Hindawi
Security and Communication Networks
Volume 2018, Article ID 6868506, 9 pages
https://doi.org/10.1155/2018/6868506

WILEY | Hindawi

Research Article

Research on Plaintext Restoration of AES Based on Neural Network

Xinyi Hu[1] **and Yaqun Zhao**[1,2]

[1]*State Key Laboratory of Mathematical Engineering and Advanced Computing, Wuxi, China*
[2]*State Key Laboratory of Cryptography and Science, Beijing, China*

Correspondence should be addressed to Xinyi Hu; 290760485@qq.com

Received 12 July 2018; Revised 16 October 2018; Accepted 31 October 2018; Published 18 November 2018

Guest Editor: Ching-Nung Yang

Known plaintext attack is a common attack method in cryptographic attack. For ciphertext, only known part of the plaintext but unknown key, how to restore the rest of the plaintext is an important part of the known plaintext attack. This paper uses backpropagation neural networks to perform cryptanalysis on AES in an attempt to restore plaintext. The results show that the neural network can restore the entire byte with a probability of more than 40%, restoring more than half of the plaintext bytes with a probability of more than 63% and restoring more than half of the bytes above 89%.

1. Introduction

With the development of machine learning technology, the research of cryptanalysis is not limited to traditional manual deciphering. The intelligent deciphering based on machine learning, especially the emergence of intelligent deciphering based on neural network, provides a new development direction for cryptanalysis.

Neural network-based cryptanalysis can solve the shortcomings of traditional cryptanalysis methods in terms of attack difficulty and the amount of data required for attacks. First of all, neural network as an ideal black box recognition tool can effectively analyze and simulate the black box problem (cryptanalysis problem), infinitely approach the cryptanalysis problem, and finally get the algorithm equivalent to the encryption and decryption algorithm to achieve cryptanalysis. In this case, the neural network does not need to restore the key of the algorithm and the specific setting parameters and only needs to input the ciphertext, and after training, the corresponding plaintext can be obtained with a certain probability. Secondly, as a machine learning method, neural network can effectively achieve the corresponding cryptanalysis results and can achieve the final deciphering algorithm with less training set (plaintext-ciphertext pair). Therefore, neural networks are rapidly recognized by the industry as a method of cryptanalysis and are gradually called a new research direction in the field of cryptanalysis.

The application of neural network in cryptanalysis is mainly used for the global deduction of cryptographic algorithms [1] (the algorithm that the attacker obtains and encrypts and decrypts may not know the key) and the complete crack [1] (the attacker obtains the key).

In 2008, Bafghi et al. [2] used a neural network model to represent the differential operation of block ciphers to find the difference features. The differential feature space of the block cipher can be represented by a multilevel weighted directed graph, so the problem of finding the best differential feature can be transformed into the problem of finding the least weight multibranch path between two known nodes in the directed graph. In this paper, the cyclic neural network (RNN) is used to find the path by minimizing the network cost function in the differential operation graph of the block cipher. In 2010, Alallayah et al. [3–5] performed neural network-based cryptanalysis on classical cryptography, sequence ciphers, and simplified DES (SDES). They regard cryptanalysis as a black box problem, using neural networks as an ideal tool for identifying black boxes, combining system identification technology with adaptive system technology, and constructing a neuroidentifier that simulates the target cryptosystem. A neural model and the key can be determined

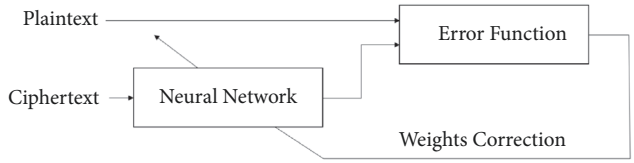

Figure 1: System structure of neural network cryptanalysis.

from a given pair of ciphertexts. In 2012, Alani et al. [6, 7] implemented a known plaintext attack based on neural networks for DES and 3DES. In the experiment, they trained a neural network to retrieve plaintext from ciphertext without having to retrieve the keys used in encryption. This kind of attack is successfully applied to DES and 3DES. For DES, an average of 2^{11} pairs of ciphertext pairs are needed, and cryptanalysis can be completed in about 51 minutes to achieve a global deductive effect. For the 3DES cryptanalysis, only an average of 2^{12} plaintext-ciphertext pairs are needed, and the analysis can be completed in about 72 minutes. Corresponding results significantly reduce the number of known plaintexts required and reduce the time required to perform a full attack compared to other attacks. In 2013, Bahubali Akiwate et al. [8] proposed a neural network-based cryptanalysis of the DES algorithm, aiming to analyze the nonlinear characteristics of DES through neural networks. The text data is used as the plaintext in the analysis, and the ciphertext encrypted by DES is used as the input of the neural network and is trained together with the plaintext to obtain the corresponding output. The neural output is compared with the plaintext, and performance error indicators are established for analyzing the data to improve efficiency. In 2014, Danziger et al. [9] used the method in [5] to find out the mapping relationship between plaintext, ciphertext, and key in SDES. When experiments are performed using 102,400 sample data, the keys of the first, second, and fifth bits of the 10-bit key can be obtained; when the number of samples is reduced to 2000, the keys of the first and second bits can be obtained. Together with the differential analysis, they improved the S-box of SDES, which reduced the correlation between adjacent keys in the key space, making it resistant to key restoration attacks in neural network cryptanalysis.

