SELF-SUPERVISED BAYESIAN DEEP LEARNING FOR IMAGE DENOISING

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep learning is currently one prominent approach for image denoising, and most of existing works train a denoising neural network (NN) on many pairs of noisy images and their clean counterparts. Recent studies showed that it is possible to train a denoising NN on a dataset consisting of only noisy images. This paper took one step further to study how to train a powerful denoising NN for a given image without any training samples, which is appealing to the applications where collecting training samples is challenging. For instance, biological imaging and medical imaging. Built on the Bayesian neural network (BNN), this paper proposed a self-supervised deep learning method for denoising a single image, in the absence of training samples. The experiments showed that the performance of our self-supervised method is very competitive to those state-of-the-art supervised ones.

1 INTRODUCTION

Image denoising refers to removing measurement noise from images to have better signal-to-noise-ratio. It is a very basic task seen in a wide range of applications in imaging systems. Also, it plays an essential role in image recovery as it serves as one fundamental module in many image recovery methods. A noisy image $\boldsymbol{y}$ is usually modeled by

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{n}, \tag{1}$$

where $\boldsymbol{n}$ denotes the random measurement noise and $\boldsymbol{x}$ denotes the clean image. In last few decades, image denoising has been extensively studied with an abundant literature. Recently, deep learning, especially supervised deep learning (see *e.g*. Vemulapalli et al. (2016); Zhang et al. (2017; 2018)), has become one promising tool for image denoising. These supervised deep learning methods train an NN over many pairs of noisy images and their clean counterparts (noisy/clean) to learn the mapping between the noisy one and its clean counterpart. The performance of such supervised methods is heavily dependent on the availability of a great amount of high-quality noisy/clean image pairs that are relevant to the images for processing. The collection of such a training dataset is often time-consuming and expensive. In many fields of applications, it is very challenging and even impossible, *e.g*. medical images of patients and scientific imaging of biological particles.

Very recently, there has been a rapid progress on developing deep-learning-based image denoisers that do not require a training dataset with noisy/clean image pairs. For instance, Noise2Noise (N2N) (Lehtinen et al., 2018) trains a denoising NN on a dataset with the pairs of noisy images of the same scenes (noisy/noisy). Noise2Void (N2V) (Krull et al., 2019), Noise2Self (N2S) (Batson & Royer, 2019), SURE (Soltanayev & Chun, 2018) and Laine et al. (2019a) train denoising NNs on a set of noisy images without correspondence. Generative adversarial network (GAN) based methods (Cha et al., 2019; Chen et al., 2018) synthesize paired noisy/clean (or noisy/noisy) images from un-paired noisy/clean images (or un-paired noisy images) for supervising the training of a denoising NN. The *deep image prior* (DIP) proposed by Ulyanov et al. (2018) showed that by using *early stopping*, it is possible to train an NN to denoise an input image, without any external training samples.

1.1 DISCUSSION

A self-supervised deep-learning-based image denoiser that does not require any training sample except the target noisy image itself, is of great practical value. The DIP (Ulyanov et al., 2018) showed

that one can train an NN for denoising an image without any external sample. Indeed, it is not surprising from the viewpoint of non-local methods. Different from the classification task that relies on the extraction of high-level global features, image denoising focuses more on local details of images. Take the well-known non-local denoising method, BM3D (Dabov et al., 2007), for example. It treats an image as the collection of many noisy local patches with strong recurrence. For each target patch, the BM3D method finds a group of similar patches. By viewing such a group of patches as multiple instances of the same patch corrupted by independent measurement noise, the BM3D method estimates the clean target patch with the state-of-the-art performance. In other words, a single image itself provides sufficient statistical information for denoising each local patch in the image. Thus, it is possible to train an NN for image denoising only on the input noisy image itself.

When training an NN on only the input noisy image $\boldsymbol{y}$, one issue needed to be addressed is how to avoid the convergence to the identity map $\boldsymbol{I}$. Denote the denoising NN by $\mathcal{F}_{\boldsymbol{\theta}}$. It can be seen that

$$\|\mathcal{F}_{\boldsymbol{\theta}}(\boldsymbol{y}) - \boldsymbol{y}\| = \|\boldsymbol{I}(\boldsymbol{y}) - \boldsymbol{y}\| = \|\boldsymbol{y} - \boldsymbol{y}\| = \boldsymbol{0}, \text{ when } \mathcal{F}_{\boldsymbol{\theta}} = \boldsymbol{I}. \tag{2}$$

One simple yet effective technique is calling some data augmentation to avoid the convergence to the identity map. Conceptually, the blind-spot technique in N2V Krull et al. (2019) and N2S Batson & Royer (2019) generates a set of noisy training samples $\{(\widehat{\boldsymbol{y}}_m, \overline{\boldsymbol{y}}_m)\}_{m=1}^M$ by:

$$\widehat{\boldsymbol{y}}_m := \boldsymbol{y} \odot \boldsymbol{b}_m, \quad \overline{\boldsymbol{y}}_m = \boldsymbol{y} \odot (\boldsymbol{1} - \boldsymbol{b}_m), \quad 1 \leq m \leq M, \tag{3}$$

where $\odot$ denotes entry-wise multiplication, $\boldsymbol{b}_m$s are the instances sampled from the binary Bernoulli matrix $\boldsymbol{b}$ with probability p. The idea is to predict the values of a set of randomly chosen pixels by their neighboring pixel values, *i.e.*, predicting the clean version of $\overline{\boldsymbol{y}}_m$ by using $\widehat{\boldsymbol{y}}_m$ as the NN input. With only the generated samples $\{(\widehat{\boldsymbol{y}}_m, \overline{\boldsymbol{y}}_m)\}_{m=1}^M$ from the single noisy image $\boldsymbol{y}$, the NN is trained by finding the maximum likelihood estimate as follows:

