Scientific Journal of Impact Factor (SJIF): 4.72
e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 4, Issue 12, December -2017

Implementation of cryptography using Jordan network

Charu Gupta[1]

ME-Digital Communication, Department of Electronics & Communication Engineering, MBM Engineering College, JNV University, Jodhpur-342011

Abstract: *A Neural Network is a machine that is designed to model the way in which the brain performs a task or function of interest. It has the ability to perform complex computations with ease. The objective of this project was to investigate the use of ANNs in the field of Cryptography. Using a Jordan (Recurrent network), trained by back-propagation algorithm, a finite state sequential machine was successfully implemented. The sequential machine thus obtained was used for encryption with the starting key being the key for decryption process.*

Keywords *- Cryptography, Artificial neural network*

I. INTRODUCTION

I.I NEURAL NETWORKS

The term neural network was traditionally used to refer to a network or circuit of biological neurons. The modern usage of the term often refers to artificial neural networks, which are composed of artificial neurons or nodes. Thus the term has two distinct usages:

- Biological neural networks are made up of real biological neurons that are connected or functionally related in a nervous system. In the field of neuroscience, they are often identified as groups of neurons that perform a specific physiological function in laboratory analysis.
- Artificial neural networks are composed of interconnecting artificial neurons (programming constructs that mimic the properties of biological neurons). Artificial neural networks may either be used to gain an understanding of biological neural networks, or for solving artificial intelligence problems without necessarily creating a model of a real biological system.

II. CRYPTOGRAPHY

from Greek κρυπτός, "hidden, secret"; and γράφειν, graphein, "writing", or -λογία, -logia, "study", respectively) is the practice and study of techniques for secure communication in the presence of third parties (called adversaries). More generally, it is about constructing and analyzing protocols that overcome the influence of adversaries and which are related to various aspects in information security such as data confidentiality, data integrity, authentication, and non-repudiation. Modern cryptography intersects the disciplines of mathematics, computer science, and electrical engineering. Applications of cryptography include ATM cards, computer passwords, and electronic commerce.

- **Classic cryptography**

The main classical cipher types are transposition ciphers, which rearrange the order of letters in a message (e.g., 'hello world' becomes 'ehlol owrdl' in a trivially simple rearrangement scheme), and substitution ciphers, which systematically replace letters or groups of letters with other letters or groups of letters (e.g., 'fly at once' becomes 'gmz bu podf' by replacing each letter with the one following it in the Latin alphabet). Simple versions of either have never offered much confidentiality from enterprising opponents. Cipher texts produced by a classical cipher (and some modern ciphers) always reveal statistical information about the plaintext, which can often be used to break them.

- **Modern cryptography**

The modern field of cryptography can be divided into several areas of study. Two of them are described here:

- **Symmetric-key cryptography**

Symmetric-key cryptography refers to encryption methods in which both the sender and receiver share the same key (or, less commonly, in which their keys are different, but related in an easily computable way). Symmetric key ciphers are implemented as either block ciphers or stream ciphers. A block cipher enciphers input in blocks of plaintext as opposed to individual characters, the input form used by a stream cipher.

- **Public-key cryptography**

A public key system is so constructed that calculation of one key (the 'private key') is computationally infeasible from the other (the 'public key'), even though they are necessarily related. Instead, both keys are generated secretly, as an interrelated pair. In public-key cryptosystems, the public key may be freely distributed, while its paired private key must remain secret. In a public-key encryption system, the public key is used for encryption, while the private or secret key is used for decryption.

III. ARTIFICIAL NEURAL NETWORKS

A neural network is a massively parallel distributed processor made up of simple processing units, which has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects:
1. Knowledge is acquired by the network from its environment through a learning process.
2. Interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge.
The procedure used to perform the learning process is called a learning algorithm, the function of which is to modify the synaptic weights of the network in an orderly fashion to attain a desired design objective. Neural networks are also referred to in literature as neuro computers, connectionist networks, and parallel distributed processors.

