2017 RISP International Workshop on Nonlinear Circuits,
Communications and Signal Processing (NCSP'17)
Guam, USA, February 28th to March 3rd, 2017

Denoising Auto Encoder with Logistic Map

Ryuta YOSHIMURA, Shinsaburo KITTAKA, Yoko UWATE, and Yoshifumi NISHIO

†Dept. of Electrical and Electronic Engineering, Tokushima University
2-1 Minami-Josanjima, Tokushima 770–8506, Japan
Email: {yoshimura,kittaka,uwate,nishio}@ee.tokushima-u.ac.jp

Abstract

In this study, we investigate denoising auto encoder as prior learning. We show results using chaotic noises with denoising auto encoder. We investigate value and parameter of network by using a denoising auto encoder.

1. Introduction

Recentry, the method of machine learning is Deep Learning. It has already used as commercial services. The companies which have advanced technology about Deep Learning have used it as image search. The reason why Deep Learning becomes hot topic is that A team who uses the new machine learning method of the Deep Learning won the championship in competition of the image recognition in 2012.

The reason why the team using Deep Learning won is that Deep Learning has a multi neural network. A neural network with two or more hidden layers is called a deep neural network. Neural network looks like human brain cells. It is expressive because Deep Learning has very complicated relationships between their inputs and outputs if it has more and more hidden layers. However, When there are two or more hidden layers, it usually does not work because it takes more time to have the whole network to learn by error back propagation and overfitting also becomes more easy. If it is difficult to learn the whole by error back propagation, we should do prior learning each layer and give initial values to make feature extraction [1].

The idea of auto encoder is an effectively prior learning with each layer of the multilayer neural network. When we stack two or more hidden layers, after auto encoder first we make it with only one hidden layer, then remove the output layer, consider the hidden layer as the input layer and stack one more layer in order to make deep neural network. We build the network for the purpose of using it as input of other models as a certain potentiality vector by reducing the dimension of the hidden layer. Furthermore, When we use input data with noise insted of original data, we obtain good initial values. This method is called denoising auto encoder.

In this study, we show results using chaotic noises with denoising auto encoder. The noises are made up from the logistic map.

2. Auto Encoder

Auto encoder is used as prior learning. It is unknown that each layer should do, what kind of feature extraction at the time of prior learning. Therefore, the auto encoder is unsupervised learning. An initial value can be said that it is suitable if an output from hidden layer can restore input. In other words, we hope the network does not lose information in the hidden layer. The structure of the auto encoder has three layers in neural network consisting of an input layer, hidden layer and output layer. How to obtain a good value is minimizing the error function. By making the unit of the hidden layer smaller than the unit of the input layer, the dimensions of the input information are reduced in Fig. 1. Then, when hidden layer is taken out, an important fetures remain.

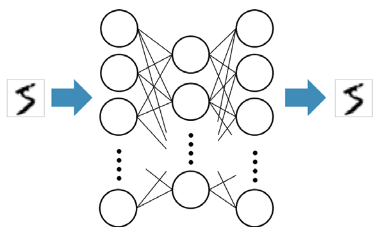

Figure 1: Auto encoder.

3. Denoising Auto Encoder

In addition to the above, it is possible to obtain a better initial value by setting constraint conditions on the auto encoder. The constraint conditions is to add a noise to each input. Applying noise increases the number of samples and there is no danger of relying on certain features. Also, the features extracted based on many samples are important parameters throughout the network. In other words, it is good feature extraction to work the auto encoder under strict conditions. It is called denoising auto encoder. We obtain an initial value with the higher generalization ability by performing prior learning with them. Denoising auto encoder is a model that removes noise from data by learning a network that uses the original data as a correct answer.