$$\min_{\boldsymbol{\theta}} \log p(\boldsymbol{y}|\boldsymbol{\theta}) \Longleftrightarrow \min_{\boldsymbol{\theta}} \log p(\{(\widehat{\boldsymbol{y}}_m, \overline{\boldsymbol{y}}_m)\}_{m=1}^M|\boldsymbol{\theta}) \Longleftrightarrow \min_{\boldsymbol{\theta}} \sum_{m=1}^{M} \|(\boldsymbol{1} - \boldsymbol{b}_m) \odot \mathcal{F}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{y}}_m) - \overline{\boldsymbol{y}}_m)\|_2^2, \tag{4}$$

for Gaussian noise removal. Unfortunately, it is empirically observed that the NN trained by the augmented samples $\{(\widehat{\boldsymbol{y}}_m, \overline{\boldsymbol{y}}_m)\}_{m=1}^M$ still suffers a lot from the overfitting effect; see Table 1 for the experimental results (labeled as "N2V(1)" and "N2S(1)"). In comparison to the N2V trained over a set of noisy images, such a significant performance loss is not surprising. From the viewpoint of statistical inference, reducing the number of samples will significantly increase the variance of the estimate. As all training samples are augmented from a single noisy image $\boldsymbol{y}$, such a set does not provide sufficient diversity for network training. A typical approach to reducing the variance of prediction is adding some regularization to the estimator.

1.2 Main idea

This paper aims at developing an estimator for image denoising with competitive performance to those state-of-the-art denoising methods, while no external training sample is required. We propose to use BNN (Blundell et al., 2015), an NN whose weights are random variables, to approximate the minimum mean square error (MMSE) estimate of the truth image $\boldsymbol{x}$. The main idea of using BNN for self-supervised denoising is listed as follows.

- It introduces weight uncertainty to reduce the possible model bias caused by the NN architecture; see *e.g.* Barber & Bishop (1998); Blundell et al. (2015); Kendall & Gal (2017); Gal & Ghahramani (2016).
- It trains an ensemble of NNs and provides a Bayesian averaging prediction that has lower variance than a single prediction; see *e.g.* Baldi & Sadowski (2013); Lakshminarayanan et al. (2017).

Recall that for a BNN, denoted by $\mathcal{F}_{\boldsymbol{\theta}}$, the weights $\boldsymbol{\theta} = \{\theta_i\}$ are random variables. The network input $\widehat{\boldsymbol{y}}$ defined in (3) is also a random vector parameterized by $\boldsymbol{b}$. In Bayesian statistics, the truth $\boldsymbol{x}$ is also represented as a random variable. Here it is related to $\boldsymbol{\theta}$ and $\widehat{\boldsymbol{y}}$ by

$$\boldsymbol{x} = \mathcal{F}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{y}}). \tag{5}$$

Our aim is to find an MMSE estimate of $\boldsymbol{x}$:

$$\widetilde{\boldsymbol{x}} := \arg\min_{\boldsymbol{u}} \mathbb{E}_{(\boldsymbol{x}|\boldsymbol{y})} \|\boldsymbol{u} - \boldsymbol{x}\|_2^2 = \mathbb{E}_{(\boldsymbol{x}|\boldsymbol{y})}(\boldsymbol{x}|\boldsymbol{y}) = \int \boldsymbol{x} p(\boldsymbol{x}|\boldsymbol{y}) d\boldsymbol{x}. \tag{6}$$

It is further given by

$$\widetilde{\boldsymbol{x}} = \iiint \boldsymbol{x} p(\boldsymbol{x}|\boldsymbol{\theta}, \widehat{\boldsymbol{y}}) p(\boldsymbol{\theta}, \widehat{\boldsymbol{y}}|\boldsymbol{y}) d\boldsymbol{\theta} d\widehat{\boldsymbol{y}} d\boldsymbol{x} = \iint \mathcal{F}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{y}}) p(\boldsymbol{\theta}, \widehat{\boldsymbol{y}}|\boldsymbol{y}) d\widehat{\boldsymbol{y}} d\boldsymbol{\theta} = \iiint \mathcal{F}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{y}}) p(\boldsymbol{\theta}|\boldsymbol{y}) p(\boldsymbol{b}) d\boldsymbol{b} d\boldsymbol{\theta}, \tag{7}$$

where $p(\boldsymbol{\theta}|\boldsymbol{y})$ is the posterior probability distribution function of $\boldsymbol{\theta}$. As the explicit form of $p(\boldsymbol{\theta}|\boldsymbol{y})$ is intractable in practice, it is approximated by the joint distribution of independent normal distributions $q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\sigma})$:

$$q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\sigma}): \ \ \theta_i \sim \mathcal{N}(\mu_i, \sigma_i^2),\ \boldsymbol{\theta} = \{\theta_i\},\ \boldsymbol{\mu} = \{\mu_i\},\ \boldsymbol{\sigma} = \{\sigma_i\}. \tag{8}$$

