Секция 7

Нейросетевые системы обработки данных, распознавания образов и управления

S.V. BEZOBRAZOV, J.M. BLACKLEDGE
Brest State Technical University, Belarus
School of Mathematics, Statistics and Computer Science, University of KwaZulu-Natal, South Africa, Durban
bescase@gmail.com, dvcresearch@ukzn.ac.za

PERSONAL CRYPTOGRAPHY USING ARTIFICIAL NEURAL NETWORKS

This paper presents a method of generating encryption algorithms using neural networks. Based on the application of natural noise sources we 'teach' a system to approximate the input noise with the aim of generating an output nonlinear sequence. This output is then treated as an iterator which is subjected to a range of tests to check for potential cryptographic strength in terms of metric. This approach provides the potential for generating an unlimited number of unique Pseudo Random Number Generator (PRNG) that can be used on a 'one-to-one' basis.

Keywords: *coding and encryption, artificial neural networks, multiple algorithms, personalized encryption engines.*

Introduction

Protective software (i.e. antiviruses, intrusion detection systems etc.) is as an integral part of an operating system, e.g. file managers, browsers, text processors etc. However, even with use of the latest protective software products there is no guarantee that the computer and/or computing environment is 'safe'. Such infamous malware for cyber espionage as Stuxnet, Flamer, Duqu, Gauss, Regin and others gives proof that up-to-date protective solutions are powerless in the face of sophisticated cyber-weapons. The listed malware may not only still undetectable for years but also perform cyber espionage by controlling infected computers. The existing situation opens a possible arena in regard to personalized encryption algorithms and data protection systems. Personal encryption algorithms allow users to protect data even if protective software fails to detect a threat.

Our previous work in this area [1] presented an approach to using evolutionary computing for generating personal ciphers using input data streams consisting of natural noise. We showed that the proposed approach provides the potential for generating an unlimited number of unique PRNG that can be used on a 'one-to-one' basis.

In this paper we present the implementation of an Artificial Neural Network for generating personal ciphers. It will be shown that neural networks are powerful tools in cryptography, provide a route to producing a strong personal cipher and demonstrate the highest operation speed in comparison with evolutionary computation approach.

The paper is organized as follows. A state-of-the art is given in Section 2. Section 3 gives a background of how evolutionary computing can be used to generate a strong cypher. The implementation of an Artificial Neural Network, in particular, a Radial Basis Function Neural Network for cipher generation is given in Section 4. Section 5 provides the experimental results. The final section concludes this paper.

I. State-of-the-art

Neural cryptography is a direction in cryptography that applies Artificial Neural Networks (ANN) for encryption and cryptanalysis. Recent works in the use of neural networks in cryptography can be categorized into three major parts.

1. *Synchronization neural networks.* The first approach of applying an ANN in cryptography is through the creation of two topologically identical neural networks. Neural networks can synchronize by learning from each other and full synchronization can be achieved in a finite number of steps. The main idea is as follows: the user A wants to communicate with user B, but they cannot exchange a secret key through a secure channel; so they create two topologically identical neural networks but with different random weights and evaluate them with the same inputs until the weights of both ANN's match [2] – [4]. The process of synchronization of the networks can be considered as the key generation process in cryptography. The common identical weights of synchronized networks for two partners can be used as a key for encryption.

2. *Chaos-based cryptography* combines two research fields, i.e., chaos (nonlinear dynamic systems) and cryptography and becomes more practical in the secure transmission of large multi-media files over public data communication networks. In this field, the implementation of neural networks composed of chaotic neurons opens a wide area for developing strong cryptosystems. W. Yu and al. proposed an encryption technique based on the chaotic Hopfield neural network with time varying delay [5]. The chaotic neural network is used for

generating binary sequences for masking the plaintext. The binary value of the binary sequence chooses the logistic map randomly, used for generated the binary sequences. The plaintext is masked by switching the chaotic neural network maps and the permutation of generated binary sequences.

In the area of image cryptography, a triple key chaotic neural network was used by S. Suryawanshi et al. [6]. A Triple Key means that three parameters are used as control parameters producing a hexadecimal sequence. The triple parameters are used to perform the various operations on an image so as to scramble the data.

T. Fadil et al. presented an algorithm coupled with a chaotic neural network to encrypt MPEG-2 video codecs [7]. The algorithm supports quality and bit rate control that is required by many video transmission applications. The logistics map is used with a neural network to produce a combination of chaotic neural networks based on a binary sequence generated from the logistic map, the biases and weights of neurons are modified and taken to be the secret key.

