NEURAL ADAPTIVE IMAGE DENOISER

Sungmin Cha and Taesup Moon

School of Electronic and Electrical Engineering,
College of Information and Communication Engineering,
Sungkyunkwan University (SKKU)
E-mail: {csm9493@gmail.com, tsmoon@skku.edu}

ABSTRACT

We propose a novel neural network-based adaptive image denoiser, dubbased as Neural AIDE. Unlike other neural network-based denoisers, which typically apply supervised training to learn a mapping from a noisy patch to a clean patch, we formulate to train a neural network to learn context-based *affine* mappings that get applied to each noisy pixel. Our formulation enables using SURE (Stein's Unbiased Risk Estimator)-like estimated losses of those mappings as empirical risks to minimize. In results, we can combine both supervised training of the network parameters from a separate dataset and *adaptive* fine-tuning of them using the given noisy image subject to denoising. Our algorithm with a plain fully connected architecture is shown to attain a competitive denoising performance on benchmark datasets compared to the strong baselines. Furthermore, Neural AIDE can robustly correct the mismatched noise level in the supervised learning via fine-tuning, of which adaptivity is absent in other neural network-based denoisers.

Index Terms— image denoising, neural networks, unbiased estimate, adaptive

1. INTRODUCTION

Image denoising is one of the oldest problems in image processing, and various denoising methods have been proposed over the past several decades, e.g., wavelet shrinkage [1], non-local means [2], BM3D [3], field of experts [4], sparse-coding based methods [5, 6], WNNM [7], EPLL [8], CSF [9], MLP [10], and DnCNN [11], etc.

Of particular interest among above are the methods based on deep neural networks. Particularly, [11, 12] recently have applied the convolutional neural network-based residual learning to image denoising and impressively surpassed the previous state-of-the-arts. However, there is one drawback on those methods; they are solely based on offline batch training of the neural network and lacks adaptivity to the given noisy image. Such lack of adaptivity, which is typically possessed in other methods, *e.g.*, [1, 2, 6, 7], could be problematic in practice when the characteristics of the given noisy image, *e.g.*, noise level, is different from those included in the training set. While [11, 12] train blind denoising models by training with multiple noise levels, such models could again fail to perform well when the test noise level is outside the range used for the supervised training.

To that end, we propose a novel framework for devising a neural network-based Adaptive Image DEnoiser (Neural AIDE). That is, we first formulate to learn an adaptive, context-based *affine* denoising mapping for each pixel with a neural network. Then, by utilizing the SURE (Stein's Unbiased Risk Estimator [13])-like estimated losses of such affine mappings as empirical risks to minimize, we *adaptively* train the network parameters solely based on the noisy image. Such framework is compatible with supervised training of the parameters on a separate dataset, in which the adaptive training step becomes *fine-tuning* (with the given noisy image) the pre-trained parameters. Our approach is inspired by the recent work in discrete denoising [14]. The experimental results are promising that Neural AIDE with simple fully connected architecture becomes competitive with the strong baselines and enjoys adaptivity that gives an edge to other neural net-based denoisers.

2. NOTATIONS AND PROBLEM SETTING

We denote $x^{n\times n}$ as the clean grayscale image, and each pixel $x_i \in [0, 255]$ is corrupted by an independent additive noise to result in a noisy pixel Z_i, *i.e.*, $Z_i = x_i + N_i, \quad i = 1, \ldots, n^2$, where the continuous noise variables N_i's are independent (not necessarily identically distributed nor Gaussian) over i and $\mathbb{E}(N_i) = 0, \mathbb{E}(N_i^2) = \sigma^2$ for all i. We treat the clean image $x^{n\times n}$ as an *individual* image without any probabilistic model and only treat $Z^{n\times n}$ as random.

