American Association for Science and Technology
AASCIT Communications
Volume 1, Issue 2
July 20, 2014 online

A Novel Approach to Synchronization Problem of Artificial Neural Network in Cryptography

Ömer Faruk Ertuğrul

Keywords: Cryptography, Synchronization, Artificial Neural Network

It is becoming increasingly difficult to have data security nowadays. There have been used various cryptography methods in literature, but recent developments in computational area have heightened the need of new methods. In this study the feed-forward artificial neural network (FFNN) was used with a different perspective by using the structure of artificial neural network as a key as a solution of synchronization problem of FFNNs in cryptography. The proposed method was employed for text, audio and image data and the results were found acceptable. Also, the results of FFNN were assessed with the results of probabilistic, radial basis artificial neural networks and k nearest neighbor and wavelet transforms. The FFNN was faster than these methods and had 100% decryption accuracy.

Introduction

In the history of human development, knowledge has been thought of as a key factor in various areas and for some cases the key that makes the knowledge important is depends on keeping it secret [1]. Data security involves cryptography, ensuring integrity of data, verification of sender and prevention of denial. Cryptography algorithms are classified into private (symmetric) key (DES, AES, RC4, etc ...), public (asymmetric) key (RSA, Robin, etc ...) and mixed cryptography. However, a major problem with this kind of application is the increase in the number of new attack methods and computational capacity. Therefore novel methods as an alternative of existing cryptography methods must be developed for being more secure against attacks.

Kinzel and Kanter showed that the artificial neural network (ANN) remains secure for attacks and they suggested that ANN may be lead in cryptography systems depend on its speed and simplicity. Additionally they mentioned that each message can be encrypted by a new key easily by ANN [2]. Also Mislovaty et al. reported that ANN was secure against brute-force attacks [3]. Therefore, there has been an increasing interest in the applicability of ANN in cryptography and a large and growing body of literature has investigated about this, such as multi-layer [4], feed-forward [5], radial-based [6], recurrent [7], chaotic [8-10], hopfield [11] ANNs. The ANN is successful against attacks because of (1) high learning and generalization capability, (2) quick or parallel processing capability, (3) hardware realizing, (4) being easily adapted to different disciplines, (5) difficultly decrypting by brute-force attack. In previous studies ANN was used in cryptography, for data [4, 10], image [7], and sound [12] encryption and also, as a public key generator [13].

The structure of ANN depends on the distribution of data in the training set which is an important advantage of using ANN as a cryptography method, and also a big obstacle to widespread ANN cryptography therefore the encrypt and decrypt ANNs must be synchronized [14] by using a unique characteristic due to the training set. Also Rosen-Zvi et al. pointed out that the synchronization as a question [15]. Volna et al. indicates the synchronization of ANNs as a complex dynamical process in their review [16]. The high need of cryptography is depends on the problem of having a secure channel inside the transmitter and receiver and sending the whole training data in unsecure channel. Additionally still two ANN may be optimized with different weights and biases because of using the same training depend on the ANN training parameters such as learning rate and momentum and also network structure such as the number of hidden layers and neurons. As a summary one of the most significant discussions in ANN cryptography is synchronization of ANNs. Several studies have proposed to synchronize the ANNs, but there is not still a simple and fast method for this. Kanter et al. proposed a method for synchronizing two feed forward artificial neural network (FFNN) by exchanging and learning their mutual outputs for giving common inputs. They also performed that their proposed synchronization method is fast and needs less bits than the number of components of the weights [17, 18]. Ruttor et al. proposed a synchronization method that generates the queries in a determined order for both encryption and decryption ANNs [19]. Klein et al. used mutual learning for ANN synchronization [20]. Adel et al. reviewed the usage of ANN and the proposed solutions of synchronization of ANNs in their review in detail [21]. The aim of this study was to evaluate and validate a new and simple approach to overcome the synchronization of ANNs by using