In other words, we consider the BNN whose weights follow a distribution parameterized by $\boldsymbol{\mu}$ and $\boldsymbol{\theta}$. Then, The objective of training a BNN is about minimizing the KL divergence between $q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\sigma})$ and $p(\boldsymbol{\theta}|\boldsymbol{y})$:

$$(\boldsymbol{\mu}^*, \boldsymbol{\sigma}^*) = \arg\min_{\boldsymbol{\mu}, \boldsymbol{\sigma}} \mathrm{KL}(q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\sigma}) \| p(\boldsymbol{\theta}|\boldsymbol{y})). \tag{9}$$

See Buntine & Weigend (1991); MacKay (1992) for more details on BNN. Once the model is trained with learned distribution parameters $\boldsymbol{\mu}^*$ and $\boldsymbol{\sigma}^*$, we have an approximation to the MMSE estimate:

$$\boldsymbol{x}^* = \iint \mathcal{F}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{y}}) q(\boldsymbol{\theta}|\boldsymbol{\mu}^*, \boldsymbol{\sigma}^*) p(\boldsymbol{b}) d\boldsymbol{b} d\boldsymbol{\theta}. \tag{10}$$

See Section 3 for more details.

In summary, built on the BNN-based approximation to the MMSE estimation, this paper proposed a self-supervised learning method for image denoising, which does not call any external training sample. The experiments on both Gaussian noise removal and real noise removal showed that the proposed one outperformed existing non-learning methods and un-supervised methods in the same setting. When compared to the deep learning methods trained on a set of training samples (either paired or un-paired), the proposed method still provided competitive performance.

2 Related work

Non-learning based methods. In the past, many non-learning methods were proposed for image denoising by imposing certain empirical prior on the target image. For example, the ℓ_1-norm relating regularization methods (Chambolle, 2004; Rudin et al., 1992) which impose Hyper-Laplacian prior on image gradients. The non-local methods (Dabov et al., 2007; Gu et al., 2014; Dong et al., 2012) process the stack of similar image patches to exploit the recurrence prior of local image patches over the image. These non-local methods indeed provide state-of-the-art results among non-learning methods.

Deep-learning-based denoisers supervised on image pairs. In recent years, deep learning has become one powerful tool for image denoising. The majority of these deep learning methods require a large dataset of noisy/clean image pairs to train the NNs that map the noisy image to its clean counterpart; see *e.g*. Burger et al. (2012); Zhang et al. (2017); Chen et al. (2018); Vemulapalli et al. (2016); Zhang et al. (2018); Lefkimmiatis (2018); Guo et al. (2019); Jia et al. (2019). Among them, DnCNN (Zhang et al., 2017) is often used as the baseline method for the performance evaluation of image denoising methods. Instead of using noisy/clean image pairs for training, the N2N (Lehtinen et al., 2018) uses a set of noisy/noisy image pairs corresponding to the same scene but with independent measurement noise. It is shown that the performance of the NN trained on noisy/noisy pairs is competitive to those trained on noisy/clean pairs.

Deep-learning-based denoisers trained on un-paired external images. Recently, there has been a rapid progress on deep learning based denoising methods that only require a set of noisy images for NN training. One approach is based on GAN, *e.g*. Chen et al. (2018) learns denoisers with un-paired noisy and clean training images and Cha et al. (2019) with a set of un-paired noisy images. The basic idea is using GAN to build a paired dataset for training the denoiser. Another approach is the so-called self-supervised denoisers (Batson & Royer, 2019; Krull et al., 2019; Laine et al., 2019b)

which use a set of un-paired noisy images to train the NNs. The basic idea is designing a specific loss function to avoid the convergence of the NN to the identify map. N2V (Krull et al., 2019) proposed a blind-spot strategy that predicts the central pixel by its neighborhood pixels. N2S (Batson & Royer, 2019) used a similar technique and Laine et al. (2019b) designed a special blind-spot NN architecture to exclude the center pixel in its receptive field. Based on Stein's Unbiased Risk Estimator (SURE), some works (Soltanayev & Chun, 2018; Metzler et al., 2018) proposes to regularize the NN by penalizing the divergence of the gradient.

Denoisers learned without any external image. Our method falls into this category. These methods tackle the problem of how to learn a denoiser with only the input noisy image available. Earlier approaches are based on sparse coding (Elad & Aharon, 2006; Bao et al., 2015; Papyan et al., 2017), which learn a dictionary from the input noisy image such that image patches have a sparse representation under the learned dictionary. For deep learning, there are a few studies along this line. The seminal work is the DIP method (Ulyanov et al., 2018), which showed that an NN can learn regular image patterns prior to random noise during the training. Thus, by using the early stopping strategy during training, one can obtain an NN that generates the target clean image without noise. One issue of DIP is that when to stop before noise showing up is not easy to determine. Another more serious one is that DIP is not competitive to the state-of-the-art non-learning methods. The aforementioned N2V and N2S can be extended to train on the single noisy image. Similarly, their performance is not competitive either.

3 Main Body

This section gives a detailed discussion on the proposed BNN for denoising a single image without any external training samples. Recall that noisy image $\boldsymbol{y}$ and the clean image $\boldsymbol{x}$ are related by

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{n}, \tag{11}$$

where $\boldsymbol{n}$ denotes random noise.

