Robust Image Denoising through Adversarial Frequency Mixup

Donghun Ryou[2] Inju Ha[1] Hyewon Yoo[2] Dongwan Kim[1] Bohyung Han[1,2]
Computer Vision Laboratory, [1]ECE & [2]IPAI, Seoul National University
{dhryou, hij1112, yoohyewony, dongwan123, bhhan}@snu.ac.kr

Abstract

Image denoising approaches based on deep neural networks often struggle with overfitting to specific noise distributions present in training data. This challenge persists in existing real-world denoising networks, which are trained using a limited spectrum of real noise distributions, and thus, show poor robustness to out-of-distribution real noise types. To alleviate this issue, we develop a novel training framework called Adversarial Frequency Mixup (AFM). AFM leverages mixup in the frequency domain to generate noisy images with distinctive and challenging noise characteristics, all the while preserving the properties of authentic real-world noise. Subsequently, incorporating these noisy images into the training pipeline enhances the denoising network's robustness to variations in noise distributions. Extensive experiments and analyses, conducted on a wide range of real noise benchmarks demonstrate that denoising networks trained with our proposed framework exhibit significant improvements in robustness to unseen noise distributions. The code is available at https://github.com/dhryougit/AFM.

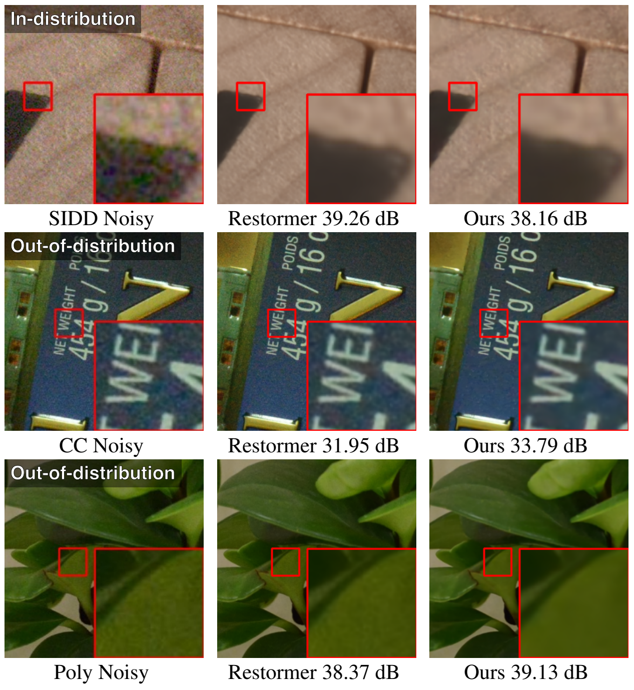


Figure 1. The difference in real-world denoising results for in-distribution and out-of-distribution samples. While the state-of-the-art Restormer [28] performs well on in-distribution samples from SIDD [1], its performance is degraded on out-of-distribution examples from CC [16] and Poly [23]. In contrast, a DnCNN [30] model trained with our AFM exhibits robustness for both samples.

1. Introduction

Image denoising based on deep neural networks [5, 6, 22, 26–28, 30, 31] has witnessed unprecedented success benefiting from the simplicity of the problem formulation and the construction of new datasets. Traditionally, denoising networks have been trained using synthetic noise models, such as Gaussian or Poisson noise, which are artificially added to images for training and evaluation purposes. However, real-world noise, influenced by various factors within the Image Signal Processing (ISP) pipeline such as demosaicing and gamma correction, exhibits a distinct signal dependency and often follows distributions that differ from synthetic counterparts. This divergence between synthetic and real-world noise distributions raises significant generalization issues when applying denoising models to real noisy images.

While efforts have been devoted to creating datasets with clean and noisy image pairs in the real-world, collecting these datasets poses a significant challenge. Thus, various self-supervised approaches [10, 12, 18] have emerged as promising solutions for image denoising, aiming to reduce the dependence on paired noisy-clean image datasets. Nevertheless, these methods often fall short in performance when compared to supervised learning approaches. As a result, supervision on real noise datasets still remains favorable in practical scenarios.

Image denoisers trained with supervision on real noise datasets, however, come with their own set of flaws. Most

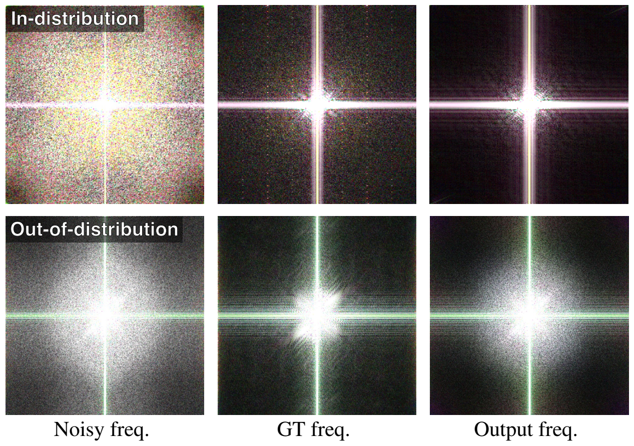


Figure 2. Visualizing the frequency magnitudes of denoising results for an in-distribution (SIDD [1], top row) and out-of-distribution (Poly [23], bottom row) sample, using a model trained without AFM. For the in-distribution sample, the model yields a cleaner frequency map after denoising than the ground truth, but for the out-of-distribution sample, the model struggles to denoise certain regions in the frequency map.

notably, we observe that even state-of-the-art denoisers encounter difficulty generalizing to variations in noise distributions, which arise from factors such as different camera sensor types, shooting environments, and ISP processes. We illustrate this phenomenon in Figure 1, and provide a frequency analysis in Figure 2. While this lack of real noise generalization has a detrimental effect on the widespread use of denoising models, it is a relatively unexplored issue in the field of image denoising research.

