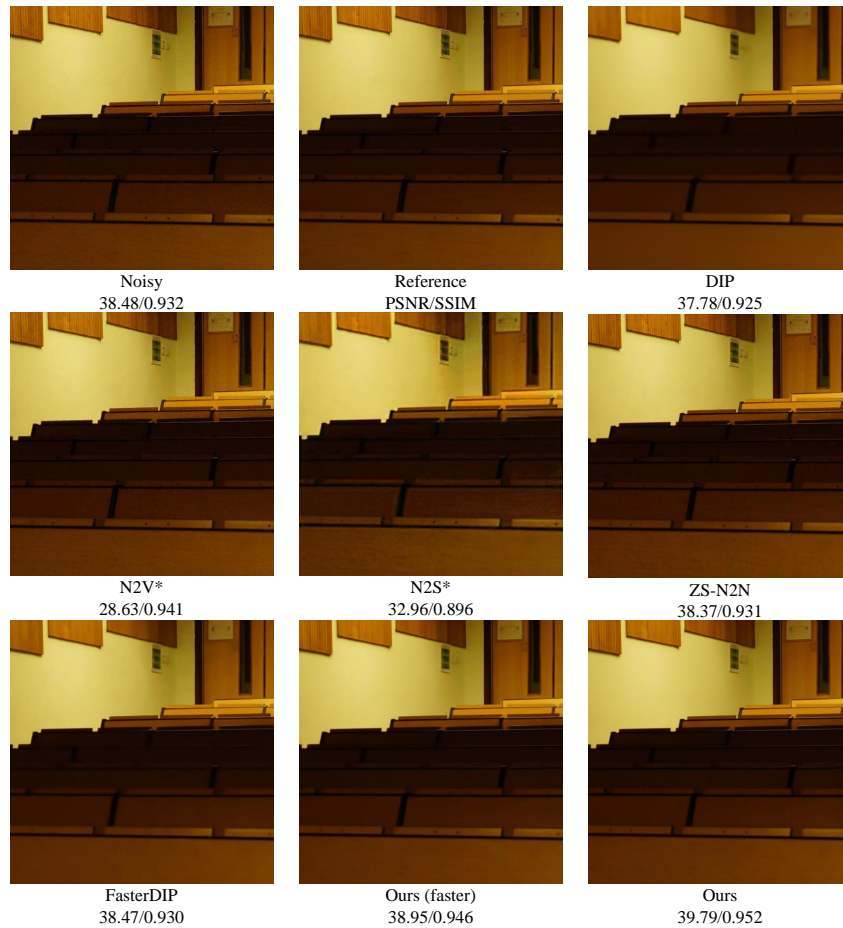
Figure 32: Qualitative comparison of realnoise on PolyU [45]. Noisy patch is from Sony_3-5_200_1600_classroom_14.
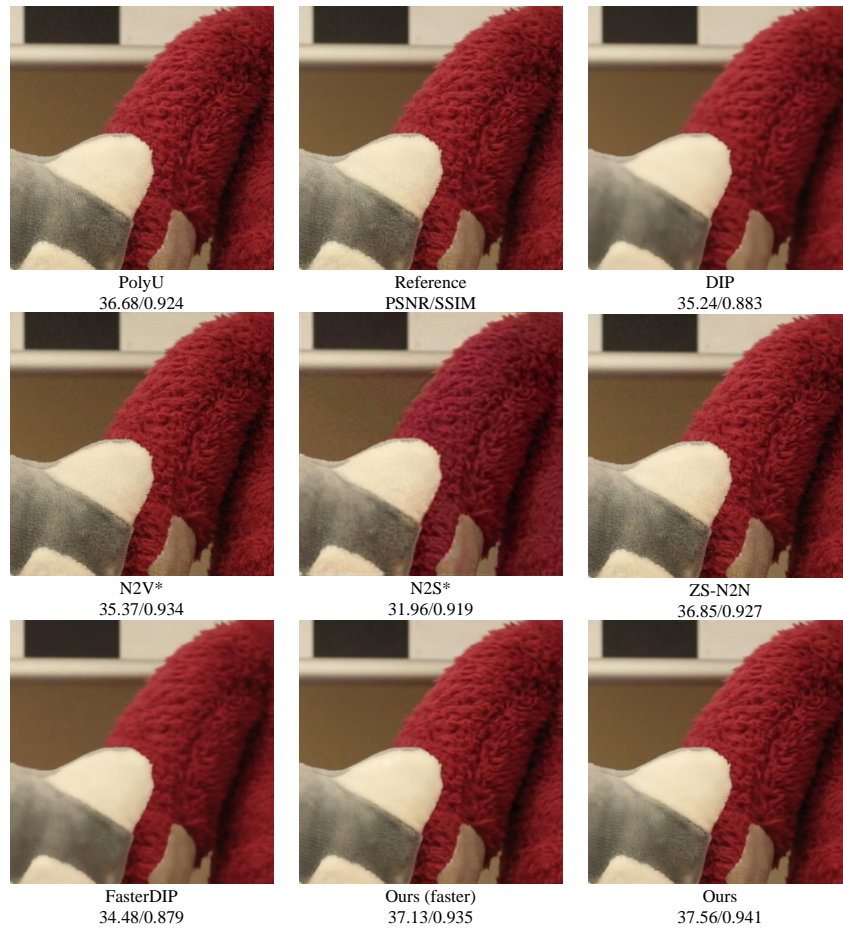
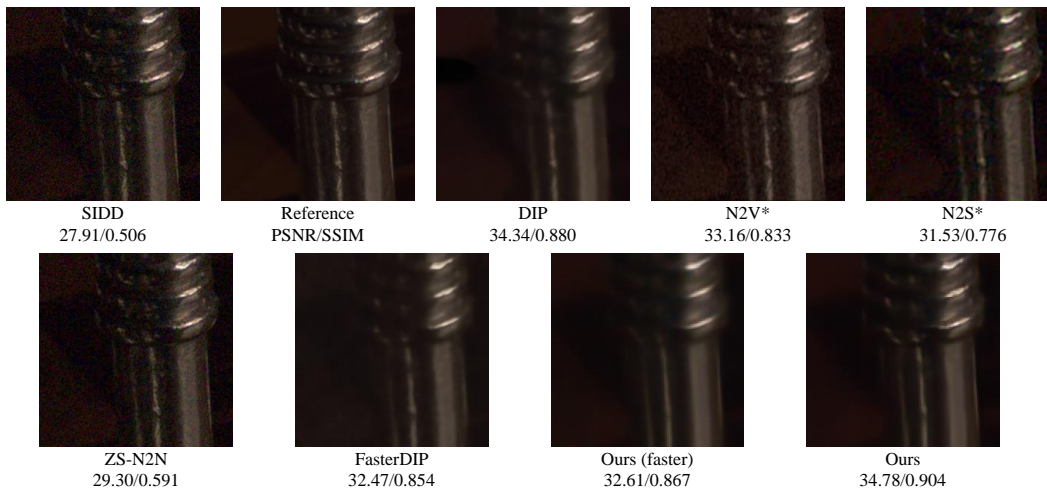Figure 33: Qualitative comparison of realnoise on PolyU [45]. Noisy patch is from Canon5D2_5_200_3200_toy_1.