Let $\mathcal{F}_{\boldsymbol{\theta}}$ denote the BNN whose weights $\boldsymbol{\theta} = \{\theta_i\}$ are random variables. The network is trained over the generated paired samples $(\widehat{\boldsymbol{y}}, \overline{\boldsymbol{y}})$:

$$\widehat{\boldsymbol{y}} = \boldsymbol{b} \odot \boldsymbol{y}, \quad \overline{\boldsymbol{y}} = (\boldsymbol{1} - \boldsymbol{b}) \odot \boldsymbol{y}, \quad \boldsymbol{b} \sim \mathcal{B}(p). \tag{12}$$

where $\widehat{\boldsymbol{y}}$ is the network input, $\overline{\boldsymbol{y}}$ is the target and $\boldsymbol{b}$ is randomly sampled from Bernoulli matrix $\mathcal{B}(p)$. Our goal is to approximate the MMSE estimate of $\boldsymbol{x}$, denoted by $\widetilde{\boldsymbol{x}}$, which can be expressed as

$$\widetilde{\boldsymbol{x}} = \int\int \mathcal{F}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{y}}) p(\boldsymbol{\theta}|\boldsymbol{y}) p(\boldsymbol{b}) d\boldsymbol{b} d\boldsymbol{\theta}. \tag{13}$$

See (6) and (7) for the derivation. Recall that $\boldsymbol{\theta}$ is the vector of network weights whose dimension is tremendously high and the network $\mathcal{F}_{\boldsymbol{\theta}}$ is also highly complex and non-linear. Thus, the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{y})$ is intractable in practice. Instead, we approximate $p(\boldsymbol{\theta}|\boldsymbol{y})$ by the independent joint normal distribution $q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\sigma})$:

$$\theta_i \sim \mathcal{N}(\mu_i, \sigma_i), \tag{14}$$

where the mean $\boldsymbol{\mu} = \{\mu_i\}$ and standard deviation $\boldsymbol{\sigma} = \{\sigma_i\}$ of the normal distribution are the parameters to be estimated.

3.1 Training

Our training goal is to optimize the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ to make $q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\sigma})$ as close as possible to the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{y})$. This goal is achieved by minimizing the KL divergence between $q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\sigma})$ and $p(\boldsymbol{\theta}|\boldsymbol{y})$:

$$\min_{\boldsymbol{\mu}, \boldsymbol{\sigma}} \mathrm{KL}(q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\sigma}) \| p(\boldsymbol{\theta}|\boldsymbol{y})). \tag{15}$$

The KL divergence can be rewritten as

$$\begin{aligned} & \mathrm{KL}(q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\sigma}) \| p(\boldsymbol{\theta}|\boldsymbol{y})) \\ = \; & \mathrm{KL}(q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\sigma}) \| p(\boldsymbol{\theta})) - \mathbb{E}_{\boldsymbol{\theta} \sim q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\sigma})} \log p(\boldsymbol{y}|\boldsymbol{\theta}) + \mathrm{const}, \end{aligned} \tag{16}$$

where $p(\boldsymbol{\theta})$ is the prior distribution of the NN weights $\boldsymbol{\theta}$. However, the KL-divergence between $q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\sigma})$ and $p(\boldsymbol{\theta})$ in (16) is still difficult to estimate for a general distribution $p(\boldsymbol{\theta})$. Thus, we

further simplify the optimization problem by assuming that the prior distribution $p(\boldsymbol{\theta})$ is a joint distribution of i.i.d. normal distributions with zero mean and standard deviation $\widetilde{\sigma}$.

As for the likelihood function term $\mathbb{E}_{\boldsymbol{\theta}\sim q(\boldsymbol{\theta}|\boldsymbol{\mu},\boldsymbol{\sigma})} \log p(\boldsymbol{y}|\boldsymbol{\theta})$, it depends on the statistics of $\boldsymbol{n}$. Suppose the components of $\boldsymbol{n}$ follow i.i.d. normal distribution $\mathcal{N}(0, \overline{\sigma})$. We obtain the following proposition.

Proposition 1. Suppose $p(\boldsymbol{n}) \sim \prod_i \exp(\frac{-\boldsymbol{n}_i^2}{2\overline{\sigma}^2})$ and $p(\boldsymbol{\theta}) \sim \prod_i \exp(\frac{-\boldsymbol{\theta}_i^2}{2\widetilde{\sigma}^2})$. Then, we have

$$\begin{aligned} & \min_{\boldsymbol{\mu},\boldsymbol{\sigma}} \mathrm{KL}(q(\boldsymbol{\theta}|\boldsymbol{\mu},\boldsymbol{\sigma})\|p(\boldsymbol{\theta}|\boldsymbol{y})) \\ \Longleftrightarrow \quad & \min_{\boldsymbol{\mu},\boldsymbol{\sigma}} \sum_{\boldsymbol{b}\sim\mathcal{B}(p)} \mathbb{E}_{\boldsymbol{\theta}\sim q(\boldsymbol{\theta}|\boldsymbol{\mu},\boldsymbol{\sigma})} \|(\mathbf{1}-\boldsymbol{b}) \odot (\mathcal{F}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{y}}) - \boldsymbol{y})\|_2^2 + \lambda_1(\|\boldsymbol{\mu}\|_2^2 + \|\boldsymbol{\sigma}\|_2^2) - \lambda_2 \textstyle\sum_i \log \sigma_i, \end{aligned} \tag{17}$$

where $\lambda_1 = \overline{\sigma}^2/\widetilde{\sigma}^2$ and $\lambda_2 = 2\overline{\sigma}^2$.

Proof. See the supplementary file for the detailed derivation. □

In implementation, we only sample one instance of $\boldsymbol{\theta}$ from the distribution $q(\boldsymbol{\theta}|\boldsymbol{\mu},\boldsymbol{\sigma})$ at each iteration to approximate the expectation in the first data-dependent term of (17). In addition, the mini-batch gradient descent with mini-batch size of 1, that is, the stochastic gradient descent, is employed to update the model when using the generated Bernoulli masks from $\mathcal{B}(p)$.