the structure of ANNs such as weights and bias of neurons as a key which can be assigned randomly or user-defined. The proposed method was adapted to a public key, private key and mixed cryptography algorithm, and the results were found successful. Furthermore the results of FFNN were compared with probabilistic ANN (PN), radial-based ANN (RBN), k nearest neighbor (kNN) and wavelet transform (WT) methods. The major importance of this study is the weights and biases were used as a key to a solution of synchronization. Additionally, any sized keys can be employed for encrypting any data type, such as text, audio, image with FFNN. Section 2 begins by laying out the theoretical dimensions of the ANN, how the weights change depending on the training set and why the synchronization is an important issue. Section 3 describes the analysis of the proposed approach and asses the results of the new approach with PN, RBN, kNN and WT. The last chapter concludes the paper.

Method

In this study a new approach of using feed-forward ANN was proposed to overcome the synchronizing problem of encrypt and decrypt ANNs by using the network structure as a key. The artificial neural network (ANN) method, which is also called multilayer perceptron, is based on the findings of the biological nervous system [22]. In this artificial neural system, there are nerve cells, which are joined together in a variety of ways to form networks. Perceptrons, which can be used for classifying data only two classes and is a single layer network with threshold, were offered in 1959 by Rosenblatt that models a biological neuron [23]. Later on Minsky and Papert showed that a single perceptron can not model even a simple exclusive-or function, but two layer feed forward network can overcome these problems with optimized network parameters in 1969 [24]. Therefore ANN has become popular since 1986 that Rumelhart et al. offered back propagation learning algorithm that can be used for optimizing Feed forward neural network (FFNN) parameters [25]. FFNN is still one of the popular ANN algorithms that the inputs are taken into account in forward direction. The FFNN consists of layers, namely the input layer, the inner layers (hidden layers) and the output layer. The input layer receives data from the external world. The output layer presents the data to the user. The hidden layers between these 2 layers are where the data is processed. Also, there is not any connection in neurons that are in the same layer. The number of the nerve cells in the hidden layer is significant for the performance as well as the length of the network. Usually the output of each neuron is determined by using a nonlinear activation function such as a sigmoid or hyperbolic tangent. ANN is trained by experience, when applied to an unknown input in the network; it can generalize from past experiences and produce a new result [26].

To date various methods have been developed and introduced to train ANNs, the back propagation learning algorithm is the traditional learning method used to optimize network parameters, weights and biases. The back propagation learning algorithm [27], a gradient descent method, offered by Rumelhart et al. [25] that is also known as generalized delta rule. In BP optimum weights are determined by back propagating the errors from output layer. As a simple way the errors are taken into account in backward direction. Back propagation learning algorithm can be employed to networks with any number of layers. There are many different parameters that should be decided when designing an ANN such as the number of layers, the number of neurons per layer, and the number of training iterations, learning rate, and activation function.

As it is obvious, the order of the training data and parameters, and also the network parameters are strongly related to the network parameters (weights and biases). Two same structured (number of hidden layers and neurons in hidden layers) networks that are trained by the same dataset with different order may be optimized for different weights. To solve this situation the training dataset must also be transmitted from encryption side to decryption side, however this is another problem for having secure link. The proposed method was overcome this drawback of ANN. Here in two different keys can be used one is the structure of the network and the other is network parameters.

The general algorithm of the proposed method is shown in Figure 1.a and Figure 1.b.

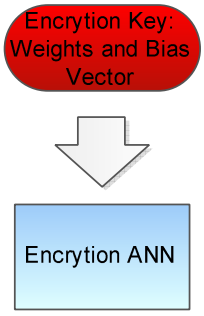

Figure 1a. *Encryption.*

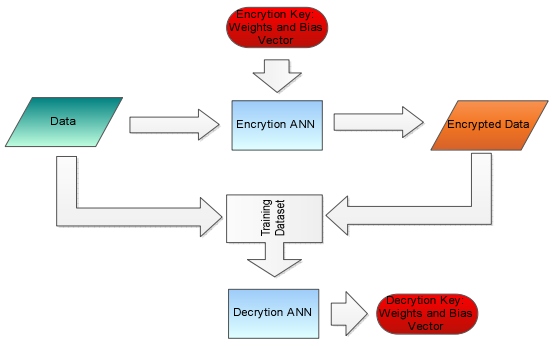

Figure 1b. *Decryption.*

As seen in Figure 1.a the encryption ANN does not require training. The creation of decryption ANN is shown in Figure 1.b. As seen in decryption stage, also the encryption ANN is created for obtaining the training dataset of decryption ANN. The encryption and decryption ANNs can be easily copied or cloned without the need of any further training. The codes of proposed approach and its output are in appendix.