A denoiser is generally denoted as $\hat{X}^{n\times n} = \{\hat{X}_i(Z^{n\times n})\}_{i=1}^{n^2}$ denoting that each reconstruction at location i is a function of the noisy image $Z^{n\times n}$. The standard loss function used to measure the denoising quality is the mean-squared error (MSE) denoted as $\mathbf{\Lambda}_{\hat{X}^{n\times n}}(x^{n\times n}, Z^{n\times n}) = \frac{1}{n^2}\sum_{i=1}^{n^2} \mathbf{\Lambda}\big(x_i, \hat{X}_i(Z^{n\times n})\big)$ where $\mathbf{\Lambda}(x, \hat{x}) = (x - \hat{x})^2$ is the

per-symbol squared-error. Conventionally, the MSE is compared in the dB-scale using the Peak Signal-to-Noise-Ratio (PSNR) defined as $10\log_{10}(255^2/\mathbf{\Lambda}_{\hat{X}^{n\times n}}(x^{n\times n}, Z^{n\times n}))$.

2.1. Estimated loss function for the affine denoiser

In this paper, we consider the denoiser of the form $\hat{X}_i(Z^{n\times n}) = a(Z^{\setminus i}) \cdot Z_i + b(Z^{\setminus i})$ for each i, in which $Z^{\setminus i}$ stands for the entire noisy image *except* for Z_i. Namely, the reconstruction at location i is an *affine* function of Z_i, but the slope and the intercept parameters, *i.e.*, $a(Z^{\setminus i})$ and $b(Z^{\setminus i})$, can be functions of the surrounding pixels. Hence, different parameters can be used for each location. Following lemma motivates considering such form of denoisers.

Lemma 1 *Suppose $Z = x+N$ with $\mathbb{E}(N) = 0$ and $\mathbb{E}(N^2) = \sigma^2$, and consider a mapping of form $\hat{X}(Z) = aZ + b$. Then,*

$$\mathbf{L}(Z, (a, b); \sigma^2) = (Z - (aZ + b))^2 + 2a\sigma^2 \tag{1}$$

is an unbiased estimate of $\mathbb{E}\mathbf{\Lambda}(x, \hat{X}(Z)) + \sigma^2$.

Remark: We note (1) is equivalent to the SURE [13] for $\hat{X}(Z)$ although N may not be a Gaussian. While the true MSE, $\mathbf{\Lambda}(x, \hat{X}(Z))$, can be evaluated only when the clean symbol x is known, the estimated loss $\mathbf{L}(Z, (a, b); \sigma^2)$ can be evaluated soley with Z, the affine mapping (a, b) and the noisy variance σ^2, thus, plays a key role in adaptively learning the neural network-based denoiser. The proof of lemma is omitted due to the page limit. ■

From Lemma 1, we can also show that for $\hat{X}_i(Z^{n\times n}) = a(Z^{\setminus i}) \cdot Z_i + b(Z^{\setminus i})$, given $Z^{\setminus i}$, $\mathbf{L}(Z_i, (a(Z^{\setminus i}), b(Z^{\setminus i})); \sigma^2)$ is an unbiased estimate of $\mathbb{E}_{Z_i}\left(\mathbf{\Lambda}(x_i, \hat{X}_i(Z^{n\times n}))\middle| Z^{\setminus i}\right) + \sigma^2$ since the noise is independent over i.

3. NEURAL AIDE

Our proposed Neural AIDE is defined to be

$$\hat{X}_i(Z^{n\times n}) = a(\mathbf{C}_{k\times k}^{\setminus i}) \cdot Z_i + b(\mathbf{C}_{k\times k}^{\setminus i}), \tag{2}$$

in which $\mathbf{C}_{k\times k}^{\setminus i}$ stands for the noisy image patch, or the context, of size $k \times k$ surrounding Z_i that does *not* include Z_i. Thus, the patch has a hole in the center. Then, as depicted in Figure 1, we define a neural network

$$\mathbf{g}(\mathbf{w}, \cdot) : [0, 1]^{k^2-1} \to \mathbb{R}^2 \tag{3}$$

that takes the context $\mathbf{C}_{k\times k}^{\setminus i}$ as input and outputs the slope and intercept parameters $a(\mathbf{C}_{k\times k}^{\setminus i})$ and $b(\mathbf{C}_{k\times k}^{\setminus i})$ for each location i. Thus, although having an affine function form, (2) is a highly nonlinear function in $Z^{n\times n}$. We denote $\mathbf{w}$ as the parameters of the network and use the plain fully connected neural network with ReLU activations.