3.1 Model of a neuron

A neuron is an information-processing unit that is fundamental to the operation of a neural network. The block diagram in Fig. shows the model of a neuron, which forms the basis for designing (artificial) neural networks. Here we identify three basic elements of the neuronal model:

- A set of synapses or connecting links, each of which is characterized by a weight or strength of its own. Specifically, a signal X_j a t the input of synapse j connected to neuron k is multiplied by the synaptic weight w_{kj}" It is important to make a note of the manner in which the subscripts of the synaptic weight w_{kj} are written. The first subscript refers to the neuron in question and the second subscript refers to the input end of the synapse to which the weight refers.
- An adder for summing the input signals, weighted by the respective synapses of the neuron; the operations described here constitutes a linear combiner.

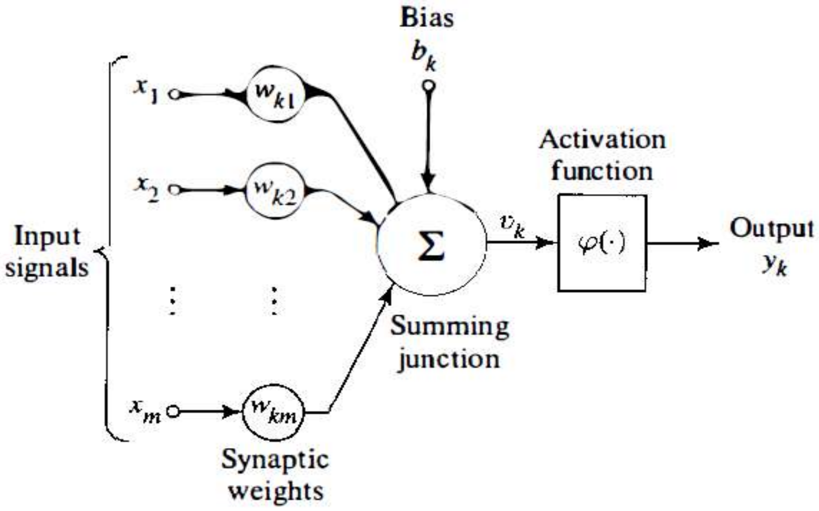

Figure-1: Nonlinear model of a neuron.

- An activation function for limiting the amplitude of the output of a neuron. The activation function is also referred to as a squashing function in that it squashes (limits) the permissible amplitude range of the output signal to some finite value.

3.2 Network architecture:
The manner in which the neurons of a neural network are structured is intimately linked with the learning algorithm used to train the network. We may therefore speak of learning algorithms (rules) used in the design of neural networks as being structured.