In this work, we propose Adversarial Frequency Mixup (AFM), a model-agnostic training framework that improves the generalizability of denoising networks to variations in real noise distributions. To this end, AFM constructs images with noise that is unique and difficult to denoise, while maintaining the properties of real-world noise. Then, by integrating these noisy images into the training pipeline, the denoising network becomes robust to variations in real noise distributions. More specifically, AFM works by mixing up a noisy and denoised image according to a mixup mask in the frequency domain. This mixup mask is generated by a separate lightweight neural network, which yields two variants of our approach: AFM-E and AFM-B. AFM-E generates the mixup mask in an element-wise manner, while AFM-B generates the mask by assigning a mixup value for each frequency band. Moreover, AFM is trained using an adversarial loss and can generate adversarial mixup masks.

Overall, our contributions are summarized as follows:

- We propose AFM, a model-agnostic training framework that improves the denoising network's generalization and robustness to real-world out-of-distribution noise images by generating new noisy images through an adversarial mixup in the frequency domain.
- We design two adversarial mask generation networks, AFM-E and AFM-B. AFM-E generates mixup masks in an element-wise fashion while AFM-B generates mixup masks for individual bands.
- We demonstrate the effectiveness of AFM on multiple real-world out-of-distribution image denoising benchmarks. Our method consistently outperforms ordinary training on various denoising architectures and even state-of-the-art denoising models by significant margins.

2. Related Works

This section reviews existing image denoising models based on deep neural networks with and without supervision and discusses recent efforts for robust image denoising.

2.1. Supervised Image Denoising

In recent years, there have been significant advancements in the area of supervised image denoising, where paired noisy and clean images are available for training. The initial breakthroughs were largely driven by CNN-based models, with Denoising Convolutional Neural Network (DnCNN) [30] leading the stage for further innovations in this domain. Building upon this foundation, the U-Net-based architectures [4, 5, 26, 27] have emerged as prominent models, leveraging skip connections to effectively combine local and global contextual information. Additionally, the introduction of transformer-based models [22, 28] marked a paradigm shift in denoising strategies. Equipped with attention mechanisms, transformer-based models excel at identifying complex dependencies, effectively diminishing noise distortions in the process. These models have shown exceptional skill in reducing noise, substantially mitigating the impact of specific noise distributions they have been trained on using paired datasets. Yet, they encounter challenges in effectively managing noise distributions that diverge from those they have been trained on, which poses hurdles for their application in real-world scenarios.

2.2. Self-supervised Image Denoising

Collecting clean and noisy pair data from the real-world is cost-intensive. To mitigate these issues, there has been an active research on exploring unsupervised and self-supervised learning techniques like N2N [12], N2S [2], N2V [10], and R2R [18]. These approaches demonstrate the feasibility of training denoising networks using datasets comprising solely noisy pairs or individual noisy images, without the need for a clean and noisy pair dataset. Moreover, DIP [21], Self2Self [19] and Neighbor2Neighbor [7] have introduced novel approaches to train denoising networks even in the absence of training data, by using a single noisy image to produce a clean counterpart. However, these

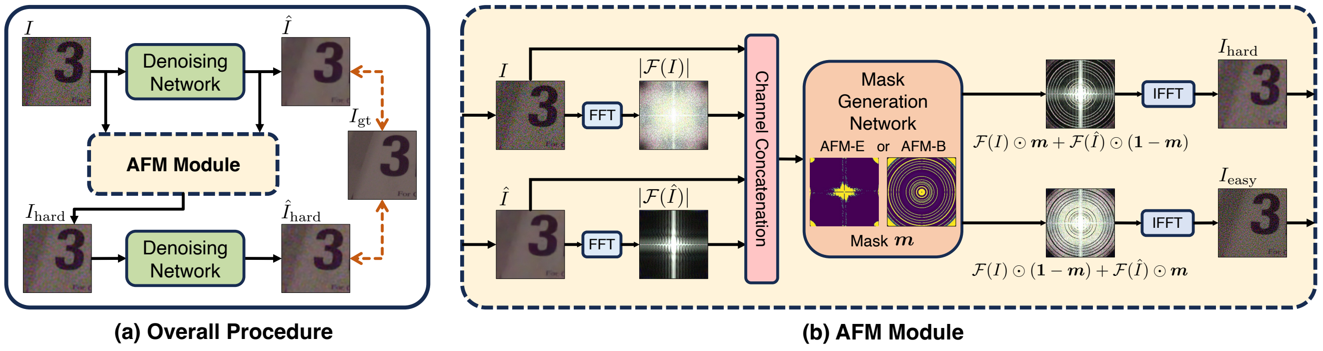


Figure 3. An overview of our Adversarial Frequency Mixup (AFM) framework. (a) Overall procedure. The input image I is fed to the denoising network to produce a denoised image, $\hat{I}$. Then, I and $\hat{I}$ are passed to the AFM module, which generates a new image I_{hard}. Both I and I_{hard} are used to train the denoising network. (b) Details of the AFM module, Two images, I and $\hat{I}$, are mixed in the frequency domain according to the mask $\boldsymbol{m}$ generated by the mask generator. Note that, while the AFM module generates both I_{hard} and I_{easy}, only I_{hard} is used to train the denoising network and I_{easy} is used to update the mask generator. (The process of training the mask generator is not illustrated in this figure.)

methods are plagued by inefficiency of significant time consumption, as the network needs to be retrained for each distinct image. Importantly, their performance often falls short when compared to supervised methods.

2.3. Generalization for Denoising

Existing denoising models have a generalization issue, overfitting to training noise distribution. Only a few works have been addressed this challenge [3, 15]. Mohan *et al.* [15] observe that the noise overfitting is caused by bias terms and remove all the biases from the network. However, this method is targeted to enhance robustness within various levels of noise, but not noise types. Chen *et al.* [3] introduce the input pixel and attention feature masks during training to focus on reconstructing a clean image itself rather than denoising, but its performance on a real noise still remains insufficient.