There are two sharp differences between our Neural AIDE and other neural network-based denoisers, *e.g.*, [10, 11, 12, 15]. First, the other schemes take the full noisy image patch (including the center) as input to the network, and the network is trained to directly infer the corresponding clean image patch. In contrast, Neural AIDE is trained to first learn an affine mapping based on $\mathbf{C}_{k\times k}^{\setminus i}$, then the learned mapping is applied to Z_i to obtain the recostruction $\hat{X}_i$. Such difference enables deriving the SURE-like estimated loss in Lemma 1 and the adaptive training of the network as described in the next section. The principle of first learning a mapping then applying it to the noisy symbol for denoising or filtering has also been utilized in [16, 17, 14]. Second, unlike other schemes, in which the patch-level reconstructions should somehow be aggregated to generate the final denoised image, Neural AIDE simply generates the final pixel-by-pixel reconstructions. Thus, there is no need for a step to aggregate multiple number of reconstructed patches, which simplifies the denoising step.

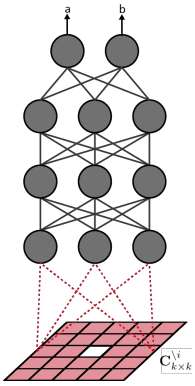


Fig. 1. The architecture of Neural AIDE

3.1. Adaptive training with noisy image

We first describe how the network parameters $\mathbf{w}$ can be adaptively learned from the given noisy image $Z^{n\times n}$ without any additional training data. That is, by denoting each output element of the neural network $\mathbf{g}(\mathbf{w}, \cdot)$ for the context $\mathbf{C}_{k\times k}^{\setminus i}$ as $\mathbf{g}(\mathbf{w}, \mathbf{C}_{k\times k}^{\setminus i})_1 \triangleq a(\mathbf{C}_{k\times k}^{\setminus i})$ and $\mathbf{g}(\mathbf{w}, \mathbf{C}_{k\times k}^{\setminus i})_2 \triangleq b(\mathbf{C}_{k\times k}^{\setminus i})$, we can define an objective function, $\mathcal{L}_{\text{adaptive}}(\mathbf{w}, Z^{n\times n})$, for the neural network to minimize as

$$\frac{1}{n^2}\sum_{i=1}^{n^2} \mathbf{L}\Big(Z_i, (\mathbf{g}(\mathbf{w}, \mathbf{C}_{k\times k}^{\setminus i})_1, \mathbf{g}(\mathbf{w}, \mathbf{C}_{k\times k}^{\setminus i})_2); \sigma^2\Big), \tag{4}$$

by using the definition of $\mathbf{L}(Z, (a, b); \sigma^2)$ in (1). The training process using (4) is identical to the ordinary neural network learning, *i.e.*, start with randomly initiallized $\mathbf{w}$, then use backpropagation and variants of mini-batch SGD, *e.g.*, [18], for updating the parameters. The formulation (4) may seem similar to training a neural network for a regression problem; namely, $\{(\mathbf{C}_{k\times k}^{\setminus i}, Z_i)\}_{i=1}^{n^2}$ can be analogously thought of as the input-target label pairs for the supervised regression. However, note that (4) only depends on $Z^{n\times n}$ and $\mathbf{w}$ (and σ^2), thus makes the learning adaptive.

The rationale behind using $\mathbf{L}(Z, (a, b); \sigma^2)$ in (4) is similar to other SURE-based estimators; minimize the unbiased estimate such that the true MSE may be also mini-

mized. However, unlike typical SURE-based estimators, *e.g.*, [1, 19, 20], that choose a few tunable hyperparameters via minimizing the unbiased estimate, we use $\mathbf{L}(Z,(a,b);\sigma^2)$ as an empirical risk in the empirical risk minimization (ERM) framework to learn the entire parametric model (*i.e.*, the neural network). Our approach is inspired by a recent work in discrete denoising [14] that works with the unbiased estimated losses for sliding-window denoisers.

