An adaptive image denoising method based on Deep Rectified Denoising Auto-Encoder

Hongqiang MA, Shiping MA, Yuelei XU, Mingming ZHU
Graduate School, Air Force Engineering University, Xian, China

E-mail: 18049025189@163.com

Abstract. Stacked Sparse Denoising Auto-Encoder (SSDA) has been successfully applied to image denoising, which is superior to the most of the traditional image denoising algorithms. However, the algorithm has low training convergence speed and poor universality. To address these limitations, we present an adaptive image denoising method based on Deep Rectified Denoising Auto-Encoder. To reduce training difficulty and speed up convergence, the rectified linear units (ReLU) is used as activation function of the network, and batch normalization (BN) is used to normalizes the input data of the middle layer in network. Different from the previous image denoising model, the proposed model is trained to learn the mapping relationship between noisy image and noise by residual learning. To overcome the problem that the new model cannot effectively deal with noise not seen in training, we perform adaptive training of multiple channels on the improved model to obtain the optimal channel weights and jointly output the denoised image. The experimental results show that the proposed algorithm can not only outperform SSDA in the convergence speed, but also adaptively remove the noise that is not seen in training.

1. Introduction
Image denoising is a classical yet still active topic in low level vision since it is an indispensable step in many practical applications [1], such as image segmentation, target tracking and recognition, etc. However, during the process of formation and transmission, images are often disturbed by different levels of noises, which degrade image quality. Therefore, it is of great theoretical and practical value to study the image denoising algorithms.

At present, there are mainly three kinds of image denoising algorithms. One is method based on wavelet decomposition or dictionary learning. 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 [2]. The dictionary-based method is to sparsely represent the noisy image on an overcomplete atom library so as to remove unnecessary information in the image and achieve denoising [3], such as KSVD. Another one is methods based on global image statistics or other image properties, such as self-similarity. BM3D is generally considered one of the state-of-the-art image denoising algorithms. It uses a collaborative form of Wiener filter for high dimensional block of patches by grouping similar 2D blocks into a 3D data array [4]. The last one is methods based on deep learning. In recent years, deep learning has received extensive attention from researchers and has gradually become an upsurge of internet and artificial intelligence [5]. It also provides new ideas for image denoising. Harmeling et al. [6] trains the a multi plain perceptron to learn the mapping relationship between noisy images and clean images, and its denoising performance is superior to most traditional
denoising algorithms. Later, SSDA is successfully used to remove noise from corrupted images [7-8], especially when the noise intensity is large, the denoising performance is better than that of KSVD and BM3D, PSNR is higher and image blurring has been further improved.

The image denoising method based on deep learning has improved both visual effects and objective evaluation metrics compared with traditional methods [9]. However, deep learning is to learn feature expression from massive data and fit complex nonlinear functions. Therefore, the network is very deep, the parameter is huge, and the gradient dispersion phenomenon is serious. It takes a long time to adjust the parameters and often depends on the prior knowledge. Secondly, this method usually only adds a single type noise to train, and it can effectively remove the noise that appears in the training. If we want to remove other noise types, we have to spend a lot of time to retrain.

In view of this, we make the following three improvements to SSDA. 1) To prevent gradient dispersion, we adopt ReLu function instead of the sigmoid function as the activation function of hidden layers in SSDA, and the sparse constraint term is removed. 2) we use BN and residual learning to facilitate the adjustment of the parameters and speed up the convergence of the network. 3) We perform multi-channel training on the improved model (Deep Rectified Denoising Auto-Encoder, DRDA) to achieve adaptive image denoising and improve the robustness of the model to different types of noise.

2. Stacked Sparse Denoising Auto-Encoder (SSDA)

Vincent et al. proposed a denoising auto-encoder (DAE) in 2008. 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 $s_f$ to get the feature representation $h$ of the input, and last through the decoding function $s_g$ maps $h$ to the output layer to obtain the reconstructed input data $y$.

The encoding and decoding process of the DAE network is respectively $h = s_f(\tilde{x} + b)$, $y = s_g(\tilde{x} + b')$. Where, $s_f$ and $s_g$ is a nonlinear activation function (for which sigmoid function $s(z) = \frac{1}{1 + e(-z)}$ is often used). $w$ and $w'$ is respectively the encoding and decoding weights, $b$ and $b'$ is respectively the encoding and decoding bias. $\theta = \{w, w', b, b'\}$ is the parameter of the model DAE. Given training data $D = \{x^{(i)}\}_{i=1}^{N}$, the DAE is trained by backpropagation to minimize the reconstruction loss given by

$$L_{DAE}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \|y^{(i)} - \hat{x}^{(i)}\|^2 + \lambda \frac{1}{2} \|w\|^2 + \|w'\|^2$$

Where, $\lambda$ is the weight of decay term, which can reduce the magnitude of the weight and prevent overfitting.