3. *Multilayer neural network in the cryptography.* K. Noaman et al. presented an encryption system based on General Regression Neural Network [8]. Here a neural network is used to construct an efficient encryption system by using a permanently changing key. The simulation results provide a relatively better performance than traditional encryption methods.

E. Volna et al. studied the use of back-propagation neural networks in cryptography [9]. The model converts the input message in to ASCII code and then gets the sequence of bits for each code which is divided into 6 bit blocks which are used as input for the encryption process. The cipher key is the neural network structure containing an input layer, hidden layer, output layer, and updated weights.

In this paper we propose a different technique that is based on the ability of an ANN to simulate a high entropy input with the aim of transforming the result into a low entropy output. However, this process can be reversed to generate a high entropy output from a low entropy input. In this sense, an ANN can be used to generate a cipher by simulating natural noise once it has been trained to do so.

II. Cryptography using evolutionary computation

Evolutionary Computing is associated with the field of Computational Intelligence, and, like Artificial Intelligence, involves the process of continuous and combinatorial optimizations. It is inspired from biological mechanisms of evolution and uses iterative processes in a parallel manner to achieve the goal.

The application of evolutionary algorithms to cryptology as presented in [1] is, to the best of the author's knowledge, an original concept. The proposed technology is the first of its kind to break away from the Kerckhoff principle, a

principle which states that «*A cryptosystem system should be secure even if everything about the system, except the key, is public knowledge*». We use the principles of evolutionary computing to automatically generate an algorithm that is unique to the user of the algorithm and the keys that initiate it. The technology generates ciphers of a high cryptographic strength using input data streams consisting of natural noise source such as atmospheric noise, cosmic noise, radioactive decay and so on. Fig. 1 demonstrates the basic steps of the proposed approach in schematic form.

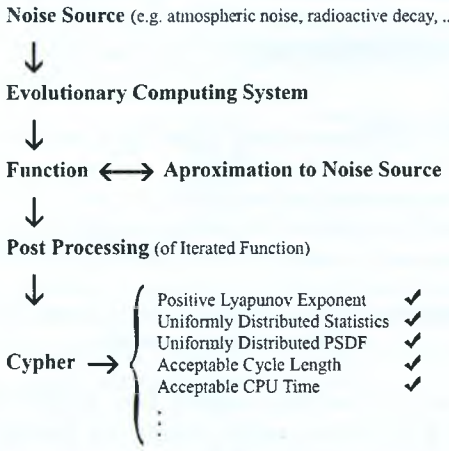

Fig. 1. Schematic of the processes for evolving a cipher

Evolutionary computing systems, which use mechanisms such as reproduction, mutation, recombination, selection etc., try to find the best equation for an optimal approximation to a given data stream. We use the *Eureqa* system [10] developed by Nutonian Inc. for detecting equations and hidden mathematical relationships in the data. This system uses evolutionary computations (in particular, symbolic regression) and iteratively develops a nonlinear function to described stochastic input signals usually associated with experimental data output from a chaotic system, for example. We input a noise source to the system with the purpose of 'forcing' the system to output a result (a nonlinear function) that is an approximation to the input noise. If genuine random (delta uncorrelated) noise is input into the system, then, from a theoretical point of view, no nonlinear function should be found on an evolutionary basis. Thus, inputting natural noise is a way of 'cheating' the system to 'force' it to provide a result that may be suitable (on an iterative basis) as a PRNG. The output is then

treated as an iterated function which is subjected to a range of tests to check for potential cryptographic strength.

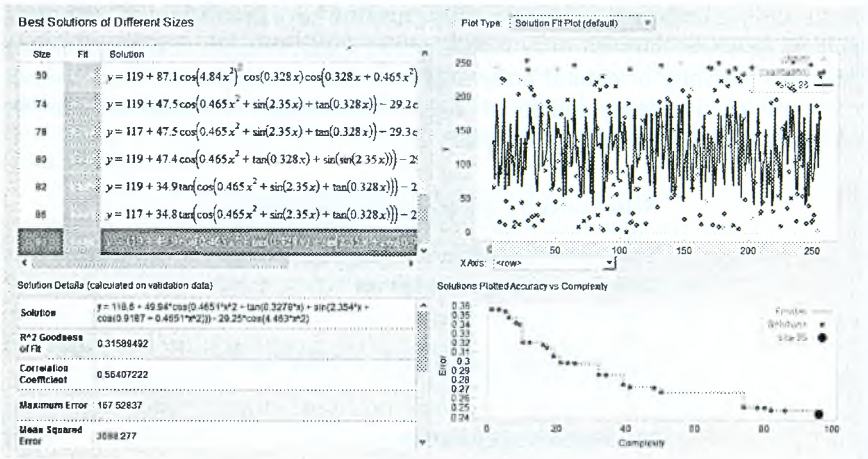

Fig. 2. Screen shot of *Eureqa* used for evolving nonlinear functions suitable for cipher generation