Once the training is done, we can then denoise the same noisy image $Z^{n\times n}$ used for training by applying the affine mapping at each location as follows; denoting $\mathbf{w}^*$ as the learned parameter by minimizing (4), the reconstruction at location i by Neural AIDE is

$$\hat{X}_{i,\texttt{N-AIDE}}(Z^{n\times n}) = \mathbf{g}(\mathbf{w}^*, \mathbf{C}^{\setminus i}_{k\times k})_1 \cdot Z_i + \mathbf{g}(\mathbf{w}^*, \mathbf{C}^{\setminus i}_{k\times k})_2. \quad (5)$$

3.2. Supervised training and adaptive fine-tuning

While the formulation in (4) gives an effective way of adaptively training a denoiser based on the given noisy image $Z^{n\times n}$, the specific form of the denoiser in (2) makes it possible to carry out the supervised pre-training of $\mathbf{w}$ before the adaptive training. That is, we can collect abundant clean images, $\tilde{x}^{n\times n}$, from various image sources (*e.g.*, World Wide Web) and corrupt them with the assumed additive noise with known variance σ^2 to generate the correspoding noisy images, $\tilde{Z}^{n\times n}$, and the labelled training data of size N, $\mathcal{D} = \{(\tilde{x}_i, \tilde{\mathbf{C}}_{i,k\times k})\}_{i=1}^N$. Note $\tilde{\mathbf{C}}_{i,k\times k}$ stands for the noisy image patch of size $k\times k$ at location i that *includes* the noisy symbol $\tilde{Z}_i$, and $\tilde{x}_i$ is the clean symbol corresponding to $\tilde{Z}_i$.

Now, the subtle point is that, unlike the usual supervised learning that may directly learn a mapping from $\tilde{\mathbf{C}}_{i,k\times k}$ to $\tilde{x}_i$, we remain in using the neural network defined in (3) and learn $\mathbf{w}$ by minimizing $\mathcal{L}_{\text{supervised}}(\mathbf{w},\mathcal{D})$ that equals to

$$\frac{1}{N}\sum_{i=1}^{N} \mathbf{\Lambda}\Big(\tilde{x}_i, \mathbf{g}(\mathbf{w}, \tilde{\mathbf{C}}^{\setminus i}_{k\times k})_1 \cdot \tilde{Z}_i + \mathbf{g}(\mathbf{w}, \tilde{\mathbf{C}}^{\setminus i}_{k\times k})_2\Big). \quad (6)$$

Once the objective function (6) converges after sufficient iteration of weight updates, we denote the converged parameter as $\tilde{\mathbf{w}}$. Then, for a given noisy image to denoise, $Z^{n\times n}$, we can further update $\tilde{\mathbf{w}}$ adaptively for $Z^{n\times n}$ by minimizing $\mathcal{L}_{\text{adaptive}}(\mathbf{w}, Z^{n\times n})$ in (4) starting from $\tilde{\mathbf{w}}$. That is, we adaptively *fine-tune* $\tilde{\mathbf{w}}$ until $\mathcal{L}_{\text{adaptive}}(\mathbf{w}, Z^{n\times n})$ converges, then denoise $Z^{n\times n}$ with the converged parameter as (5). This capability of adaptively fine-tuning the supervised trained $\tilde{\mathbf{w}}$ is the unique characteristic of Neural AIDE that differentiates it from other neural network-based denoisers.

4. EXPERIMENTAL RESULTS

We compared the denoising performance of Neural AIDE with several state-of-the-arts, such as BM3D [3], MLP [10], EPLL [8], WNNM [7], and DnCNN [11]. We could not compare with [12] since no source code was available.

4.1. Data and experimental setup

For the supervised training, we generated the labelled training set using 2000 publicly available images, out of which 300 were taken from train/validation sets in the Berkeley Segmentation Dataset (BSD) [21] and the remaining 1700 were taken from Pascal VOC 2012 Dataset [22]. For the Pascal VOC images, we resized them to match the resolution of the BSD, 481×321. We corrupted the images with additive Gaussian noise and tested with multiple noise levels, $\sigma = 5, 10, 15, 20, 25, 50, 75, 100$. Namely, we built a separate training set with 2000 images for each noise level. The total number of training data points (*i.e.*, N) in each dataset was about 308 million. We evaluated the denoising performance with standard 11 images, {`Barbara`, `Boat`, `C.man`, `Couple`, `F.print`, `Hill`, `House`, `Lena`, `Man`, `Montage`, `Peppers`}, and 68 Berkeley images [4].

