Deep Marginalized Sparse Denoising Auto-Encoder for Image Denoising

Hongqiang MA1, *, Shiping MA1, Yuelei XU1 and Mingming ZHU1
1Aeronautics Engineering College, Air Force Engineering University, Xian, China
*18049025189@163.com

Abstract. Stacked Sparse Denoising Auto-Encoder (SSDA) has been successfully applied to image denoising. As a deep network, the SSDA network with powerful data feature learning ability is superior to the traditional image denoising algorithms. However, the algorithm has high computational complexity and slow convergence rate in the training. To address this limitation, we present a method of image denoising based on Deep Marginalized Sparse Denoising Auto-Encoder (DMSDA). The loss function of Sparse Denoising Auto-Encoder is marginalized so that it satisfies both sparseness and marginality. The experimental results show that the proposed algorithm can not only outperform SSDA in the convergence speed and training time, but also has better denoising performance than the current excellent denoising algorithms, including both the subjective and objective evaluation of image denoising.

1. Introduction
During the process of formation and transmission, digital images are often disturbed by different levels of noises, which degrade image quality. Image denoising is a very fundamental but crucial procedure in image processing, and its processing results directly influence the final effect of various tasks such as edge detection, target recognition and so on. Therefore, it is of great theoretical and practical value to study the image denoising algorithms and optimize the performance of the algorithms.

In recent years, a variety of image denoising methods has been proposed, typically including the following 3 categories. One approach is linear or non-linear filtering methods which are a relatively simple approach based on smoothing, such as median filtering and Wiener filtering. Another one is methods based on wavelet or dictionary decomposition of the image. Wavelet decomposition is to transform the image signal into wavelet domain for multi-layers decomposition, where they can be more easily separated from the noise, such as BLS-GSM [1]. The dictionary-based methods are to denoise by approximating the noisy patch using a sparse linear combination of atoms, including KSVD, which is an iterative algorithm that learns a dictionary on the noisy image at hand. [2]. The last one is methods based on global image statistics or other image properties, such as self-similarity. BM3D is generally considered the state-of-the-art algorithm for image noise removal. It uses a collaborative form of Wiener filter for high dimensional block of patches by grouping similar 2D blocks into a 3D data array [3].

While these methods have been successfully applied in practice, they share a shallow linear structure. When the noise intensity is small, the denoising effect is remarkable; and when the noise intensity is large, the denoised images often lose the edge, texture and other details information. Therefore, it is necessary to find a method to denoise effectively in high noise environment. Recent
research suggests, however, that non-linear deep models can achieve superior performance in various realistic problems. A few of deep models have also been applied to image denoising [4,5,6]. Deep learning, as a deep network, through the establishment of a hierarchical model similar to the human brain structure and the feature extraction of input data from bottom to top step by step, which can well establish the mapping relation from bottom signal to high-level semantics [7]. The auto-encoder is a classic model of deep learning. We can use it to learn the compression and distributed feature representation of a given dataset to reconstruct the input data [8]. Later, the researchers added some constraints on it, and derived a variety of deformation auto-encoders, including Sparse Auto-Encoder(SAE), Denoising Auto-Encoder(DAE), Marginalized Denoising Auto-Encoder(MDA)[9] and Stacked Sparse Denoising Auto-Encoder(SSDA). Xie et al. [10] had success at removing noise from corrupted images with SSDA, especially when the noise intensity is large, the performance of noise reduction is better than that of KSVD and BM3D. However, as a deep structure neural network model, it needs to be trained layer by layer, resulting in a large amount of calculation, a slow training speed and a long time to optimize the parameters.

In this paper, we marginalize the loss function of SSDA network and form Deep Marginalized Sparse Denoising Auto-Encoder (DMSDA), which is applied to image denoising. The experiment results show that the proposed algorithm not only improves the training speed, but also has a higher PSNR and the more detail information.

2. Related model description

2.1 Denoising auto-encoder (DAE)

The traditional AE can easily copy the input directly to the output without any constraints, the model usually has very poor performance [11]. Therefore, Vincent et al. proposed the DAE algorithm to improve the robustness of the network. DAE is a 3 layers neural network, composed of input layer, hidden layer and output layer. The algorithm flow of DAE is shown in Figure 1. The original input data $x$ first adds noise to get corrupted version $\tilde{x}$, then through the coding function $g$ to get the feature representation $h$ of the input, and last through the decoding function $f$ maps $h$ to the output layer to obtain the reconstructed input data $y$. The reconstruction error is defined by a loss function $L(x, y)$.