Results and Discussion

The new approach was successfully applied to private and public key cryptography. Private key cryptography; encryption and decryption keys are same where they are different in public key cryptography. Private key cryptography was basically formed with two separate modules (encryption and decryption modules) and a training set was created only creates for the training decryption module. In public key cryptography; the ANN was also used as a public key generator that creates encryption or decryption keys separately depend on the other one. Text, audio and video files were encrypted and decrypted successfully with 100% accuracy. There is a 399x600 sized image sample in Figure 2.a and its encrypted form by FFNN is shown in Figure 2.b.

Figure 2a. *An Image.*

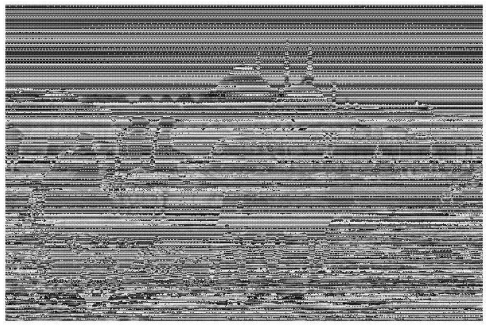

Figure 2b. *Encrypted Image.*

A 1x300 vector sized audio signal is shown in Figure 3.a and the encrypted of the same audio with FFNN is illustrated in Figure 3.b.

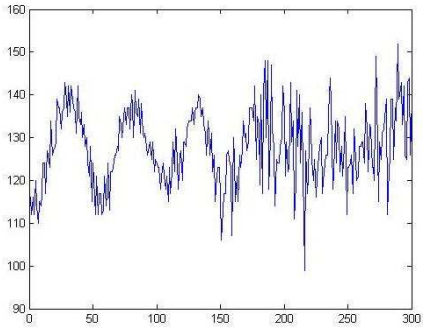

Figure 3a. *Audio Sample.*

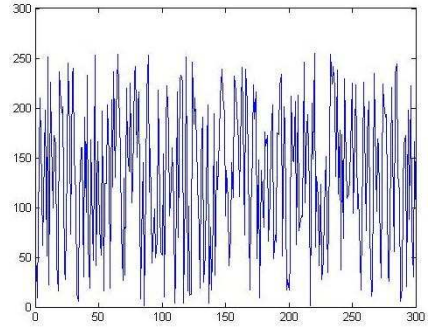

Figure 3b. *Encrypted Audio Sample.*

Image and audio were encrypted and decrypted successfully. The encryption and decryption speeds of different sized networks are sorted in Table 1. The tests were done by Intel (R) Core (TM) 2 Duo CPU, P8400, 2.26 GHz and 1.97 GB RAM computer.

The grown in the size and complexity of ANN causes a slower encryption and decryption stage as seen in Table 1. The FFNN is fast because it does not need any training in encryption module. In this study the key, which may be user-defined or randomly generated, can be at desired amount, size and any data type (text, audio and image). The decryption key is generated by using the encryption key. The key generation durations are listed in Table 2.

