CRYPTOGRAPHY USING NEURAL NETWORK

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS OF THE DEGREE OF INTEGRATED M.SC.

IN

MATHEMATICS

SUBMITTED BY:

PRIYA RAJ (410MA5042)

Under the supervision of

PROF. S. CHAKRAVERTY

Department of Mathematics

National Institute of Technology Rourkela

Odisha, India

2014 - 2015.

NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA

DECLARATION

I, the undersigned, declare that the work contained in this thesis entitled **Cryptography Using Neural Network**, in partial fulfilment of the requirement for the award of the degree of Master of Science, submitted in the Department of Mathematics, National Institute of Technology, Rourkela, is entirely my own work and has not previously in its entirety or part been submitted at any university for a degree, and that all the sources I have used or quoted have been indicated and appropriately acknowledged by complete references.

PRIYA RAJ

May 2015

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Dr. S. CHAKRAVERTY
Professor, Department of Mathematics
National Institute of Technology
Rourkela 769008
Odisha, India

ACKNOWLEDGEMENTS

I would like to express my profound gratitude and regards to my project guide, **Prof. S. Chakraverty**, for his exemplary guidance and monitoring. The help, motivation, ideas and blessings have been the biggest milestone throughout the thesis.

I would also extend my sincere thanks to all the seniors and friends, for providing valuable information and support. I am grateful to them for their cooperation during the assignment of this project.

Lastly, I thank my parents, brothers, sisters and all close ones for their constant support without whom, this project would not be possible.

PRIYA RAJ

(410MA5042)

ABSTRACT

The project is aimed to implement artificial neural network method in cryptography. Cryptography is a technique to encrypt simple message into cipher text for secure transmission over any channel. The training of the network has been done using the input output set generated by the cryptosystem, which include shift and RSA ciphers. The training patterns are observed and analysed by varying the parameters of Levenberg Marqaurdt method and the number of neurons in the hidden layer. Using the converged network, the model is first trained, and one may obtain the desired result with required accuracy. In this respect, simulations are shown to validate the proposed model. As such, the investigation gives an idea to use the trained neural network for encryption and decryption in cryptography.

CONTENTS

1. INTRODUCTION

1.1 MOTIVATION

The rising growth of technology in the communication sector has always created an increased demand of secure channel for the transmission of data. Cryptography has always served as a successful means to build such channels. These channels find numerous applications, as in mobile phones, internet, digital watermarking etc. and also for secure transmission protocols. There are several encryption-decryption techniques which may be improvised for a secure transfer of data, like public and private key cryptosystems. However, the risk of attack by an intruder is still very high. A novel approach has been adopted here by applying neural network to cryptography. As such, in case of shift ciphers, the transfer of message would not be safe if the key is public. So sending it over a neural network, where in, keeping the key private, the transfer becomes secure. Also, in the case of RSA cryptosystem, where two keys are involved which may be easily retrieved by solving the factor problem, the implementation of neural network serves as an efficient method.

1.2 LITERATURE REVIEW

Recently many investigations have been carried out by various researchers in Cryptography using Neural Networks. As such, few literatures are discussed below:

Zurada [1] has discussed artificial neural network with respect to different learning methods and network properties. Supervised and unsupervised learning has been elaborated in detail with the help of network architecture. The usage of parameters for training is illustrated. The minimization of error functions in multilayer feedforward networks has been explained using the backpropagation algorithm. Koshy [2] has emphasized on the problem solving techniques and their applications. With the help of Fermat's Little Theorem, we may find the least residues. Different cryptosystems and their algorithms illustrate the encryption-decryption methods. Depending on the key usage, the cryptosystem has been subdivided and explained in detail.

Kanter and Kinzel [3] presented the theory of neural networks and cryptography based on a new method by the synchronisation of neural networks for the secure transmission of secret messages. The encryption based on synchronisation of neural networks by mutual learning has been used which involves construction of two neural networks, where the synaptic weights are synchronised by the exchange and learning of mutual outputs for the given inputs. The network of one may be trained by the output of the other. In case, the outputs do not comply with each other, the weights are adjusted and updated using the Hebbian learning rule. The synchronisation of those two networks occurs in a definite time which tends to decrease with the increasing size of inputs. The author focuses on accelerating the synchronisation process from hundred of time steps to the least possible value and maintaining the security of the network at the same time.

Laskari et al. [4] studied the performance of artificial neural networks on problems related to cryptography based on different types of cryptosystems which are computationally intractable. They have illustrated various methods to address such problems using artificial neural networks and obtain better solutions. The efficiency of a cryptosystem may be judged by its computational intractability. This paper deals with the study of three problems, namely, discrete logarithmic problem, Diffie-Hellman key exchange protocol problem and factorisation problem. The artificial neural networks have been used to train a feedforward network for the plain and ciphered text using backpropagation technique. It aims to assign proper weights to the network in order to minimise the difference between the actual and desired output. The normalised data is fed to the network and then its performance is evaluated. The percentage of trained data and its near measure is evaluated.

Meletiou et al. [5] has discussed RSA cryptography and its susceptibility to various attacks. The author has used the artificial neural network for the computation of the euler totient function in the determination of deciphering key and hence, RSA cryptography may be easily forged. The multilayer feedforward network is used for training the data set with backpropagation of errors. Learning rate of network may not be ideal but is asymptotically approachable. The network performance is measured by using the complete and near measure of errors. Also the result has been verified for prime numbers ranging from high to low values.

