GAN2GAN: GENERATIVE NOISE LEARNING FOR BLIND DENOISING WITH SINGLE NOISY IMAGES

Sungmin Cha¹, Taeeon Park¹, Byeongjoon Kim², Jongduk Baek² and Taesup Moon³∗
Sungkyunkwan University¹, Yonsei University², Seoul National University³, South Korea
{csm9493, pte1236}@skku.edu, bjkim2006@naver.com, jongdukaekeb@yonsei.ac.kr, tsmoon@snu.ac.kr

ABSTRACT

We tackle a challenging blind image denoising problem, in which only single distinct noisy images are available for training a denoiser, and no information about noise is known, except for it being zero-mean, additive, and independent of the clean image. In such a setting, which often occurs in practice, it is not possible to train a denoiser with the standard discriminative training or with the recently developed Noise2Noise (N2N) training; the former requires the underlying clean image for the given noisy image, and the latter requires two independently realized noisy image pairs for a clean image. To that end, we propose GAN2GAN (Generated-Artificial-Noise to Generated-Artificial-Noise) method that first learns a generative model that can 1) simulate the noise in the given noisy images and 2) generate a rough, noisy estimates of the clean images, then 3) iteratively trains a denoiser with subsequently synthesized noisy image pairs (as in N2N), obtained from the generative model. In results, we show the denoiser trained with our GAN2GAN achieves an impressive denoising performance on both synthetic and real-world datasets for the blind denoising setting; it almost approaches the performance of the standard discriminatively-trained or N2N-trained models that have more information than ours, and it significantly outperforms the recent baseline for the same setting, e.g., Noise2Void, and a more conventional yet strong one, BM3D. The official code of our method is available at https://github.com/csm9493/GAN2GAN.

1 INTRODUCTION

Image denoising is one of the oldest problems in image processing and low-level computer vision, yet it still attracts lots of attention due to the fundamental nature of the problem. A vast number of algorithms have been proposed over the past several decades, and recently, the CNN-based methods, e.g., Cha & Moon (2019); Zhang et al. (2017); Tai et al. (2017); Liu et al. (2018), became the throne-holders in terms of the PSNR performance. The main approach of the most CNN-based denoisers is to apply the discriminative learning framework with (clean, noisy) image pairs and known noise distribution assumption. While being effective, such framework also possesses a couple of limitations that become critical in practice; the assumed noise distribution may be mismatched to the actual noise in the data or obtaining the noise-free clean target images is not always possible or very expensive, e.g., medical imaging (CT or MRI) or astrophotographs.

Several attempts have been made to resolve above issues. For the noise uncertainty, the so-called blind training have been proposed. Namely, a denoiser can be trained with a composite training set that contains images corrupted with multiple, pre-defined noise levels or distributions, and such blindly trained denoisers, e.g., DnCNN-B in Zhang et al. (2017), were shown to alleviate the mismatch scenarios to some extent. However, the second limitation, i.e., the requirement of clean images for building the training set, still remains. As an attempt to address this second limitation, Lehtinen et al. (2018) recently proposed the Noise2Noise (N2N) method. It has been shown that a denoiser, which has a negligible performance loss, can be trained without the clean target images, as long as two independent noisy image realizations for the same underlying clean image are available. Despite its

∗Corresponding author (E-mail: tsmoon@snu.ac.kr)
effectiveness, the requirement of the two independently realized noisy image pair for a single clean image, which may hardly be available in practice, is a critical limiting factor for N2N.

In this paper, we consider a setting in which neither of above approach is applicable, namely, the pure unsupervised blind denoising setting where only single distinct noisy images are available for training. Namely, nothing is known about the noise other than it being zero-mean, additive, and independent of the clean image, and neither the clean target images for blind training nor the noisy image pairs for N2N training is available. While some recent work, e.g., Krull et al. (2019); Batson & Royer (2019); Laine et al. (2019), took the self-supervised learning (SSL) approach for the same setting, we take a generative learning approach. The crux of our method is to first learn a Wasserstein GAN (Arjovsky et al., 2017)-based generative model that can 1) learn and simulate the noise in the given noisy images and 2) generate rough, initially denoised images. Using such generative model, we then synthesize noisy image pairs by corrupting each of the initially denoised images with the simulated noise twice and use them to train a CNN denoiser as in the N2N training (i.e., Noisy N2N). We further show that iterative N2N training with refined denoised images can significantly improve the final denoising performance. We dubbed our method as GAN2GAN (Generated-Artifical-Noise to Generated-Artificial-Noise) and show that the denoiser trained with our method can achieve (sometimes, even outperform) the performance of the standard supervised-trained or N2N-trained blind denoisers for the white Gaussian noise case. Furthermore, for mixture/correlated noise or real-world noise in microscopy/CT images, for which the exact distributions are hard to know a priori, we show our denoiser significantly outperforms those standard blind denoisers, which are mismatch-trained with white Gaussian noise, as well as other baselines that operate in the same condition as ours: the SSL baseline, N2V (Krull et al., 2019), and a more conventional BM3D (Dabov et al., 2007).

2 Related Work

Several works have been proposed to overcome the limitation of the vanilla supervised learning based denoising. As mentioned above, Noise2Self (N2S) (Batson & Royer, 2019) and Noise2Void (N2V) (Krull et al., 2019) recently applied self-supervised learning (SSL) approach to train a denoiser only with single noisy images. Their settings exactly coincide with ours, but we show later that our GAN2GAN significantly outperforms them. More recently, Laine et al. (2019) improved N2V by incorporating specific noise likelihood models with Bayesian framework, however, their method required to know the exact noise model and could not be applied to more general, unknown noise settings. Similarly, Soltanayev & Chun (2018) proposed SURE (Stein’s Unbiased Risk Estimator)-based denoiser that can also be trained with single noisy images, but it worked only with the Gaussian noise. Their work was extended in Zhussip et al. (2019), but it required noisy image pairs as in N2N as well as the Gaussian noise constraint. Chen et al. (2018) devised GCBD method to learn and generate noise in the given noisy images using W-GAN Arjovsky et al. (2017) and utilized the unpaired clean images to build a supervised training set. Our GAN2GAN is related to Chen et al. (2018), but we significantly improve their noise learning step and do not use the clean data at all. Table 1 summarizes and compares the settings among the above mentioned recent baselines. We clearly see that only our GAN2GAN and N2V do not utilize any “sidekicks” that other methods use.

Table 1: Summary of different settings among the recent baselines.

| Alg. | Requirements | Clean image | Noisy pairs | Noise model |
|------|---------------|-------------|-------------|-------------|
| N2N  | [Lehtinen et al. (2018)] | X | ✓ | X |
| HQ SSL [Laine et al. (2019)] | X | ✓ | ✓ |
| SURE [Soltanayev & Chun (2018)] | X | X | ✓ |
| Ext. SURE [Zhussip et al. (2019)] | X | ✓ | ✓ |
| GCBD [Chen et al. (2018)] | ✓ | ✓ | X |
| N2V  [Krull et al. (2019)] | ✓ | ✓ | ✓ |
| GAN2GAN (Ours) | X | X | X |

Additionally, there are recently published papers on blind image denoising but these also have a difference with ours. Anwar & Barnes (2019); Zhang et al. (2018) suggest effective CNN architectures for denoising, however, they only consider the setting in which clean images are necessary for training. Zamir et al. (2020) considers the denoising of specific camera settings, and it also requires clean sRGB images as well as the knowledge of the noise level. Thus, it cannot be applied to the complete blind setting as ours, in which no information on the specific noise distribution or clean images is available.
More classical denoising methods are capable of denoising solely based on the single noisy images by applying various principles, e.g., filtering-based Buades et al. (2005); Dabov et al. (2007), optimization-based Elad & Aharon (2006); Mairal et al. (2009), Wavelet-based Donoho & Johnstone (1995), and effective prior-based Zoran & Weiss (2011). Those methods typically are, however, computationally intensive during the inference time and cannot be trained from a separate set of noisy images, which limits their denoising performance. Another line of recent work worth mentioning is the deep learning-based priors or regularizers, e.g., Ulyanov et al. (2018); Yeh et al. (2018); Lunz et al. (2018), but their PSNRs still fell short of the supervised trained CNN-based denoisers.

3 Motivation

In order to develop the core intuition for motivating our method, we first consider a simple, single-letter Gaussian noise setting. Let \( Z = X + N \) be the noisy observation of \( X \sim \mathcal{N}(0, \sigma_X^2) \), corrupted by the \( N \sim \mathcal{N}(0, \sigma_N^2) \). It is well known that the minimum MSE (MMSE) estimator of \( X \) given \( Z \) is \( f_{\text{MMSE}}(Z) = \mathbb{E}(X|Z) = \frac{\sigma_X^2}{\sigma_X^2 + \sigma_N^2} Z \). We now identify the optimality of N2N in this setting.