Table 1. *Encryption and Decryption Speed.*

	1 Hidden Layer		**2 Hidden Layers**		**3 Hidden Layers**	
Number of Neurons	10	20	4 // 10	10 // 20	4 // 10 // 4	10 // 20 // 10
Encryption (kb/sec)	46,51	44,94	43,76	43,10	42,28	42,10
Decryption (kb/sec)	244,72	238,10	215,65	210,53	198,02	186,91

Table 2. *Key Generation Process of FFNN.*

Number of Hidden Layers	Distribution of Neurons	Number of Neurons	Key Length (bits)	Key Generation Time (sec)
1	1	1	16	1,0907
	1 // 1	2	32	1,1782
2	5 // 1	6	128	1,1521
	10 // 1	11	248	1,1387
	1 // 1 // 1	3	48	1,1538
3	1 // 4 // 1	6	120	1,1698
	1 // 10 // 1	12	264	1,1826
	10 // 100 // 1	111	9.768	2,9799
4	1 // 10 // 10 // 1	22	1.144	1,2266
5	1 // 10 // 20 // 10 // 1	42	3.704	1,4669
6	1// 10 // 20 // 20 // 10 // 1	62	7.064	2,8280

As seen in Table 2 the key size is variable, on the other hand, when key size is increased, the key generation speed is decreased depending on the increase of the number of neurons of complexity of feed-forward ANN. As it was reported by Mislovaty et al. ANN was secure [3] because each FFNN produces its own outputs depend on its unique structure and network parameters such as training set, learning rate, moment, transfer functions, weights and biases of each particular neuron in the network. E.g. the data encrypted by a FFNN cannot be decrypted with another FFNN that optimized with another key, since each key shows the weights and biases network and network structure such as the number of hidden layers and neurons. The accuracy rate of decryption was obtained as 0.3922% for each case, as mentioned in the literature review.

Table 3. *Process Speed (kb/sec).*

Cryptography Method	Encryption	Decryption
FFNN	44,94	238,1
PN	0,94	0,98
RBN	0,87	0,95
kNN	2,53	2,68
WT	13,18	63,69

In this study the applicability of using FFNN was also tested with probabilistic ANN (PN), radial-based ANN (RBN) [6], k nearest neighbor (kNN) [28] and wavelet transform (WT) [13] methods. These algorithms were tested with the test data size was 2040 bit and key size was 520.2 kbit. The FFNN had 20 neurons in a single hidden layer, one nearest neighbor was used in kNN and 'db3' transfer function was used in WT. Speed for encryption, decryption, training encryption and decryption models are listed in Table 3.

The fastest encryption and decryption process can be done with FFNN as seen in Table 4. However FFNN uses supervised (Levenberg-Marquardt) training method therefore creating decryption key stage takes long time. Encryption and decryption speeds of FFNN in Table 4 are enough for using this method in real time applications. Further analysis showed that the accuracy of employing different datasets for training encryption and decryption modules was 0.3922 %, additionally the accuracy of decrypting an encrypted data by using wrong decryption key in private key cryptography was 0% in PN and kNN. Additionally, the success rate of employing a different number of nearest neighbors in encryption and decryption stage was 0%. Furthermore the decryption accuracy of employing different transfer functions was 0.3922% in all types of ANN and WT. The kNN is strong against attacks because of employing a unique training set and unique number of nearest neighbors. The synchronization of training dataset is highly important for kNN and also for WT.

As a summary, the ANN is a secure algorithm against attacks due to employing unique key, training set, training conditions, network structure and transfer function. To decrypt an encrypted message the main issue is to transmit these data through a public channel, therefore to decrease this weakness a new and simple approach was proposed for the synchronization of ANNs. In this approach the network parameters were used as a security key in encryption stage and a new training dataset were obtained in decryption stage by using encryption FFNN. After all this dataset were used for creating decryption ANN.

Conclusion

In this study a new approach was proposed for synchronization of encryption and decryption ANNs by using random or user-defined keys as a weight and bias vector of FFNN and the results of this method was compared with PN, RBN, kNN and WT. This study set out with the aim of assessing the applicability of this approach and it was tested with public key, private key and mixed cryptography algorithms and for text, image and video data successfully with 100%. There was a significant positive correlation between the encryption and decryption speeds of FFNN and the complexity and size of the ANN.