For this study, we use the data available from RANDOM.ORG which, to date, has generated 1,92 trillion random bits for the Internet community [11]. Fig. 2 shows an example screen shot of the *Eureqa* system used to generate the following iteration function for cipher generation:

$$c_{i+1} = 118{,}6 + 49{,}9\cos(0{,}4c_i^2 + \tan(0{,}3c_i) + \sin(2{,}3c_i + \cos(0{,}9 + 0{,}4c_i^2))) - 29{,}2\cos(4{,}4c_i^2) \quad (1)$$

While equation (1) provides a valuable iterator (subject to normalization so that $c(i) \in (0,1] \; \forall i$ and post-processing based on the tests described in Fig. 1), it is not provable that this equation is structurally stable, i.e. that a cryptographically strong cipher is guaranteed for any floating point value of c_0 between 0 and 1, say, irrespective of the precision of c_0. This is important because c_0 (which seeds and thereby initiates the cipher stream) could, for example, be generated by a Hash function from a low bit private key and possibly fail at some point in the future for lack of structural stability. However, this is in keeping with many other PRNGs especially those generated by nonlinear iterations for which the structural stability problem remains an issue.

In this way, the proposed approach pays no attention to the algorithmic complexity of the iterator which is one of the main problems in the application of chaos to cryptography. Neither does it consider the structural stability of the iterator or its algorithmic complexity. However, it does provide a practical solution to the problem of developing a large database of PRNG for the application of personalizing encryption algorithms for strictly 'one-to-one' communications or 'one-to-Cloud' (encrypted) data storage. By using evolutionary computing systems such as Eureqa seeded with noise, it is possible to generate a nonlinear function with appropriate control parameters. Using this function in an iterative form with an additional transformation say, and, a partition function, a PRNG suitable for encrypting data can be constructed. The combined effect is that of a hard-core predicate.

III. Cryptography using artificial neural networks

Artificial Neural Networks (ANN) as well as Evolutionary Computation are a subfield of Artificial Intelligence and involve various architectures of neural networks and rules of training. ANNs were inspired by biological neural networks (the central nervous systems, in particular, the brain) and can be implemented in various complex control engineering problems such as function approximation pattern recognition, data mining, prediction, machine learning and so on.

In this work we propose the encryption technique that is based on ability of an ANN to simulate a high entropy input with the aim of transforming the result into a low entropy output. However, this process can be reversed to generate a high entropy output from a low entropy input. In this sense, an ANN can be used to generate a cipher by simulating natural noise once it has been trained to do so. To use an ANN in this way, the cryptographer requires knowledge of the ANN algorithm and the weights that have been generated through the iterative training process (i.e. the input of the noise sources used to generate the weights).

We use a Radial Basis Function (RBF) network. This neural network uses the radial basis function as an activation function and provides optimal performance in the function approximation tasks. The structure of the classical RBF neural network contains three layers (Fig. 3).

The first layer of such network consists of linear neurons and just feeds the values to each neuron in the hidden layer. The second layer is a hidden layer and performs a nonlinear transformation of the input space into the hidden space according to the radial basis function

$$F(n) = e^{-n^2}. \qquad (2)$$

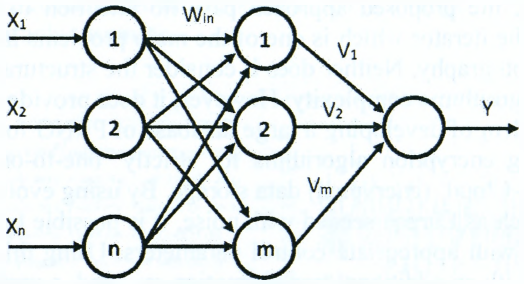

Fig. 3. RBF neural network

Each neuron in the radial basis layer will output a value according to how close the input vector is to each neuron's weight vector.