The encoding and decoding process of the DAE network is respectively $h = g(\tilde{x}) = s_h(w \tilde{x} + b)$, $y = f(h) = s_f(w' h + b')$. Where, $s_h$ and $s_f$ is a nonlinear activation function, $w$ and $b$ are respectively encoding weights and biases, $w'$ and $b'$ are respectively the decoding weights and biases. Given training data $D = \{x^{(n)}\}_{n=1}^N$, the DAE is trained by backpropagation to minimize the reconstruction loss given by

$$L_{\text{ recon}}(\theta) = \frac{1}{N} \sum_{n=1}^{N} J(x,y) + \frac{\lambda}{2} (\|w\|^2 + \|w'\|^2)$$

(1)

Where, $\theta = \{w, w', b, b'\}$ are the parameters of the model, $J(x,y) = \frac{1}{2} \|x - y\|^2$, $\lambda$ is the weight of decay term, which can reduce the magnitude of the weight and prevent overfitting.

**Figure 1.** The algorithm flow of DAE
2.2 Sparse auto-encoder (SAE)

In addition to making the output equal to the input as much as possible, the auto-encoder requires that the hidden layers must satisfy a certain sparsity (i.e., the output value of most of the neurons on the hidden layers is close to 0). Thus, the input data can be compressed and reduce dimensionality.

Supposed $\rho$ is a sparsity parameter, $\hat{\rho}_j$ is the average activation value of hidden unit $j$-th (averaged over the training set). In order to satisfy the sparsity of each neuron of the hidden layer, we approximately enforce the constraint $\hat{\rho}_j = \rho$, where, $\rho$ is typically a small value close to zero (say $\rho = 0.05$). To achieve this, an extra penalty term KL divergence is added to loss function that penalizes deviating $\hat{\rho}_j$ significantly from $\rho$. This penalty function has the property that $KL(\hat{\rho}_j, \rho) = 0$, if $\hat{\rho}_j = \rho$, and otherwise it increases monotonically as $\hat{\rho}_j$ diverges from $\rho$. So the penalty term is added to the loss function of the basic auto-encoder to form SAE.

2.3 Stacked Sparse Denoising Auto-Encoder (SSDA)

To make the auto-encoder both sparse and robust, a Sparse Denoising Auto-Encoder algorithm is proposed by the combination of SAE and DAE. The loss function of SDA is as follows:

$$L_{SDA}(\theta) = \frac{1}{N} \sum_{x \in D} J(x, y) + \beta \sum_{j=1}^{L} KL(\hat{\rho}, \rho) + \frac{\lambda}{2} \|w^l\| + \|w^{l'}\|$$

(2)

Where, $\beta$ controls the weight of the sparsity penalty term.

However, SDA is still a shallow neural network with poor learning ability. It is difficult to learn the deep essential characteristics of the data. Therefore, a series of SDAs are connected to form a Stacked Sparse Denoising Auto-Encoder—a deep network formed by feeding the hidden layer’s activations of one SDA into the input of the next SDA. New researches show that SSDA network with too many layers is easy to appear gradient dispersion phenomenon but also overfitting. In this paper, the network structure of 2 SDA is selected.

A good way to obtain good parameters for SSDA is to use Hinton’s unsupervised greedy layerwise training method proposed in [12]. The main idea of this method is: first train the first layer on raw input to obtain parameters, then use the representation of the input data to train the second layer to obtain parameters, repeat for subsequent layers, using the output of each layer as input for the subsequent layer. This method trains the parameters of each layer individually while freezing parameters for the remainder of the model. To produce better results, after this phase of training is complete, fine-tuning using backpropagation can be used to improve the results by tuning the parameters of all layers are changed at the same time. This entire network—the SSDA—is trained again by back-propagation in a fine-tuning stage, minimizing the loss given by

$$L_{SSDA}(\theta) = \frac{1}{N} \sum_{x \in D} J(x, y) + \frac{\lambda}{2} \sum_{l=1}^{L} \|w^l\| + \|w^{l'}\|$$

(3)

The sparsity-inducing term is not needed for this step because the sparsity was already incorporated in the pre-trained SDAs [13].