Inspired by [6, 7], this paper uses the methods in [6, 7] to construct an analysis framework similar to them. According to the block length and data type of AES algorithm, the data processing process suitable for AES algorithm is set. Then, according to the research object and the structural characteristics of neural network, the specific experimental system structure is designed. The AES algorithm in ECB and CBC mode is used to perform cryptanalysis using neural network. Without knowing the key, the plaintext is restored from the ciphertext, trying to achieve the effect of global deduction. The results show that the plaintext restored by the AES-128 and AES-256 algorithms over the neural network is more than 63% higher than the plaintext compared with the original plaintext.

Section 2 of this paper introduces the specific process of cryptanalysis based on neural networks. Section 3 designs the steps of the overall experiment and selects the relevant parameters. Section 4 is the experimental results. Section 5 gives the conclusion.

Table 1: Experimental equipment.

Version	macOS High Sierra 10.13.1
Processor	1.2 GHz Intel Core m5
RAM	8 GB 1867 MHz LPDDR3
Graphics Card	Intel HD Graphics 515 1536 MB

2. Cryptanalysis Process Based on Neural Network

The cryptanalysis method based on neural network utilizes the learning ability of the neural network to train the neural network with the known Ming ciphertext. After the training is completed, the neural network can restore the plaintext from the ciphertext that does not belong to the training set. The corresponding system structure is shown in Figure 1. The system contains an information forward propagation process (ciphertext ⟶ neural network ⟶ plaintext), and an error backpropagation process (plaintext ⟶ error function ⟶ weight correction ⟶ neural network).

The ciphertexts are input in the neural network, and the output results are compared with the known plaintexts to obtain an error function. The weight is continuously corrected according to the error until the neural network is successfully trained. And finally the plaintext can be restored with a certain probability. This attack method is considered to be a global deductive attack, which is functionally equivalent to the original decryption algorithm without knowing the key. This analysis method is similar to the global approximation method for multilayer feedforward neural networks in [11].

In order to train a neural network with an acceptable error rate, it is necessary to expand the network size, so it is necessary to increase the time of each training cycle. There are many related parameters that need to be set in the neural network, such as the number of neurons, the number of hidden layers, the training function, etc. These parameters will be specified in Section 3.

3. Experimental Design and Parameter Selection

3.1. Experimental Environment. The relevant experiments were carried out in MATLAB_R2016a with a neural network toolbox. The equipment used in the experiment was MacBook, and the relevant data is shown in Table 1.