❖ **Recurrent Networks:**

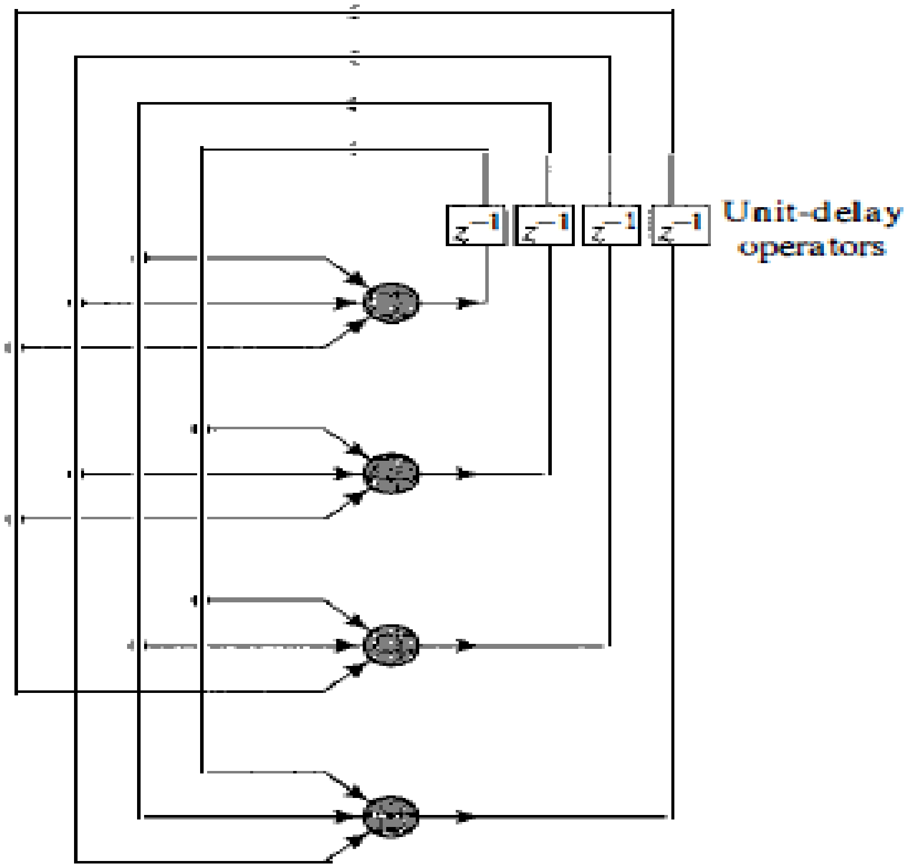

Figure-2: Recurrent network with no self feedback loops and no hidden neurons.

Recurrent neural network distinguishes itself from a feed forward neural network in that it has at least one feedback loop. The feedback loops involve the use of particular branches composed of unit-delay elements (denoted by Z^{-1}), which result in a nonlinear dynamical behavior, assuming that the neural network contains nonlinear units.

IV. IMPLEMENTATION

4.1 Sequential Machine

A ``sequential machine" is a device in which the output depends in some systematic way on variables other than the immediate inputs to the device. These ``other variables" are called the *state variables* for the machine, and depend on the *history or state* of the machine. For example, in a counter, the state variables are the values stored in the flip flops. The essence of a state table can be captured in a state diagram. A state diagram is a graph with labeled nodes and arcs; the nodes are the states, and the arcs are the possible transitions between states.

In this project we have used the fact that the output of the sequential machine depends on the state of the machine as well as the input given to the sequential machine. Therefore we have used a Jordan network in which a few outputs are used as inputs, these outputs denote the states. A multilayered neural network is designed on this basis whish has a log sigmoid in the output layer as a transfer function.

The network has 4 input layers, hidden layers and 4 output layers. The size of the input layer depends on the number of inputs and the number of outputs being used to denote the states. The learning algorithm used for this network is back propagation algorithm and the transfer function in the hidden layer is a sigmoid function. For implementation of sequential machine a serial adder and a sequential decoder is used.

4.2 Sequential Machine Implementation

A finite state sequential machine was implemented using a Jordan network is used. In the Jordan network, the activation values of the output units are fed back into the input layer through a set of extra input units called the state units. There are as many state units as there are output units in the network. The connections between the output and state units have a fixed weight of +1 and learning takes place only in the connections between input and hidden units as well as hidden and output units. Thus all the learning rules derived for the multi-layer perceptron can be used to train this network.