Figure 34: Qualitative comparison of realnoise on SIDD [1]. Noisy patch is from SIDDval_20_8.
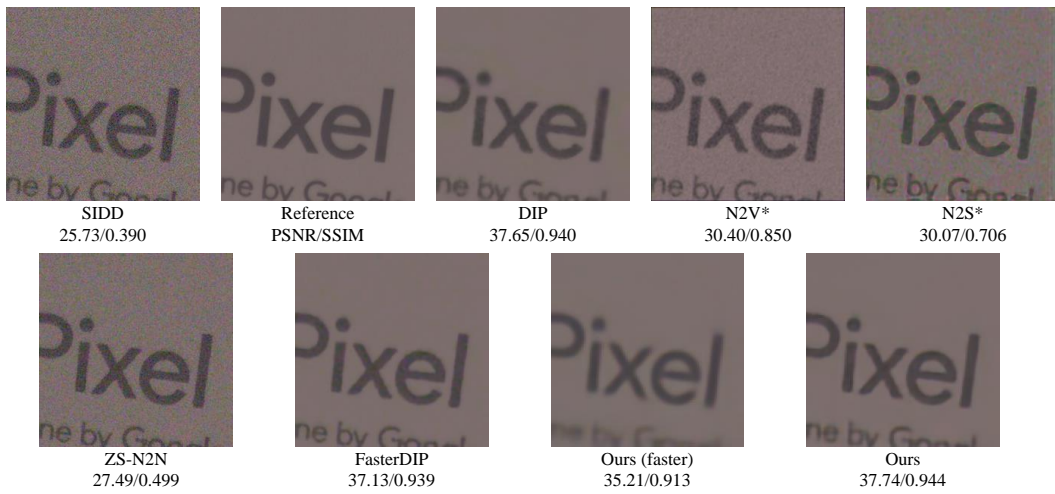
Figure 35: Qualitative comparison of realnoise on SIDD [1]. Noisy patch is from SIDDval_13_19.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: NA

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: NA

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA] .

Justification: NA

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: NA

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: NA

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: NA

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: NA

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: NA

   Guidelines:
   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: NA

   Guidelines:
   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: NA

    Guidelines:
    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: NA

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: NA

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: NA

    Guidelines:

    - The answer NA means that the paper does not release new assets.

- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: NA

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: NA

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.