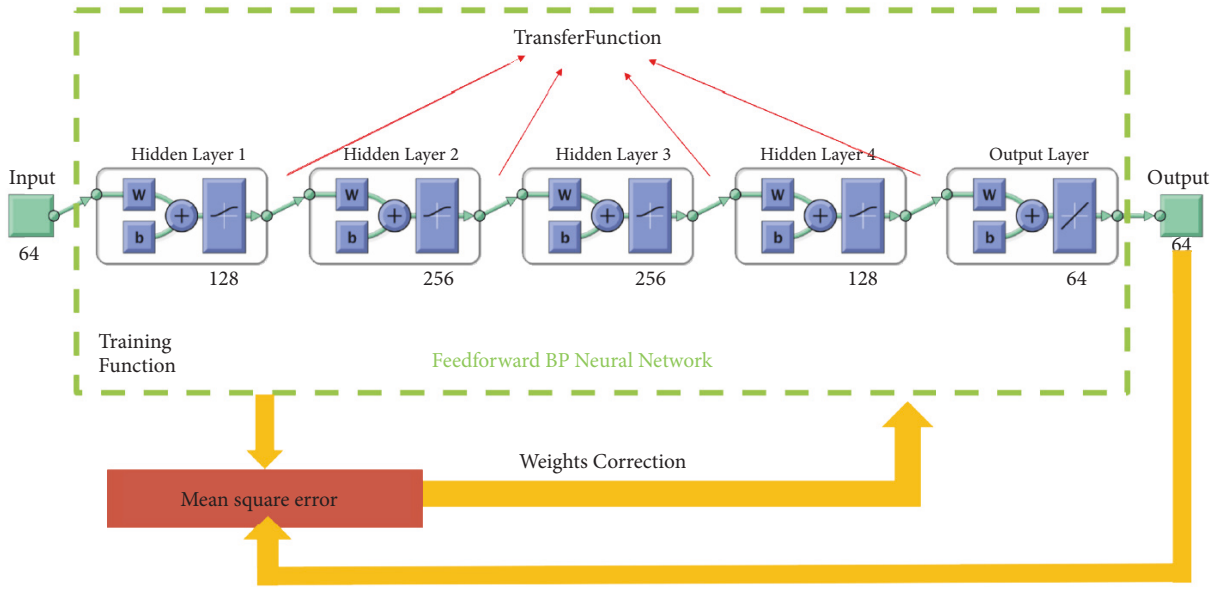

Figure 2: System structure based on feedforward BP neural network.

3.2. Data Representation. We divide the ciphertext data into 64-bit blocks, that is, get 64 channels of input and output, and represent them in matrix form in MATLAB.

$$P = \begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,k} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,k} \\ p_{3,1} & p_{3,2} & \cdots & p_{3,k} \\ \vdots & \vdots & \ddots & \vdots \\ p_{64,1} & p_{64,2} & \cdots & p_{64,k} \end{bmatrix},$$
$$C = \begin{bmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,k} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,k} \\ c_{3,1} & c_{3,2} & \cdots & c_{3,k} \\ \vdots & \vdots & \ddots & \vdots \\ c_{64,1} & c_{64,2} & \cdots & c_{64,k} \end{bmatrix} \quad (1)$$

where P is the plaintext matrix, C is the ciphertext matrix, $p_{(i,j)}$ is the i-th plaintext bit in the j-th plaintext block, $c_{(i,j)}$ is the i-th in the j-th ciphertext block A ciphertext bit, and k is the number of blocks of the ciphertext pair used in the training. Each bit in C corresponds to each bit in P. According to the number of inputs and outputs of the neural network, the number of rows in the two matrices is selected, and all ciphertexts in the ciphertext pair are encrypted by the same key.

3.3. Size and Layout of Neural Network. In the experiment, we choose feedforward backpropagation neural network and cascaded feedforward backpropagation neural network, in which the hidden layer and output layer of the feedforward BP neural network are only affected by the previous layer. The hidden layer and the output layer of the cascaded feedforward BP neural network are related to each of the previous layers. The specific experimental structure is shown in Figures 2 and 3.

The relevant parameters of the neural network are set as follows:

(i) Set four hidden layers, each of which has 128, 256, 256, and 128 neurons. Since the experiment in [6] contains several hidden layer settings. When the setting is four layers, and the neurons in each layer are 128, 256, 256, and 128, respectively, the experiment is successful and the results are better. So we choose this setting. In the future research, we will also change the settings of the hidden layer, taking into account the impact of different settings on the results.

(ii) The training function of the neural network selects the quantized conjugate gradient method 'trainscg', which is suitable for large networks and occupies storage due to the conjugate gradient method. The space is small, and the 'trainscg' quantized conjugate gradient method saves more time than other conjugate gradient methods.

(iii) The error function that selects the correction weight during training is the mean square error (MSE). The error function is also called the loss function. The two most commonly used loss functions are cross entropy and mean square error. Among them, the cross entropy characterizes the distance between two probability distributions, which is a loss function that is used more in the classification problem. Unlike the classification problem, the regression mainly solves the prediction of specific numerical values. The restoration of plaintext from the known ciphertext