Later, researchers studied the information processing mechanism of human brain, and found that when images enter into human brain from vision, only a small number of neurons are stimulated, most of the neurons are still in a state of inhibition [10]. Inspired by this, Bengio adds an extra penalty term $KL$ to the reconstruction loss function, which enables the activation of neurons in the hidden layer to satisfy a certain sparsity, that is, the activation value of most neurons is 0. The loss function of the Sparse Denoising Auto-Encoder (SDA) is as follows

$$L_{SDA}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \|y^{(i)} - \hat{x}^{(i)}\|^2 + \beta \sum_{i=1}^{N} KL(\rho) + \lambda \frac{1}{2} \|w\|^2 + \|w'\|^2$$ (2)
Where, $\text{KL}(\hat{\rho}||\rho) = \rho \log \frac{\rho}{\hat{\rho}} + (1-\rho)\log \frac{1-\rho}{1-\hat{\rho}}$, $\rho$ is a sparsity parameter, $\beta$ controls the weight of the sparsity penalty term. $\hat{\rho}_j = \frac{1}{N} \sum_{i=1}^{N} \hat{h}_j(x^{(i)})$ is the average activation value of hidden unit $j$-th (averaged over the training set).

**Figure 1.** The algorithm flow of DAE

**Figure 2.** The network diagram of SSDA

SSDA is a deep neural network stacked by a series of SDAs. Between each SDA, the activation value of the previous SDA’s hidden layer is used as the input of the next SDA. Figure 2 shows an SSDA network structure with two hidden layers.

A good way to obtain good parameters for SSDA is to use Hinton’s unsupervised greedy layer-wise training method [11], which can relieve the phenomenon of dispersion to a certain extent. That is to say, each SDA is separately trained first and then the optimal weights of the network are obtained. Then, these weights are taken as the initial values of the SSDA network weights. Finally, the Back Propagation (BP) algorithm is used to fine-tune the whole network until the optimal parameters of the whole network are obtained. The loss function of SSDA in the fine-tune phase is as follows:

$$L_{\text{SSDA}}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \left| \hat{y}^{(i)} - x^{(i)} \right|^2 + \frac{\lambda}{2} \sum_{l=1}^{l} \left| w^{(l)} \right|^2$$

(3)

Where, $l$ is the number of stacked SDA. The sparsity-inducing term is not needed for this step because the sparsity was already incorporated in the pre-trained SDAs.

3. **Adaptive image denoising**

3.1 **Deep Rectified Denoising Auto-Encoder (DRDA)**

3.1.1 **Activation function selection.** The main reason for the gradient dispersion is still the residual, and the residual expression is as follows. The sigmoid function is used in SSDA. Since the derivative value $f'$ of the sigmoid function is always less than 1 and the residual propagates backward from the output layer to the bottom layer. After each layer, the residual is multiplied by $f'$. Thus, after the residual is passed through many layers, the deep neural network cannot update network weights because of the low residual of the bottom.

$$\delta^{(n)} = -(y - a^{(n)}) \cdot f'(z^{(n)})$$

(4)

$$\delta^{(l)} = -(W^{(l)})^T \delta^{(l+1)} \cdot f'(z^{(l)})$$

(5)

$$\nabla W^{(l)} = \delta^{(l+1)} (a^{(l)})^T$$

(6)

Where, $\delta^{(n)}$ is the residual of output layer, $a^{(n)}$ is the actual output of output layer, $y$ is the expected output of output layer, $\delta^{(l)}$ is the hidden layer residual, $z$ is the input of activation function, and $a^{(l)}$ is the output of hidden layer. $\nabla W$ represents the gradient of the loss function to the network weights.
The function expression of ReLu is $\max(0, x)$, whose derivative is 1. Therefore, the gradient can flow well in the backpropagation, which can effectively relieve the gradient dispersion. Compared with the sigmoid function, ReLu has 3 main characteristics: unilateral inhibition, relatively wide exciting boundary and sparsity activation. After initialization of network weights by experience rule, almost half of the neurons in the sigmoid function are activated, which is not consistent with neuroscience research. However, ReLu can sparse negative values and adjust dynamically, which is closer to the activation model of biology [12].