3. Methods

Our goal is to train a denoising network that demonstrates strong generalization performance across various examples with unseen real noise. To accomplish this, we propose Adversarial Frequency Mixup (AFM), a novel training framework based on augmented images in the frequency domain with realistic noise distribution.

3.1. Background

Denoising networks aim to generate clean images, regardless of noise patterns imposed on input images. Such a procedure is formulated as

$$\mathcal{D}_\theta(\boldsymbol{x}+\boldsymbol{n}) = \boldsymbol{x}, \tag{1}$$

where $\mathcal{D}_\theta$ denotes the denoising network parametrized by θ and $(\boldsymbol{x}, \boldsymbol{n})$ indicates a pair of a clean image and its noise. For supervised learning, a prevalent approach is to train a denoising model using a real noise dataset such as SIDD [1], which consists of clean ($\boldsymbol{x}$) and noisy image ($\boldsymbol{x}+\boldsymbol{n}$) pairs. The objective of the problem is to optimize the model parameter θ by minimizing the following loss:

$$\mathcal{L} = \|\mathcal{D}_\theta(\boldsymbol{x}+\boldsymbol{n}) - \boldsymbol{x}\|. \tag{2}$$

However, conventional methods are prone to learning fixed mappings between clean and noisy images, which often leads to poor generalization. In practical scenarios, there is a wide range of noise variations due to the diverse characteristics of camera sensors and the Image Signal Processing (ISP) pipelines. Therefore, it is imperative for real noise denoising models to generalize to such variations in noise distributions.

3.2. Adversarial Frequency Mixup

Figure 3(a) illustrates an overview of the proposed AFM approach, designed to improve generalization to unseen real noise images. Given a noisy input image $I \in \mathbb{R}^{C\times H\times W}$, we first predict a denoised image $\hat{I}$ using a denoising network $\mathcal{D}_\theta$, which is given by

$$\hat{I} = \mathcal{D}_\theta(I). \tag{3}$$

Then, we mixup the original input I and the prediction $\hat{I}$ in the frequency domain as follows:

$$I_{\text{hard}} = \mathcal{F}^{-1}\left(\mathcal{F}(I) \odot \boldsymbol{m} + \mathcal{F}(\hat{I}) \odot (\mathbf{1}-\boldsymbol{m})\right), \tag{4}$$

where $\mathcal{F}$ is the Fast Fourier Transform (FFT), $\odot$ is an elementwise multiplication operator, $\boldsymbol{m} \in [0,1]^{1\times H\times W}$ is an arbitrary mask in the frequency domain, and I_{hard} is the resulting mixed image[1]. Since the mask $\boldsymbol{m}$ is bounded by

[1]The resulting image is termed I_{hard} for the reasons that will be discussed in Section 3.2 and 3.3

[0,1], the right-hand side of Eq. (4) represents an element-wise interpolation between I and $\hat{I}$ in the frequency domain. Given I and $\hat{I}$, the objective here is to generate a new image with some noise that is characterized by a distribution distinct from the original noisy image, while also resembling noise encountered in real-world scenarios. Our design choices in Eq. (4)—specifically, the use of frequency mixup—are directly inspired by this objective.

The Fourier Transform on I and $\hat{I}$ maps each image to the frequency domain, where it is relatively easier to manipulate the noise distribution of an image compared to the spatial domain. An intuitive visualization supporting this claim is shown in Figure 2, where the regions of noise and clean signal are clearly distinguishable in the frequency domain. Furthermore, changes in the frequency component act globally on the image, which minimizes the risk of unwanted manipulation to the underlying image content. Essentially, the resulting I_{hard} stays in the manifold of real noise images because it is an interpolation of I and $\hat{I}$. Therefore, frequency mixup prevents the construction of synthetic images with arbitrary and artificial noise.

By deriving a simplified expression of Eq. (4), we develop a more intuitive understanding of the mixup operation:

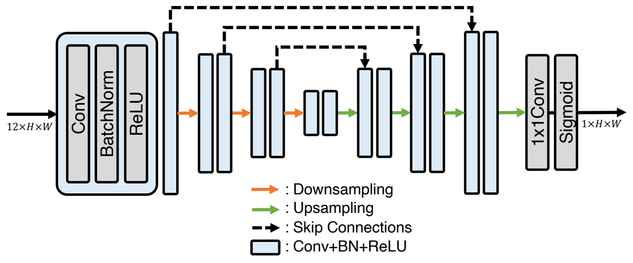
$$\begin{aligned} I_{\text{hard}} &= \mathcal{F}^{-1}\left(\mathcal{F}(\hat{I}) + \boldsymbol{m} \odot (\mathcal{F}(I) - \mathcal{F}(\hat{I}))\right) \\ &= \hat{I} + \mathcal{F}^{-1}\left(\boldsymbol{m} \odot \mathcal{F}(I - \hat{I})\right) \\ &= \hat{I} + \mathcal{F}^{-1}(\boldsymbol{m}) * \mathcal{F}^{-1}\left(\mathcal{F}(\hat{\boldsymbol{n}})\right) \\ &= \hat{I} + \mathcal{F}^{-1}(\boldsymbol{m}) * \hat{\boldsymbol{n}} \\ &= \hat{I} + \boldsymbol{h} * \hat{\boldsymbol{n}}, \end{aligned} \tag{5}$$

where $\boldsymbol{h}$ denotes a filter, $*$ denotes a convolution operator, and $\hat{\boldsymbol{n}} = I - \hat{I}$ denotes the predicted noise. Essentially, frequency mixup does not randomly alter the image, but rather *transforms* the noise component $\hat{\boldsymbol{n}}$ according to $\boldsymbol{h}$, which is defined by the inverse Fourier transform of mask $\boldsymbol{m}$.