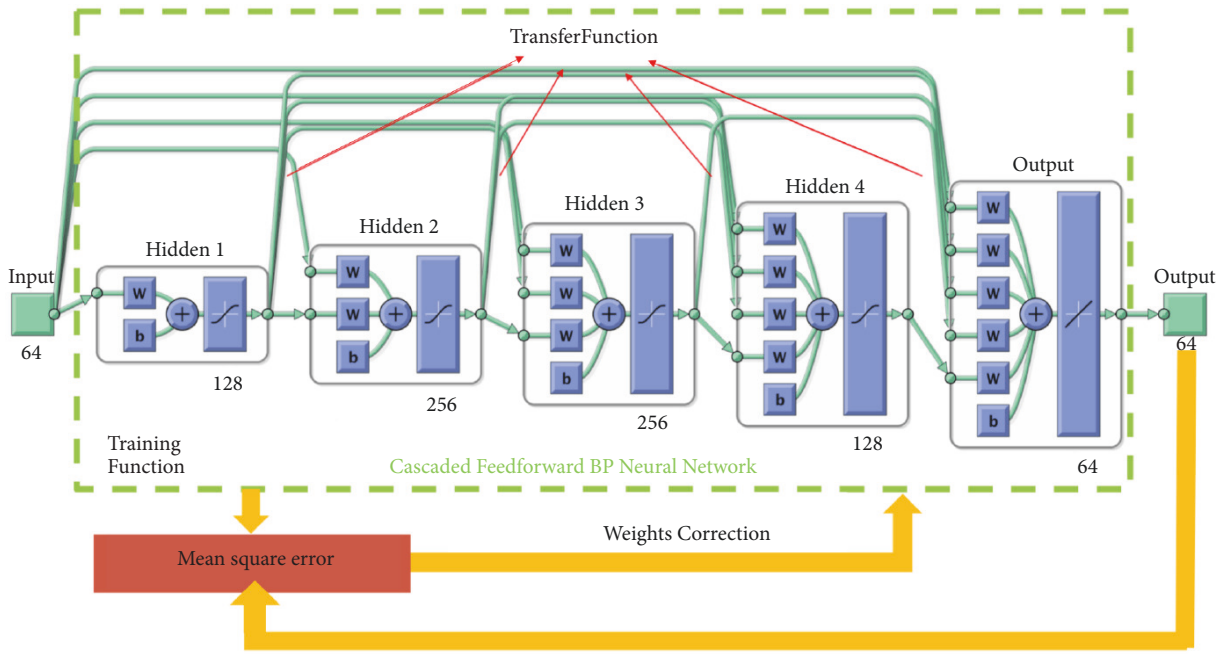

Figure 3: System structure based on cascaded feedforward BP neural network.

belongs to the regression problem. For the regression problem, the most commonly used loss function is the mean square error.

3.4. Training Stop Condition. The maximum number of training cycles is set to 500, and there are three training stop conditions, namely, (1) reaching the final training cycle number of 500; (2) the acceptable mean square error limit of 0.05; (3) continuous verification failure The maximum number of times is 20.

3.4.1. Training Phase. (1) The training process first creates a neural network and selects the network layout, that is, determines the corresponding neurons in each layer, and then sets the training stop conditions.

(2) At the beginning of the training, part of the data in the data set is used as the training set, and the rest is used as the test set. The training of the ciphertext in the training set is performed, the ciphertext is input and the corresponding result is output, and the result is XORed with the original plaintext, and the weight is continuously corrected according to the error function.

(3) The training is over and it is judged whether the training is successful. If the number of consecutive verification failures reaches 10, and the mean square error is higher than 0.1, the training fails; if the mean square error is less than 0.1, the training is successful.

(4) If the training fails, reinitialize the network and return to step (2) to start retraining.

(5) If the training is successful, enter the test process.

3.4.2. Test Phase. (1) After the training is completed, the ciphertext matrix C_{train} used in the training is input into the neural network to obtain the corresponding plaintext result P'_{train}. The P'_{train} is subjected to a bitwise XOR operation with the plaintext P_{train} of the training set, and the error percentage is calculated, which is called the training error. Training errors can be expressed by the following formula:

$$train\_error = \frac{\sum_{i=1}^{m_{train}} \sum_{j=1}^{n_{train}} p'_{train}(i,j) \oplus p_{train}(i,j)}{m_{train} \times n_{train}} \quad (2)$$

where m_{train} is the number of bits per block in the training set, n_{train} is the number of blocks in the training set, $p'_{train}(i,j)$ is the j-th bit in the i-th block of the neural network output, and $p_{train}(i,j)$ is the j-th bit in the i-th block of the plaintext used in the training. In this experiment, $m_{train} = 64$.

(2) The ciphertext C_{test} different from the training set is input into the neural network, and the corresponding result P'_{test} of the neural network is obtained. The P'_{test} and the plaintext P_{test} of the training set are XORed bit by bit, and the error percentage is calculated, which is called the test error. The test error can be expressed by the following formula:

$$test\_error = \frac{\sum_{i=1}^{m_{test}} \sum_{j=1}^{n_{test}} p'_{test}(i,j) \oplus p_{test}(i,j)}{m_{test} \times n_{test}} \quad (3)$$

where m_{test} is the number of bits per block in the test set, n_{test} is the number of blocks in the test set, $p'_{test}(i,j)$ is the j-th bit in the i-th block of the output during the test, and $p_{test}(i,j)$ is the j-th bit in the i-th block of the plaintext used in the test. In this experiment, $m_{test} = 64$.