It is noted that the standard deviation σ_i should be always positive. Thus, we adopt the re-parameterization trick in Blundell et al. (2015) here, which re-expresses σ_i by

$$\sigma_i = \log(1 + \exp(\rho_i)). \tag{18}$$

During every forward process of BNN, we generate the network weights by

$$\theta_i = \mu_i + \log(1 + \exp(\rho_i)) \cdot \epsilon_i, \tag{19}$$

where ϵ_i is sampled from the standard normal distribution $\mathcal{N}(0, 1)$. Then, at the subsequent backward procedure, each ϵ_i is fixed for the calculation of gradients. More details on the training of BNN via back-propagation can be found in the related materials.

3.2 Testing

Once the training of BNN is completed with the optimized parameters $\boldsymbol{\mu}^*, \boldsymbol{\sigma}^*$, we obtain an approximation to the posterior probability distribution $p(\boldsymbol{\theta}|\boldsymbol{y})$, *i.e.* $q(\boldsymbol{\theta}|\boldsymbol{\mu}^*, \boldsymbol{\sigma}^*)$. The approximate MMSE estimate is then given by

$$\boldsymbol{x}^* = \iint \mathcal{F}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{y}}) q(\boldsymbol{\theta}|\boldsymbol{\mu}^*, \boldsymbol{\sigma}^*) p(\boldsymbol{b}) d\boldsymbol{b} d\boldsymbol{\theta}. \tag{20}$$

Although $\mathcal{F}_{\boldsymbol{\theta}}(\cdot)$ and $q(\boldsymbol{\theta}|\boldsymbol{\mu}^*, \boldsymbol{\sigma}^*)$ have explicit forms, the above integration is still intractable. In practice, we use Monte Carlo (MC) integration to compute it:

$$\boldsymbol{x}^* \approx \frac{1}{T} \sum_{j=1}^{T} \mathcal{F}_{\boldsymbol{\theta}^j}((\mathbf{1} - \boldsymbol{b}^j) \odot \boldsymbol{y}), \tag{21}$$

where $\{\boldsymbol{\theta}^j\}$ (or $\{\boldsymbol{b}^j\}$) are sampled from the distribution $q(\boldsymbol{\theta}|\boldsymbol{\mu}^*, \boldsymbol{\sigma}^*)$ (or $\mathcal{B}(p)$) and T is the total sampling number.

4 Experiments

In this section, we evaluate the performance of the proposed method on denoising images with Gaussian white noise and real-world noisy images. Due to space limitation, only partial results are reported in main manuscript. More results can be found in the supplementary file.

Implementation details. We adopted a UNet with skip-connections shown in Figure 1. For all convolution layers, the kernel size is 3×3, and stride/padding number is 1. The upsampling is done by bi-linear interpolation. The negative slope of all leaky ReLUs is 0.01. The BNN parameter $\boldsymbol{\mu}$ is initialized using normal distribution as He et al. (2015) while the initial value of $\boldsymbol{\rho}$ is drawn uniformly

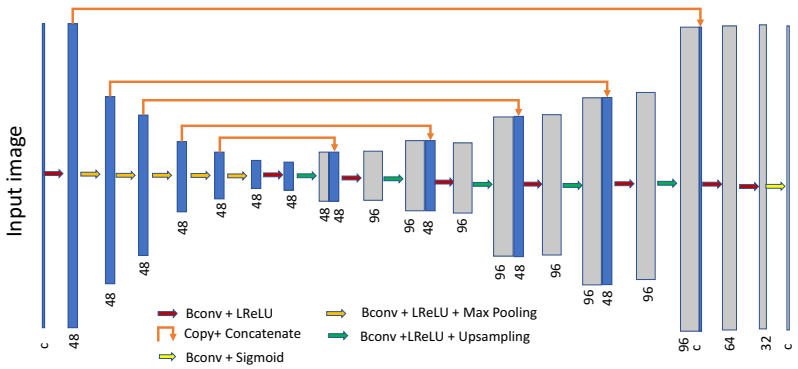


Figure 1: The NN architecture of our method on one noisy image of channel c (=1 or 3). Boxes represent the hidden layers with their channel numbers indicated on the bottom. Arrows of different colors stand for operations defined by the legends. Specifically, Bconv stands for the Bayesian convolution operation, where the weights are random variables in the form of (19).

from $[-5, -4]$. The method is implemented in Pytorch. We trained the model using Adam with learning rate 10^{-4}. The sampling number T used in the MC approximation during prediction is set to 100. Note that with the Bernoulli sampled image $\widehat{\boldsymbol{y}}$ as input, we fill the non-sampled pixels with the averaging of their 3×3 neighboring pixel values in $\widehat{\boldsymbol{y}}$. The sampling ratio p of Bernoulli matrix $\boldsymbol{b}$ is set to 0.7.

Remark 1 (Comparison among different methods)**.** For the compared methods, we cite the results directly from the literature if possible. Otherwise, we followed the authors' instructions to train the model using the codes provided by the authors and made our effort to tune the parameters to obtain the optimal results. If none is available, we leave it blank in the table.