3. Algorithm

3.1 DMSDA network model

To restore the original image to the maximum extent and improve the performance of image denoising, the SDA network loss function is marginalized to form Marginalized Sparse Denoising Auto-Encoder (MSDA).
In order to improve the robustness of the SDA network, normally, the training samples are repeatedly added with the same distributed noise to obtain as much training set as possible to train the optimal parameters. For a dataset \( D = \{ x_1, x_2, \ldots, x_n \} \), it can yield corrupted copy \( \{ x_1', x_2', \ldots, x_n' \} \) by adding noise to each \( x_i \) \( m \)-times. In equation (3), let \( m_j(x) = y \), and minimize the averaged reconstruction loss

\[
L_{SDA}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{m} \sum_{j=1}^{m} J(x_i, m_j(x'_i)) + \beta \sum_{j=1}^{m} KL(\tilde{\rho}_j \| \rho) + \frac{\lambda}{2} \left( \| w \|_2^2 + \| w' \|_2^2 \right) 
\]  

(4)

When \( m \) is large, this increase directly computational cost and training time. Therefore, the idea of limit is introduced, as \( m \rightarrow \infty \), the average loss function for formulation (4) is:

\[
L_{SDA}(\theta) = \frac{1}{N} E(J(x, m_j(x))) + \beta \sum_{j=1}^{m} KL(\tilde{\rho}_j \| \rho) + \frac{\lambda}{2} \left( \| w \|_2^2 + \| w' \|_2^2 \right) 
\]  

(5)

The loss function of SDA by its Taylor expansion with respect to up to the second-order:

\[
J(x, m_j(x)) \approx J(x, m_j(\mu)) + (\tilde{x} - \mu) J_\Delta x_J + \frac{1}{2} (\tilde{x} - \mu) J_\Delta^2 x_J (\tilde{x} - \mu) 
\]  

(6)

Where, \( \mu = E[\tilde{x}] \) is the expected value of \( \tilde{x} \), \( \Delta x_J \) and \( \Delta^2 x_J \) are respectively the first-order derivative and second-order derivate about \( \tilde{x} \). we take the mean with regard to \( J(x, m_j(x)) \)

\[
E(J(x, m_j(x))) = J(x, m_j(\mu)) + \frac{1}{2} \text{tr}(E[(\tilde{x} - \mu)(\tilde{x} - \mu)'] \Delta^2 x_J) 
\]  

(7)

Here, we supposed the matrix \( \Sigma_x = E[(\tilde{x} - \mu)(\tilde{x} - \mu)'] \), and obtain

\[
E(J(x, m_j(\tilde{x}))) = J(x, m_j(\mu)) + \frac{1}{2} \text{tr}(\Sigma_x \Delta^2 x_J) 
\]  

(8)

The matrix \( \Sigma_x \) can simplify to diagonal matrix, because the noise is added to each dimension of the input vector independently. The diagonal terms of the Hessian \( \Delta^2 x_J \) is given by

\[
\frac{\partial^2 J}{\partial x^2} = (\frac{\partial^2 J}{\partial x^2})' = (\frac{\partial J}{\partial z})' (\frac{\partial J}{\partial z})' + (\frac{\partial J}{\partial z})' (\frac{\partial^2 z}{\partial x^2})' 
\]  

(9)

Here, \( d \) represents the dimension of the input vector.

LeCun et al. [14] remove the second term in (9). The matrix \( \frac{\partial^2 J}{\partial z^2} \) is the Hessian of \( J \) with respect to \( z \), and is positive definite. Therefore, the non-negative diagonal term of the matrix can be further simplified by using the positive definite property, here, we give the final approximation

\[
\frac{\partial^2 J}{\partial x^2} = \sum_{i=1}^{n} \frac{\partial^2 J}{\partial x^2} (\frac{\partial z_{k_i}^2}{\partial x_{j_i}})^2 
\]  

(10)

Where, \( n \) is the number of hidden neurons, \( z \) is the feature representation of hidden layers. After the above simplified calculation, the marginalized loss function

\[
E(J(x, m_j(\tilde{x}))) = J(x, m_j(\mu)) + \frac{1}{2} \sum_{i=1}^{n} \sigma_{z_{k_i}}^2 \sum_{j=1}^{n} \frac{\partial^2 J}{\partial z_{k_i}^2} (\frac{\partial z_{k_i}}{\partial x_{j_i}})^2 
\]  

(11)
Where, $\sigma^2_{id}$ is the corruption variance of the $d^{th}$ input dimension. The final loss function of MSDA is deduced:

$$J_{MSDA}(\theta) = \frac{1}{N} \sum_{i=0}^{N} (J(x, m_g(\mu)) + \frac{1}{2} \sum_{d=1}^{D} \sigma^2_{id} \sum_{k=1}^{K} \frac{\partial^2 J}{\partial z_k^d} (\frac{\partial z_k^d}{\partial x_d}) + \beta \sum_{j=1}^{J} KL(\hat{\beta}, \| \rho ) \right)$$  \quad (12)$$