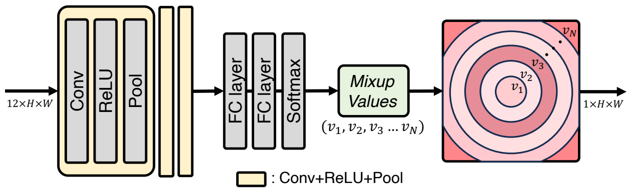
What remains is the choice of $\boldsymbol{m}$, or more specifically, how to create an appropriate mask for the frequency mixup. As will be discussed with much more detail in Section 3.3, we opt for an adversarial design of $\boldsymbol{m}$, which results in a real noise image that the denoising network considers as *hard* (hence the term I_{hard}). Then, we can incorporate I_{hard} into the training pipeline to improve the real-world generalization of the denoising network.

3.3. Mask Design

The mask $\boldsymbol{m}$ is adversarially generated such that the difference between the denoised output of I_{hard}, $\mathcal{D}_\theta(I_{\text{hard}})$, and the ground truth, I_{gt}, is maximized. As illustrated in Figure 3(b), this is done with the mask generation network G_ϕ, which can be implemented by either AFM-E or AFM-B.



(a) Element-wise mask generation network (AFM-E)



(b) Band-wise mask generation network (AFM-B)

Figure 4. Architecture details of the two mask generation networks.

Regardless of implementation choices, $\boldsymbol{m}$ is expressed as:

$$\boldsymbol{m} = G_\phi\left(\left[I, \hat{I}, |\mathcal{F}(I)|, |\mathcal{F}(\hat{I})|\right]\right), \tag{6}$$

where $|\mathcal{F}(I)|$ and $|\mathcal{F}(\hat{I})|$ are the frequency magnitudes of I and $\hat{I}$, respectively, and $[\cdot,\cdot]$ is a concatenation operator along the channel dimension. Given the mask $\boldsymbol{m}$, we construct I_{hard} following Eq. (4). Furthermore, to help stabilize the training of G_ϕ, we also construct I_{easy} using the opposite mask, $\mathbf{1} - \boldsymbol{m}$:

$$I_{\text{easy}} = \mathcal{F}^{-1}\left(\mathcal{F}(I) \odot (\mathbf{1} - \boldsymbol{m}) + \mathcal{F}(\hat{I}) \odot \boldsymbol{m}\right). \tag{7}$$

We formulate the following loss function to make I_{hard} adversarial and train the mask generation network, G_ϕ:

$$\mathcal{L}_{\text{AFM}} = \text{PSNR}(\hat{I}_{\text{hard}}, I_{\text{gt}}) - \gamma \text{PSNR}(\hat{I}_{\text{easy}}, I_{\text{gt}}), \tag{8}$$

where $\hat{I}_{\text{hard}} = \mathcal{D}_\theta(I_{\text{hard}})$, $\hat{I}_{\text{easy}} = \mathcal{D}_\theta(I_{\text{easy}})$, and γ is a hyperparameter that balances the two terms. The first term satisfies our main purpose: generating the adversarial case I_{hard} by minimizing the Peak Signal-to-Noise Ratio (PSNR) between $\hat{I}_{\text{hard}}$ and I_{gt}. The second term aims to maximize the PSNR between $\hat{I}_{\text{easy}}$ and I_{gt}, since I_{easy}, which is a complement term of I_{hard}, should be generated as a relatively easy input for the denoising network. Although I_{easy} is not strictly necessary in Eq. (8) to train an adversarial G_ϕ, we empirically find that it helps prevent the generation of trivial masks.

3.3.1 AFM-E: Element-wise Mask Generation

AFM-E employs pixel segmentation to construct a mask with the same size as the input image. Thus, each element

Algorithm 1 AFM Training Procedure

```
Require: $I$: Input noisy image
Require: $\mathcal{D}_\theta(\cdot)$: Denoising network
Require: $G_\phi$: Mask generation network
Require: AFM$(\cdot\,|G_\phi)$: AFM module
 1: for $t = 1$ to $T$ do
 2:     $\hat{I} \leftarrow \mathcal{D}_\theta(I)$
 3:     $I_{\text{hard}} \leftarrow \text{AFM}(I, \hat{I}\,|G_\phi)$
 4:     $\hat{I}_{\text{hard}} \leftarrow \mathcal{D}_\theta(I_{\text{hard}})$
 5:     Update $D_\theta$ by $\mathcal{L}_{\text{D}}$                ▷ Eq. (9)
 6:     $\hat{I} \leftarrow \mathcal{D}_\theta(I)$
 7:     $I_{\text{hard}}, I_{\text{easy}} \leftarrow \text{AFM}(I, \hat{I}\,|G_\phi)$
 8:     $\hat{I}_{\text{hard}} \leftarrow \mathcal{D}_\theta(I_{\text{hard}})$
 9:     $\hat{I}_{\text{easy}} \leftarrow \mathcal{D}_\theta(I_{\text{easy}})$
10:     Update $G_\phi$ by $\mathcal{L}_{\text{AFM}}$                ▷ Eq. (8)
11: end for
```

in $\mathcal{F}(I)$ and $\mathcal{F}(\hat{I})$, may be mixed with a different mixup ratio. As depicted in Figure 4(a), the combined input goes through an encoder-decoder architecture with skip connections based on UNet [20]. More details regarding the architecture can be found in the Appendix.

3.3.2 AFM-B: Band-wise Mask Generation

While AFM-E is an effective method to generate a mixup mask, we observe that masks generated by AFM-E naturally exhibit some level of rotational invariance, *i.e.* mask elements that correspond to the same frequency share similar mixup values. Thus, we introduce a simplified but powerful alternative: band-wise mask generation. As depicted in Figure 4(b), the network is constructed using few convolutional and fully-connected layers, which yields a set of N mixup values. The resulting mask is formed with N circular bands that are equally distributed along the polar axis. All elements within the area of a single band correspond to a mixup value produced by the network.