4.1 Removing Gaussian white noise from images

Two datasets are used for performance evaluation, *i.e.*, Set9 used in Ulyanov et al. (2018) with 9 color images and BSD68 used in Krull et al. (2019) with 68 gray-scale images. Following others, the experiments on noise level $\overline{\sigma} = 25, 50, 75, 100$ are conducted on Set9 and $\overline{\sigma} = 25, 50$ on BSD68. The parameters λ_1, λ_2 in our training model (17) are updated as follows:

$$\lambda_1 = 0.01 \times \overline{\sigma}^2, \ \lambda_2 = 0.05 \times \overline{\sigma}^2, \tag{22}$$

with the noise level $\overline{\sigma}$ given. The maximum iteration K for training is set to be 10^5.

Several representative single-image-based denoising methods with published codes are selected for performance comparison: KSVD (Elad & Aharon, 2006), BM3D (Dabov et al., 2007) and DIP (Ulyanov et al., 2018). Since DIP is sensitive to the iteration number for different noise levels (see Figure 2), we revised it to improve the performance by stopping the iteration when the residual reaches the given noise level, termed by DIP*.

Except the single-image based methods, the recent dataset-based deep learning methods are also tested for comparison, including N2V (Krull et al., 2019), N2S (Batson & Royer, 2019), SURE (Soltanayev & Chun, 2018), Laine et al. (2019a), N2N (Lehtinen et al., 2018) and DnCNN (Zhang et al., 2017). Recall that N2V, N2S, SURE and Laine et al. (2019a) are trained on unorganized noisy images, N2N on paired noisy images, and DnCNN on noisy/clean image pairs. For N2V, N2S, N2V, Laine *et al*. and DnCNN, we use their published codes and follow the corresponding data generation scheme to train models with specific noise level on CBSD400 and CBSD400's gray-scale version for color/gray-scale image denoising respectively. For SURE, we cite the results from the original paper and leave it blank if no result is available in its publication. In addition, we also compare our method to the single-image extension of N2V and N2S, which is denoted by N2V(1) and N2S(1).

See Table 1 for the comparison to non-learning methods and the methods without calling external training images. The results showed that our method is the best performer among all with noticeable advantage over others. See Table 2 for the comparison to other learning methods which call external

datasets for training. Our method remains the best performer on Set9 and has medium performance on BSD68. Note that such a comparison is not very fair: no training data used in ours vs. training dataset used in others. With such an inherent disadvantage, ours still provides competitive performance.

Table 1: Average PSNR(dB)/SSIM of the methods w/o training samples for removing Gaussian noise

Dataset	$\overline{\sigma}$	KSVD	BM3D	N2V(1)	N2S(1)	DIP*	Ours
Set9	25	30.00/0.935	31.67/0.955	28.12/0.912	29.30/0.940	30.77/0.942	**31.68/0.958**
	50	26.50/0.870	28.95/0.922	26.01/0.875	27.25/0.904	28.23/0.910	**29.39/0.930**
	75	24.29/0.810	27.36/0.895	24.18/0.827	25.85/0.861	26.64/0.883	**27.88/ 0.909**
	100	23.12/0.770	26.04/0.868	23.55/0.780	24.67/0.848	25.41/0.858	**26.58/0.889**
BSD68	25	28.42/0.796	28.56/0.801	25.34/0.681	27.19/0.769	27.96/0.774	**28.57/0.802**
	50	25.08/0.653	25.62/0.687	23.85/0.618	24.53/0.642	25.04/0.645	**25.93/0.698**

Table 2: Average PSNR(dB)/SSIM of the methods w/ training data for removing Gaussian noise

Dataset	$\overline{\sigma}$	N2V	N2S	Laine *et al.*	SURE	N2N	DnCNN	Ours
Set9	25	30.660.947	30.05/0.944	30.89/0.953	31.19/–	31.33/0.957	31.56/0.958	**31.68/0.958**
	50	27.81/0.912	27.51/0.905	28.03/0.916	28.55/–	28.94/0.929	28.67/0.924	**29.39/0.930**
	75	25.99/0.875	26.49/0.882	26.39/0.884	–/–	27.42/0.905	27.08/0.895	**27.88/ 0.909**
	100	25.37/0.858	25.46/0.857	24.77/0.841	–/–	26.45/0.886	25.82/0.865	**26.58/0.889**
BSD68	25	27.72/0.794	28.12/0.792	28.84/0.814	28.97/–	28.86/0.823	**29.19/0.827**	28.57/0.802
	50	25.12/0.684	25.62/0.678	25.78/0.698	25.93/–	25.77/0.700	**26.20/0.718**	25.93/0.698

4.2 Removing real-world image noise

For real-world image denoising, we test the dataset CC (Nam et al., 2016) which contains 15 real noisy images captured by several different consumer cameras. As our training model (17) is derived under the Gaussian white noise assumption, we made some modifications for denoising real-wold images. We corrupt the real-world noisy image $\boldsymbol{y}$ by independent Gaussian white noise of standard deviation $\widehat{\sigma}$ at each iteration:

$$\boldsymbol{y}' = \boldsymbol{y} + \boldsymbol{n}', \quad \widehat{\boldsymbol{y}}' = \boldsymbol{b} \circ \boldsymbol{y}', \quad \boldsymbol{n}'_i \sim \mathcal{N}(0, \widehat{\sigma}), \tag{23}$$

and then feed $\widehat{\boldsymbol{y}}'$ to model (17):

$$\min_{\boldsymbol{\mu},\boldsymbol{\sigma}} \sum_{\boldsymbol{b},\boldsymbol{n}'} \mathbb{E}_{\boldsymbol{\theta}\sim q(\boldsymbol{\theta}|\boldsymbol{\mu},\boldsymbol{\sigma})} \|(\mathbf{1}-\boldsymbol{b}) \odot (\mathcal{F}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{y}}') - \boldsymbol{y})\|_2^2 + \lambda_1(\|\boldsymbol{\mu}\|_2^2 + \|\boldsymbol{\sigma}\|_2^2) - \lambda_2 \sum_i \log \sigma_i. \tag{24}$$