Table 1 summarizes two loss function and their corresponding derivatives. In this paper, we choose square error as reconstruction loss function. In the case of additive Gaussian white noise, the second item in formulation (11) is transformed to:

$$R_z = \sum_{h} \sum_{d} h(h_d) \sum_{h_d} (1 - z_{h_d}) \sum_{h_d} w_{h_d}$$

In contrast to typical measuring model parameters with L2 norms, the regularizer is faster than $w_{h_d}$. So it accelerates the convergence speed of the network, makes the output closer to the input.

| Type              | $J(x, y)$          | $\frac{\partial^2 J}{\partial z_{h_d}^d}$ | $\frac{\partial z_k^d}{\partial x_d}$ |
|-------------------|--------------------|------------------------------------------|---------------------------------------|
| Squared loss      | $\parallel x - y \parallel$ | $2 \sum_{d} w_{h_d}^d$                  | $z_{h_d}(1 - z_{h_d})w_{h_d}$          |
| Cross-entropy loss | $-x^T \log(y) - (1 - x)^T \log(1 - y)$ | $\sum_{d} y_d(1 - y_d)w_{h_d}^d$       | $z_{h_d}(1 - z_{h_d})w_{h_d}$          |

MSDA is still a shallow neural network, and it is difficult to effectively extract the potential hierarchical features of data. Compared with the SSDA algorithm, in this paper, the two MSDA are stacked to form DMSDA. The network diagram of DMSDA is shown in Figure 2.

The network DMSDA is still trained by greedy layer-wise method. First, each MSDA is trained individually, and then the whole deep network is fine-tuned. DMSDA after fine-tuning has the basic characteristics of the biological nervous system, which reflects some functions of the human brain to some extent and simulates the biological system better.

We choose the following loss function to fine-tune the whole parameters of the DMSDA.

$$J_{DMSDA}(\theta) = \frac{1}{N} \sum_{i=1}^{N} | | f(x^{(i)}) - f(g(x^{(i)})) | |$$  \quad (13)$$

There is no weight of decay term and the sparsity-inducing term in eq. (13). This is due to the fact that both have been taken into account when training a single MSDA. There is not a significant change in performance when they are included.

3.2 Specific algorithm steps

The specific implementation process of image denoising based on DMSDA is as follows. The algorithm flow is shown in Figure 3.

Step 1 70000 patches are randomly selected from natural images, and the corresponding noisy image patches are obtained as training samples.

Step 2 The optimal network parameters are obtained by using the training samples to train the first MSDA and minimizing its loss function. At the same time, we need to calculate the output of the first hidden layer.

Step 3 The output obtained by step (2) is used as the input of second MSDA, and the parameters of second hidden layer network is trained by the same method.

Step 4 These parameters of the two layers are used as the initialization weights of the DMSDA network, and then use the BP algorithm to fine-tune all the layers to obtain the optimal parameters of the whole network. At this point, DMSDA training is over.

Step 5 The noisy image is divided into patches that are consistent with the size of the samples to input the trained DMSDA network for testing. Finally, the denoised image patches are then combined into a denoised image.
4. Results and discussion

In this paper, we will demonstrate the effectiveness of the proposed algorithm from two aspects of convergence speed and denoising performance. 80 pieces of $512 \times 512$ natural images are selected as training sets. To denoise the whole image is extremely difficult. In view of this, this paper will select small image patches from the entire image as training samples. The number of hidden neurons in the experiment was 40, $\lambda = 10^{-5}$, $\beta = 10^{-3}$.

4.1 Convergence speed

SSDA and DMSDA are usually trained to get a better deep network by three stages, including the first layer, the second layer and the fine-tuning. In order to show that DMSDA has faster convergence rate than SSDA, this paper will carry out simulation experiments from the above three stages. The experiment results are shown in Figure 4.

![Figure 2. The network diagram of DMSD](image)

![Figure 3. The algorithm flow based on DMSDA](image)

4.2 Denoising performance

In order to verify the better denoising effect of DMSDA, this paper will evaluate the quality of denoised images from both objective and subjective aspects. At present, an important objective criterion for the evaluation of denoising performance is PSNR, the higher the PSNR, the better the effect of noise reduction. Subjective evaluation is to see whether the denoised image is too smooth, to lose the details information of the original images.
4.2.1 **Objective evaluation.** 5 standard natural images are selected in the experiment to effectively illustrate the image denoising algorithm DMSDA has higher PSNR. The standard deviation of Gaussian white noise is respectively set to 25, 35, 50, 75. Then use the above four methods for noise reduction, and take the average PSNR. Table 2 is the average PSNR of various algorithms under different noise standard deviations.