3.4. Training Procedure

In our training procedure, detailed in Algorithm 1, we concurrently train the denoising network and the AFM network. At each iteration, we employ a two-step approach. In the first step, we keep the mask generation network fixed and update the denoising network with the following loss:

$$\begin{aligned}\mathcal{L}_{\text{D}} &= -\text{PSNR}(\hat{I}, I_{\text{gt}}) - \lambda \text{PSNR}(\hat{I}_{\text{hard}}, I_{\text{gt}}) \\ &= \mathcal{L}_{\text{rec}} + \lambda \mathcal{L}_{\text{hard}},\end{aligned} \tag{9}$$

where λ is a hyperparameter. In the second step, we keep the denoising network fixed and update the AFM module with the adversarial loss function defined in Eq. (8).

It is worth noting that our AFM framework only affects the training process. At inference, we simply omit the AFM module and make predictions with the denoising network without any additional memory or computational costs.

4. Experiments

We apply our framework, AFM, to various image denoising architectures and evaluate on multiple out-of-distribution (OOD) benchmarks. Our results demonstrate that AFM significantly improves the OOD generalization of image denoising networks regardless of architecture (Table 1), and even outperforms state-of-the-art denoising networks in terms of OOD generalization (Table 2). Finally, we present some qualitative results.

4.1. Experimental Settings

Datasets Across all experiments, we train the denoising network exclusively on the Smartphone Image Denoising Dataset (SIDD) Medium [1] dataset. We measure the in-distribution (ID) performance on the SIDD validation set, and evaluate the out-of-distribution (OOD) performance across five real noise benchmarks to ensure a robust assessment across various data domains. The five OOD benchmarks include Poly [23], CC [16], HighISO [24], iPhone [9], and Huawei [9]. The image size of Poly,CC and HighISO is 512×512 and the image size of iPhone and Huawei is 1024×1024.

Training details We train all models for 200K iterations with a batch size of 32 and a training patch size of 256×256. For the denoising network and the mask generation network, we employ the AdamW [13] optimizer with an initial learning rate of 10^{-3}, which decreases to 10^{-6} following a cosine annealing schedule. Finally, we set $\lambda = 0.8$ and $\gamma = 0.3$.

4.2. Results

In Table 1, we compare the ID and OOD performance of existing denoising architectures with and without using the proposed framework. We test on four different architectures, including DnCNN [30], CBDNet [6], NAFNet [5], and MPRNet [27][2]. The results show that incorporating AFM consistently yields performance improvements across nearly all OOD benchmarks and architectures, while retaining competitiveness for ID performance. For example, integrating AFM leads to average OOD PSNR improvements of +0.75 dB, +0.33 dB, +0.54 dB, and +1.00 dB on DnCNN, CBDNet, MPRNet, and NAFNet, respectively. This demonstrates that AFM can be applied to various denoising networks to improve the network's generalization and robustness to unseen noise distributions.

[2]For large-scale architectures such as MPRNet and NAFNet, we reduce the number of blocks and channels to reduce training costs.

Table 1. Quantitative comparisons of denoising networks with and without our proposed AFM-B, on the SIDD [1] validation set (in-distribution) and other real noise benchmarks (out-of-distribution). We present performance in terms of PSNR↑ (dB) and SSIM (Structural Similarity Index Measure)↑. Dagger (†) denotes architectures that have been reimplemented with fewer model parameters.

		In-distribution	Out-of-distribution					
Architecture	Metric	SIDD [1]	Poly [23]	CC [16]	HighISO [24]	iPhone [9]	Huawei [9]	OOD Avg.
DnCNN [30]	PSNR	38.62	37.36	35.69	37.85	39.87	38.26	37.81
	SSIM	0.9501	0.9740	0.9755	0.9703	0.9688	0.9654	0.9708
DnCNN-Ours	PSNR	38.35	**37.75**	**36.84**	**39.17**	**40.65**	**38.39**	**38.56**
	SSIM	0.9478	**0.9804**	**0.9830**	**0.9801**	**0.9777**	**0.9683**	**0.9779**
CBDNet [6]	PSNR	38.35	37.83	36.25	38.18	**40.63**	38.35	38.25
	SSIM	0.9476	0.9809	0.9780	0.9722	**0.9759**	0.9656	0.9745
CBDNet-Ours	PSNR	39.27	**37.86**	**36.75**	**39.00**	40.51	**38.76**	**38.58**
	SSIM	0.9551	**0.9812**	**0.9843**	**0.9789**	0.9757	**0.9691**	**0.9778**
MPRNet† [27]	PSNR	39.55	37.54	35.96	38.01	40.41	38.17	38.02
	SSIM	0.9572	0.9795	0.9792	0.9753	**0.9771**	0.9683	0.9759
MPRNet†-Ours	PSNR	39.41	**37.92**	**36.56**	**39.11**	**40.62**	**38.60**	**38.56**
	SSIM	0.9568	**0.9807**	**0.9806**	**0.9788**	0.9765	**0.9689**	**0.9771**
NAFNet† [5]	PSNR	39.84	37.10	35.66	38.10	37.75	37.65	37.27
	SSIM	0.9592	0.9788	0.9807	0.9774	0.9111	0.9679	0.9631
NAFNet†-Ours	PSNR	39.81	**37.70**	**36.56**	**38.34**	**40.10**	**38.64**	**38.27**
	SSIM	0.9591	**0.9792**	**0.9823**	**0.9760**	**0.9723**	**0.9684**	**0.9756**

Table 2. Quantitative comparisons between our AFM-B and state-of-the-art supervised and self-supervised image denoising networks on the SIDD [1] validation set (in-distribution) and other real-noise benchmarks (out-of-distribution). We present performance in terms of PSNR↑ (dB) and SSIM↑, and also report the number of parameters and MACs at inference. The values of MACs are estimated by an input with the spatial size of 256×256. Networks marked with asterisk (*) are evaluated using official out-of-the-box models.

