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, dubbed 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 em-
pirical risks to minimize. In results, we can combine both
supervised training of the network parameters from a sepa-
rate 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 pro-
cessing, 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
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 im-
age, 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 (Neu-
ral 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 Unbi-
ased 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 pa-
rameters on a separate dataset, in which the adaptive train-
ing 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 ar-
chitecture 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 \( Z_i = x_i + N_i \), \( i = 1, \ldots, n^2 \),
where the continuous noise variables \( N_i \)'s are independent
(not necessarily identically distributed nor Gaussian) over \( i \)
and \( E(N_i) = 0, E(N_i^2) = \sigma^2 \) 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 func-
tion of the noisy image \( Z_{n\times n} \). The standard loss func-
tion used to measure the denoising quality is the mean-
squared error (MSE) denoted as
\[
\Lambda(X_{n\times n}, Z_{n\times n}) = \frac{1}{n^2} \sum_{i=1}^{n^2} \Lambda(x_i, \hat{x}_i(Z_{n\times n}))
\]
where \( \Lambda(x, \hat{x}) = (x - \hat{x})^2 \) 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 $10 \log_{10}(255^2/A_{\hat{X},n,n}(x_{n \times n}, Z_{n \times n}))$.

2.1. Estimated loss function for the affine denoiser

In this paper, we consider the denoiser of the form $\hat{X}_i(Z_{n \times n}) = a(Z_i) \cdot Z_i + b(Z_i)$ for each $i$, in which $Z_i$ 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., $a(Z_i)$ and $b(Z_i)$, 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 $E(N) = 0$ and $E(N^2) = \sigma^2$, and consider a mapping of form $\hat{X}(Z) = aZ + b$. Then,

$$L(Z, (a, b); \sigma^2) = (Z - (aZ + b))^2 + 2a\sigma^2$$

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

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, $\Lambda(x, \hat{X}(Z))$, can be evaluated only when the clean symbol $x$ is known, the estimated loss $L(Z, (a, b); \sigma^2)$ can be evaluated solely with $Z$, the affine mapping $(a, b)$ and the noisy variance $\sigma^2$, 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 $\hat{X}_i(Z_{n \times n}) = a(Z_i) \cdot Z_i + b(Z_i)$, given $Z_i$, $L(Z_i, (a(Z_i), b(Z_i)); \sigma^2)$ is an unbiased estimate of $E_{Z_i}(\Lambda(z_i, \hat{X}_i(Z_{n \times n}))) + \sigma^2$ since the noise is independent over $i$.

3. NEURAL AIDE

Our proposed Neural AIDE is defined to be

$$\hat{X}_i(Z_{n \times n}) = a(C_{i \times k \times k}) \cdot Z_i + b(C_{i \times k \times k}),$$

in which $C_{i \times k \times k}$ 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

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

that takes the context $C_{i \times k \times k}$ as input and outputs the slope and intercept parameters $a(C_{i \times k \times k})$ and $b(C_{i \times k \times k})$ for each location $i$. Thus, although having an affine function form, (2) is a highly nonlinear function in $Z_{n \times n}$. We denote $w$ 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 $C_{i \times k \times k}$, then the learned mapping is applied to $Z_i$ to obtain the reconstruction $\hat{X}_i$. 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.

3.1. Adaptive training with noisy image

We first describe how the network parameters $w$ 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 $g(w, \cdot)$ for the context $C_{i \times k \times k}$ as $g(w, C_{i \times k \times k})_1 \triangleq a(C_{i \times k \times k})$ and $g(w, C_{i \times k \times k})_2 \triangleq b(C_{i \times k \times k})$, we can define an objective function, $\mathcal{L}_{\text{adaptive}}(w, Z_{n \times n})$, for the neural network to minimize as

$$\frac{1}{n^2} \sum_{i=1}^{n^2} L(Z_i, (g(w, C_{i \times k \times k})_1, g(w, C_{i \times k \times k})_2); \sigma^2),$$

by using the definition of $L(Z, (a, b); \sigma^2)$ in (1). The training process using (4) is identical to the ordinary neural network learning, i.e., start with randomly initialized $w$, 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, $\{(C_{i \times k \times k}, Z_i)\}_{i=1}^{T_2}$ 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 $w$ (and $\sigma^2$), thus makes the learning adaptive.

The rationale behind using $L(Z, (a, b); \sigma^2)$ 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 \( L(Z, (a, b); \sigma^2) \) 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 \( w^* \) as the learned parameter by minimizing (4), the reconstruction at location \( i \) by Neural AIDE is

\[
\hat{X}_{\text{N-AIDE}}(Z^{n \times n}) = g(w^*, C_{\hat{\sigma}_k^{i,k}})1 \cdot Z_i + g(w^*, C_{\hat{\sigma}_k^{i,k}})2.
\] (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 \( w \) 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 \( \sigma^2 \) to generate the corresponding noisy images, \( \tilde{Z}^{n \times n} \), and the labelled training data of size \( N \),

\[
D = \{(\tilde{x}_i, \tilde{C}_{\hat{\sigma}_k^{i,k}})\}_{i=1}^N.
\]

Note \( C_{\hat{\sigma}_k^{i,k}} \) 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 training that may directly learn a mapping from \( C_{\hat{\sigma}_k^{i,k}} \) to \( \tilde{x}_i \), we remain in using the neural network defined in (3) and learn \( w \) by minimizing \( L_{\text{supervised}}(w, \tilde{D}) \) that equals to

\[
\frac{1}{N} \sum_{i=1}^{N} \Lambda(\tilde{x}_i, g(w, C_{\hat{\sigma}_k^{i,k}})1 \cdot \tilde{Z}_i + g(w, C_{\hat{\sigma}_k^{i,k}})2). \] (6)