The radial basis function has a maximum of '*1*' when its input is '*0*'. Thus a radial basis neuron from hidden layer generates '*1*' whenever the input vector is identical to its weight vector. In contrast, if the input vector differs from the weight vector of a radial basis neuron it produces '*0*' (in fact, a neuron produces the values that are close to '*1*' and '*0,5*' or lower). The hidden neurons pass their values to the output layer. The output layer consists of linear neurons and represents the results of calculation. With enough neurons in the hidden layer, a Radial Basis Network can fit any function with any desired accuracy.

The standard training process of the RBF can be described as follows:

1) initially the hidden radial basis layer has no neurons;
2) the train vector is inputted to the network;
3) the input vector with the greatest error is found;
4) a hidden radial basis neuron is added with weights equal to that vector;
5) the output linear layer weights are redesigned to minimize error;
6) the steps 2-5 are repeated until the network's mean squared error falls below goal.

The proposed technique for cipher creation that is based on ANN can be represent as the next algorithm:

1) receiving a stochastic data stream (atmospheric noise, radioactive decay etc.);
2) creation and training, using stochastic data, an ANN;
3) checking an trained ANN to cryptographic strength;
4) using an ANN for strong cypher generation in personal cryptography.

Fig. 4 represents in schematic form the described algorithm for strong cipher generation.

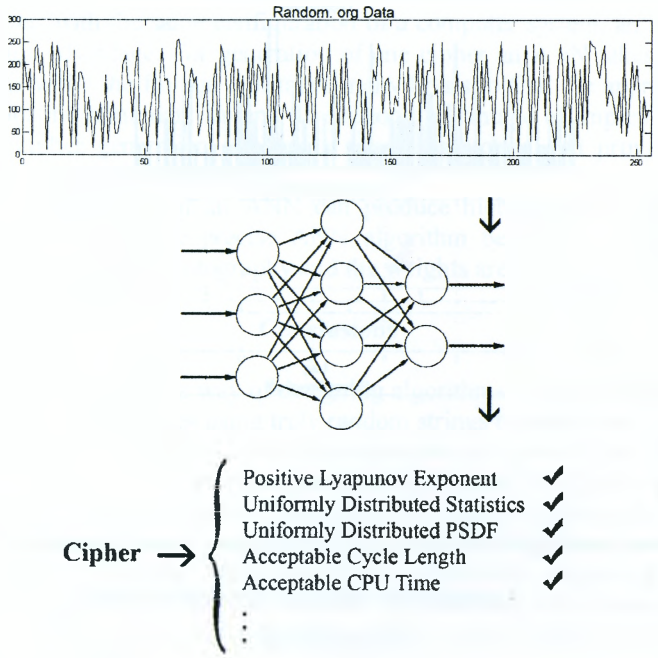

Fig. 4. ANN-based technique for cipher generation

IV. Experimental Results

The non-linear iteration function given by equation (1) is the result of *Eureqa* undertaking over 100 iterations (using 255 noise samples randomly selected from the data bases available at RANDOM.ORG) to evolve the result, taking approximately 27 hours using an Intel Core i3 1,7x2 GHz CPU to do so. Fig. 5 demonstrates some cryptographic strength characteristics of the received equation: uniform distribution, power spectrum and autocorrelation function.

In comparison with the Evolutionary Computation approach an ANN provides analogous results in cryptography strength (uniformly distributed statistics, uniformly distributed power spectrum, positive Lyapunov exponent, information entropy etc.). Fig. 6 compares the results from from the noise from RANDOM.ORG to the noise from the Eureqa output and the ANN.