3.1.2 Residual learning and batch normalization. The goal of image denoising is to recover a clean image $x$ from a noisy observation $\tilde{x}$ which follows an image degradation model $\tilde{x} = x + n$. Rather than directly outputing the denoised image $\hat{x}$, in this paper, the algorithm we proposed is designed to predict the residual image $\hat{n}$, i.e., the difference between the noisy observation and the latent clean image. We can find a mapping $f$ that minimizes the cost function:

$$\arg\min_f \|f(\tilde{x}) - n\|^2$$

(7)

Therefore, we use residual learning, the input is a noisy image, and the output is the extracted noise. The denoised image can be obtained by formula $x = \tilde{x} - f(\tilde{x})$.

The purpose of the BN algorithm is to batch and normalize the input data of each intermediate layer to reduce the influence of changes in the distribution of data in the middle of the network on the training of neural network parameters [13], thereby improving the stability of the neural network. Supposed that the input data collection of an intermediate layer of the network is $\{x_1, \cdots, x_n\}$, and the algorithm process is as follows:

$$\mu = \frac{1}{m} \sum_{i=1}^{m} x_i$$

(8)

$$\sigma = \frac{1}{m} \sum_{i=1}^{m} (x_i - \mu)^2$$

(9)

Where, $\mu$ and $\sigma$ is respectively the mean and variance, and then the data are normalized to data $\hat{x}_i$ with a mean of 0 and a variance of 1. $\epsilon$ is a constant to avoid disagreement when the variance is zero.

$$\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

(10)

In order to avoid damage to the feature distribution due to normalization of the data, it is necessary to restore the original feature distribution by reconstruction shift.

$$z_i = \text{BN}_{\alpha, \beta}(\hat{x}_i) = \alpha \hat{x}_i + \beta$$

(11)

$\alpha$ and $\beta$ are reconstruction parameters introduced by the reconstruction shift, which can obtain through network training.

According to [14], stochastic gradient descent algorithm is simple and efficient for deep network training, but it needs artificial choice of parameters. The choice of parameters is crucial to the training results so that a lot of time is wasted on the parameters adjustment. The use of BN greatly reduces the difficulty of network training and does not need to adjust the parameters deliberately to achieve a better performance.

In summary, we first propose a DRDA network. This model uses ReLu as an activation function and removes the sparse constraints in the SSDA loss function, using direct training method. At the same time, the network introduces the BN algorithm to normalize the middle layer data, uses the
residual learning to learn the mapping between the noisy image and the noise, and then obtains the denoised image by
\[ x = \tilde{x} - f(\tilde{x}) \]. The network model DRDA is shown in Figure 3.

The loss function of DRDA is shown as follows
\[
L_{\text{DRDA}}(\theta) = \frac{1}{N} \sum_{n=1}^{N} \left\| y^{(i)} - (\tilde{x}^{(i)} - x^{(i)}) \right\|^2 + \frac{\lambda}{2} \sum_{c=1}^{C} \left\| w^{(i)} \right\|^2 + \left\| w^{(i)} \right\|^2 \tag{12}
\]

3.2 Adaptive Deep Rectified Denoising Auto-Encoder (ADRDA)

The image denoising methods using deep learning have achieved good results in recent years. However, this kind of method is to learn the mapping between the noisy image and the clean image by adding a single type of noise (such as Gaussian white noise), which is only effective for the same noise in the test phase. In view of this, we train adaptively the improved network model—DRDA to remove the noise that is not trained. The universality and robustness of the algorithm have been improved.

ADRDA is the linear combination of several DRDAs, or channels, each trained on a single type of noise can learn the optimized features generated by the activation of the DRDA’s hidden layers. Then we use the all features as inputs to a neural network-based regression component, referred to here as the weight prediction module. this module then uses these features to compute the optimal channel weights used to linearly combine the column outputs into a weighted average. The network model is shown in Figure 4.

![Figure 3. The network diagram of DRDA](image)

The ADRDA has three training phases: training the multi-channel DRDAs, finding ideal channel weights and training weight prediction model. The multi-channel DRDAs are trained as discussed in Section 3.1, with each DRDA provided a noisy training set, corrupted by a single noise type along with the original versions of those images as a target set.