(3) The ciphertext C_{total} of the entire data set is input into the neural network, and the corresponding result P'_{total} is obtained, and the output P'_{total} and the original plaintext P_{total} are XORed bit by bit, and the calculated error percentage is

Table 2: 2^{11}-size file in the feedforward BP neural network error byte 0-8 bits per byte (unit: byte).

Algorithms	0	1	2	3	4	5	6	7	8	total
AES-128_ECB	118	46	24	23	22	15	7	1	0	256
AES-128_CBC	126	45	20	21	21	13	7	2	0	256
AES-256_ECB	127	44	18	21	20	16	8	2	0	256
AES-256_CBC	117	48	24	21	21	16	7	2	0	256

Table 3: 2^{11}-size file in the cascaded feedforward BP neural network error byte 0-8 bits per byte (unit: byte).

Algorithms	0	1	2	3	4	5	6	7	8	total
AES-128_ECB	128	31	22	24	25	16	8	2	0	256
AES-128_CBC	104	48	29	23	24	17	9	2	0	256
AES-256_ECB	151	24	15	18	22	15	9	2	0	256
AES-256_CBC	104	35	32	33	24	18	7	3	0	256

called the overall error. The overall error can be expressed by the following formula:

$$total\_error = \frac{\sum_{i=1}^{m_{total}} \sum_{j=1}^{n_{total}} p'_{total}(i,j) \oplus p_{total}(i,j)}{m_{total} \times n_{total}} \quad (4)$$

where m_{total} is the number of bits per block in the overall data set, n_{total} is the number of blocks in the data set, $p'_{total}(i,j)$ is the j-th bit in the i-th block of the output, and $p_{total}(i,j)$ is plaintext of the j-th bit in the i-th block. In this experiment, $m_{total} = 64$.

(4) Compare the output result P'_{total} with the original plaintext P_{total}, and count the number of pairs of each byte and the number of errors 1-8 bits, and count the number of consecutive two bytes, four bytes, and eight bytes. The reason for counting the total number of consecutive bytes is that, in Chinese, one Chinese character is two bytes, and the words are generally composed of two Chinese characters, that is, four bytes, and four Chinese characters (eight bytes) may form a phrase or a sentence. So if the plaintext is composed of Chinese, part of the plaintext can be understood by restoring some of the Chinese characters. The same applies to English and other languages. The more the consecutive pairs of bytes correct, the better the understanding of the content of the plaintext.

4. Experimental Result

Experiments are specific to the AES-128, ECB, and CBC modes of the AES-256 algorithm. According to the experimental results in [6] and our current ciphertext data, the experiment consists of two parts: one is to select the image in the Caltech Image Database [10] and splicing into 1000 files of 512 KB size. In plain text, the above four kinds of cryptographic algorithms are used to encrypt each, 1,000 ciphertext files are, respectively, obtained, then the ciphertext files are truncated, the file length is still controlled to 512 KB, a total of 4,000 ciphertext files of 512 KB size are obtained, and then the ciphertext is obtained. Converted to 01 sequence, respectively, and intercepted the first 2^{11} digits for experiment. Second, Python randomly generated 1,000 files of 2^{17} characters as plaintext, respectively, encrypted with AES-128, AES-256 algorithm ECB and CBC modes to get 1,000 ciphertext files, and then cut off the ciphertext files, control the file length to 2^{17} characters. A total of 4,000 ciphertext files with a length of 2^{17} characters are obtained, and then the ciphertexts are converted into 01 sequences. A 01 sequence with a size of 2^{20} is obtained. The result of the experimental output is between $[0, 1]$, we round it up, the result of the result less than 0.5 is 0, and the result of the result greater than or equal to 0.5 is 1.

In the specific experiment, in the first part, we select the first 85% of the data set as the training set and the last 15% as the test set and count the output results of each size of 2^{11}, each byte error 0-8 bits number. The corresponding results are shown in Tables 2 and 3. Since [3, 8] and this paper are all aimed at restoring plaintext, the restoration accuracy of this paper and [3, 8] is compared. The corresponding results are shown in Table 4. In the second part, we select the first 2^{11}-, 2^{12}-, 2^{13}-, 2^{14}-, 2^{15}-, and 2^{16}-bit training neural networks in a data set and count each 0-8-bit error in each byte of the output result of 2^{20}. The corresponding results are shown in Figures 4 and 5. And the number of consecutive two bytes, four bytes, eight pairs of pairs, and the corresponding results are shown in Figures 6, 7, and 8.

It can be seen from Tables 2 and 3 that the two neural networks have little difference in the plaintext restoration results of the four algorithms, and the full pair of bytes can be more than 100, accounting for more than 40% of all bytes. The number of more than half of the bits in each byte is more than 89% of all bytes. Therefore, these two neural networks have a good effect on the plaintext restoration in cryptanalysis, which can effectively reduce the number of exhaustive, significantly shorten the analysis time, and greatly reduce the required resources.