**N2N** Assume that we have two i.i.d. copies of the noise \( N \): \( N_1 \) and \( N_2 \). Then, let \( Z_1 = X + N_1 \) and \( Z_2 = X + N_2 \) be the two independent noisy observation pairs of \( X \). The N2N in this setting corresponds to obtaining the MMSE estimator of \( Z_2 \) given \( Z_1 \),

\[
f_{\text{N2N}}(Z_1) \triangleq \arg\min_f \mathbb{E}(Z_2 - f(Z_1))^2 = \mathbb{E}(Z_2|Z_1) = \mathbb{E}(X + N_2|Z_1) \overset{(a)}{=} \mathbb{E}(X|Z_1) = \frac{\sigma_X^2}{\sigma_X^2 + \sigma_N^2} Z_1,
\]

in which (a) follows from \( N_2 \) being independent of \( Z_1 \). Note (1) has the exact same form as \( f_{\text{MMSE}}(Z) \), hence, estimating \( X \) with \( f_{\text{N2N}}(Z) \) also achieves the MMSE, in line with (Lehtinen et al., 2018).

**"Noisy" N2N** Now, consider the case in which we again have the two i.i.d. \( N_1 \) and \( N_2 \), but the noisy observations are of a *noisy* version of \( X \). Namely, let \( X' = X + N_0 \), in which \( N_0 \sim \mathcal{N}(0, \sigma_0^2) \), and denote \( Z_1' = X' + N_1 \) and \( Z_2' = X' + N_2 \) as the noisy observation pairs. Then, we can define a "Noisy" N2N estimator as the MMSE estimator of \( Z_2' \) given \( Z_1' \),

\[
f_{\text{Noisy N2N}}(Z_1', y) \triangleq \arg\min_f \mathbb{E}(Z_2' - f(Z_1'))^2 = \mathbb{E}(X'|Z_1') = \frac{\sigma_X^2(1 + y)}{\sigma_X^2(1 + y) + \sigma_N^2} Z_1',
\]

in which we denote \( y \triangleq \sigma_0^2/\sigma_X^2 \) and assume that \( 0 < y < 1 \). Note clearly (2) coincides with (1) when \( y = \sigma_0^2 = 0 \). Following N2N, (2) is essentially estimating \( X' \) based on \( Z' = X' + N \). An interesting subtle question is what happens when we use the mapping \( f_{\text{Noisy N2N}}(Z, y) \) for estimating \( X \) given \( Z = X + N \), not \( X' \) given \( Z' \). Our theorem below, of which proof is in the Supplementary Material (S.M.), shows that for a sufficiently large \( \sigma_0^2 \), \( f_{\text{Noisy N2N}}(Z, y) \) gives a better estimate of \( X \) than \( X' \).

**Theorem 1** Consider the single-letter Gaussian setting and \( f_{\text{Noisy N2N}}(Z, y) \) obtained in (2). Also, assume \( 0 < y < 1 \). Then, there exists some \( y_0 \) s.t. \( \forall y \in (y_0, 1) \), \( \mathbb{E}(X - f_{\text{Noisy N2N}}(Z, y))^2 < \sigma_0^2 \).

Theorem 1 provides a simple, but useful, intuition that motivates our method; if simulating the noise in the images is possible, we may carry out the N2N training iteratively, provided that a rough noisy estimate of the clean image is initially available. Namely, we can first simulate the noise to generate noisy observation pairs of the initial noisy estimate, then do the Noisy N2N training with them to obtain a denoiser that may result in a better estimate of the clean image when applied to the actual noisy image subject to denoising (as in Theorem 1). Then, we can refine the estimates by *iterating* the Noisy N2N training with the generated noisy observation pairs of the previous step’s estimate of the clean image, until convergence.

To check whether above intuition is valid, we carry out a feasibility experiment. Figure 1 shows the denoising results on BSD68 (Roth & Black, 2009) for Gaussian noise with \( \sigma = 25 \). The blue line is the PSNR of the N2N
model trained with noisy observation pairs of the clean images in the BSD training set, serving as an upper bound. The orange line, in contrast, is the PSNR of the Noisy N2N model that is trained with the noisy observation pairs of the noisy images, which were set to be another Gaussian noise-corrupted training images. The standard deviations (σ₀) of the Gaussian for generating the noisy estimates are given in the horizontal axis, and the corresponding PSNRs of the estimates are given in the parentheses. Although Noisy N2N clearly lies much lower than the N2N upper bound, we note its PSNR is still higher than that of the initial noisy estimates, which is in line with Theorem 1. Now, if we iterate the Noisy N2N with the previous step’s denoised images (i.e., Noisy-N2N2/Noisy-N2N3 for second/third iterations, respectively), we observe that the PSNR significantly improves and approaches the ordinary N2N for most of the initial σ₀ values. Thus, we observe the intuition from Theorem 1 generalizes well to the image denoising case in an ideal setting, where the noise can be perfectly simulated, and the initial noisy estimates are Gaussian corrupted versions. The remaining question is whether we can also obtain similar results for the blind image denoising setting. We show our generative model-based approach in details in the next section.

4 MAIN METHOD: THREE COMPONENTS OF GAN2GAN

To concretely describe our method, we first set the notations. We assume the noisy image Z is generated by \( Z = x + N \), in which \( x \) denotes the underlying clean image and \( N \) denotes the zero-mean, additive noise that is independent of \( x \). For training a denoiser, we do not assume either the distribution or the covariance of \( N \) is known. Moreover, we assume only a database of \( n \) distinct noisy images, \( D = \{Z^{(i)}\}_{i=1}^n \), is available for learning a denoiser. A CNN-based denoiser is denoted as \( \hat{X}_\phi(Z) \) with \( \phi \) being the model parameter, and we use the standard quality metrics, PSNR/SSIM, for evaluation. Our method consists of three parts; 1) smooth noisy patch extraction, 2) training a generative model, and 3) iterative GAN2GAN training of \( \hat{X}_\phi(Z) \), each of which we elaborate below.

4.1 SMOOTH NOISY PATCH EXTRACTION

The first step is to extract the noisy image patches from \( D \) that correspond to smooth, homogeneous areas. Our extraction method is similar to that of the GCBD proposed in (Chen et al., 2018), but we make a critical improvement. The GCBD determines a patch \( p \) of (pre-determined size) is smooth if it satisfies the following for all of its smaller sub-patches, \( q_j \), with some hyperparameters \( \mu, \gamma \in (0, 1) \):

\[
|\mathbb{E}(q_j) − \mathbb{E}(p)| ≤ \mu \mathbb{E}(p), \quad |\mathbb{V}(q_j) − \mathbb{V}(p)| ≤ \gamma \mathbb{V}(p),
\]

in which \( \mathbb{E}(\cdot) \) and \( \mathbb{V}(\cdot) \) are the empirical mean and variance of the pixel values.

![Figure 2: Comparison of smooth noisy patch extraction rules.](image)

While (3) works for extracting smooth patches to some extent, as we show in Figure 2(b), it does not rule out choosing patches with high-frequency repeating patterns, which are far from being smooth. Thus, we instead use the 2D discrete wavelet transform (DWT) for a new extraction rule; namely, we determine \( p \) is smooth if its four sub-band decompositions obtained by DWT, \( \{W_k(p)\}_{k=1}^4 \), satisfy

\[
\frac{1}{4} \sum_{k=1}^4 \left| \hat{\sigma}(W_k(p)) - \mathbb{E}[\hat{\sigma}_W(p)] \right| ≤ \lambda \mathbb{E}[\hat{\sigma}_W(p)],
\]

in which \( \hat{\sigma}(\cdot) \) is the empirical standard deviation of the wavelet coefficients, \( \mathbb{E}[\hat{\sigma}_W(p)] \triangleq \frac{1}{4} \sum_{k=1}^4 \hat{\sigma}(W_k(p)) \), and \( \lambda \in (0, 1) \) is a hyperparameter. This rule is much simpler than (3), which

...
Figure 3: Overall structure of the W-GAN based generative model.

has to be evaluated for all the sub-patches, \( \{ q_i \} \). Once \( N \) patches are extracted from \( D \) using (4), we subtract each patch with its mean pixel value, and obtain a set of “noise” patches, \( \mathcal{N} = \{ n^{(j)} \}_{j=1}^N \). Such subtraction is valid since all the pixel values should be close to their mean in a smooth patch, and the noise is assumed to be zero-mean, additive.

Figure 2 compares the rules (3) and (4) by showing the quality of the “noise” patches extracted from 1,000 Gaussian-corrupted images. The two plots in Figure 2(a) show the normalized histograms of the empirical standard deviations, \( \hat{\sigma} \), of the extracted patches when true \( \sigma = \{ 25, 50 \} \). We clearly observe that while the \( \hat{\sigma} \)’s for (4) are mostly concentrated on true \( \sigma \), those of (3) have much higher variation. In addition, Figure 2(b) visualizes the randomly sampled patches of which \( \hat{\sigma} \)’s were above the 90-th percentile among the extracted patches for each rule (when \( \sigma = 25 \)). Again, it is obvious that (3) also may result in selecting the patches with high-frequency patterns, whereas (4) is much more effective for extracting accurate noise patches. Later, we show (in Figure 5) that such improved quality of the noise patches by our (4) plays an essential role; namely, our pure unsupervised learning based denoiser using (4) even outperforms the clean target image based denoiser in (Chen et al., 2018) using (3).