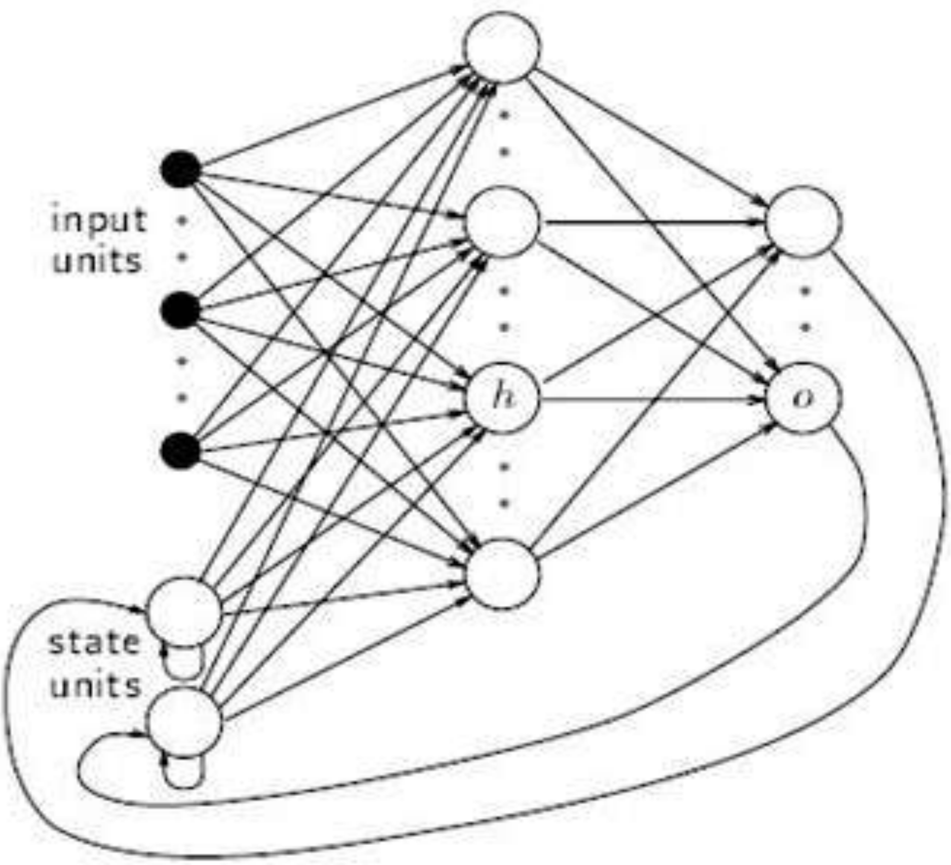

Figure-3: Jordan Network

To train the Jordan network back propagation algorithm was used. Back propagation algorithm is discussed here below.

❖ **Back propagation network:**

The feed forward back propagation network is a very popular model in neural networks. In multi layer feed forward networks, the processing elements are arranged in layers and only the elements in the adjacent layers are connected. It has minimum three layers of elements (i) input layer, (ii) the middle or the hidden layer and (iii) the output layer.

The information propagation is only in forward direction and there are no feedback loops. Even though it does not have feedback loops errors are back propagated during training. The name back propagation arrives from the fact that the computations are passed from the input to output layer following which errors are propagated back in the other direction to change the weights to obtain the better performance.

The training algorithm of back propagation is in four stages:

- ❖ Initialization of weights
- ❖ Feed forward
- ❖ Back propagation of errors
- ❖ Updation of the weights and biases.

Step by step procedure of training algorithm is given below:

1. Initialize the weight to small random values
2. While stopping condition is false, perform steps 3-10
3. For each training pair do steps 4-9

Feed-forward:

4. Each input unit receives the input signal x_i and transmits this signals to all units in the layer above it i.e. hidden units
5. Each hidden unit sums its weighted input signals and applies the input signal and sends the signal to all units in the layer above i.e. output units.

6. Each output unit sums its weighted input signals and applies its activation function to calculate the output signals

   Back-propagation of errors:

7. Each output unit receives a target pattern corresponding to an input pattern, error information term is calculated.
8. Each hidden unit sums its delta inputs from units in the layer above and the error information term is calculated.

   Updation of weights and biases:

9. Each input and output unit updates its bias and weights.
10. Test the stopping condition. The stopping condition may be minimization of errors, no. of epochs etc.

For the implementation of the sequential machine the state table is used as input and the outputs as well as next states are used as the combined output for the Jordan network. Depending upon the size of the dataset the size of the hidden layer is changed as the complexity of the sequential machine increases.

4.3 Implementation

The application procedure of back propagation is a shown below:

1. initialize weights(from training algorithm)
2. For each input vector perform steps 3-5
3. For i=1,..n, set activation of input unit
4. For j=1,..p, calculate the sum of its weighted input signals
5. For k=1,..m, calculate the sum of weighted input units at output unit and apply the activation function.

Using this algorithm two types of forward neural network architectures were used. The first type was a feed-forward neural network that was fully connected. The second type partitions the problem into smaller independent subtasks and a neural network is designed to implement each subtask.