3.2.1 Finding ideal channel weights.

Once the DRDAs are trained, we make full use of the feature learning capability of the DRDA network and construct a new feature training set extracted from the hidden layers of the DRDAs. Specifically, for DRDA column \( c \), the activations of hidden layers \( h^{(i)}; \cdots; h^{(i)} \) are concatenated into a feature vector \( f_{i} \), and \( f_{i}^{1}, f_{i}^{2}; \cdots; f_{i}^{c} \) are concatenated to form the entire feature training set \( \hat{\phi} = [f_{i}^{1}; \cdots; f_{i}^{c}] \), where, \( i \) represents the number of stacked DRDA and \( C \) is the number of channels.

In addition, for each input image, each channel produces an output. \( \hat{Y} = [y_{1}, \cdots, y_{C}] \) represents all channel outputs, \( n = [n_{1}, \cdots, n_{C}] \) is the expected noise of the corresponding channel. In order to find the channel weight of the ADRDA, the following loss function is minimized by:
\[
\arg\min_{\mathbf{s}} \frac{1}{2} \| \mathbf{s} - \mathbf{n} \|^2 
\]  
\[
0 \leq s_c \leq 1, \forall c 
\]  
\[
1 - \delta \leq \sum_{c=1}^{C} s_c \leq 1 + \delta 
\]

Here, $\mathbf{s}$ is the vector of channel weights, and $s_c$ represents the weight of the channel $c$. The constraint in Eq. [15] helps to avoid degenerate cases where some weights are too large or too small due to the sum of weights. Although making the weights sum exactly to one is more consistent with human intuition, we found that the performance slightly improved when given some flexibility. In this paper, $\delta=0.04$ is used.

3.2.2 Training weight prediction model. The last stage is the training of the weight prediction model. Since a radial basis function (RBF) network can approximate any nonlinear function, it can handle the hard analytic regularity in the system and has good generalization ability [15]. Therefore, we choose an RBF network as the weight prediction module that is trained to take the feature training set as input and fit the ideal channel weight calculated by 3.2.1.

After the above three phases, the ADRDA training has been completed. A noisy image $\mathbf{x}$ is supplied as input to each of the channel, which together produce the output $\hat{\mathbf{Y}}$, which is an approximation of noise. The feature vector $\phi$ is created from the activation of the hidden layers of each DRDA and fed into the weight prediction module, which then computes the predicted column weights $\mathbf{s}^*$ according to the deep features of the data. The final denoised image of the proposed model is $\mathbf{x} - \hat{\mathbf{Y}} \mathbf{s}^*$.

4. Results and discussion

4.1 Experimental data

The experimental data in this paper are taken from natural image data sets commonly used in image denoising. We follow [1] to use 400 images of size 180×180 for training, called Train400. Because deep neural network training requires massive data samples, to enrich the training data and improve the generalization ability of the model, we use the data augmentation method to amplify the original database. That is, for Train 400, 90°, 180° rotation and 270° rotation and flip is performed respectively, and a total of 1600 images are used as the final training samples. We use BSD68 (a dataset containing 68 natural images from Berkeley segmentation dataset) as validation set and Set12 (a dataset containing 12 natural images with rich details and textures) as test set.

4.2 Analysis of convergence

In order to prove the validity of the model DRDA, we first compare DRDA and SSDA. In this paper, we set the number of layers in SSDA and DRDA to be 4. 8000 small image patches of size $8 \times 8$ are selected as the input of the network. The number of hidden neurons in the experiment was 40, $\lambda=10^{-4}$, $\rho=10^{-3}$, $\rho=0.01$. The experiment was trained by the method of momentum gradient descent. The batch size is 1000, epoch is set to 10, momentum is [0.9 0.95 0.99], which is updated every 3 epochs. The initial value of learning rate is set to 0.1, and it is reduced by 10 times after 3 epochs.

We record the training curves of two models in Figure 5. We can see that DRDA can better fit the sample data than the SSDA network. First, both the training set and the Validation set, it can achieve a very low mean square error (MSE) which the SSDA network may not be able to achieve, and the number of iterations required is less than SSDA. The MSE reflects the difference between the denoised image and the clean image, which indirectly verifies that the denoised images using DRDA retain more information of the original images and has better denoising performance. Second, the mean square error
curves of the two networks for train set and validation set are basically the same, which explains the rationality of the hyperparameters selection and the network generalization ability.