4.2 Training a W-GAN Based Generative Model

Equipped with \( D = \{ Z^{(i)} \}_{i=1}^n \) and the extracted noise patches \( \mathcal{N} = \{ n^{(j)} \}_{j=1}^N \), we train a generative model, which can learn and simulate the noise as well as generate initial noisy estimates of the clean images, hence, realize the Noisy N2N training explained in Section 3. As shown in Figure 3, our model has three generators, \( \{ g_{w_1}, g_{w_2}, g_{w_3} \} \), and two critics, \( \{ f_{w_1}, f_{w_2} \} \), in which the subscripts stand for the model parameters. The loss functions associated with the components of our model are:

\[
\mathcal{L}_n(\theta_1, w_1) \triangleq \mathbb{E}_n \left[ f_{w_1}(n) \right] - \mathbb{E}_r \left[ f_{w_1}(g_{\theta_1}(r)) \right]
\]

(5)

\[
\mathcal{L}_Z(\theta_1, \theta_2, w_2) \triangleq \mathbb{E}_Z \left[ f_{w_2}(Z) \right] - \mathbb{E}_{Z,r} \left[ f_{w_2}(g_{\theta_2}(Z) + g_{\theta_1}(r)) \right]
\]

(6)

\[
\mathcal{L}_{\text{cyc}}(\theta_2, \theta_3) \triangleq \mathbb{E}_Z \left[ \| z - g_{\theta_3}(g_{\theta_2}(Z)) \|_1 \right].
\]

(7)

The loss (5) is a standard W-GAN (Arjovsky et al., 2017) loss for training the first generator-critic pair, \( (g_{w_1}, \ f_{w_1}) \), of which \( g_{w_1} \) learns to generate the independent realization of the noise mimicking the patches in \( \mathcal{N} = \{ n^{(j)} \}_{j=1}^N \), taking the random vector \( r \sim \mathcal{N}(0, I) \) as input. The second loss (6) links the two generators, \( g_{w_1} \) and \( g_{w_2} \), with the second critic, \( f_{w_2} \). The second generator \( g_{w_2} \) is intended to generate the estimate of the underlying clean patch for \( Z \), i.e., coarsely denoise \( Z \), and the critic \( f_{w_2} \) determines how close the distribution of the generated noisy image, \( g_{w_2}(Z) + g_{\theta_1}(r) \), is to the that of \( Z \). Our intuition is, if \( g_{w_1} \) can realistically simulate the noise, then enforcing \( g_{w_2}(Z) + g_{\theta_1}(r) \) to mimick \( Z \) would result in learning a reasonable initial denoiser \( g_{w_2} \). One important detail regarding \( g_{w_2} \) is its final activation must be the sigmoid function for stable training. The third loss (7), which

\footnote{We assume \( g_{w_2} \) implicitly has the cropping step for \( Z \) such that the dimension of \( g_{w_2}(Z) \) and \( g_{\theta_1}(r) \) match.}
resembles the cycle loss in (Zhu et al., 2017), imposes the encoder-decoder structure between \( g_{\theta_2} \) and \( g_{\theta_1} \), hence, helps \( g_{\theta_2} \) to compress the most redundant part of \( Z \), i.e., the noise, and carry out the initial denoising. Once the losses are defined, training the generators and critics are done in an alternating manner, as in the training of W-GAN (Arjovsky et al., 2017), to approximately solve
\[
\min_{\theta_1, \theta_2, \theta_3} \max_{w_1, w_2} \left[ \alpha L_n(\theta_1, w_1) + \beta L_Z(\theta_1, \theta_2, w_2) + \gamma L_{cyc}(\theta_2, \theta_3) \right],
\]
in which \( (\alpha, \beta, \gamma) \) are hyperparameters to control the trade-offs between the loss functions. The pseudo algorithm for training a generative model is given in Algorithm 1. There are a couple of subtle points for training with the overall objective (8), and we describe the full details on model architectures and hyperparameters in the S.M.

Algorithm 1 Training a generative model, all experiments in this paper used the defaults values, \( n_{critic} = 5, n_{epoch} = 30, m = 64, \alpha_g = 4e^{-4}, \alpha_{critic} = 5e^{-5}, \alpha = 5, \beta = 1, \gamma = 10 \)

1: Require \( \mathcal{D}, \lambda \)
2: \( \mathcal{N} \leftarrow \text{NoisePatchExtraction}(\mathcal{D}, \lambda) \)
3: for \( e_{\text{PGAN}} \leftarrow 1, n_{\text{epoch}} \) do
4: \( \{n(i)\}_{i=1}^m \sim \mathcal{N}, \{r(i)\}_{i=1}^m \sim N(0, I), \{Z(i)\}_{i=1}^m \sim \mathcal{D} \)
5: for \( e_{\text{critic}} \leftarrow 1, n_{\text{critic}} \) do
6: \( g_{\theta_1} \leftarrow \nabla_w L[Z(\theta_1, w_1)], g_{\theta_2} \leftarrow \nabla_w [L_Z(\theta_1, \theta_2, w_2)] \)
7: \( w_1 \leftarrow \text{Clip}(w_1 + \alpha_{\text{critic}} \cdot \text{Adam}(w_1, g_{\theta_1}, -c, c)) \)
8: \( w_2 \leftarrow \text{Clip}(w_2 + \alpha_{\text{critic}} \cdot \text{Adam}(w_2, g_{\theta_2}, -c, c)) \)
9: end for
10: \( g_{\theta_1}, g_{\theta_2}, g_{\theta_3} \leftarrow \nabla_{\theta_1, \theta_2, \theta_3} [\alpha L_n(\theta_1, w_1) + \beta L_Z(\theta_1, \theta_2, w_2) + \gamma L_{cyc}(\theta_2, \theta_3)] \)
11: \( \theta_1 \leftarrow \theta_1 - \alpha_g \cdot \text{Adam}(\theta_1, g_{\theta_1}), \theta_2 \leftarrow \theta_2 - \alpha_g \cdot \text{Adam}(\theta_2, g_{\theta_2}), \theta_3 \leftarrow \theta_3 - \alpha_g \cdot \text{Adam}(\theta_3, g_{\theta_3}) \)
12: end for
13: return \( \theta_1, \theta_2 \)

4.3 Iterative GAN2GAN Training of a Denoiser

With our generative model, we then carry out the iterative Noisy N2N training as described in Section 3, with the generated noisy images. Namely, given each \( Z(i) \in \mathcal{D} \), we generate the pair
\[
(Z_{11}^{(i)}, Z_{12}^{(i)}) \triangleq (g_{\theta_2}(Z(i)) + g_{\theta_1}(r_{11}^{(i)}), g_{\theta_2}(Z(i)) + g_{\theta_1}(r_{12}^{(i)})),
\]
in which \( r_{11}^{(i)}, r_{12}^{(i)} \in \mathbb{R}^{128} \) are i.i.d. \( N(0, I) \). In contrast to the ideal case in Section 3, each generated image in (9) is a noise-corrupted version of \( g_{\theta_2}(Z(i)) \), in which the corruption is done by the simulated noise \( g_{\theta_1}(r) \). Denoting the set of such pairs as \( \mathcal{D}_1 = \{(Z_{11}^{(i)}, Z_{12}^{(i)})\}_{i=1}^n \), a denoiser \( \hat{X}_\phi(Z) \) is trained by minimizing \( L_{\text{G2G}}(\phi, \mathcal{D}_1) \triangleq \frac{1}{n} \sum_{i=1}^n (Z_{11}^{(i)} - \hat{X}_\phi(Z_{12}^{(i)}))^2 \). In \( L_{\text{G2G}}(\cdot) \), we only use the generated noisy images and do not use the actual observed \( Z(i) \), hence, we dubbed our training as GAN2GAN (G2G) training. Now, denoting the learned denoiser as \( G2G_1 \) (with parameter \( \phi_1 \)), we can iterate the G2G training. For the \( j \)-th iteration (with \( j \geq 2 \)), we generate
\[
(Z_{j1}^{(i)}, Z_{j2}^{(i)}) \triangleq (\hat{X}_{\phi_{j-1}}(Z(i)) + g_{\theta_1}(r_{j1}^{(i)}), \hat{X}_{\phi_{j-1}}(Z(i)) + g_{\theta_1}(r_{j2}^{(i)}))
\]
for each \( Z^{(i)} \) and denote the resulting set of the pairs as \( \mathcal{D}_j \). Note in (10), we update the noisy estimate of the clean image with the output of G2G \( j-1 \). Then, the new denoiser \( G2G_j \) is obtained by computing \( \phi_j = \arg \min_{\phi} L_{\text{G2G}}(\phi, \mathcal{D}_j) \), where the minimization is done via warm-starting from \( \phi_{j-1} \). In our experiments, we show the sequence, \( G2G_{\geq 1} \), successively refines the denoising quality and significantly improves the initial noisy estimate, similarly as in Figure 1. Moreover, we identify the benefit of the iterative G2G training becomes greater when noise is more sophisticated; i.e., for synthetic noise, the performance of \( G2G_{\geq 1} \) converges after \( 1 \sim 3 \) iterations, whereas for the real-world microscopy noise, the performance keeps increasing until larger number of iterations.