4. Proposed Method

Denoising auto encoder uses input data with the noise and assume original data as a correct answer. Furthermore, it learns the model which removes a noise from the data. We have denoising auto encoder to be able to reconstructed the origin data reproducing noise by adjusting a parameter. We will show an update equation of parameter.

$$y = s(Wx + b) \tag{1}$$

$$z = s(W'y + b') \tag{2}$$

Equation (1) shows encode. Equation (2) shows decode. The process of converting x to y is called encode, and the process of converting y to z is called decode. z means reconstructing data from x. We update weight so that output z nears input x. s means sigmoid function. W and W' mean weight. In this study, we set a restriction that W_x equel W'_x. It is called tied weight and reduse eight to calculate. How close the z to x is defined by an error function. It is the learning of the denoising auto encoder to update the parameter W, b, b' by the error back propagation in order to minimize this error function. The cross entropy error function which is shown by equation (3) is used as error function.

$$L_H(x, z) = -x \log z - (1 - x) \log(1 - z) \tag{3}$$

In order to minimize the error function, we should calculate a slope of L_H for W, b and b'. After calculating slopes of W, b and b', we obtain slopes which are shown by equation (4).

$$\begin{cases} W^{new} = W^{old} + \dfrac{\partial \epsilon}{N} \displaystyle\sum_{n=1}^{N} \dfrac{\partial L}{\partial W} \\ b^{new} = b^{old} + \dfrac{\partial \epsilon}{N} \displaystyle\sum_{n=1}^{N} \dfrac{\partial L}{\partial b} \\ b'^{old} + \dfrac{\partial \epsilon}{N} \displaystyle\sum_{n=1}^{N} \dfrac{\partial L}{\partial b'} \end{cases} \tag{4}$$

When input data of the N unit are given, update of the parameter is given by stochastic gradient descent in denoising auto encoder. Equation (5) shows the update of the parameter. ϵ means a learning rate. The proposed system uses a logistic map for chaos function to decide the probability.

$$x_{n+1} = a x_n (1 - x_n) \tag{5}$$

x_n means input. x_{n+1} means output. Parameter a makes the logistic map constant value, periodic vibration or aperiodic complicated behavior that is called the chaos. The change in the value of x_n with to increase the n. We determine the parameter a as 3.828327.

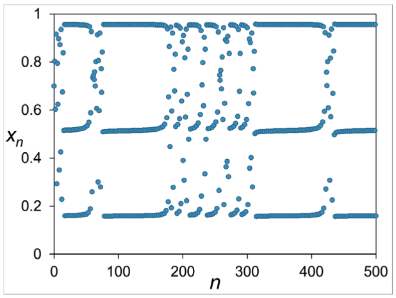

Figure 2: Logistic map at the time we decide the parameter a as 3.828327.

Figure 2 shows a behavior of intermittency chaos. The part where the irregular behavior is occurring and where the periodic behavior occurs alternately are called intermittency chaos. When we determine the parameter a as 3.828327, the logistic map shows intermittency chaos. We generate a random number by the logistic map and compare it with the threshold that we set. When it does not exceed the threshold, it output 1. We mutilate it to each pixel of the input data. The pixel which is multiplied 0 is painted black and become a noise.

5. Simulation results

We forcus on the following two points. They are difference due to learning loops and noise application rate. We use MNIST data set. MNSIT has 28 x 28 pixels image. Our network has input layer 784 units, hidden layer 500 units and output layer 784 units. First, we explain simulation result difference due to learning loops. The noise application rate is 60 %. We show cost which is the output from the error function in table 1 at any learning loops.

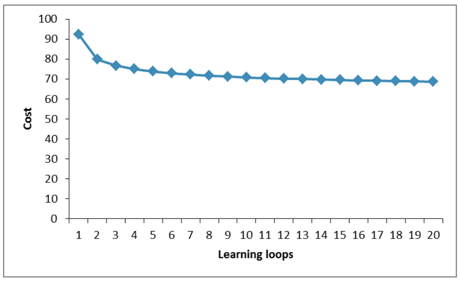

Figure 3: A cost at learning loops.

Figure 3 shows that the cost decreases and denoising auto encoder can learn it. The error converges after the 6th time. We will show a visuals of the weight and reconstructed images.

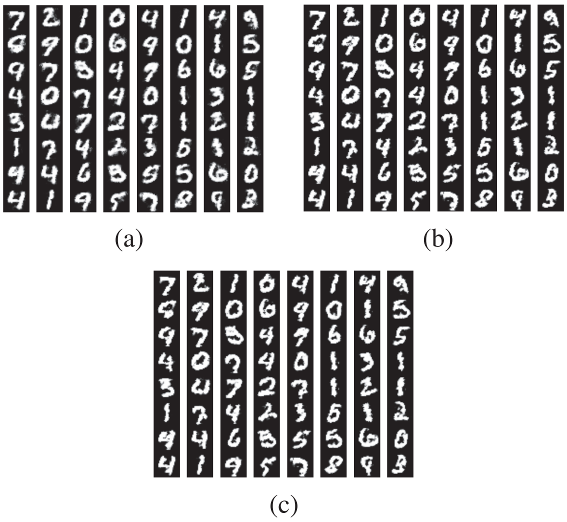

Figure 4: Reconstructed image of studying difference due to learning loops. (a) Learning loops is 1. (b) Learning loops is 5. (c) Learning loops is 20.