Our network had 9 fully connected layers with 512 nodes in each layer and used Adam [18] as the optimizer for training. The number of epochs, learning rates, the context size k, and the regularization parameters were determined via cross-validation. All our experiments used Keras [1] with Tensorflow [23] backend and NVIDIA GeForce GTX1080 with CUDA 8.0.

4.2. Quantitative evaluation

We first carried out the adaptive training for various k values solely with the given noisy image as described in Section 3.1. The best average PSNR on the 11 standard images was 28.62 (dB) for $\sigma = 25$ with $k = 7$. While the result is decent, we note some PSNR gap exists compared to the state-of-the-arts shown in Table 1. Then, we carried out the supervised training only with 2000 images described above. We observed the supervised training alone can achieve much higher PSNR, 30.32 (dB) for $\sigma = 25$ with $k = 17$, for the 11 images than the adaptive training, and become close to the state-of-the-arts. Finally, we fixed $k = 17$ and combined the adaptive fine-tuning with the supervised model as described in Section 3.2, of which results are summarized below.

Table 1 summarizes PSNRs of Neural AIDE compared to the recent state-of-the-arts on the standard 11 test images for various noise levels. For the baseline methods, we downloaded the codes from the authors' webpages and ran the code on the noisy images so that the numbers can be compared fairly. (MLP and DnCNN-S could run only on selected noise levels, which is the reason for the missing values in tables.) DnCNN-S and DnCNN-B of [11] stand for the model trained on separated training dataset with the correct σ and the blindly trained denoiser, respectively. $\texttt{N-AIDE}_{\texttt{S}}$ is the Neural AIDE with supervised training only, and $\texttt{N-AIDE}_{\texttt{S+FT}}$ is supervised training combined with the adaptive fine-tuning. The best performance for each noise level is denoted with bold.

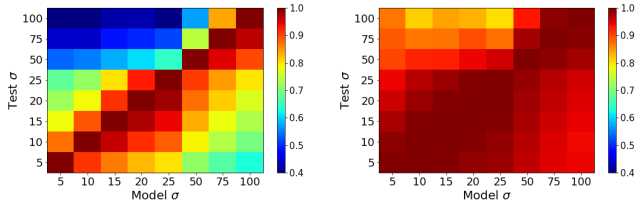
[1] `http://keras.io`

Table 1. PSNR(dB) on the 11 standard benchmark images.

σ	BM3D	MLP	EPLL	WNNM	DnCNN-B	Dn-CNN-S	N-AIDE$_{\text{S}}$	N-AIDE$_{\text{S+FT}}$
5	38.23	-	37.90	38.45	37.80	-	38.21	**38.47**
10	34.69	34.45	34.26	**34.95**	34.66	34.88	34.71	34.92
15	32.74	-	32.27	32.99	32.86	**33.02**	32.79	32.98
20	31.40	-	30.90	31.63	31.60	**31.67**	31.43	31.63
25	30.33	30.25	29.79	30.57	30.55	**30.62**	30.32	30.53
50	27.08	-	26.52	27.39	27.43	**27.39**	26.98	27.23
75	25.21	-	24.62	**25.51**	18.62	25.38	25.00	25.27
100	23.96	-	23.42	**24.27**	14.04	-	23.73	23.95

From the table, we see N-AIDE$_{\text{S+FT}}$ is competitive with the state-of-the-arts, WNNM and DnCNN, and mostly outperforms BM3D, MLP, and EPLL. Particularly, N-AIDE$_{\text{S+FT}}$ is much better than MLP, another plain fully connected neural network-based denoiser. Also, by comparing N-AIDE$_{\text{S}}$ with N-AIDE$_{\text{S+FT}}$, we can clearly see the effectiveness of the adaptive fine-tuning.