It can be seen from Table 4 that the epoch of the four AES algorithms are all below 50, the MSE is more than 0.05, the epoch of DES and 3DES are both above 200, and the MSE is less than 0.04. This is because the AES algorithm is the latest data encryption standard, and the security is higher than DES and 3DES. It is difficult to restore the plaintext with high

Table 4: Accuracy comparison of AES, DES, and 3DES.

Algorithms	AES-128_ECB	AES-256_ECB	AES-128_CBC	AES-256_CBC	DES_ECB	3DES_ECB
Average epoch	44	37	46	45	352	239
Average MSE	0.0501	0.0536	0.0569	0.0611	0.0308	0.0372
Average error	0.1768	0.2095	0.1699	0.1909	0.083	0.114
Experiment data source and size	[10], 2^{11}bit				unexplained, 2^{20}bit	
Average size of data required for successful experiments	≈1741bit				2^{11}bit	2^{20}bit
Results source	this paper				[6]	[7]

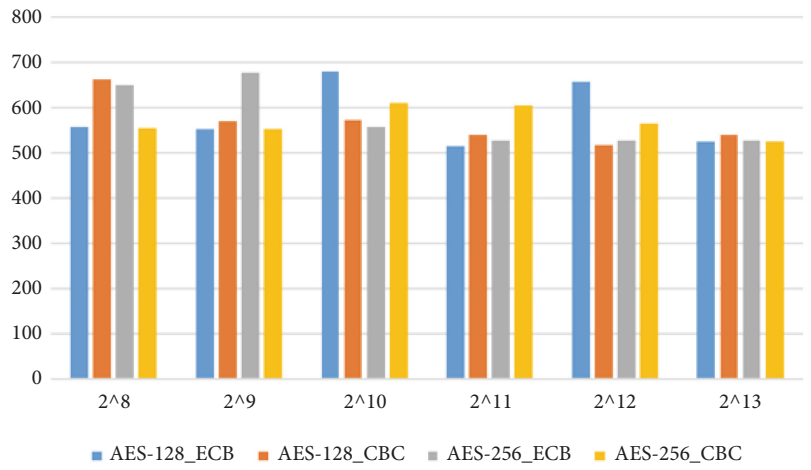

Figure 4: Number of bytes all correct of feedforward BP neural networks.

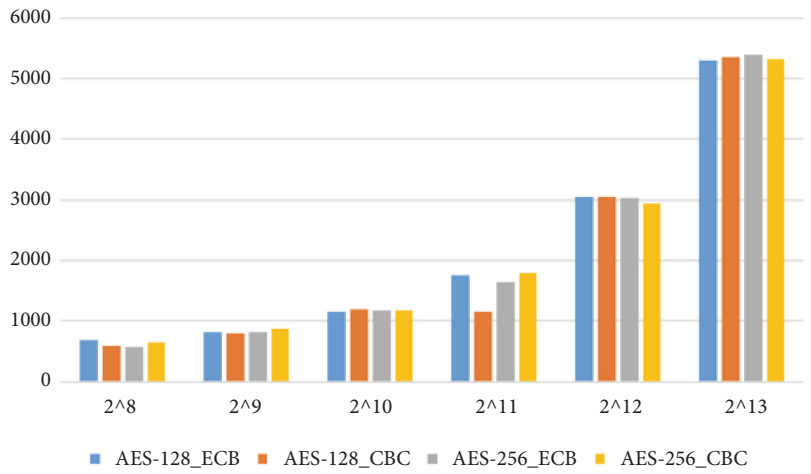

Figure 5: Number of bytes all correct of cascaded feedforward BP neural networks.

accuracy only through 2048 plaintext-ciphertext pairs, and this experiment only uses personal laptop for experiment; memory and performance are limited. Therefore, the epoch is small, resulting in higher error than DES and 3DES. However, only the DES and 3DES algorithms in the ECB mode are considered, the CBC mode is not considered, and the block length of the algorithms and the experimental data source are not explained in the [6, 7].