5 Experimental Results

5.1 Data and Experimental Settings

Data & training details In synthetic noise experiments, we always used the noisy training images from BSD400 (Martin et al., 2001). For evaluation, we used the standard BSD68 (Roth & Black,
2009) as a test set. For real-noise experiment, we experimented on two data sets: the WF set in the microscopy image datasets in (Zhang et al., 2019) and the reconstructed CT dataset. For both datasets, we trained/tested on each (Avg = n) and each dose level, respectively, which corresponds to different noise levels. For the generative model training, the patch size used for D and N was 96 × 96, and n and N were set to 20,000 (BSD) and 40,000 (microscopy), respectively. For the iterative G2G training, the patch size for D was 120 × 120 and n = 20,500, and in every mini-batch, we generated new noisy pairs with gθ, as in the noise augmentation of (Zhang et al., 2017). The architecture of G2G was set to 17-layer DnCNN in (Zhang et al., 2017). We put full details on training, model architectures, and hyperparameters as well as the software platforms in the S.M.

**Baselines** The baselines were BM3D (Dabov et al., 2007), DnCNN-B (Zhang et al., 2018), N2N (Lehtinen et al., 2018), and N2V (Krull et al., 2019). We reproduced and trained DnCNN-B, N2N and N2V using the publicly available source codes on the exactly same training data as our iterative G2G training. For DnCNN-B and N2N, which use either clean targets or two independent noisy image copies, we used 20-layers DnCNN model with composite additive white Gaussian noise with σ ∈ [0, 55]. N2N considers the same setting as ours and uses the exact same architecture as G2G; more details on N2V are also given in the S.M. We could not compare with the scheme in (Laine et al., 2019), since their code cannot run beyond white Gaussian noise case in our experiments and they had an unfair advantage: they newly generate noisy images by corrupting given clean images for every mini-batch whereas we assume the given noisy images are fixed once for all. It is known that such noise augmentation significantly can increase the performance, and their code could not run in our setting in which the noisy images are fixed once given. As an upper bound, we implemented N2C(Eq.(4)), denoting a 17-layer DnCNN trained with clean target images in BSD400 and their noisy counterpart, which is corrupted by our gθ, learned with (4).

### 5.2 Denoising results on synthetic noise

**White Gaussian noise** Table 2 shows the results on BSD68 corrupted by white Gaussian noise with different σ’s. Several variations of our G2G, gθ, and the G2G iterates, G2G_j≥1, are shown for two different training data versions for learning the generative model. Firstly, we clearly observe the iterative G2G training is very effective; namely, it significantly improves the initial noisy estimate gθ, particularly when the quality of the initial estimate is not good enough. This result confirms the result of Figure 1 indeed carries over to the blind denoising setting with our method. Secondly, we note G2G_j already considerably outperforms N2V, which is trained with the exact same model architecture and dataset. Finally, the performance of G2G_j is very strong; it outperforms BM3D, which knows true σ, and even sometimes outperforms the blindly trained DnCNN-B and N2N, which is trained with the same BSD400 dataset, but with more information. This somewhat counter-intuitive result is possible since our G2G_j accurately learns the correct noise level in the image, while DnCNN-B and N2N are trained with the composite noise levels, σ ∈ [0, 55].

**Mixture and correlated noise** Table 3 shows the results on mixture and correlated noise beyond white Gaussian. Note our G2G_j does not assume any distribution or correlation structure of the noise, hence, it can still run as long as the assumption on the noise holds. In the table, the G2G results are for (BSD) as specified above. Moreover, DnCNN-B and N2N are still blindly trained with the mismatched white Gaussian noise. For mixture noise, we tested with two cases. Case A corresponds to the same setting as given in (Chen et al., 2018), i.e., 70% ~ N(0, 0.12), 20% ~ N(0, 1), and...
10% ∼ Unif[−s, s] which means the random variable that is uniformly distributed between [−s, s] with s = 15, 25. For case B, we tested with larger variances, i.e., 70% Gaussian N(0, 15²), 20% Gaussian N(0, 25²), and 10% Uniform [−s, s] with s = 30, 50. For correlated noise, we generated the following noise for each ℓ-th pixel,

\[ N_ℓ = ηM_ℓ + (1 − η)\left(\frac{1}{\sqrt{|NB_ℓ|}} \sum_{m \in NB_ℓ} M_m\right), \quad ℓ = 1, 2, \ldots \]

in which \{M_ℓ\} are white Gaussian \(N(0, \sigma^2)\), \(NB_ℓ\) is the \(k \times k\) neighborhood patch except for the pixel ℓ, and η is a mixture parameter. We set \(η = 1/\sqrt{2}\) such that the marginal distribution of \(N_ℓ\) is also \(N(0, \sigma^2)\) and set \(k = 16\). Note in this case, \(N_ℓ\) has a spatial correlation, and we tested with \(σ = 15, 25\). From the table, we first note that DnCNN-B and N2N suffer from serious performance degradation for both mixture and correlated noises due to noise mismatch, and the conventional BM3D outperforms them for some cases (e.g., Case A for mixture noise). However, we note our G2G can still denoise very well after just two iterations and outperforms all the baselines for all noise types. Note N2V seriously suffers and is not comparable to ours. Finally, N2C(Eq.(4)) is a sound upper bound for all noise types, confirming the correctness of the extraction rule (4).

5.3 Denoising results on real noise

We also test our method on the real-world noise. While some popular real noise is known to have source-dependent characteristics, there are also cases in which the noise is source-independent and pixel-wise correlated, which satisfies the assumption of our method. We tested on two such datasets, the Wide-Focal (WF) set in the microscopy image dataset (Zhang et al., 2019) and a Reconstructed CT dataset. A more detailed description and analysis on these two datasets are in S.M. The WF and Reconstructed CT data has 5 sets (Avg = 1, 2, 4, 8, 16) and 4 sets (Dose=25, 50, 75, 100) with different noise levels, respectively. We did not exploit the fact that the images are multiple noisy measurements of a clean image, which enables employing N2N, but treated them as noisy images of distinct clean images. Figure 4(a) and 4(b) shows the PSNR of all methods for each dataset, respectively, averaged over all sets. The baselines were DnCNN-B, BM3D and N2V. We note BM3D estimated noise \(σ\) using the method in Chen et al. (2015). We iterated until G2G_B and N2C(Eq.(4)) was an upper bound for each set. We clearly observe that the performance of G2G, significantly improves (over \(θ_B\)) as the iteration continues. In results, G2G_B becomes significantly better than DnCNN-B and N2V as well as BM3D, still one of the strongest baselines for real-world noise denoising when no clean target images are available, for both datasets. We report more detailed experimental results (including SSIM) on both datasets in S.M. Moreover, the inference time for BM3D is about 4.5~5.0 seconds per image since a noise estimation has to be done for each image separately, whereas that for G2G_B is only 4 ms (on GPU), which is another significant advantage of our method. Figure 4(c) shows the visualizations on the WF, and we give more examples in the S.M.

5.4 Ablation study

Noise patch extraction Here, we evaluate the effect of the noisy patch extraction rules (3) and (4) in the final denoising performance. Figure 5 compares the PSNR of N2C(GCBD Eq.(3)), a re-implementation of (Chen et al., 2018), N2C(Ours Eq.(4)) and the best G2G, for each dataset.
We note neither source code nor training data of (Chen et al., 2018) is publicly available, and the PSNR in (Chen et al., 2018) could not be reproduced (with the exact same $\eta$ and $\gamma$ as in (Chen et al., 2018)). From the figure, we clearly observe the significant gap between N2C(Our Eq.(4)) and N2C(GCBD Eq.(3)), particularly when the noise is not white Gaussian. Moreover, our pure unsupervised G2G with (4) even outperforms N2C(GCBD Eq.(3)) that utilizes the clean target images, confirming the quality difference shown in Figure 2(b) significantly affects learning noise and a denoiser.

**Generative model and iterative G2G training** Figure 6(a) shows the PSNRs of $g_{\theta_2}$ on BSD68/Gaussian($\sigma = 25$) trained with three variations; “No $\mathcal{L}_Z$” for no $f_{w_2}$, “No $\mathcal{L}_{cyc}$” for no (7) and $g_{\theta_1}$, and “No sigmoid” for no sigmoid activation at the output layer of $g_{\theta_2}$. We confirm that our proposed architecture achieves the highest PSNR for $g_{\theta_2}$, the sigmoid activation and $f_{w_2}$ are essential, and the cycle loss (7) is also important. Achieving a decent PSNR for $g_{\theta_2}$ is beneficial for saving the number of G2G iterations and achieving high final PSNR. More detailed analyses on the generative model architecture are in the S.M. Figure 6(b) and 6(c) show the effect of the quality of the initial estimate for the iterative G2G training. From Figure 1, one may ask whether $g_{\theta_2}$ is indeed necessary, since even when $\sigma_0 \approx \sigma$, the iterating the Noisy N2N can mostly achieve the upper bound. Hence, for samples of synthetic and real microscopy data, we evaluate how G2G$_p$ performs when the iteration simply starts with $Z$. Figure 6(b) shows a somewhat surprising result that for synthetic noises, starting from $Z$ achieves essentially the same performance as starting from $g_{\theta_2}$ with a couple more G2G iterations. However, for real microscopy noise case in Figure 6(c), $WF(Avg=1, \sigma_2)$ in which starting from $Z$ achieves far lower performance than starting from $g_{\theta_2}$, justifying our generative model for attaining the initial noisy estimate.