Furthermore, as mentioned in the Introduction, one of the main drawbacks of the other neural network-based denoisers, *e.g.*, MLP [10] and DnCNN-S [11], is that the networks have to be trained separately for all noise levels, and the mismatch of σ can significantly hurt the denoising performance. While N-AIDE$_{\text{S}}$ is also trained in a similar way, Figure 2 show that the adaptive fine-tuning can be very effective in overcoming such limitation. Figure 2(a) shows the PSNR of the mismatched N-AIDE$_{\text{S}}$ models before fine-tuning. "Model σ" stands for the σ used for the supervised training, and "Test σ" stands the σ of the true noise in the noisy image. Each row is normalized with the PSNR of the matched case, *i.e.*, the diagonal element, and is color-coded. We clearly see the sensitivity of PSNR in the mismatch of σ as the off-diagonal values show significant gaps compared to the diagonal ones in each row. On the other hand, Figure 2(b) shows the PSNR of N-AIDE$_{\text{S+FT}}$'s that have mismatched N-AIDE$_{\text{S}}$ models, but are adaptively fine-tuned with the correct Test σ's. We observe that the PSNR gaps of the mismatched N-AIDE$_{\text{S}}$ models can be significantly closed by the adaptive fine-tuning as long as the true Test σ is known at the fine-tuning stage.



(a) PSNR of N-AIDE$_{\text{S}}$ (b) PSNR of N-AIDE$_{\text{S+FT}}$

Fig. 2. PSNR(dB) of mismatched models

To overcome the σ mismatch problem of the supervised models, DnCNN-B [11] trains a single, blindly trained supervised model with a data that has mixture of broad range of σ values between $[0, 55]$. Table 1 shows that such model has strong performance when the Test σ is included in the $[0, 55]$ range. However, we can see that for the Test σ of 75 (18.62dB) and 100 (14.04dB), the performance of DnCNN-B dramatically deteriorates.

Table 2. PSNR(dB) on the 11 images for Test $\sigma = 75$.

Model σ for N-AIDE$_{\text{S}}$	5	10	15	20	25	50
Fine-tuning $\sigma = 75$	22.47	21.99	22.04	22.65	22.12	24.77
Fine-tuning $\sigma = 100$	22.27	21.05	21.64	22.04	21.56	24.61

Table 2, results for Test $\sigma = 75$, shows that N-AIDE$_{\text{S+FT}}$ can effectively correct the mismatch of the N-AIDE$_{\text{S}}$ models even when the σ for the fine-tuning is also mismatched, *i.e.*, *doubly* mismatched. Row 1 of the table is identical to the first 6 values of the second row of the matrix in Figure 2(b), *i.e.*, mismatched Model σ, but a matched "Fine-tuning" σ. Row 2 is when Fine-tuning σ is also mismatched. We observe that while the second row results are slightly worse than those of the first row, the doubly mismatched N-AIDE$_{\text{S+FT}}$'s still achieve much higher PSNRs than that of DnCNN-B (*i.e.*, 18.62dB). Particularly, when the Model σ is 50 and the Fine-tuning σ is 100, the PSNR of N-AIDE$_{\text{S+FT}}$ (24.61 dB) is just 2.6% worse than the matched case (25.27dB in Table 1), whereas the mismatched DnCNN-B is 26.3% worse. Note there is no way to fix DnCNN-B in this case other than re-training the whole network from scratch with a new supervised set including $\sigma = 75$, since the model is not adaptive. Clearly, such re-training is expensive in practice.

Table 3. PSNR(dB) on the 68 standard Berkeley images.

σ	BM3D	MLP	EPLL	WNNM	DnCNN-B	Dn-CNN-S	N-AIDE$_{\text{S}}$	N-AIDE$_{\text{S+FT}}$
5	37.58	-	37.55	37.76	37.65	-	37.81	**37.88**
10	33.31	33.49	33.37	33.55	33.71	**33.86**	33.66	33.75
15	31.07	-	31.19	31.32	31.60	**31.72**	31.47	31.57
20	29.61	-	29.73	29.83	30.19	**30.25**	30.02	30.13
25	28.56	28.95	28.67	28.81	29.15	**29.22**	28.95	29.06
50	25.60	-	25.66	25.89	26.20	**26.21**	25.90	26.03
75	24.19	-	24.09	24.36	18.68	**24.62**	24.31	24.46
100	23.23	-	23.05	23.38	14.29	-	23.24	**23.41**