![Figure 5. The convergence effect comparison](image)

4.3 Adaptive image denoising

In order to better train the ADRDA model, we add Gaussian noise, speckle noise, and salt & pepper noise to the training samples, and each type of noise has 6 levels. In total, 18 channels of DRDA need to be trained. Among them, the variance of Gaussian noise is respectively 0.02, 0.06, 0.10, 0.14, 0.18, 0.22. The density of speckle noise is respectively 0.05, 0.10, 0.15, 0.20, 0.25, 0.30. The density of the salt & pepper noise is the same as the speckle noise’s. At the same time, to facilitate the experimental comparison and analysis, we respectively use the same type of noise images with all noise levels as training sample to train model. Here, we respectively refer to these as G-DRDA, S&P-DRDA, S-DRDA. MC-DRDA refers to training 18 channels, but the weight of each channel is equal. When testing the performance of the ADRDA model, Gaussian noise, speckle noise, salt and pepper noise, Poisson noise, and uniform noise is respectively added to the data set Set12. Each type of noise has two degrees, which are as follows: G1: 0.01, G2: 0.07, S1: 0.1, S2: 0.15, S&P1: 0.1, S&P2: 0.2, P1: 0.5, P2: 1.5, U1: 0.2, U2: 0.4.

![Figure 6. The comparison of denoising performance of different algorithms](image)
In the subjective evaluation, we select three typical images in the data set Set12 as an example to demonstrate the denoising effect of the proposed algorithm. Additive Gaussian white noise, salt and pepper noise, and Poisson noise are added from top to bottom in Figure 6. It can be clearly seen that G-DRDA and BM3D have a good performance of removing Gaussian noise. The detailed features of denoised image such as edge and texture are well preserved, but they are not ideal to remove salt &pepper noise and still remain a lot of noise. We can also see that they can remove Poisson noise, but they will make the non-noise pixels of the image too smooth and the details lost. Although S&P-DRDA can remove these three types of noise, the denoised image remains to be improved. Compared with other algorithms, the algorithm ADRDA is effective for all three types of noise. The denoised image has clear outlines and the texture information remains more complete.

Table 1. The average PSNR/SSIM of the experimental results of different algorithms on the Set12 data set.

| Noisy Type | G-DRDA | S&P-DRDA | S-DRDA | MC-DRDA | ADRDA | BM3D |
|-----------|--------|----------|--------|---------|-------|------|
| G1        | 27.63/0.8156 | 25.39/0.7504 | 25.41/0.7561 | 27.44/0.8075 | 29.87/0.8517 | 29.88/0.8486 |
| G2        | 25.39/0.7233 | 23.13/0.6194 | 22.78/0.6003 | 25.38/0.7171 | 25.44/0.7321 | 25.39/0.7232 |
| S&P1      | 28.27/0.8406 | 34.90/0.9851 | 23.9/0.6795 | 26.62/0.8002 | 36.87/0.9863 | 19.50/0.4326 |
| S&P2      | 25.75/0.7998 | 31.66/0.9661 | 22.92/0.6284 | 23.79/0.7347 | 33.66/0.9711 | 13.71/0.1646 |
| S1        | 23.87/0.6851 | 24.45/0.6971 | 28.58/0.8411 | 24.91/0.7029 | 28.22/0.8396 | 19.40/0.4743 |
| S2        | 23.6/0.6596 | 23.85/0.6644 | 27.75/0.8147 | 24.24/0.6713 | 27.77/0.8136 | 16.64/0.3769 |
| P1        | 23.79/0.6845 | 24.44/0.7420 | 23.90/0.6877 | 24.26/0.6720 | 28.83/0.8479 | 27.13/0.8289 |
| P2        | 23.4/0.6520 | 23.77/0.6561 | 22.92/0.6310 | 23.40/0.6463 | 25.63/0.7517 | 20.61/0.7454 |
| U1        | 23.65/0.6770 | 24.11/0.6738 | 25.05/0.6739 | 24.14/0.6815 | 25.05/0.6669 | 16.98/0.3098 |
| U2        | 19.28/0.6368 | 19.41/0.6366 | 20.21/0.6500 | 19.88/0.6443 | 21.10/0.6610 | 14.13/0.3036 |
| Average   | 24.46/0.7210 | 25.51/0.7391 | 24.80/0.7014 | 24.35/0.6463 | 25.82/0.8122 | 20.34/0.5208 |