Appendix

Code

```
% Key: It can be also user
defined
r=randperm(1000);
wba=r(1:31)'/1000; %This is
the key

% Message
Message='AASCIT
Communication';%Message
Ph=double(Message);%Digitali
zation of Message
disp(['Original Message:
',Message]);
disp(['Original Message:
',int2str(Ph)]);

%% Encryption
```

```
% Generating Encryption ANN
net1ff = newff([1 255],[10
1],{'purelin' 'purelin'});
% Configuring ANN by using
key
net1ff=setx(net1ff, wba);

%% Encrypting Message
Y1ff = sim(net1ff,Ph);
En_mes=char(round(Y1ff));
disp(['Encrypted Message:
',En_mes]);
disp(['Encrypted Message: ',
int2str(Y1ff)]);

%% Decryption
% Generating Encryption ANN
netx1ff = newff([1 255],[10
1],{'purelin' 'purelin'});
netx1ff=setx(netx1ff, wba);

%Generating Training Dataset
Pc=1:255;%ASCII Characters
Pg=sim(netx1ff,Pc);%
Encrypted ASCII Characters

%Generating Decryption ANN
netx2ff = newff([1 255],[10
1],{'purelin' 'purelin'});
%Training Decryption ANN
netx2ff.trainParam.goal =
0.01;
netx2ff.trainParam.epochs =
1000;
netx2ff =
train(netx2ff,Pg,Pc);

wbb=getx(netx2ff); %Decrypti
on key

% Generating new
synchronized ANNs if
required
net2ff = newff([1 255],[10
1],{'purelin' 'purelin'});
net2ff=setx(net2ff, wbb);

%% Decrypting Message
Y2ff = sim(netx2ff,Y1ff);
Y1ff = sim(net1ff,Ph);
De_mes=char(round(Y2ff));
disp(['Decrypted Message:
',De_mes]);
disp(['Decrypted Message: ',
int2str(Y2ff)]);
```

The Output of Code

Original Message: AASCIT Communication
Original Message: [65 65 83 67 73 84 32 67 111 109 109 117 110 105 99 97 116 105 111 110]

Encrypted Message: ffir4i¬©©¶«£´£¬«
Encrypted Message: [102 102 130 105 114 131 52 105 172 169 169 182 171 163 154 151 180 163 172 171]

Decrypted Message: AASCIT Communication
Decrypted Message: [65 65 83 67 73 84 32 67 111 109 109 117 110 105 99 97 116 105 111 110]. ■

Ömer Faruk Ertuğrul
Department of Electrical and Electronics Engineering, Batman University, 72060, Batman, Türkiye
omerfarukertugrul@gmail.com

References

[1] Agrawal, H., & Sharma, M. A Review of Text Encryption Techniques. *Asian Journal Of Computer Science & Information Technology*. 4(5): 47-54 (2014)

[2] Kinzel, Wolfgang, and Ido Kanter. "Neural cryptography." *arXiv preprint cond-mat/0208453* (2002).

[3] R. Mislovaty, E. Klein, I. Kanter ve W. Kinzel, "Security of neural cryptography." *sign (hi)* 1 (2004): 1.

[4] Yee, Liew Pol, and Liyanage C. De Silva. "Application of multilayer perceptron networks in public key cryptography." *Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on*. Vol. 2. IEEE, 2002.

[5] Sağıroğlu, Şeref, and Necla Özkaya. "Neural Solutions for Information Security." *Gazi Üniversitesi Politeknik Dergisi* 10.1 (2007).

[6] Zhou Kaili, Kang Yaohong, Huang Yan ve Feng Erli. "Encrypting Algorithm Based on RBF Neural Network."*Natural Computation, 2007. ICNC 2007. Third International Conference on*. Vol. 1. IEEE, 2007.

[7] Arvandi, M., S. Wu, and A. Sadeghian. "On the use of recurrent neural networks to design symmetric ciphers." *Computational Intelligence Magazine, IEEE* 3.2 (2008): 42-53.