The other features added to enhance the learning were:-

1. Random weights were used to help the network start at different places for the same network architecture. The weights were usually between 0.01 and .99**.**

2. A digital error was added such that if the output was between 0.0 and 0.2 it was considered a low and if it was between 0.7 and 1.0 it was a high. Thus a noise margin was added between 0.2 and 0.4, as with most digital circuits. This allowed the program to stop early, instead of finding a minimum error

V. WORKING

A sequential machine using a *Jordan* network has been implemented using the back-propagation algorithm. For use of sequential machine for encryption and decryption, a state diagram is drawn and a state table is obtained. Using the state table, a training set is generated. The input set includes all the possible inputs and states possible whereas the output consists of the encrypted/decrypted output and the next state.

The reason for using sequential machine for implementation is that the output and input can have any type of relationship and the output depends on the starting state. The starting state is used as a key for encryption and decryption. If the starting state is not known, it is not possible to retrieve the data by decryption even if the state table or the working of the sequential state is known. For training of the neural network, any type of sequential machine can be used with the key showing the complexity or the level of security obtained.

The working of a simple sequential machine for encryption is explained here. The sequential machine has *n* states and the input is an *m* bit input. The output state can be anything according to the user. The encrypted data will depend upon the present state of the machine. Therefore, the starting state along with the input will generate an output and then the state will change according to the state table. In case of two states, if it not known whether the state is"0" or "1", the data cannot be decrypted and hence the starting state acts as a key.