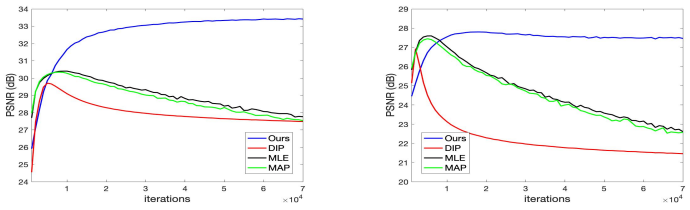
The injected noise level $\widehat{\sigma}$ is set to 30. The parameters λ_1 and λ_2 are set as (22) with estimated $\overline{\sigma} = 5$. The maximum iteration number K is 5×10^3.

Table 3: Average PSNR(dB)/SSIM of real-world noise removal results on CC.

KSVD	CBM3D	N2V(1)	N2S(1)	DIP
36.41/0.946	35.19/0.858	32.27/0.862	33.38/0.846	35.69/0.926
MCWNNM	**TWSC**	**NC**	**DnCNN**	**Ours**
37.70/0.954	37.81/**0.959**	36.43/0.936	33.86/0.864	**37.85**/0.955

As supervised learning is sensitive to the noise patterns in training samples, we mainly compare our method with those that do not call any external training data, except the pre-trained model of DnCNN. Recall that DnCNN has the state-of-the-art performance on Gaussian noise removal. In addition, we also include three methods specifically designed for denoising real-world images: multi-channel weighted nuclear norm minimization (MCWNNM) (Xu et al., 2017), trilateral weighted sparse coding (TWSC) (Xu et al., 2018), and "noise clinic" (NC) method (Lebrun et al., 2015).

See Table 3 for the comparison. Our method noticeably outperformed traditional non-learning methods (KSVD and CBM3D) and deep learning methods (N2V(1), N2S(1) and DIP) trained without



(a) $\overline{\sigma} = 50$ (b) $\overline{\sigma} = 100$

Figure 2: PSNR over iterations of different methods for Gaussian white noise removal on the natural image "F16" in Set9. Our method is more stable to the iteration number than the other three in terms of PSNR value.

external images. In comparison to three state-of-the-art methods designed for real noise removal, the performance of our method is also very competitive. Note that the unsatisfactory performance of DnCNN is caused by different noise characteristic between training samples and tested images: one is Gaussian noise and the other is real-world noise.

4.3 Ablation Study

Ablation studies of the proposed approach are conducted on Set9 for Gaussian white denoising. This study is to check how much performance improvement weight certainty of BNN actually contributes. Two deterministic versions of the BNN are used as baselines for the comparison. One is the maximum likelihood estimator (MLE) which trains the NN with deterministic weights by

$$\boldsymbol{\theta}^{\mathrm{MLE}} = \arg\min_{\boldsymbol{\theta}} \log p(\boldsymbol{y}|\boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \sum_{\boldsymbol{b}} \|(\mathbf{1}-\boldsymbol{b}) \odot (\mathcal{F}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{y}}) - \boldsymbol{y})\|_2^2, \tag{25}$$

and makes prediction with $\boldsymbol{x}^{\mathrm{MLE}} = \mathbb{E}_{\boldsymbol{b}} \mathcal{F}_{\boldsymbol{\theta}^{\mathrm{MLE}}}(\widehat{\boldsymbol{y}})$. The other is the maximum a posterior (MAP) estimator which assumes a Gaussian prior on the weights of the deterministic NN:

$$\begin{aligned} \boldsymbol{\theta}^{\mathrm{MAP}} &= \arg\min_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}|\boldsymbol{y}) = \arg\min_{\boldsymbol{\theta}} \log p(\boldsymbol{y}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) \\ &= \arg\min_{\boldsymbol{\theta}} \textstyle\sum_{\boldsymbol{b}} \|(\mathbf{1}-\boldsymbol{b})(\odot \mathcal{F}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{y}}) - \boldsymbol{y})\|_2^2 + \gamma \overline{\sigma}^2 \|\boldsymbol{\theta}\|_2^2, \end{aligned} \tag{26}$$

where γ is set to 0.01 after rigorous tuning-up and $\boldsymbol{x}^{\mathrm{MAP}} = \mathbb{E}_{\boldsymbol{b}} \mathcal{F}_{\boldsymbol{\theta}^{\mathrm{MAP}}}(\widehat{\boldsymbol{y}})$. Similar as DIP, MLE and MAP are sensitive to the iteration number. See Figure 2 for illustration. So we stop the iteration for MLE and MAP if the residual reaches the given noise level. See Table 4 for the comparison with our method. The BNN with random weights significantly outperformed the other two versions of deterministic NN. Such a comparison clearly indicates the effectiveness of weight uncertainty in BNN on handling the overfitting and correcting the model bias for self-supervised image denoising.

Table 4: Average PSNR(dB)/SSIM of ablation studies on Set9 for Gaussian noise removal.