6 CONCLUDING REMARK

Motivated by a novel observation on Noisy N2N, we proposed a novel GAN2GAN method, which can tackle the challenging blind image denoising problem solely with single noisy images. Our method showed impressive denoising performance that even sometimes outperform the methods with more information as well as VST+BM3D for real noise denoising. As a future work, we plan to extend our framework to more explicitly handle the source-dependent real-world noise.

ACKNOWLEDGMENT

This work was supported in part by NRF Mid-Career Research Program [NRF-2021R1A2C2007884] and IITP grant [No.2019- 0-01396, Development of framework for analyzing, detecting, mitigating of bias in AI model and training data], funded by the Korean government (MSIT).
REFERENCES

Saeed Anwar and Nick Barnes. Real image denoising with feature attention. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 3155–3164, 2019.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In International Conference on Machine Learning (ICML), 2017.

Joshua Batson and Loïc Royer. Noise2self: Blind denoising by self-supervision. In International Conference on Machine Learning (ICML), 2019.

A. Buades, B. Coll, and J. M. Morel. A review of image denoising algorithms, with a new one. SIAM Journal on Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal, 2005.

Sungmin Cha and Taesup Moon. Fully convolutional pixel adaptive image denoiser. In International Conference on Computer Vision (ICCV), 2019.

Guangyong Chen, Fengyuan Zhu, and Pheng Ann Heng. An efficient statistical method for image noise level estimation. In Proceedings of the IEEE International Conference on Computer Vision (CVPR), pp. 477–485, 2015.

Jingwen Chen, Jiawei Chen, Hongyang Chao, and Ming Yang. Image blind denoising with generative adversarial network based noise modeling. In Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

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

D. Donoho and I. Johnstone. Adapting to unknown smoothness via wavelet shrinkage. Journal of American Statistical Association, 90(432):1200–1224, 1995.

M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. IEEE Trans. Image Processing, 54(12):3736–3745, 2006.

Alexander Krull, Tim-Oliver Buchholz, and Florian Jug. Noise2void-learning denoising from single noisy images. In Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

Samuli Laine, Tero Karras, Jaakko Lehtinen, and Timo Aila. High-quality self-supervised deep image denoising. In Advances in Neural Information Processing Systems (NeurIPS), pp. 6968–6978, 2019.

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 (ICML), 2018.

Ding Liu, Bihan Wen, Yuchen Fan, Chen C. Loy, and Thomas S. Huang. Non-local recurrent network for image restoration. In Neural Information Processing Systems (NeurIPS), 2018.

S. Lunz, O. Öktem, and C.-B. Schönlieb. Adversarial regularizers in inverse problems. In Neural Information Processing Systems (NeurIPS), 2018.

J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In International Conference on Computer Vision (ICCV), 2009.

D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In International Conference on Computer Vision (ICCV), 2001.

S. Roth and M.J. Black. Field of experts. International Journal of Computer Vision, 82(2):205–229, 2009.

Shakarim Soltanayev and Se Young Chun. Training deep learning based denoisers without ground truth data. In Advances in Neural Information Processing Systems (NeurIPS), pp. 3257–3267, 2018.
Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu. Memnet: A persistent memory network for image restoration. In *International Conference on Computer Vision (ICCV)*, 2017.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

R.A. Yeh, T. Y. Lim, C. Chen, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do. Image restoration with deep generative models. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Cycleisp: Real image restoration via improved data synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2696–2705, 2020.

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

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 *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image restoration. *arXiv preprint arXiv:1812.10477*, 2018.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *International Conference on Computer Vision (ICCV)*, pp. 2223–2232, 2017.

Magauia Zhussip, Shakarim Soltanayev, and Se Young Chun. Extending stein’s unbiased risk estimator to train deep denoisers with correlated pairs of noisy images. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 1465–1475, 2019.

D. Zoran and Y. Weiss. From learning models of natural image patches to whole image restoration. *International Conference on Computer Vision (ICCV)*, 2011.
SUPPLEMENTARY MATERIALS FOR
GAN2GAN: GENERATIVE NOISE LEARNING FOR
BLIND DENOISING WITH SINGLE NOISY IMAGE

Sungmin Cha$^1$, Taeeon Park$^1$, Byeongjoon Kim$^2$, Jongduk Baek$^2$ and Taesup Moon$^3$∗
Sungkyunkwan University$^1$, Yonsei University$^2$, Seoul National University$^3$, South Korea
{csm9493,pte1236}@skku.edu, bjkim2006@naver.com,
jongdukebak@yonsei.ac.kr, tsmoon@snu.ac.kr

1 PROOF OF THEOREM 1

Theorem 1 Consider the single-letter Gaussian setting and $f_{\text{Noisy N2N}}(Z, y)$ obtained in (Eq.(2), manuscript). Also, assume $0 < y = \sigma_0^2/\sigma_N^2 < 1$. Then, there exists some $y_0$ s.t. $\forall y \in (y_0, 1), \mathbb{E}(X - f_{\text{Noisy N2N}}(Z, y))^2 < \sigma_0^2$.

Proof: We consider the following chain of equalities:
\[
\begin{align*}
\sigma_0^2 - \mathbb{E}(X - f_{\text{Noisy N2N}}(Z, y))^2 &= \sigma_0^2 - \mathbb{E}\left(X - \frac{\sigma_X^2(1 + y)}{\sigma_X^2(1 + y) + \sigma_N^2}(X + N)^2\right) \\
&= \sigma_0^2 - \mathbb{E}\left(\frac{\sigma_N^2}{\sigma_X^2(1 + y) + \sigma_N^2} X - \frac{\sigma_N^2(1 + y)}{\sigma_X^2(1 + y) + \sigma_N^2} N\right)^2 \\
&= \sigma_X^2\left(y - \frac{\sigma_N^2}{(\sigma_X^2(1 + y) + \sigma_N^2)^2} - \frac{\sigma_N^2(1 + y)^2}{(\sigma_X^2(1 + y) + \sigma_N^2)^2}\right) \\
&= \sigma_X^2\left(y(\sigma_X^2(1 + y) + \sigma_N^2)^2 - \sigma_N^2 - \sigma_N^2(1 + y)^2\right) \\
&= \sigma_X^2(\sigma_X^2 - \sigma_N^2) - \sigma_N^2(1 + y)^2.
\end{align*}
\]

Now, by denoting the numerator of (5) as $g(y)$, we have
\[
\begin{align*}
g(y) &= y(\sigma_X^2(1 + y)^2 + \sigma_N^2 + 2\sigma_X^2 \sigma_N^2(1 + y)) - \sigma_N^2 \sigma_X^2(y^2 + 2y + 1) - \sigma_N^4 \\
&= \sigma_X^2 y^2 + (2\sigma_X^2 + \sigma_N^2 \sigma_X^2)y^2 + (\sigma_X^4 + \sigma_N^4)y - \sigma_N^2 \sigma_X^2 - \sigma_N^4.
\end{align*}
\]

Then, we can easily see that
\[
\begin{align*}
g(0) &= -\sigma_N^2 \sigma_X^2 - \sigma_N^4 < 0 \\
g(1) &= 4\sigma_X^4 > 0 \\
g(y) &= 3\sigma_X^2 y^2 + 2(2\sigma_X^4 + \sigma_N^2 \sigma_X^2)y + \sigma_X^4 + \sigma_N^4 > 0.
\end{align*}
\]

Therefore, in $0 < y < 1$, we can see $g(y)$ is an increasing function and has a root $y_0$ in the interval. Hence, the claim of the theorem: for all $y \in (y_0, 1), \mathbb{E}(X - f_{\text{Noisy N2N}}(Z, y))^2 < \sigma_0^2$ holds. ■

2 DETAILS ON THE EXPERIMENTAL SETTINGS

2.1 Smooth noisy patch extraction

2.1.1 GCBD (CHEN ET AL., 2018) RULE Eq.(3)

The original GCBD paper (Chen et al., 2018) did not provide any source code or training data, hence, we reproduced their noisy patch extraction algorithm. There are six hyperparameters for the rule...
and we used the exact same hyperparameters given in their paper, which are shown in Table 1. \(d\) and \(h\) denote the size of a patch, \(p\), and its sub-patches, \(q_j\), given in [Eq.(3), Manuscript], respectively. \(s_p\) and \(s_q\) are the stride sizes for extracting the patches, \(p\) and \{\(q_j\)}, from a given image. \(\mu\) and \(\lambda\) are the hyperparameters of the rule for selecting the smooth patches shown in [Eq.(3), Manuscript].