VI. RESULT& CONCLUSION

Cryptography using ANN based Sequential machine

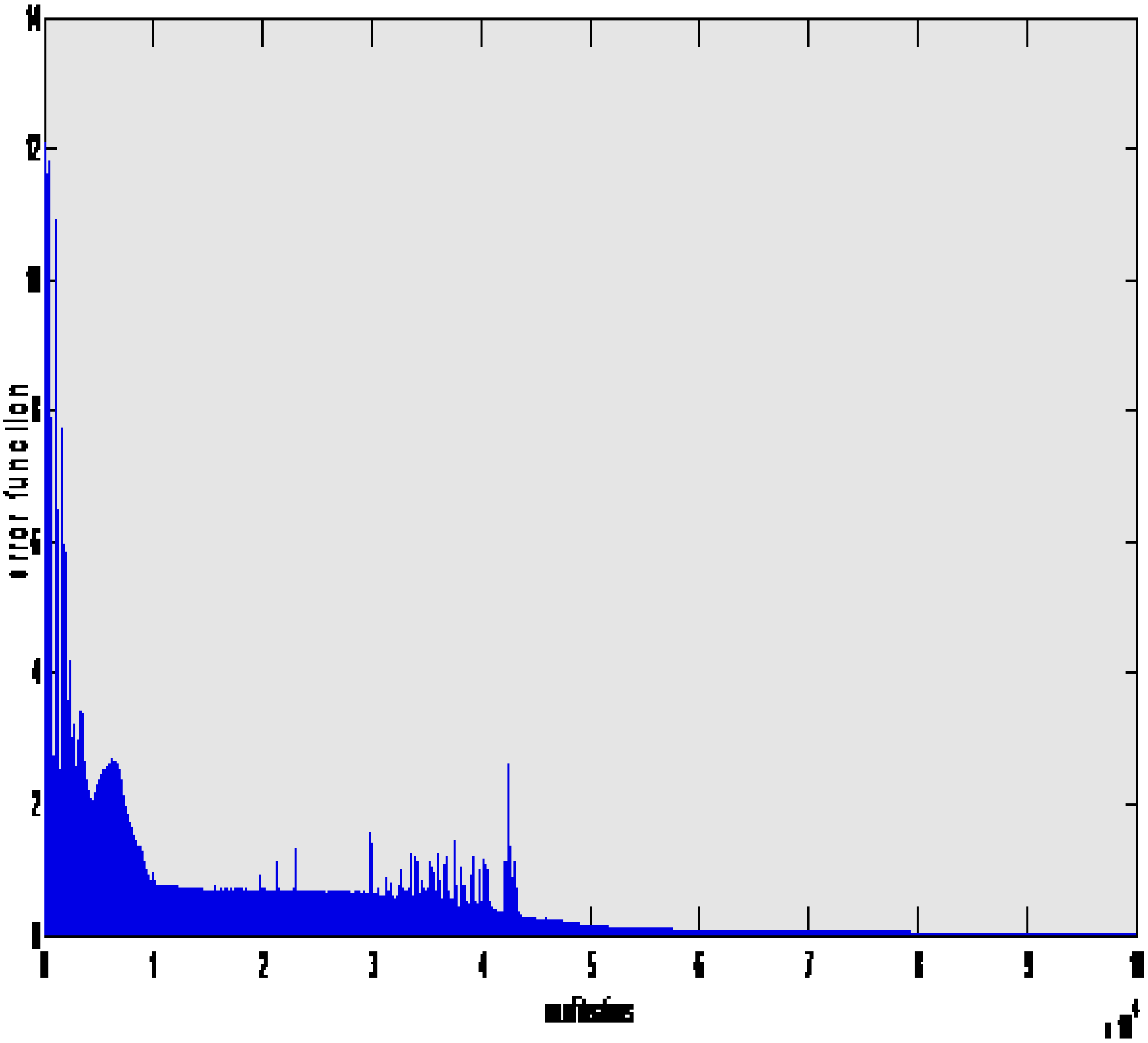

Figure-4: The plotted graph of the error function after the learning process

Figure shows the plot of the error function against the number of iterations. A 4-bit encryption machine was successfully built using an ANN based sequential machine. The input is the 4-bit data to be encrypted. Letters" A" to "P" were used to represent all the possible 4-bit inputs. If the state is 1, the letter is shifted by 2. During this operation, the state is automatically switched.

VII. CONCLUSION

Artificial Neural Networks is a simple yet powerful technique which has the ability to emulate highly complex computational machines. In this project, we have used this technique to built simple combinational logic and sequential machine using back-propagation algorithm. Data security is a prime concern in data communication systems. The use of ANN in the field of Cryptography is investigated. A sequential machine based method for encryption of data is designed .Better results can be achieved by improvement of code or by use of better training algorithms. Thus, Artificial Neural Network can be used as a new method of encryption and decryption of data.

REFERENCES

[1] Schneier B. Applied Cryptography—Protocols, Algorithms, and Source Code in C. 2nd Ed. New York, USA: John Wiley & Sons, Inc, 1996.

[2] Hu Y P, Zhang Y Q, Xiao G Z. Symmetric Key Cryptography. Beijing: China Machine Press, 2002. (in Chinese)

[3] M. E. Smid and D. K. Branstad, “The Data Encryption Standard: Past and Future,” Proceedings of The IEEE, vol. 76, no. 5, pp. 550-559, 1988.

[4] C. Boyd, “Modem Data Encryption,” Electronics & Communication Journal, pp. 271-278, Oct. 1993. 131 N. Bourbakis and C. Alexopoulos, “Picture Data Encryption Using SC4N Pattern,” Pattern Recognition, vol. 25, no. 6, pp. 567-581, 1992.

[5] J. C. Yen and J. I. GUO, “A New Image Encryption Algorithm and Its VLSI Architecture,” 1999 IEEE Workshop on Signal Procs. Systems, Grand Hotel, Taipei, Taiwan, Oct. 18-22, pp. 430-437, 1999.

[6] C. J. Kuo and M. S. Chen, “A New Signal Encryption Technique and Its Attack Study,” IEEE International Conference on Security Technology, Taipei, Taiwan

[7] C. W. Wu and N. F. Rulkov, “Studying chaos via 1-D maps - A tutorial,” IEEE Trans. on Circuits and Systems I-Fundamental Theory and Applications, vol. 40, no. 10, pp. 707-721, 1993.

[8] T. S. Parker and L. 0. Chua, “Chaos - A tutorial for engineers,” IEEE Proc., vol. 75, pp. 982-1008, 1987.

[9] Haykin, Simon. Neural Networks, A Comprehensive Foundation. MacMillin College Publishing CO, New York. 1994.