[8] Su, Scott, Alvin Lin, and Jui-Cheng Yen. "Design and realization of a new chaotic neural encryption/decryption network." *Circuits and Systems, 2000. IEEE APCCAS 2000. The 2000 IEEE Asia-Pacific Conference on*. IEEE, 2000.

[9] Zhang, Yunpeng, et al. "The Improvement of Public Key Cryptography Based on Chaotic Neural Networks." *2008 Eighth International Conference on Intelligent Systems Design and Applications*. Vol. 3. 2008.

[10] Dalkiran, Ilker, and Kenan Danisman. "Artificial neural network based chaotic generator for cryptology." *Turkish Journal of Electrical Engineering and Computer Sciences* 18.2: 225-240, 2010.

[11] Liu, Niansheng, and Donghui Guo. "Security analysis of public-key encryption scheme based on neural networks and its implementing." *Computational Intelligence and Security*. Springer Berlin Heidelberg, 2007. 443-450.

[12] Liu, Niansheng, and Donghui Guo. "Security analysis of public-key encryption scheme based on neural networks and its implementing." *Computational Intelligence and Security*. Springer Berlin Heidelberg, 2007. 443-450.

[13] Meghdad Ashtiyani, Soroor Behbahani, Saeed Asadi ve Parmida Moradi Birgani, "Transmitting Encrypted Data by Wavelet Transform and Neural Network, *2007 IEEE International Symposium on Signal Processing and Information Technology, IEEE*, Pages:385-389, 2007.

[14] Godhavari, T., N. R. Alamelu, and R. Soundararajan. "Cryptography using neural network." *INDICON, 2005 Annual IEEE*. IEEE, 2005.

[15] Kanter, Ido, and Wolfgang Kinzel. "The Theory of Neural Networks and Cryptography." *Quantum Computers and Computing* 5.1 (2005): 130-140.

[16] Volna, E., Kotyrba, M., Kocian, V., & Janosek, M. Cryptography Based On Neural Network. *In Proceedings 26th European Conference on Modelling and Simulation.* 386-391 (2012).

[17] Rosen-Zvi, Michal, Ido Kanter, and Wolfgang Kinzel. "Cryptography based on neural networks—analytical results." *Journal of Physics A: Mathematical and General* 35.47 (2002): L707.

[18] Kanter, Ido, Wolfgang Kinzel, and Eran Kanter. "Secure exchange of information by synchronization of

neural networks." *EPL (Europhysics Letters)*57.1 (2002): 141.

[19] Ruttor, Andreas, Wolfgang Kinzel, and Ido Kanter. "Neural cryptography with queries." *Journal of Statistical Mechanics: Theory and Experiment* 2005.01 (2005): P01009.

[20] Klein, E., Mislovaty, R., Kanter, I., Ruttor, A., & Kinzel, W. "Synchronization of neural networks by mutual learning and its application to cryptography." *Advances in Neural Information Processing Systems*. 2004.

[21] Adel A. El-Zoghabi, Amr H. Yassin, Hany H. Hussien. *Survey Report on Cryptography Based on Neural Network. International Journal of Emerging Technology and Advanced Engineering,* 3(12):456-462, 2013.

[22] Bishop, Christopher M. "Neural networks for pattern recognition." (1995): 5.

[23] Rosenblatt F., 1959. Principles of Neurodynamics. *New York. Spartan Books*:23-26

[24] Minsky M., Papert S. Perceptrons: An introduction to Computational Geometry. *The MIT Press*: 1969, 13-33

[25] Rumelhat DE., Hinon GE., Williams RJ. Learning representations by back-propagating errors. *Nature,* 1986, 323:533-536

[26] Fausett, L, Fundamentals of Neural Networks, *New York: Prentice Hall,* 1994.

[27] Ben Kröse, Patrick van der Smagt, "An introduction to Neural Networks", *The University of Amsterdam,* Eighth edition, 1996

[28] Zhan, J., Chang, L., & Matwin, S. Building k-nearest neighbor classifiers on vertically partitioned private data. *IEEE International Conference on Granular Computing*, 2: 708-711, 2005