Figure 4 shows reconstructed images. As learning loops increase we can see that it is clearly reconstructed. It is thought that a tendency to correct the black place noised to white becomes strong. When the learning loops increase, this tendency becomes remarkable. We can see this tendency too much at the learning loops of 20.

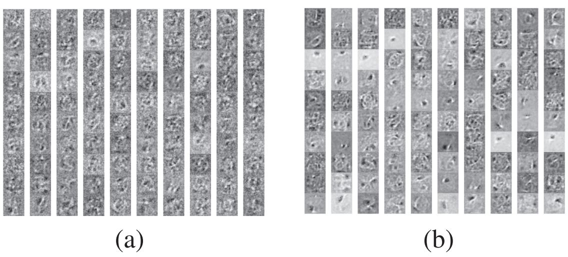

Figure 5: Weight visualization of studying difference due to learning loops. (a) Learning loops is 1. (b) Learning loops is 5.

Figure 5 shows that the tendency of weight is different with learning loops. We thought each weight gains features by increasing the learning loops. Each weight visualizes clearer at the leaning loops of 20 than the learning loops of 1.

Next, we explain difference due to noise application rate. The parameter a of the logistic map is 3.828327. Learning loops is 10. Noise application rate is different. We show cost with some rate which is the output from error function in Fig. 6.

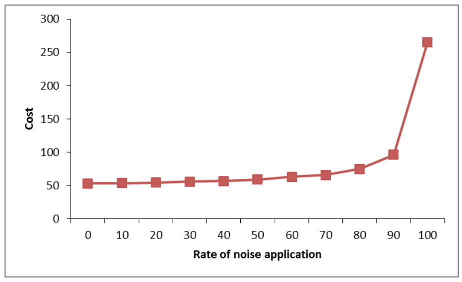

Figure 6: A cost at rate of noise application.

Figure 6 shows that cost increases when rate increases. We will show some images with noise. Increasing the noise rate increases the cost. This is because it is an error between the noise images and the original images.

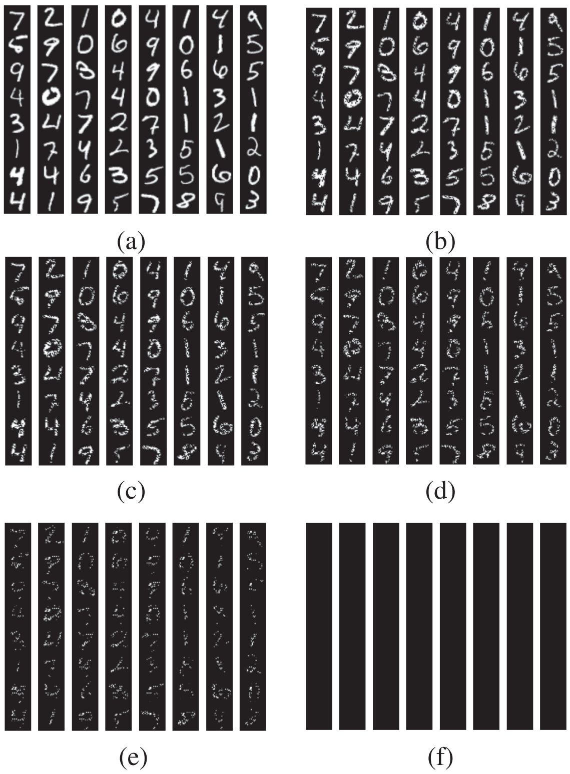

Figure 7: Corrupted image of studying difference due to noise application rate. (a) 0%. (b) 20%. (c) 40%. (d) 60%. (e) 80%. (f) 100%.

Figure 7 shows corrupted images due to noise application rate. Corrupted image at rate of 0% means input data for auto encoder.