$\overline{\sigma}$	25	50	75	100
MLE	30.58/0.941	28.17/0.915	25.88/0.864	24.59/0.845
MAP	30.61/0.945	28.25/0.916	26.17/0.872	25.03/0.861
Ours	**31.68/0.958**	**29.39/0.930**	**27.88/ 0.909**	**26.58/0.889**

5 Conclusion

Based on the framework of BNN, we proposed a new self-supervised deep learning method for image denoising, which does not require any external image for training. The uncertainty introduced by the BNN provides substantial benefit on resolving the overfitting caused by too few training samples. The experiment results show that our method noticeably outperformed non-learning methods and the other learning methods in the same category. Despite the disadvantage on training samples, our method still provided competitive performance to those deep learning methods that require (structured or unstructured) external dataset for training. The idea in this paper has the potential to see its applications in other image recovery tasks.

REFERENCES

Pierre Baldi and Peter J Sadowski. Understanding dropout. In *NeurIPS*, pp. 2814–2822, 2013.

Chenglong Bao, Hui Ji, Yuhui Quan, and Zuowei Shen. Dictionary learning for sparse coding: Algorithms and convergence analysis. *Trans. Pattern Anal. Mach. Intell.*, 38(7):1356–1369, 2015.

David Barber and Christopher M Bishop. Ensemble learning in bayesian neural networks. *Nato ASI Series F Computer and Systems Sciences*, 168:215–238, 1998.

Joshua Batson and Loic Royer. Noise2self: Blind denoising by self-supervision. *Proc. ICML*, 2019.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *ICML*, pp. 1613–1622, 2015.

Wray L Buntine and Andreas S Weigend. Bayesian back-propagation. *Complex systems*, 5(6): 603–643, 1991.

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

Sungmin Cha, Taeeon Park, and Taesup Moon. Gan2gan: Generative noise learning for blind image denoising with single noisy images. *arXiv preprint arXiv:1905.10488*, 2019.

Antonin Chambolle. An algorithm for total variation minimization and applications. *J. Math. Imaging Vision*, 20(1-2):89–97, 2004.

Jingwen Chen, Jiawei Chen, Hongyang Chao, and Ming Yang. Image blind denoising with generative adversarial network based noise modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3155–3164, 2018.

Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen O Egiazarian. Color image denoising via sparse 3d collaborative filtering with grouping constraint in luminance-chrominance space. In *Proc. ICIP*, pp. 313–316, 2007.

Weisheng Dong, Lei Zhang, Guangming Shi, and Xin Li. Nonlocally centralized sparse representation for image restoration. *IEEE transactions on Image Processing*, 22(4):1620–1630, 2012.

Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. Image Process.*, 15(12):3736–3745, 2006.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, pp. 1050–1059, 2016.

Shuhang Gu, Lei Zhang, Wangmeng Zuo, and Xiangchu Feng. Weighted nuclear norm minimization with application to image denoising. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2862–2869, 2014.

Shi Guo, Zifei Yan, Kai Zhang, Wangmeng Zuo, and Lei Zhang. Toward convolutional blind denoising of real photographs. In *Proc. CVPR*, June 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, pp. 1026–1034, 2015.

Xixi Jia, Sanyang Liu, Xiangchu Feng, and Lei Zhang. Focnet: A fractional optimal control network for image denoising. In *Proc. CVPR*, pp. 6054–6063, 2019.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *NeurIPS*, pp. 5574–5584, 2017.

Alexander Krull, Tim-Oliver Buchholz, and Florian Jug. Noise2void-learning denoising from single noisy images. In *Proc. CVPR*, pp. 2129–2137, 2019.

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

Samuli Laine, Jaakko Lehtinen, and Timo Aila. Self-supervised deep image denoising. *arXiv preprint arXiv:1901.10277*, 2019b.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*, pp. 6402–6413, 2017.

Marc Lebrun, Miguel Colom, and Jean-Michel Morel. The noise clinic: a blind image denoising algorithm. *Image Processing On Line*, 5:1–54, 2015.

Stamatios Lefkimmiatis. Universal denoising networks: a novel cnn architecture for image denoising. In *Proc. CVPR*, pp. 3204–3213, 2018.

Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. Noise2noise: Learning image restoration without clean data. *Proc. ICML*, 2018.

David JC MacKay. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

Christopher A Metzler, Ali Mousavi, Reinhard Heckel, and Richard G Baraniuk. Unsupervised learning with stein's unbiased risk estimator. *arXiv preprint arXiv:1805.10531*, 2018.

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*, pp. 1683–1691, 2016.

Vardan Papyan, Yaniv Romano, Jeremias Sulam, and Michael Elad. Convolutional dictionary learning via local processing. In *Proc. ICCV*, pp. 5296–5304, 2017.

Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.

Shakarim Soltanayev and Se Young Chun. Training deep learning based denoisers without ground truth data. In *Advances in Neural Information Processing Systems*, pp. 3257–3267, 2018.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proc. CVPR*, pp. 9446–9454, 2018.

Raviteja Vemulapalli, Oncel Tuzel, and Ming-Yu Liu. Deep gaussian conditional random field network: A model-based deep network for discriminative denoising. In *Proc. CVPR*, pp. 4801–4809, 2016.

Jun Xu, Lei Zhang, David Zhang, and Xiangchu Feng. Multi-channel weighted nuclear norm minimization for real color image denoising. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1096–1104, 2017.

Jun Xu, Lei Zhang, and David Zhang. A trilateral weighted sparse coding scheme for real-world image denoising. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 20–36, 2018.

Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Trans. Image Process.*, 26(7):3142–3155, 2017.

Kai Zhang, Wangmeng Zuo, and Lei Zhang. Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. *IEEE Trans. Image Process.*, 27(9):4608–4622, 2018.