From Figure 4, in the experimental results of the feedforward BP neural network, the number of bytes in the pair is between 500 and 700, and the number of more than half of the bits in each byte accounts for more than 63%

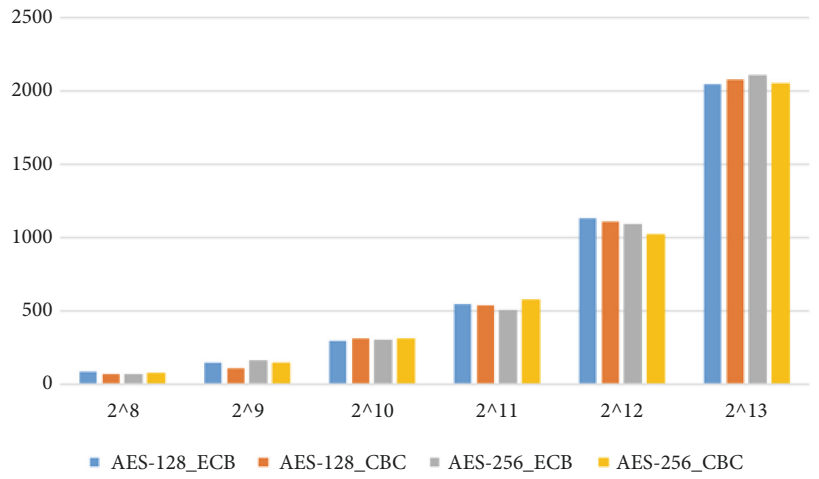

Figure 6: Number of two consecutive bytes all correct of cascaded feedforward BP neural networks.

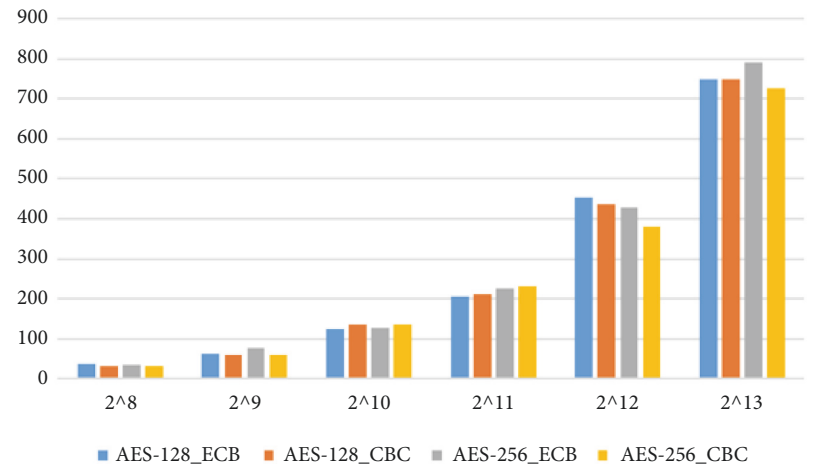

Figure 7: Number of four consecutive bytes all correct of cascaded feedforward BP neural networks.

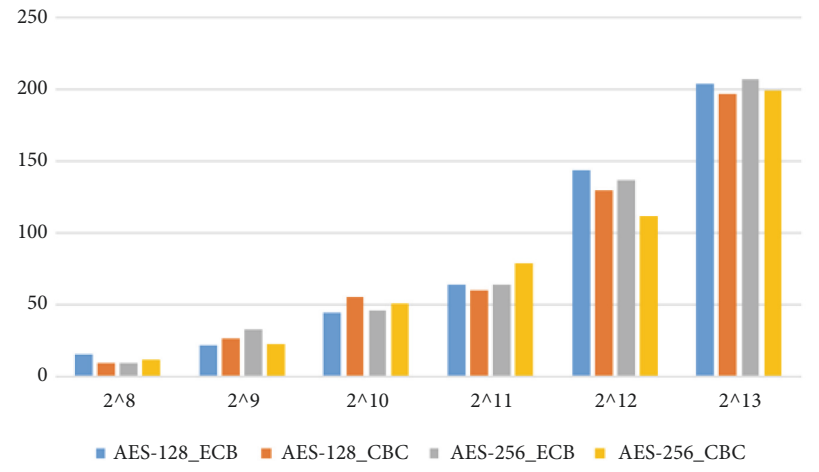

Figure 8: Number of eight consecutive bytes all correct of cascaded feedforward BP neural networks.

of all bytes. And the statistical number of each algorithm is normally distributed. It can be seen from Figure 5 that the cascaded feedforward BP neural network also has a normal distribution for each algorithm, and as the training set data increases, the number of bytes of the pair increases. In the experiment, we also found that for the feedforward BP neural network, when the training set is increased to 2^{10} bytes, the training continues to fail, and the MSE is always above 0.1; while the training success rate of the cascaded feedforward BP neural network is always, it stays above 99%, and as the training set increases, Epoch also increases. For the feedforward BP neural network, there is no obvious linear relationship between the size of the training set data and the number of bytes of the full pair. For the cascade feedforward BP neural network, the size of the training set data, and the pair of words there is a significant positive correlation between the number of sections. As the training set data increases, the number of bytes in the pair increases. Therefore, when the amount of training data increases to a certain extent, the cascade feedforward BP neural network can effectively improve the efficiency of plaintext restoration and reduce the number of exhaustive times required.