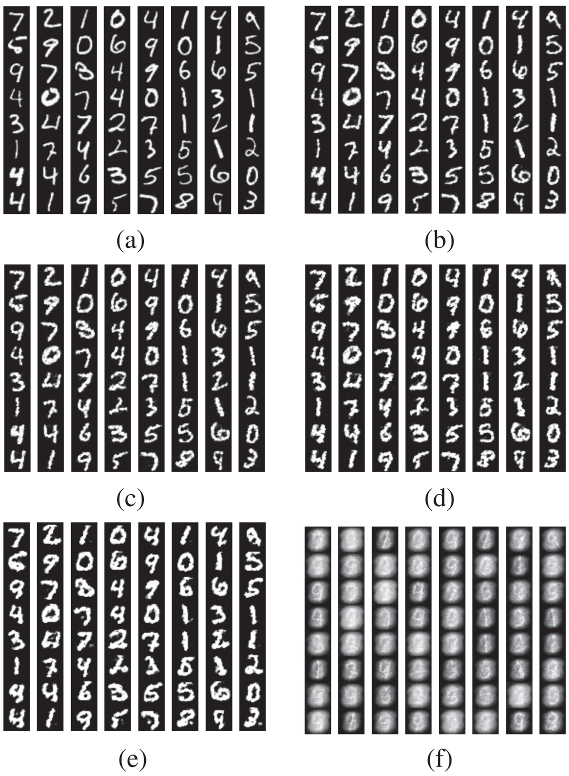

Figure 8: Reconstructed image of studying difference due to noise application rate. (a) 0%. (b) 20%. (c) 40%. (d) 60%. (e) 80%. (f) 100%.

Figure 8 shows reconstructed images. The more the rate of noise is increased, the stronger the tendency to correct noise to white. Therefore, the letters are visually thick.

Figure 9 shows tendency of weight is different due to noise application rate. When it is 80% it seems the weight has the most characteristic amount like letter of number.

6. Conclusion

In this study, we investigated denoising auto encoder as prior learning. We used some noises made up from the logistic map. We used intermittency chaos are made from logistic map by adjusting parameter. We investigated cost, weight and reconstructed images. When learning loops increased, the images were reconstructed clearly. It was thought that this is because a tendency to correct black noise to white becomes strong. Visualized weights showed features that varied in each when learning loops increased. The letters of the reconstructed images were bigger than an original and were reconstructed clearly when the rate of noise increased. Some of the visualized weights showed features like number.

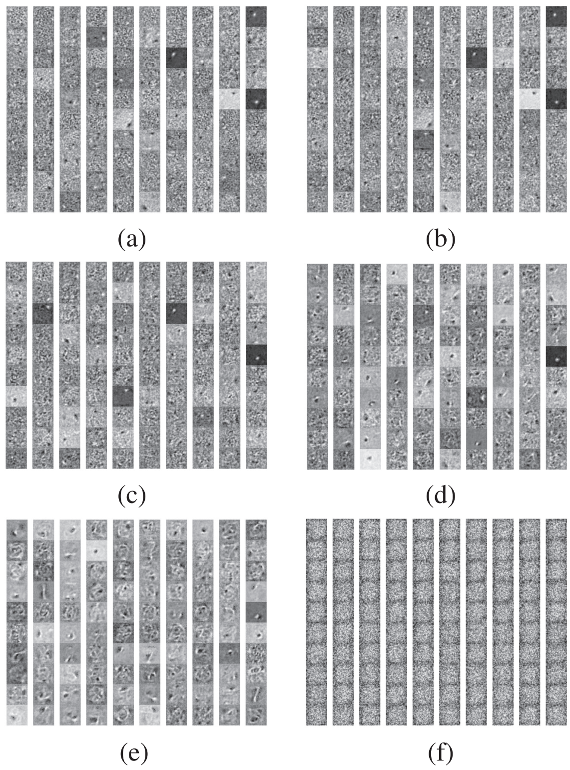

Figure 9: Weight visualization of studying difference due to noise application rate. (a) 0%. (b) 20%. (c) 40%. (d) 60%. (e) 80%. (f) 100%.

Acknowledgment

This work was partly supported by JSPS Grant-in-Aid for Scientific Research 16K06357.

References

[1] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever and Ruslan Salakhutdinov,*"Dropout: A Simple Way to Prevent Neural Networks from Overfitting*,"Journal of Machine Learning, pp.1929-1958, Jun. 2014.