**Table 2.** The average PSNR of various algorithms under different noise standard deviations.

| Algorithms | $\sigma = 25$ | $\sigma = 35$ | $\sigma = 50$ | $\sigma = 75$ |
|------------|---------------|---------------|---------------|---------------|
| Noisy images | 22.18 | 19.30 | 16.27 | 13.02 |
| KSVD | 31.38 | 29.71 | 27.48 | 26.37 |
| BM3D | 32.22 | 30.93 | 28.29 | 26.53 |
| SSDA | 30.61 | 29.69 | 28.55 | 27.02 |
| DMSDA | 30.66 | 29.73 | 28.86 | 27.33 |

It is shown from Table 2 that the PSNR of KSVD and BM3D is higher than that of SSDA and DMSDA, when the noise intensity is small (such as $\sigma = 25$, $\sigma = 35$). But with the noise intensity increasing ($\sigma \geq 50$), the PSNR of DMSDA is higher than the former three algorithms.

4.2.2 **Subjective evaluation.** In this paper, 4 natural images with rich texture and detail are selected to test, when $\sigma = 25$. The experiment results are compared with three algorithms of KSVD, BM3D and SSDA, which have better denoising effect. The experimental results are shown in Figure 5.

**Figure 5.** The details comparison of the different denoising algorithms

5. **Conclusion and future work**
Aiming at the problem that the SSDA algorithm is not easy to converge, the computational complexity is high and the training time is long, we present DMSDA deep network by marginalizing the loss function of the SDA. Results show that the proposed algorithm not only in the convergence rate is
faster than that of SSDA, but also can effectively remove the noise and keep original image details, compared with the excellent denoising algorithm, such as KSVD, BM3D and the classical SSDA in deep learning.

References
[1] Portilla J, Strela V, Wainwright M J, et al. Image denoising using scale mixtures of Gaussians in the wavelet domain[J]. IEEE Transactions on Image Processing, 2003, 12(11) pp 1338-51.
[2] Hua Zhisheng, Fu Lihua. K-SVD image denoising based on noisy image blocks classification and dictionary optimization. Computer Engineering and Applications, 2017,53(16) pp 187-192.
[3] Dabov, A. Foi, V. Katkovnik and K. Egiazarian, “Image denoising by sparse 3-D transform-domain collaborative _itering”, IEEE Transactions on Image Processing (TIP), vol. 16, no. 8, (2007) pp 2080-95.
[4] Tang Y, Salakhutdinov R, Hinton G E, et al. Robust Boltzmann Machines for recognition and denoising[C]// Proceedings of the 19th IEEE International Conference on computer vision and pattern recognition(CVPR).
[5] Vincent P, Larochelle H, Bengio Y, et al. Extracting and composing robust features with denoising auto encoders[C]//Proceedings of the 25th international conference on Machine learning. New York, USA: ACM Press, pp 1096-1103.
[6] Keyvanrad M A, Pezeshki M, Homayounpour M A, et al. Deep Belief Networks for Image Denoising[C]// Proceedings of the International Conference on Learning Representation. Berlin, Germany: Springer, 2014.
[7] Yu Kai, Jia Lei, et al. Deep Learning: Yesterday, Today, and Tomorrow[J]. Journal of Computer Research and Development, 50(9) pp 1799-1804.
[8] Rumelhart D E, Hinton G E, Williams R J. Learning Re-presentations by Back-propagating Errors[J]. Nature, 1986, 323(9) pp 533-6.
[9] Chen M, Weinberger K, Sha F, et al. Marginalized de-noising auto-encoders for nonlinear representations [C]//Proceedings of the 2014 31th International Conference on Machine Learning. New York: ACM Press, pp 1476-84.
[10] Xie Junyuan, Xu Linli, Chen Enhong, et al. Image de-noising and inpainting with deep neural networks[C]// Proceedings of the 25th International Conference on neural information processing systems. New York, USA: ACM Press, pp 341-9.
[11] Vincent P, Larochelle, Lajoie I, et al. Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion[J]. The Journal of Machine Learning Research, 2010(9999) pp 3371-3408.
[12] Hinton G E, Osindero S, The Y W. A fast learning algorithm for deep belief nets[J]. Neural Computation, 2006, 18(7) pp 1527 -54.
[13] Agostinelli F, Anderson M R, Lee H, et al. Adaptive mul-ti-column deep neural networks with application to robust image denoising[C]//neural information processing systems, 2013 pp 1493-1501.
[14] Lecun Y, Bottou L, Orr G B, et al. Efficient BackProp[M]. Orr G B, M Ller K-R. Neural Networks: Tricks of the Trade, LNCS 1524. Berlin: Springer, 1998.