Table 3 shows the PSNR results on the 68 standard Berkeley images [4]. We now see N-AIDE$_{\text{S+FT}}$ outperforms all baselines other than DnCNN. While DnCNN-S has slightly superior performance than ours (about 0.15dB), we believe N-AIDE$_{\text{S+FT}}$ has much more room to improve since we have not extensively tested with more modern network architectures, such as CNN with skipped connections [12], residual learning [24], and batch normalization [25], as in [11, 12]

5. CONCLUDING REMARKS

We devised a novel neural network-based adaptive image denoiser, Neural AIDE, based on SURE-like estimated loss minimization. For future work, we plan to explore more modern network architectures, try nonlinear mappings other than the affine mappings, and work with different types of noise.

Acknowledgments

This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea [NRF-2016R1C1B2012170] and by the ICT R&D program of MSIP/IITP [2016-0-00563].

6. REFERENCES

[1] D. Donoho and I. Johnstone, "Adapting to unknown smoothness via wavelet shrinkage," *Journal of American Statistical Association*, vol. 90, no. 432, pp. 1200–1224, 1995.

[2] 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.

[3] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Trans. Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

[4] S. Roth and M.J Black, "Field of experts," *International Journal of Computer Vision*, vol. 82, no. 2, pp. 205–229, 2009.

[5] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Processing*, vol. 54, no. 12, pp. 3736–3745, 2006.

[6] 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.

[7] S. Gu, L. Zhang, W. Zuo, and X. Feng, "Weighted nuclear norm minimization with applicaitons to image denoising," in *Computer Vision and Pattern Recognition (CVPR)*, 2014.

[8] D. Zoran and Y. Weiss, "From learning models of natural image patches to whole image restoration," *International Conference on Computer Vision (ICCV)*, 2011.

[9] U. Schmidt and S. Roth, "Shrinkage fields for effective image restoration," in *Computer Vision and Pattern Recognition (CVPR)*, 2014.

[10] H. Burger, C. Schuler, and S. Harmeling, "Image denoising: Can plain neural networks compete with BM3D?," in *Computer Vision and Pattern Recognition (CVPR)*, 2012.

[11] 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*, vol. 26, no. 7, pp. 3142 – 3155, 2017.

[12] X. Mao, C. Shen, and Y-B. Yang, "Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections," *Neural Information Processing Systems (NIPS)*, 2016.

[13] C. Stein, "Estimation of the mean of a multivariate normal distribution," *The Annals of Statistics*, vol. 9, no. 6, pp. 1135–1151, 1981.

[14] T. Moon, S. Min, B. Lee, and S. Yoon, "Neural universal discrete denosier," in *Neural Information Processing Systems (NIPS)*, 2016.

[15] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," in *Neural Information Processing Systems (NIPS)*, 2012.

[16] T. Weissman, E. Ordentlich, M. Weinberger, A. Somekh-Baruch, and N. Merhav, "Universal filtering via prediction," *IEEE Trans. Inform. Theory*, vol. 53, no. 4, pp. 1253–1264, 2007.

[17] T. Moon and T. Weissman, "Universal FIR MMSE filtering," *IEEE Transactions on Signal Processing*, vol. 57, no. 3, pp. 1068–1083, 2009.

[18] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[19] X. Xie, S. Kou, and L. Brown, "Sure estimates for a heteroscedastic hierarchical model," *Jouranl of the American Statistical Association*, vol. 107, no. 500, pp. 1465–1479, 2012.

[20] Y. Eldar, "Rethinking biased estimation: Improving maximum likelihood and the Cramer-Rao bound," *Foundations and Trends in Signal Processing*, vol. 1, no. 4, pp. 305–449, 2008.

[21] 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.

[22] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," `http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html`.

[23] M. Abadi et. al., "Tensorflow: A system for large-scale machine learning," *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pp. 265–283, 2016.

[24] K. He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[25] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *International Conference on Machine Learning (ICML)*, pp. 448–456, 2015.