Once the objective function (6) converges after sufficient iteration of weight updates, we denote the converged parameter as \( \tilde{w} \). Then, for a given noisy image to denoise, \( Z^{n \times n} \), we can further update \( w \) adaptively for \( Z^{n \times n} \) by minimizing \( L_{\text{adaptive}}(w, Z^{n \times n}) \) in (4) starting from \( \tilde{w} \). That is, we adaptively fine-tune \( w \) until \( L_{\text{adaptive}}(w, Z^{n \times n}) \) converges, then denoise \( Z^{n \times n} \) with the converged parameter as (5). This capability of adaptively fine-tuning the supervised trained \( \tilde{w} \) 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, \( 481 \times 321 \). 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 \( \sigma \) and the blindly trained denoiser, respectively. N-AIDE\(_S\) is the Neural AIDE with supervised training only, and N-AIDE\(_{S+FT}\) is supervised training combined with the adaptive fine-tuning. The best performance for each noise level is denoted with bold.

\[\text{http://keras.io}\]
From the table, we see N-AIDE_{S+FT} is competitive with the state-of-the-arts, WNNM and DnCNN, and mostly outperforms BM3D, MLP, and EPLL. Particularly, N-AIDE_{S+FT} is much better than MLP, another plain fully connected neural network-based denoiser. Also, by comparing the trained N-AIDE_{S} with N-AIDE_{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_{S} is also trained in a similar way, Figure 2 shows that the adaptive fine-tuning can be very effective in overcoming such limitation. Figure 2(a) shows the PSNR of the mismatched N-AIDE_{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_{S+FT}’s that have mismatched N-AIDE_{S} models, but are adaptively fine-tuned with the correct Test σ’s. We observe that the PSNR gaps of the mismatched N-AIDE_{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_{S} 
(b) PSNR of N-AIDE_{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 1. PSNR(dB) on the 11 standard benchmark images.

<table>
<thead>
<tr>
<th>σ</th>
<th>BM3D</th>
<th>MLP</th>
<th>EPLL</th>
<th>WNNM</th>
<th>DnCNN-B</th>
<th>DnCNN-S</th>
<th>N-AIDE_{S}</th>
<th>N-AIDE_{S+FT}</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>38.23</td>
<td>37.99</td>
<td>36.45</td>
<td>37.80</td>
<td>38.21</td>
<td>38.47</td>
<td>38.21</td>
<td>38.47</td>
</tr>
<tr>
<td>10</td>
<td>34.69</td>
<td>34.45</td>
<td>34.26</td>
<td>34.84</td>
<td>34.66</td>
<td>34.88</td>
<td>34.71</td>
<td>34.92</td>
</tr>
<tr>
<td>15</td>
<td>32.74</td>
<td>32.27</td>
<td>32.99</td>
<td>32.86</td>
<td>33.02</td>
<td>32.79</td>
<td>32.98</td>
<td>33.02</td>
</tr>
<tr>
<td>20</td>
<td>31.40</td>
<td>30.90</td>
<td>31.60</td>
<td>31.67</td>
<td>31.43</td>
<td>31.63</td>
<td>31.63</td>
<td>31.63</td>
</tr>
<tr>
<td>25</td>
<td>30.33</td>
<td>30.25</td>
<td>30.79</td>
<td>30.57</td>
<td>30.62</td>
<td>30.32</td>
<td>30.55</td>
<td>30.55</td>
</tr>
<tr>
<td>30</td>
<td>27.08</td>
<td>26.52</td>
<td>27.19</td>
<td>27.43</td>
<td>27.39</td>
<td>26.98</td>
<td>27.23</td>
<td>27.23</td>
</tr>
<tr>
<td>35</td>
<td>25.31</td>
<td>24.62</td>
<td>25.51</td>
<td>18.62</td>
<td>25.36</td>
<td>25.00</td>
<td>25.27</td>
<td>25.27</td>
</tr>
<tr>
<td>40</td>
<td>23.96</td>
<td>23.42</td>
<td>24.27</td>
<td>14.04</td>
<td>-</td>
<td>23.73</td>
<td>23.95</td>
<td>23.95</td>
</tr>
<tr>
<td>45</td>
<td>22.47</td>
<td>21.99</td>
<td>22.47</td>
<td>14.04</td>
<td>-</td>
<td>23.73</td>
<td>23.95</td>
<td>23.95</td>
</tr>
</tbody>
</table>

Table 2. PSNR(dB) on the 11 images for Test σ = 75.

<table>
<thead>
<tr>
<th>Model σ for N-AIDE_{S}</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fine-tuning σ = 75</td>
<td>22.47</td>
<td>21.99</td>
<td>22.04</td>
<td>22.65</td>
<td>22.12</td>
<td>24.77</td>
</tr>
<tr>
<td>Fine-tuning σ = 100</td>
<td>22.27</td>
<td>21.05</td>
<td>21.64</td>
<td>22.04</td>
<td>21.56</td>
<td>24.61</td>
</tr>
</tbody>
</table>

Table 3 shows the PSNR results on the 68 standard Berkeley images. We now see N-AIDE_{S+FT} outperforms all baselines other than DnCNN. While DnCNN-S has slightly superior performance than ours (about 0.15dB), we believe N-AIDE_{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