			In-distribution	Out-of-distribution							
	Architecture	Metric	SIDD [1]	Poly [23]	CC [16]	HighISO [24]	iPhone [9]	Huawei [9]	OOD Avg.	Params (M)	MACs (G)
Supervised	MIRNet-v2* [29]	PSNR	39.84	37.43	35.96	38.19	40.50	38.11	38.04	5.9	140.3
		SSIM	0.9593	0.9802	0.9798	0.9777	**0.9789**	**0.9685**	0.9770		
	Uformer* [22]	PSNR	39.89	37.48	36.02	38.14	40.31	38.37	38.06	50.9	89.5
		SSIM	0.9594	0.9794	0.9794	0.9763	0.9751	0.9684	0.9757		
	Restormer* [28]	PSNR	40.02	37.66	36.33	38.29	40.13	**38.42**	38.17	26.1	141.0
		SSIM	0.9603	0.9793	0.9807	0.9756	0.9734	0.9675	0.9753		
Self-supervised	R2R* [18]	PSNR	35.09	36.84	35.28	37.37	39.25	38.35	37.42	0.7	44.0
		SSIM	0.9154	0.9726	0.9758	0.9716	0.9614	0.9667	0.9696		
	AP-BSN* [11]	PSNR	35.49	36.28	33.51	37.48	39.76	36.26	36.66	3.1	216.2
		SSIM	0.9085	0.9735	0.9729	0.9740	0.9728	0.9499	0.9688		
	CVF-SID* [17]	PSNR	34.20	33.06	29.11	33.31	36.90	33.11	33.10	1.2	77.9
		SSIM	0.913	0.9531	0.9372	0.9521	0.9540	0.9270	0.9447		
Supervised	**DnCNN-Ours**	PSNR	38.35	**37.75**	**36.84**	**39.17**	**40.65**	38.39	**38.56**	0.7	43.8
		SSIM	0.9478	**0.9804**	**0.9830**	**0.9801**	0.9777	0.9683	**0.9779**		

We also conduct comparisons with state-of-the-art (SOTA) supervised denoising networks, such as MIRNet-v2 [26], Uformer [22], and Restormer [28], and present the results in Table 2. We evaluate all SOTA networks using the officially published weights, and compare these networks with a DnCNN architechture trained using the AFM-B framework. Remarkably, despite requiring less memory and computation at inference, DnCNN with AFM-B outperforms all SOTA models by at least +0.39 dB (Restormer) in terms of PSNR, and +0.009 (MIRNet-v2) in terms of SSIM.

We also compare the denoising performance with other SOTA dataset-based self-supervised methods. As shown in Table 2, models trained with a dataset-based self-supervised approach using the SIDD dataset show poor performance in both in-distribution and out-of-distribution scenarios, while our method significantly outperforms in both.

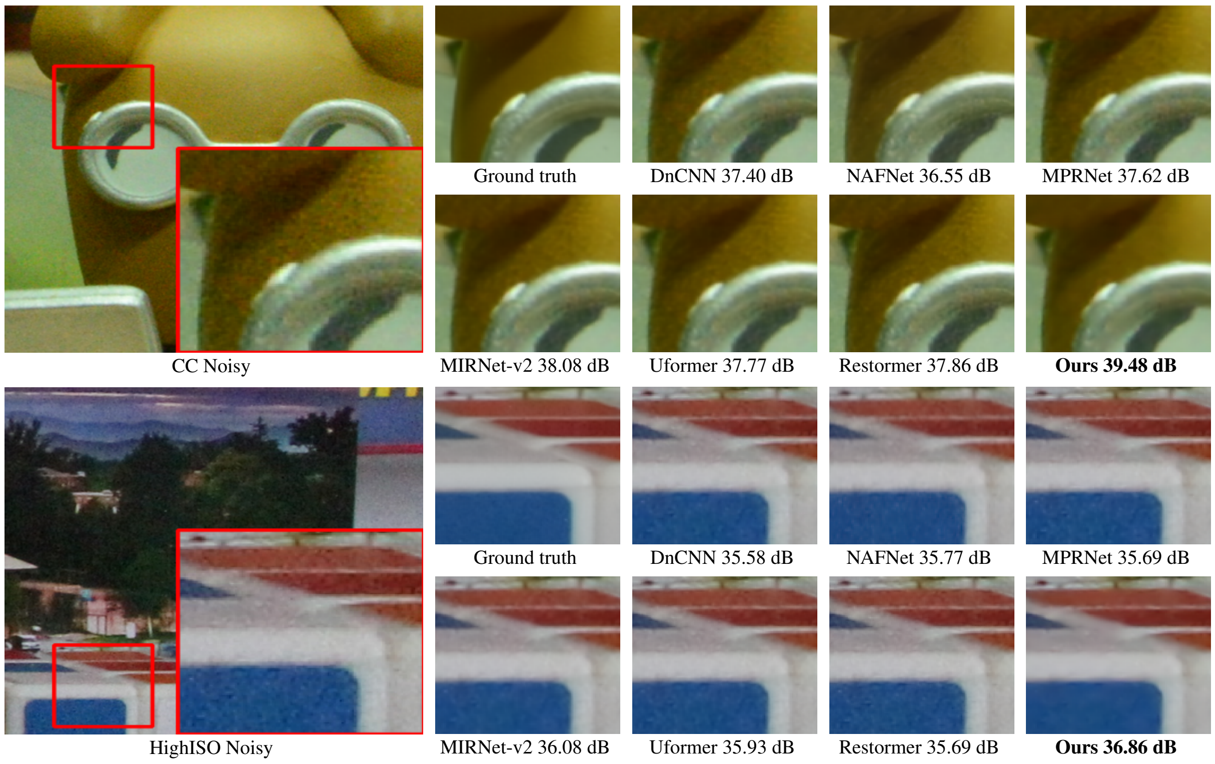


Figure 5. Comparison between the denoised outputs of various denoising networks including ours (DnCNN trained with AFM-B), on the out-of-distribution (OOD) datasets. DnCNN with AFM-B displays cleaner outputs compared to other networks that are trained without AFM.