### Table 1: Hyperparameters for the patch extraction rule of GCBD

| Hyperparameters | \(d\) | \(h\) | \(s_p\) | \(s_q\) | \(\mu\) | \(\gamma\) |
|----------------|-------|-------|--------|--------|-------|--------|
| Values         | 64    | 16    | 32     | 16     | 0.1   | 0.25   |

#### 2.1.2 G2G RULE [Eq.(4)]

There are three hyperparameters for our extraction rule [Eq.(4), Manuscript], \(\lambda\), \(d\) (the patch size), and \(s_p\) (the stride size for extracting patches from an image). The choices for our experiments are shown in Table 3. Moreover, we stress that we did not tune \(\lambda\) using clean images, but the different \(\lambda\) values in the table are determined by the pre-determined number of extracted patches by applying our rule [Eq.(4), Manuscript]. Moreover, as argued in Section 3.1 (manuscript), we do not require any sub-patches to be extracted, hence, have only half the hyperparameters compared to the GCBD rule.

### Table 2: Hyperparameters for the extraction rule of G2G

| Gaussian Noise | Mixture Noise | Correlated Noise | WF | Medical |
|----------------|---------------|------------------|----|---------|
| \(\lambda\)    | 0.03          | 0.1              | 0.15 | 0.42    | 0.015  |
| \(d\)           | 96            |                  |     |         |        |
| \(s_p\)         | 24            |                  |     |         |        |

#### 2.1.3 EFFECT OF \(\lambda\)

Table 3 shows the effect of \(\lambda\) in [Eq.(4), Manuscript] on the final performance of G2G. Note the smaller the \(\lambda\), the less number of patches are extracted, but the homogeneity increases. The table shows \(\lambda\) clearly affects the denoising performance of \(g_{\theta_1}\), but as the iterative G2G training continues, the performance of G2G becomes not very sensitive to \(\lambda\). Hence, in our experiments, we did not optimize \(\lambda\) based on any clean validation set, but just set \(\lambda\) based on the number of extracted patches and checking the visual qualities of the patches.

### Table 3: Effects of varying \(\lambda\) on the denoising performance.

| \(\lambda\)  | # of patches | \(g_{\theta_1}\) | \(\lambda\)  | # of patches | \(g_{\theta_1}\) | \(\lambda\)  | # of patches | \(g_{\theta_1}\) | \(\lambda\)  | # of patches | \(g_{\theta_1}\) |
|-------------|--------------|----------------|-------------|--------------|----------------|-------------|--------------|----------------|-------------|--------------|----------------|
| 0.04        | 80,000       | 28.84/0.315    | 0.05        | 80,000       | 28.84/0.315    | 0.06        | 80,000       | 28.84/0.315    | 0.07        | 80,000       | 28.84/0.315    |
| 0.08        | 80,000       | 28.84/0.315    | 0.10        | 80,000       | 28.84/0.315    | 0.12        | 80,000       | 28.84/0.315    | 0.14        | 80,000       | 28.84/0.315    |

#### 2.2 TRAINING A W-GAN BASED GENERATIVE MODEL

Here, we elaborate a couple of subtle points for training our generative model as mentioned in Section 3.2 (manuscript).

Firstly, given the overall optimization objective [Eq.(8), manuscript], we use \((\alpha, \beta, \gamma) = (1, 1, 0)\) for the inner maximization for critics, and use \((\alpha, \beta, \gamma) = (5, 1, 10)\) for the outer minimization for generators. The main intuition for using different \((\alpha, \beta, \gamma)\) for training the generators is due to different levels of confidence in the generator loss terms. Namely, we assign the largest weight to [Eq.(7), Manuscript] since it is a deterministic loss and its value has a clear meaning. The generator loss [Eq.(5), Manuscript], which is in the form of the standard W-GAN loss, gets the medium level weight since the meaning of its value is less certain than [Eq.(6), Manuscript]. In contrast, the generator loss in [Eq.(5), Manuscript], which consists of two generators, can become somewhat unstable during training, hence, it gets the least weight. Figure 2.2 compares the performance of
Figure 1: Ablation study on \((\alpha, \beta, \gamma)\)

\(g_\theta^2\)'s on BSD68 \((\sigma = 25)\) when using \((\alpha, \beta, \gamma) = (5, 1, 10)\), for the outer minimization, as proposed, and using \((\alpha, \beta, \gamma) = (1, 1, 1)\). We observe there is a significant gap between the two.

Secondly, the output layer of \(g_\theta\) must have the sigmoid activation function. Note \(g_\theta\) itself can be thought of another denoiser, but since we are not training it with any target, we need to ensure the outputs of \(g_\theta\) have values between \([0, 1]\) to prevent from obvious errors of generating negative or out-of-bound pixel values. Without the sigmoid activation, it turned out all the generators cannot be trained properly at all.

Finally, using the right architectures for the generators and critics, e.g., number of layers and filters, was critical since the training procedure got very sensitive to the architectural variations. Tables 4 shows the details on the architecture of our first generator, \(g_\theta^1\), which aims to generate noise patches. The dimension of \(r\) (the input random vector) was set to 128, and \(C\) denotes the channel of the generated noise patch. The architectures of the \(g_\theta^2\) and \(g_\theta^3\) in our generative model are equal to that of the DnCNN model (Zhang et al., 2018), however, \(g_\theta^2\) had 15 layers with sigmoid activation in the output layer, and \(g_\theta^3\) had 17 layers and linear activation in the output layer. In addition, the architectures of the two critics, \(\{f_{w_1}, f_{w_2}\}\), in our generative model are given in Table 5.

Table 4: Architectural details on \(g_\theta^1\).

| Layer Num | Layer composition | Input channel | Output channel | Kernel size | Stride | Padding |
|-----------|------------------|---------------|----------------|-------------|--------|---------|
| 1         | DeConv + BatchNorm + ReLU | 128           | 64             | 4           | 1      | 0       |
| 2         | DeConv + BatchNorm + ReLU | 64            | 32             | 4           | 2      | 1       |
| 3         | DeConv + BatchNorm + ReLU | 32            | 16             | 4           | 2      | 1       |
| 4         | DeConv + BatchNorm + ReLU | 16            | 8              | 4           | 1      | 1       |
| 5         | Conv + Tanh       | 8             | \(C\)          | 4           | 2      | 1       |

Output shape : \((64\times64\times C)\)

Table 5: Architectural details on the critics, \(\{f_{w_1}, f_{w_2}\}\).

| Layer Num | Layer composition | Input channel | Output channel | Kernel size | Stride | Padding | \(\alpha\) |
|-----------|------------------|---------------|----------------|-------------|--------|---------|----------|
| 1         | Conv + BatchNorm + LeakyReLU | 64            | 128            | 4           | 2      | 1       | 0.2      |
| 2         | Conv + BatchNorm + LeakyReLU | 256           | 256            | 4           | 2      | 1       | 0.2      |
| 3         | Conv + BatchNorm + LeakyReLU | 512           | 512            | 4           | 2      | 1       | 0.2      |

Output shape : \((64\times64\times1)\)

For training, we carry out the random cropping of the given patches to the size of \(64 \times 64\), and the data augmentation was done by flipping the cropped patches horizontally and vertically. For optimization, we used Adam (Kingma & Ba, 2015) optimizer for the three generators and RMSProp (Tieleman & Hinton, 2012) optimizer for the two critics. The initial learning rates were set to 0.0004 and 0.0005 for Adam and RMSProp, respectively. Also, the learning rate decay, dropping the learning rate linearly starting from epoch 10, is applied to the Adam optimizer. The parameter clipping was done for the critics and the range was set to \([-0.02, 0.02]\), and the number of training iterations for the critics was 5. The total number of training epochs was 30 and the mini-batch size was 64.
2.2.1 ABLATION STUDY ON $\mathcal{L}_{\text{cyc}}$

As shown in the synthetic noise case of Figure 6(b) (manuscript), the iterative G2G training is powerful such that there is a negligible performance difference between the schemes with and without $g_{\theta_2}$, when the number of iterations is sufficiently large. Consequently, the cycle loss $\mathcal{L}_{\text{cyc}}$ also does not have significant effect in the final performance for the synthetic noise case. However, for the real noise case, $\mathcal{L}_{\text{cyc}}$ becomes more critical. As shown in Figure 2(a) and 2(b), on WF(Avg = 1) dataset, we observe that when there is no $\mathcal{L}_{\text{cyc}}$ in our generative model, the final PSNR or SSIM performances cannot reach the model with $\mathcal{L}_{\text{cyc}}$ even after many iterations of G2G training. Hence, this result shows the necessity of $\mathcal{L}_{\text{cyc}}$.