Since the feedforward BP neural network is not good for restoring consecutive bytes, this paper only shows the restoration results of the cascade feedforward BP neural network. It can be seen from Figures 6–8 that as the training set increases, the number of bytes all correct increases. When the training set size is the same, the total number of bytes all correct of the four algorithms does not differ much. Figure 6 shows the number of consecutive two bytes all correct. When the training set reaches 2^{13} bytes, there are an average of more than 2,000 two bytes all correct, accounting for more than 3% of the whole. Figure 7 shows the number of four consecutive bytes all correct, when the training set reaches 2^{13} bytes, there are more than 700 four bytes all correct, accounting for more than 2.2% of the whole. Figure 8 shows the number of consecutive eight bytes all correct. When the training set reaches 2^{13} bytes, there are about 200 eight bytes all correct, accounting for about 1.2% of the whole.

5. Conclusion

This paper discusses the global deductive study of AES-128 and AES-256 algorithms. For the research goal, we use the feedforward BP neural network and the cascade feedforward BP neural network to restore the ciphertexts of the AES-128 and AES-256 algorithms in ECB and CBC modes. As a new method for cryptanalysis, neural network can restore the corresponding plaintext according to ciphertext. In the restored result, the number of bytes in all pairs is above 40%, and the number of bytes in more than half is 89%. Above, and for the cascade feedforward BP neural network, as the training set data increases, the error rate decreases, and the number of pairs of all pairs increases. In the global deductive study, we found that different neural networks will get different results, and different data types will lead to differences in error rates. Therefore, we will use this as an opportunity to continue to study different neural networks and consider the impact of more data types on error rates.

As an emerging cryptanalysis method, the cryptanalysis of plaintext restoration based on neural network is still in the experimental stage. The research on AES is also a new attempt. To get better results, we need to know more plaintext in advance and set more restrictions. Future research may require specific structural features specific to the AES algorithm to perform cryptanalysis more efficiently.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Disclosure

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the Open Project Program of the State Key Laboratory of Mathematical Engineering and Advanced Computing; National Key Research and Development Project 2016-2018 (2016YFE0100600); Open Fund Project of Information Assurance Technology (KJ-15-008); State Key Laboratory of Cryptography and Science.

References

[1] L. R. Knudsen, "Block ciphers—a survey," in *State of the art in applied cryptography (Leuven, 1997)*, vol. 1528 of *Lecture Notes in Comput. Sci.*, pp. 18–48, Springer, Berlin, 1998.

[2] A. G. Bafghi, R. Safabakhsh, and B. Sadeghiyan, "Finding the differential characteristics of block ciphers with neural networks," *Information Sciences*, vol. 178, no. 15, pp. 3117–3131, 2008.

[3] A. Hussein Al-Hamami, K. Alallayah, A. Mohamed, and W. Abdelwahed, "Applying neural Networks for simplified data Encryption Standards (SDES) Cipher System Cryptoanalysis," *International Arab Journal of Information Technology*, vol. 9, no. 2, pp. 2423–2432, 2012.

[4] K. Alallayah, M. Amin, W. A. El-Wahed, and A. Alhamami, "Attack and construction of simulator for some of cipher systems using Neuro-Identifier," *International Arab Journal of Information Technolog*, vol. 7, no. 4, pp. 365–372, 2010.

[5] K. M. Alallayah, W. F. Abd El-Wahed, M. Amin, and A. H. Alhamami, "Attack of against simplified data encryption standard cipher system using neural networks," *Journal of Computer Science*, vol. 6, no. 1, pp. 29–35, 2010.

[6] M. M. Alani, "Neuro-Cryptanalysis of DES and Triple-DES," in *Neural Information Processing*, vol. 7667 of *Lecture Notes*

in Computer Science, pp. 637–646, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[7] M. M. Alani, "Neuro-cryptanalysis of DES," in *Proceedings of the World Congress on Internet Security, WorldCIS-2012*, pp. 23–27, Canada, June 2012.

[8] A. Bahubali and V. Desai, "Artificial Neural Networks for Cryptanalysis of DES," *International Journal of Innovations in Engineering and Technology*, vol. 2, no. 4, pp. 11–17, 2013.

[9] M. Danziger and M. A. Henriques, "Improved cryptanalysis combining differential and artificial neural network schemes," in *Proceedings of the 2014 International Telecommunications Symposium (ITS)*, pp. 1–5, Sao Paulo, Brazil, August 2014.

[10] G. Griffin, A. Holub, and P. Perona, *Caltech-256 Object Category Dataset*, California Institute of Technology, 2007.

[11] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.