In objective evaluation, PSNR and SSIM are used to evaluate the quality of denoised image. The PSNR reflects the degree of noise reduction of the image, and the SSIM reflects the integrity of the structure information. Table 1 shows the denoising performance of different algorithms for different noise types. We can conclude that G-DRDA, S&P-DRDA and S-DRDA algorithm have higher PSNR and SSIM when removing the corresponding types of noise, while the PSNR and SSIM are lower for other noise types. The algorithm ADRDA is slightly worse than S-DRDA in removing speckle noise. However, its PSNR and SSIM are all higher than other methods, whether the types of noise seen in training (such as Gaussian, salt and pepper noise), or the noise that does not participate in training (such as Poisson and uniform noise). At the same time, in order to verify the effectiveness of the proposed algorithm, compared with BM3D that is considered as one of the state-of-the-art image denoising methods, we can find that BM3D is only effective for Gaussian noise, while the two objective values for other noise types are lower than ADRDA.

In summary, the algorithm ADRDA has a better denoising performance, which can be improved both in subjective visual and objective evaluation indicators. In addition, the algorithm overcomes the disadvantages of most denoising algorithms, and it can also deal with different types of noise within a certain range without acquiring the prior knowledge of test images.

5. Conclusion and future work

In this paper, we present an adaptive image denoising method based on Deep Rectified Denoising Auto-Encoder. The network uses ReLu as an activation function to effectively alleviate the gradient dispersion. In addition, the joint use of BN and residual learning greatly reduces the difficulty of network training and improves the training speed and image denoising performance. In order to overcome the problem that the new model DRDA can only effectively remove the single type of noise seen in the training, we take advantage of feature learning ability of the model to carry out multi-channel adaptive training. The experimental results show that ADRDA algorithm is better than other methods in terms of subjective visual and objective evaluation indicators, and that can adaptively handle different noise within a certain
range. It has certain reference significance and practical value. In the future, we will train the convolution neural network (CNN) adaptively to further improve the image denoising performance.

References

[1] Zhang K, Chen Y, Chen Y, et al. Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising[J]. IEEE Transactions on Image Processing, 2016, 26(7):3142.

[2] 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): 1338.

[3] Yan X, Yang B, Zhang W, et al. An Improved Denoising Algorithm of Feather and Down Image Based on KSVD[C]// International Conference on Information Technology in Medicine and Education. 2016 pp 419-423.

[4] Dabov K, Foi A, Katkovnik V, et al. Image restoration by sparse 3D transform-domain collaborative filtering[J]. 2008, 6812(8): 681207.

[5] Yu K, Jia L, Chen Y Q, et al. Deep learning: yesterday, today and tomorrow[J]. Journal of Computer Research and Development, 2013, 50(9) pp 1799-1804.

[6] Harmeling S, Schuler C J, Burger H C. Image denoising: Can plain neural networks compete with BM3D?[C]// Computer Vision and Pattern Recognition. IEEE, 2012 pp 2392-2399.

[7] Xie J Y, Xu L L, Chen E H, 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, 2012 pp 341-349.

[8] Li H M. Deep Learning for Image Denoising[J]. International Journal of Signal Processing Image Processing and Pattern Recognition, 2014, 7(3) pp 171-180.

[9] Zhang Y G, Yi B S, Wu C Y, et al. Low-dose CT image denoising method based on convolutional neural network[J]. Acta Optica Sinica, 2018, 38(4): 0410003.

[10] Xu J, Xiang L, Liu Q, et al. Stacked Sparse Autoencoder (SSAE) for Nuclei Detection on Breast Cancer Histopathology Images[J]. IEEE Transactions on Medical Imaging, 2016, 35(1) pp 119-130.

[11] 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.

[12] Glorot X, Bordes A, Bengio Y, et al. Deep Sparse Rectifier Neural Networks[C]// International Conference on Artificial Intelligence and Statistics. 2012 pp 315-323.

[13] Zhu W, Qu J Y, Wu R B. Straight convolutional neural networks algorithm based on batch normalization for image classification[J]. Journal of Computer-Aided Design & Computer Graphics, 2017, 29(9) pp 1650-1657.

[14] Ioffe S, Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift[J]. 2015 pp 448-456.

[15] Gai J F, Zhao G R, Song C. Model predictive control based on linearization and neural network approach[J]. Systems engineering and electronics, 2015(2) pp 394-399.