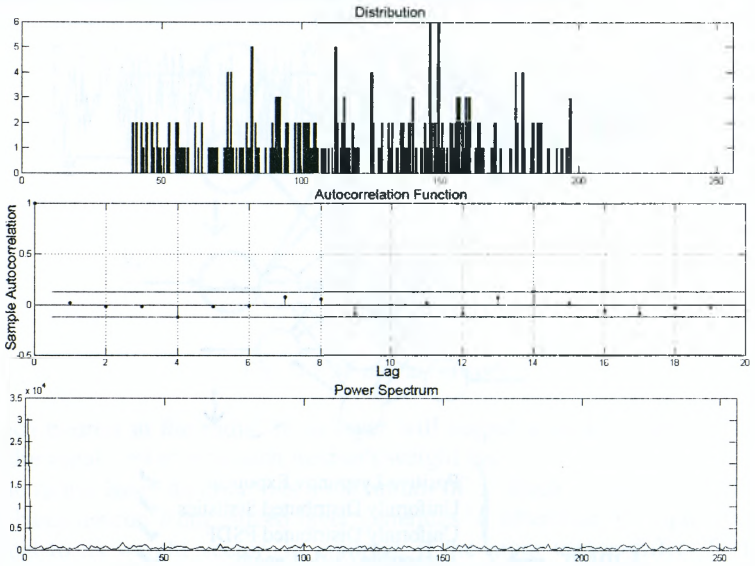

Fig. 5. Cryptographic strength of the iterated function

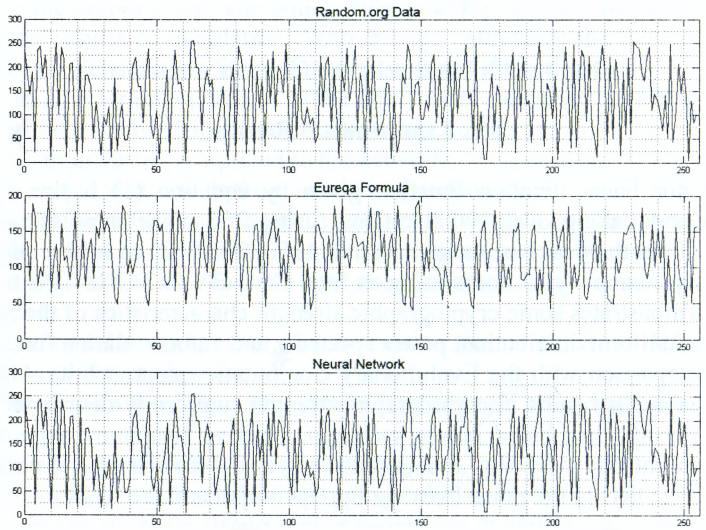

Fig. 6. Comparison of the simulation of true noise: original noise (above), *Eureqa* equation (middle) and ANN (below)

However, with the same configuration of a computer system, an ANN provides a time advantage. For generation of one cipher an ANN takes less than one minute while the Eureqa tool takes many hours to find the approximation equation for the same data stream. The second advantage of implementing an ANN for cipher generation lies in its parallel structure and provides high-distributed computational abilities.

This demonstrates that an ANN can produce highly nonlinear behavior. Given this statement, the precise ANN algorithm becomes analogous to a PRNG in conventional cryptography and the weights are equivalent to the key.

Conclusions

This paper introduces a way of designing algorithms for generating pseudo-random (chaotic) sequences using truly random strings to evolve an iterator that is taken to be an approximation to these sequences. It does provide a practical solution to the problem of developing a large database of PRNGs for the application of personalizing encryption algorithms for strictly 'one-to-one' communications or 'one-to-Cloud' (encrypted) data storage. With these provisos, the work reported in this paper demonstrates that evolutionary computing and artificial neural networks provide the potential for generating an unlimited number of ciphers which can be personalized for users to secure their 'Data on the Cloud', for example.

References

1. Blackledge J., Bezobrazov S., Tobin P., Zamora F. Cryptography using Evolutionary Computing // IET ISSC13, LYIT Letterkenny. 2013.

2. Kinzel W., Kanter I. Neural Cryptography // TH2002 Supplement. 2003. Vol. 4. P. 147-153.

3. Klein E., Mislovaty R., Kantor I., Ruttor A., Kinzel W. Synchonization of neural networks by mutual learning and its aoolication to cryptography // Advances in Neural Information Processing Systems, NIPS. 2004.

4. Jogdand R., Bisalapur S. Design of an efficient neural key generation // International Journal of Artificial Intelligence and Applications. 2011. Vol. 2. № 1. P. 60-69.

5. Yu W., Cao J. Cryptography based on delayed chaotic neural networks // Physics Letters A. 2006. Vol. 356. (4) Elsevier. P. 333-338.

6. Suryawanshi S., Nawgaje D. A triple-key Chaotic neural network for cryptography in image processing // International Journal of Engineering Sciences & Emerging Technologies. 2012. Vol. 2. Issue. 1. P. 46-50.

7. Fadil T., Yaakob S., Ahmad B., Yahya A. Encryption of mpeg-2 video signal based on chaotic neural network // Journal of Engineering and Technology. 2012. Vol. 3. P. 35-42.