Qualitative results Figure 5 visualizes the denoised outputs of DnCNN trained with AFM-B on samples of two OOD benchmarks: CC and HighISO. For comparison, we also visualize outputs of other denoising networks, trained without AFM. These qualitative comparisons clearly show that the models trained with AFM produce cleaner outputs compared to those trained without AFM.

5. Analysis and Discussions

We analyse the effectiveness of AFM when compared with other generalization methods and provide some discussions.

5.1. Comparison with Generalization Techniques

We compare our AFM framework with a few other techniques that aim to improve generalization. We briefly summarize each technique:

- **Dropout** [8] randomly drops features along the channel dimension before the last convolution layer of the network.
- **Input mask** [3] randomly masks the features after the first convolutional layer across the spatial dimension.
- **CutMix** [25] is a common data augmentation scheme to create novel data samples by cutting and pasting patches from a different image within the training dataset.
- **Adversarial training** first generates adversarial perturbations in the input image using the PGD attack [14], then trains the network to denoise the adversarial perturbations.
- **Random frequency mixup** is identical to AFM-B, except that the mixup value for each frequency band is sampled from a uniform distribution.
- **ASM-E (Adversarial Spatial Mixup)** is identical to AFM-E, except that the mixup is conducted in the spatial domain instead of the frequency domain.

We train a DnCNN network with each of the techniques mentioned above, and present the ID and average OOD results in Table 3. We first observe that Dropout and Input mask do not improve generalization performance, evidenced by a drop of average OOD performance. In addition, we notice that CutMix shows minor gains and is outperformed by both AFM-E and AFM-B, which exhibit significant improvements in OOD performance.

An intriguing observation is made in the results of adversarial training. Despite sharing some similarities with our AFM framework, we find that a naive adversarial training using the PGD attack fails to improve robustness to OOD images (-0.63 dB). We hypothesize that the difference lies in how the noise is created; while the PGD attack gener-

Table 3. Comparing the performance of AFM-E and AFM-B with other conventional generalization methods on the DnCNN architecture. We present performance in terms of PSNR↑ (dB) and SSIM↑. The full table with results on individual OOD benchmarks are available in the Appendix.

Algorithm	Metric	SIDD [1]	OOD Avg.
Normal Training	PSNR	38.62	37.81
	SSIM	0.9501	0.9708
Dropout [8]	PSNR	37.27	37.57
	SSIM	0.9294	0.9697
Input mask [3]	PSNR	37.83	37.28
	SSIM	0.9441	0.9731
CutMix [25]	PSNR	38.59	37.96
	SSIM	0.9500	0.9719
Adversarial Training	PSNR	38.43	37.18
	SSIM	0.9483	0.9668
Random Freq. Mixup	PSNR	38.53	38.17
	SSIM	0.9493	0.9714
ASM-E	PSNR	38.46	37.92
	SSIM	0.9490	0.9717
AFM-E (Ours)	PSNR	38.41	<u>38.51</u>
	SSIM	0.9485	<u>0.9774</u>
AFM-B (Ours)	PSNR	38.35	**38.56**
	SSIM	0.9478	**0.9779**

ates a *synthetic* adversarial noise, our AFM is designed to generate *realistic* adversarial noise. This implies that noise augmentation to maintain realistic properties plays a vital role in improving generalization performance. Additionally, random frequency mixup leads to a modest improvement in OOD performance (+0.36 dB), but not as much as AFM-E (+0.70 dB) or AFM-B (+0.75 dB), which highlights the importance of the adversarial mask generation network in our AFM module. Last, ASM-E fails to improve the generalization performance, which reinforces the rationale for employing mixup in the frequency domain.

5.2. Discussions

Frequency analysis Figure 6 visualizes an example of the adversarial mask $\boldsymbol{m}$, as well as the frequency magnitudes of the corresponding images, I_{hard}, $\hat{I}_{\text{hard}}$, and $\hat{I}$, from the training dataset. Since the input is an in-distribution sample, the denoised image $\hat{I}$ displays clean frequencies. However, the denoising network struggles to denoise I_{hard}; in fact, $|\mathcal{F}(\hat{I}_{\text{hard}})|$ in Figure 6 bears strong resemblance to the output frequency in Figure 2, which is sampled from an OOD dataset. This implies that our AFM module constructs training images that share many characteristics with OOD images.

Generalization in image denoising The results in Table 2 sheds light on the lack of generalization to unseen real-noise distributions, even in the best performing denoising

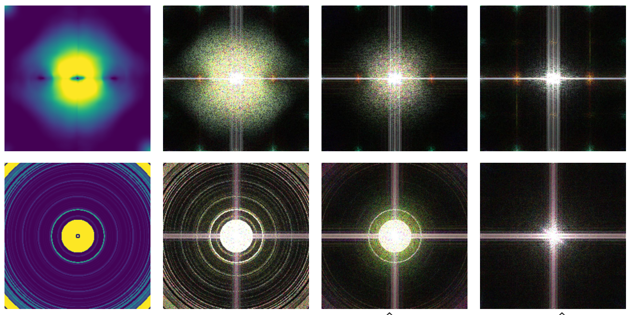


Figure 6. Visualizing the mixup mask $\boldsymbol{m}$, as well as the frequency magnitudes of I_{hard}, $\hat{I}_{\text{hard}}$, and $\hat{I}$. The top row presents the results obtained by AFM-E, while the bottom row shows those by AFM-B. In both outcomes, it is observable that $\hat{I}_{\text{hard}}$ retains more noise in the frequency domain compared to $\hat{I}$.

networks, *e.g.* Restormer. This raises a significant concern; generalization to varying real-noise distributions is often overlooked, despite the fact that generalization is a necessity for the widespread use of image denoising networks. Thus, we hope our work will inspire future image denoising research to explore this direction as well, instead of focusing exclusively on improving in-distribution performance.