2.3 ITERATIVE GAN2GAN TRAINING OF A DENOISER

We do the same random cropping and data augmentation as in the generative model training. Moreover, for every minibatch in the G2G training, we generated new synthetic noisy image pairs using our trained generators as was done in the noise augmentation of (Zhang et al., 2017). Adam optimizer with an initial learning rate $0.001$ was used, and the learning rate scheduling, which halves the learning rate every 20 epochs, was applied. The total number of training epochs was 50, and the mini-batch size was 4. We also stress that we set the architecture of $\hat{X}_{\theta}(Z)$ identical to that of 17-layers DnCNN in (Zhang et al., 2017) to make a fair comparison. The pseudo algorithm for training a generative model is in Algorithm 1.

Algorithm 1 Training G2G, all experiments in this paper used the defaults values, $n_{\text{epoch}} = 50$, $\alpha_{\text{G2G}} = 1e^{-3}$

1: Require $\mathcal{D}$, $g_{\theta_1}$, $g_{\theta_2}$, $\phi$, $n_{\text{iter}}$, $m$
2: for $j \leftarrow 1$, $n_{\text{iter}}$ do
3: for $ep \leftarrow 1$, $n_{\text{epoch}}$ do
4: Sample $\{r_{j,1}^{(i)}, r_{j,2}^{(i)}\}_{i=1}^m \sim \mathcal{N}(0, I)$, $\{Z^{(i)}\}_{i=1}^m \sim \mathcal{D}$
5: $\phi_j \leftarrow \arg\min_{\phi} \mathcal{L}_{\text{G2G}}(\phi, \hat{D}_j)$,
6: end for
7: end for
8: return $\phi_{n_{\text{iter}}}$

2.4 NOISE2VOID (KRULL ET AL., 2019)

We used the publicly available source code of Noise2Void (N2V) (Krull et al., 2019) to obtain the denoising results of N2V. Most of the hyperparameters were set to the default ones, but we changed three things to make a fair comparison with our method.

Firstly, while the CNN architecture for the original N2V was a UNet3, we used the DnCNN (Zhang et al., 2018) with 17 layers such that it has the same structure as our G2G model. Secondly, as also is

![Figure 2](image-url)
done in (Krull et al., 2019), we had to use a validation set to do a proper model selection for N2V (i.e., the best epoch), while our G2G does not require any validation set (since we always use a model at the last epoch). The reason why N2V needs a validation is that its learning curve is very unstable and a proper model selection greatly affects the final denoising performance. To that end, since we used 20,500 patches with $120 \times 120$ size for training our G2G and other baselines, we divided the 20,500 patches into 18,000 training patches and 2,500 validation patches for training and selecting the best N2V model. Thirdly, we set 'mini_batch_size' to 4 (as our G2G) and 'train_steps_per_epoch' to 'num_of_training_data / mini_batch_size', hence, 4,500. Other hyperparameters are given in Table 6.

| Hyperparameter          | Value          |
|-------------------------|----------------|
| train_steps_per_epoch   | 4,500          |
| train_loss              | ‘mse’          |
| train_scheme            | ‘Noise2Void’   |
| train_batch_size        | 4              |
| n2v_num_pix             | 64             |
| n2v_patch_shape         | (64,64)        |
| n2v_manipulator         | ‘uniform_withCP’ |
| n2v_neighborhood_radius | ‘5’            |

### 3 Comparison of the Patch Extraction Rules

Here, we make a further, thorough comparison between the GCBD smooth patch extraction rule [Eq.(3), manuscript] and ours [Eq.(4), manuscript]. We selected three noisy patches from [Figure 2(b), manuscript] and show the decision criterion of each rule for each image Figure 3. From the figure, we can observe that while our G2G rule correctly excludes the patches in Figure 3(b) and 3(c) as non-homogeneous patches, the GCBD rule wrongly determines them also as homogeneous patches. That is, we note that since the DWT transform used in our rule can successfully disaggregate the high and low frequency components in the patches, the patches with self-similar repeating patterns would have significantly varying sub-band coefficient variances as shown in the figures. Hence, our rule can exclude those patches. However, in the GCBD rule, there may exist a sub-patch $q_j$ that has similar empirical mean and variance as the original patch $p$, thus, it may determine the patches with the self-similar repeating patterns as homogeneous as well. We believe these examples clearly show the stark difference between our rule and the GBCD rule for smooth patch extraction.
Published as a conference paper at ICLR 2021

Condition for homogeneous patch

\[
\max(|\mathcal{E}(x) - \mathcal{E}(y)|) \leq \tau \times |\mathcal{E}(x)| \\
\max(|\mathcal{V}(x) - \mathcal{V}(y)|) \leq \tau \times |\mathcal{V}(x)|
\]

Homogeneous patch

Not a homogeneous patch

Figure 3: Noise patch extractions of GCBD (3) and G2G (4) rules.

(a) Example patch determined as homogeneous by the both rules.

(b) Example patch 1 that is wrongly determined as homogeneous by the GCBD rule.

(c) Example patch 2 that is wrongly determined as homogeneous by the GCBD rule.
3.1 Result Table for Real Microscopy Noise

We report the detailed experimental results on the real microscopy images in Table 7. We observe that Iterative G2G increases PSNR/SSIM in WF.

Table 7: Experimental results on the real microscopy dataset

| Noise Type | DaCNN-S | DaCNN-D | BM3D | N2C (DaCNN) | G2G | G2G | G2G | G2G | G2G | G2G | G2G | N2C (G2C-D) | N2C (cEq.4) |
|------------|---------|---------|------|-------------|-----|-----|-----|-----|-----|-----|-----|-------------|-------------|
| Raw        | 35.39/0.8738 | 25.43/0.3702 | 26.32/0.4012 | 25.31/0.3411 | 28.40/0.5261 | 30.63/0.6407 | 32.03/0.6889 | 32.21/0.7114 | 32.56/0.7672 | 32.62/0.7910 | 31.16/0.7493 | 32.26/0.8205 |
| Avg = 2    | 31.41/0.6944 | 22.54/0.3966 | 23.47/0.4366 | 22.46/0.3766 | 25.54/0.5514 | 27.77/0.6656 | 29.19/0.7134 | 29.37/0.7361 | 29.72/0.7924 | 29.88/0.8158 | 29.40/0.7808 | 30.57/0.8216 |
| Avg = 4    | 29.85/0.6467 | 21.96/0.3599 | 22.88/0.3999 | 21.89/0.3499 | 24.96/0.5246 | 27.19/0.6384 | 28.62/0.6862 | 28.80/0.7089 | 29.15/0.7652 | 29.31/0.7886 | 28.83/0.7528 | 30.00/0.8090 |
| Avg = 8    | 27.20/0.6071 | 20.31/0.3183 | 21.24/0.3583 | 20.25/0.3085 | 23.32/0.4935 | 25.54/0.5683 | 26.97/0.6161 | 27.15/0.6389 | 27.50/0.7052 | 27.66/0.7284 | 27.18/0.6918 | 29.35/0.8340 |
| Avg = 16   | 24.74/0.5675 | 19.45/0.2787 | 20.38/0.3187 | 19.39/0.2689 | 22.46/0.4537 | 24.69/0.5285 | 26.12/0.5763 | 26.30/0.6005 | 26.65/0.6678 | 26.81/0.6909 | 26.33/0.6541 | 28.52/0.7892 |
| Average    | 29.90/0.7046 | 22.01/0.3824 | 23.03/0.4224 | 22.04/0.3624 | 25.12/0.5374 | 27.35/0.6516 | 28.78/0.7094 | 28.96/0.7321 | 29.31/0.7884 | 29.47/0.8116 | 29.00/0.7758 | 30.19/0.8181 |

3.2 Description and the Result Table on Reconstructed CT Dataset

The reconstructed CT dataset consists of chest and head parts of 27 pediatric extended cardiac-torso phantoms (Segars et al., 2015), which provide a highly realistic model of the human anatomy. We extracted 60 image slices from each phantom, leading to 1620 image slices in total. The dataset was generated in the following procedure. First, noiseless projection data were acquired in a parallel-beam geometry with Siddon’s ray-driven algorithm (Sidky & Pan, 2008). To reduce view aliasing artifacts, the detector quarter-offset was used during a CT scan. Second, Poisson noise was generated and added to the noiseless projection data. Note that the mean number of detected photons was set to 2,500, 5,000, 7,500, and 10,000 to simulate 25%, 50%, 75%, and 100% of a normal dose, respectively. Finally, the images were reconstructed by filtered backprojection (Hsieh, 2003). To preserve fine anatomical structures in the images, the Ram-Lak filter was used as a reconstruction filter. Detailed simulation parameters are summarized in Table 8.

Table 8: Simulation parameters

| Parameters | Values |
|------------|--------|
| Source to iso-center distance | 595 mm |
| Source to detector distance | mm |
| Detector cell size | 0.7 mm |
| Detector array size | 736 x 1 |
| Data acquisition angle | 360 dares |
| Number of projection views | 736 |
| Reconstructed pixel width | 0.67 mm |
| Reconstructed matrix size | 512x512 |

We divided 27 phantoms into training and test data and the phantom number for each dataset is in Fig 9. Also, We visualized the first image of Female 1 in Fig 4. We can clearly see that each dose has a different noise level, and the noise is source independent and correlated. Finally, Table 10 shows the details of experimental results on Reconstructed CT dataset.