6. Conclusion

We proposed Adversarial Frequency Mixup (AFM), a novel training framework for image denoising networks that facilitates better generalization and robustness to diverse real-world noise distributions. By leveraging an adversarial mixup in the frequency domain, the AFM module generates new noisy images that retain the properties of noise encountered in the real-world. Then, these images are integrated into the training dataset to learn improved image denoising networks with the out-of-distribution robustness. Notably, image denoisers trained with our AFM framework exhibited significantly superior generalization capabilities over those trained with conventional supervised training methods. Finally, we identified that the lack of robustness in image denoising networks is an essential yet often overlooked area of research, and hope that future work will pursue this direction actively.

Acknowledgements This work was partly supported by Samsung Electronics Co., Ltd and Samsung Advanced Institute of Technology. It was also partly supported by the Bio & Medical Technology Development Program of the National Research Foundation (NRF) [No. 2021M3A9E4080782] and Institute of Information & communications Technology Planning & Evaluation (IITP) grant [No. 2021-0-01343, No. 2021-0-02068], funded by the Korea government (MSIT).

References

[1] Abdelrahman Abdelhamed, Stephen Lin, and Michael S Brown. A high-quality denoising dataset for smartphone cameras. In *CVPR*, pages 1692–1700, 2018. 1, 2, 3, 5, 6, 8

[2] Joshua Batson and Loic Royer. Noise2self: Blind denoising by self-supervision. In *ICML*, pages 524–533. PMLR, 2019. 2

[3] 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. 3, 7, 8

[4] Liangyu Chen, Xin Lu, Jie Zhang, Xiaojie Chu, and Chengpeng Chen. Hinet: Half instance normalization network for image restoration. In *CVPR*, pages 182–192, 2021. 2

[5] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. In *ECCV*, pages 17–33, 2022. 1, 2, 5, 6

[6] Shi Guo, Zifei Yan, Kai Zhang, Wangmeng Zuo, and Lei Zhang. Toward convolutional blind denoising of real photographs. In *CVPR*, pages 1712–1722, 2019. 1, 5, 6

[7] 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. 2

[8] Xiangtao Kong, Xina Liu, Jinjin Gu, Yu Qiao, and Chao Dong. Reflash dropout in image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6002–6012, 2022. 7, 8

[9] Zhaoming Kong, Fangxi Deng, Haomin Zhuang, Jun Yu, Lifang He, and Xiaowei Yang. A comparison of image denoising methods, 2023. 5, 6

[10] Alexander Krull, Tim-Oliver Buchholz, and Florian Jug. Noise2void-learning denoising from single noisy images. In *CVPR*, pages 2129–2137, 2019. 1, 2

[11] 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. 6

[12] Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. Noise2noise: Learning image restoration without clean data. *arXiv preprint arXiv:1803.04189*, 2018. 1, 2

[13] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 5

[14] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 7

[15] Sreyas Mohan, Zahra Kadkhodaie, Eero P. Simoncelli, and Carlos Fernandez-Granda. Robust and interpretable blind image denoising via bias-free convolutional neural networks. In *ICLR*, 2020. 3

[16] Seonghyeon Nam, Youngbae Hwang, Yasuyuki Matsushita, and Seon Joo Kim. A holistic approach to cross-channel image noise modeling and its application to image denoising. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1683–1691, 2016. 1, 5, 6

[17] Reyhaneh Neshatavar, Mohsen Yavartanoo, Sanghyun Son, and Kyoung Mu Lee. Cvf-sid: Cyclic multi-variate 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. 6

[18] Tongyao Pang, Huan Zheng, Yuhui Quan, and Hui Ji. Recorrupted-to-recorrupted: unsupervised deep learning for image denoising. In *CVPR*, pages 2043–2052, 2021. 1, 2, 6

[19] 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. 2

[20] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, pages 234–241. Springer, 2015. 5

[21] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9446–9454, 2018. 2

[22] Zhendong Wang, Xiaodong Cun, Jianmin Bao, Wengang Zhou, Jianzhuang Liu, and Houqiang Li. Uformer: A general u-shaped transformer for image restoration. In *CVPR*, pages 17683–17693, 2022. 1, 2, 6

[23] 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. 1, 2, 5, 6

[24] Huanjing Yue, Jianjun Liu, Jingyu Yang, Truong Q Nguyen, and Feng Wu. High iso jpeg image denoising by deep fusion of collaborative and convolutional filtering. *IEEE Transactions on Image Processing*, 28(9):4339–4353, 2019. 5, 6

[25] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019. 7, 8

[26] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Learning enriched features for real image restoration and enhancement. In *ECCV*, pages 492–511. Springer, 2020. 1, 2, 6

[27] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. In *CVPR*, pages 14821–14831, 2021. 2, 5, 6

[28] 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 *CVPR*, pages 5728–5739, 2022. 1, 2, 6

[29] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Learning enriched features for fast image restoration and enhancement. *IEEE transactions on pattern analysis and machine intelligence*, 45(2):1934–1948, 2022. 6

[30] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, 26(7):3142–3155, 2017. 1, 2, 5, 6

[31] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, 2018. 1