Table 9: Training and test data information of Reconstructed CT dataset

| Training data | Test data |
|---------------|-----------|
| Female | 1,3,4,5,6,7,8,9,10,11 | 13,14,15 |
| Male | 1,2,3,4,5,6,7,8,9,10,11 | 12,13 |

# of images 21x60 = 1260 60x3 = 300
Table 10: Experimental results on the reconstructed CT

| Data Type | Noise Type | N2C (UNet) | DnCNN | BM3D | N2V (UNet) | G2G1 | G2G2 | G2G3 |
|-----------|------------|------------|-------|------|------------|------|------|------|
| Dose 25   | 48.43      | 35.50      | 42.40 | 31.57| 34.66      | 40.49| 46.04| 47.47|
|           | /0.9609    | /0.6055    | /0.7575| /0.5416| /0.5759  | /0.8301| /0.9579| /0.9707|
| Dose 50   | 49.47      | 38.48      | 45.14 | 34.37| 37.70      | 43.38| 47.78| 48.06|
|           | /0.9600    | /0.7440    | /0.8510| /0.6931| /0.7202  | /0.9063| /0.9702| /0.9705|
| Dose 75   | 49.45      | 40.09      | 46.55 | 35.90| 39.49      | 44.96| 48.80| 49.20|
|           | /0.9591    | /0.8111    | /0.8929| /0.7756| /0.7919  | /0.9320| /0.9744| /0.9733|
| Dose 100  | 49.63      | 41.19      | 47.48 | 37.26| 40.75      | 46.11| 49.19| 48.85|
|           | /0.9565    | /0.8513    | /0.9169| /0.8118| /0.9350  | /0.9492| /0.9760| /0.9718|
| Average   | 49.15      | 38.82      | 45.39 | 34.73| 38.15      | 43.74| 47.95| 48.39|
|           | /0.9591    | /0.5730    | /0.8546| /0.7055| /0.7558  | /0.9044| /0.9696| /0.9715|

4 ANALYSIS ON REAL MICROSCOPY IMAGE

In this section, we analyze why our G2G also works well on the real microscopy image dataset (WF) Zhang et al. (2019), although the source-dependent noise does not satisfy our assumption on the noise. The real microscopy image dataset consists of three different types of dataset, which are Wide-Focal(WF), Two-Photon(TP) and Con-Focal(CF), and it is generally known that the real noise follows the Poisson-Gaussian model Zhang et al. (2019),

\[ Z_i = x_i + N_i, \quad i = 1, 2, \ldots, \]

in which \( N_i \sim \mathcal{N}(0, \sigma_i^2) \) and

\[ \sigma_i^2 = \alpha x_i + \sigma^2 \]

with a scaling factor \( \alpha > 0 \). Thus, the noise variance depends on the underlying clean source pixel value, and \( \alpha \) determines the level of the dependence.

In Table 7, we observe that our G2G performs well for WF compared to other baselines, hence, we visualize clean, noisy, noise images from each set and examine if there are any notable difference in the noise distributions. Figure 5 shows two image samples (Avg = 1 cases) from the Wide-Focal (WF) set. The noise images are obtained by subtracting the clean images from its noisy versions. Note even though the intensities in the source images change significantly among pixels (particularly for the top image), the noise images do not show any source-dependent patterns. Hence, we can deduce that \( \alpha \) may be small for the WF images. Also, we could see that there is a correlated pattern in the noise. We believe that these back the good performance of G2G for the WF set.

Figure 6, on the other hand, visualizes an image from TP set for Avg = {1, 16} cases. Comparing with Figure 5, we can clearly see the source-dependent patterns in the noise images, particularly severely for the Avg = 1 case. Also, CP set showed the similar source-dependent noise patterns. This source-dependent noise is not in our assumption so we did not apply GAN2GAN to TP and CP.
However, we want to stress out that the source independent real noise also exists and GAN2GAN shows the best result compared to any other baselines.
5 VISUALIZATIONS

5.1 VISUALIZATION OF \( \tilde{Z} \)

Figure 7 and 8 visualize the simulated noisy image pairs \((\hat{Z}_1, \hat{Z}_2)\), generated from our generative model, for synthetic and real noise cases, respectively. A close examination shows that the images are not simple copies of the original noisy image \( Z \) but are successfully synthesized with the independent noise processes.

Gaussian \((\sigma = 25)\)

Mixture \((s = 30)\)

Correlated \((\sigma = 25)\)

\[ Z + \text{Gaussian (}\sigma = 25) \]

\[ Z + \text{Mixture (}\sigma = 30) \]

\[ Z + \text{Correlated (}\sigma = 25) \]

Figure 7: Visualizations of synthesized synthetic noisy image pairs.
Figure 8: Visualizations of synthesized real noise image pairs.

5.2 Visualization of denoised images on BSD68

Figure 9 visualizes the denoising results of a BSD68 image for different types of noise. Note the clear difference in the noise characteristics for Gaussian, mixture, and correlated noises. The visualization of G2G$_3$ certainly seems better than N2V and BM3D, in line with the PSNR results. DnCNN-B and Noise2Noise use more information than G2G$_3$, but the visualization as well as the PSNR of G2G$_3$ are comparable to those of the two methods.

![Figure 9: Denoising results on the synthetic noise images.](image-url)
5.3 ADDITIONAL VISUALIZATIONS ON THE REAL MICROSCOPY IMAGES

We also visualize additional denoised images of WF images in Figure 10. We can see that the denoising results of the baselines for WF (Avg = 1) are very noisy, but G2G\textsubscript{3} shows relatively clean denoising results than others.

![Figure 10: Denoising results on the real noisy microscopy images.](image)

5.4 Additional Visualizations on the Reconstructed CT

We visualize the denoising result of a Reconstructed CT image in Figure 11. We observed that BM3D and N2V shows a still noisy result on this image but G2G\textsubscript{3} shows a clearly denoised result.

![Figure 11: Denoising results on Reconstructed CT images.](image)

6 Experimental Results of G2G\textsubscript{3} with ($\hat{Z}_{j1}^{(i)}$, $\hat{X}_{\phi_{j-1}}(Z^{(i)})$)

| PSNR / SSIM | G2G\textsubscript{3} $\hat{Z}_{j1}^{(i)}$, $\hat{X}_{\phi_{j-1}}(Z^{(i)})$ | G2G\textsubscript{3} $\hat{Z}_{j2}^{(i)}$, $\hat{Z}_{j1}^{(i)}$ |
|-------------|-------------------------------------------------|-------------------------------------------------|
| Gaussian ($\sigma = 25$) | 29.02 / 0.8153 | 28.96 / 0.8080 |
| Mixture ($s = 30$) | 30.70 / 0.8621 | 30.49 / 0.8538 |
| Correlated ($\sigma = 25$) | 24.04 / 0.8673 | 28.00 / 0.8447 |
| WF (Avg = 1) | 16.24 / 0.4490 | 34.57 / 0.7970 |
| Medical (Dose = 25) | 18.52 / 0.7397 | 47.47 / 0.9707 |

We did the experiments on G2G\textsubscript{3} with ($\hat{Z}_{j1}^{(i)}$, $\hat{X}_{\phi_{j-1}}(Z^{(i)})$) and Table 11 shows the experimental results. From the table, we observe the suggested approach (left column) can in fact achieve slightly improved results for synthetic Gaussian and Mixture noise. However, we observe that the performances of the approach for the Correlated and WF/Medical datasets deteriorate significantly compared to ours.
REFERENCES

Jingwen Chen, Jiawei Chen, Hongyang Chao, and Ming Yang. Image blind denoising with generative adversarial network based noise modeling. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1

Jiang Hsieh. *Computed tomography: principles, design, artifacts, and recent advances*, volume 114. SPIE press, 2003. 7

D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 3

Alexander Krull, Tim-Oliver Buchholz, and Florian Jug. Noise2void-learning denoising from single noisy images. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 4, 5

WP Segars, Hannah Norris, Gregory M Sturgeon, Yakun Zhang, Jason Bond, Anum Minhas, Daniel JTward, JT Ratnanather, MI Miller, D Frush, et al. The development of a population of 4d pediatric xcat phantoms for imaging research and optimization. *Medical physics*, 42(8):4719–4726, 2015. 7

Emil Y Sidky and Xiaochuan Pan. Image reconstruction in circular cone-beam computed tomography by constrained, total-variation minimization. *Physics in Medicine & Biology*, 53(17):4777, 2008. 7

Tieleman and G. Hinton. RMSProp: Divide the gradient by a running average of its recent magnitude. In *Lecture Note 6-5, University of Toronto*, 2012. 3

K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Trans. Image Processing*, 26(7):3142 – 3155, 2017. 4

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 *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 8

Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image restoration. *arXiv preprint arXiv:1812.10477*, 2018. 3, 4