1.3 GAPS

The work illustrated in the papers discussed in literature review deal with cryptography and its security. The security of the cryptosystems has been enhanced by adopting different methodologies. Accuracy of the training pattern considered in [3] is a function of time steps required to minimize time and accelerate synchronisation of the feedforward networks. This methodology may take more time and yield a tiresome process. Similarly, in [4] the training pattern has been discussed for different values of prime numbers p and q, and the variations as well as the errors have been closely observed. However, in doing so, the network topology becomes large. Also, the appropriate usage of training parameters has been ignored. In [5] the training of the neural network involves product of two prime numbers $N(= p \times q)$ and euler totient function. Using the training methods, the network has been adapted to obtain the euler totient function from given N. The training patterns and errors in [5] may be observed by varying the values of p and q. The network architecture which is varied by changing the number of hidden layers and hidden neurons, results in a complicated topology. In view of the above, time and network topology are two parameters which dictates the training of the titled problem. As such, efficient model should be developed along with network topology to handle the problem.

1.4 PROBLEM STATEMENT

The primary objective of this project is to implement encryption and decryption of shift and RSA cryptosystems, in artificial neural network. The network construction depends solely on the parameters used in the training algorithm and the number of hidden neurons. The aim is to obtain an efficient training pattern with the help of proper algorithm and parameters, such that the errors are minimised with better accuracy.

2 PRELIMINARIES

2.1 BASICS OF NEURAL NETWORK

2.1.1 Artificial Neural Network (ANN)

Artificial Neural Networks are computational models that have been inspired by human's central nervous systems [1]. They may be used to estimate or approximate functions that depend on the inputs and outputs. In an ANN, simple artificial nodes, known as neurons are connected together to form a network similar to the biological neural network. It comprises of massively parallel distributed processors made up of processing units which have the natural tendency to store the trained knowledge.

2.1.2 Certain advantages of using Artificial Neural Network (ANN)

The few advantages of using an artificial neural network may be classified as:

- Non-linearity – ANNs are capable of approximating any non linear function accuracy.
- Input-output Mapping – In an ANN, corresponding target values may be matched easily using learning phases in a way similar to the human brain.
- Adaptivity – The ANNs are highly adaptive in nature, and may even be adapted to identify the face and voice, as in the case of digital signatures and face/voice recognition.

2.1.3 Models of an Artificial Neural Network

An artificial neural network model includes the following components:

- Input layer – It consists of all the input data that has been supplied to the network.
- Hidden layer – It consists of all the passive inputs that have been supplied by the preceding layers.
- Output layers – It contains the outputs of the neural network.

- Weights and biases – They have the effect of increasing or lowering the net input of the activation function depending on whether it is positive or negative respectively.
- Epochs – The number iterations in a neural network.
- Activation functions – It is an abstraction that represents the rate of firing in the cell. It is used for transforming the input signal of a neuron into the output signal. Some of the activation functions may be defined as follows:

➢ Threshold Function – Threshold function is

$$\phi(v_k) = \begin{cases} 1, if \ v_k \ \geq 0 \\ 0, if \ v_k \ < 0 \end{cases}$$

➢ Symmetric Hard Limit Function – Symmetric hard limit function is defined as

$$\phi(v_k) = \begin{cases} 1, \ if \ v_k \ \geq 0 \\ -1, if \ v_k \ < 0 \end{cases}$$

➢ Piece wise Linear Function – Piece wise linear function may be stated as

$$\phi(v_k) = \begin{cases} 1, \ if \ v_k \ \geq 1/2 \\ -1, if \ v_k \ < 1/2 \end{cases}$$

➢ Pure Linear Function – Pure Linear Function may be written as

$$\phi(v_k) = v_k(n)$$

➢ Sigmoid activation function – Sigmoid activation function may be represented as

$$\phi(v_k) = \frac{1}{1+e^{(-v_k(n))}}$$

2.1.4 Learning Methods

There are three learning paradigms [6] which are:

1) Supervised Learning : It is a method in which a given set of data is used for training the neural network. The training function is computed by minimising the mean square error. Regression and pattern recognition come under this category.

2) Unsupervised Learning : In this method, some example pairs may be given along with the cost function, and the hidden structures are identified. It generally includes estimation problems and statistical estimations.

3) Reinforcement Learning : In this method, the data set is generated by the person's interaction with the environment. Then the observation is noted down and used for the training purpose. It includes control problems and games.

2.1.5 Feedforward Network

In a feedforward network, perceptrons may be arranged in layers. The input is taken by the first layer and output is produced by the last. Also, the middle layers are not connected to the external world and referred as the hidden layers. Each perceptron in the first layer is connected to the other perceptron in the next layer. Hence, information is always fed forward from one layer to the next. That is why it is referred as feedforward network.

2.2 BASICS OF CRYPTOGRAPHY

Cryptography is the science of hiding messages for confidential communications [2]. It is used for the secure transmission of important data from one person to the other without being forged by an intruder. It finds application in areas like electronic banking, security maintenance etc. Certain terms related to cryptography are :

- Plain text – The original message to be transferred to the other person.
- Cipher text – The secret version of the plain text which is used for transferring.
- Key – A secret code which is used to lock or unlock the plain text and the cipher text respectively.
- Encryption – The process of converting plain text to cipher text.
- Decryption – The process of converting the ciper text to plain text.

CLASSIFICATION ON THE BASIS OF KEY SELECTION

1) Symmetric key cryptosystem:

Symmetric key cryptosystem is a private key cryptosystem. In this system, the same key is used for the encryption of plain text to cipher text and decryption of cipher text to plain text. The key remains the same for both the cases.
For example: Shift ciphers.

2) Asymmetric key cryptosystem:

Asymmetric key cryptosystem is a type of public key cryptosystem. In this system, different keys are used for encryption and decryption. An enciphering key is used for encryption and deciphering key for the decryption.
For example: RSA cryptosystem.

3 DEVELOPED MODELS AND METHODS

In order to elaborate the functioning of an artificial neural network, initially a function is trained and then the results are analysed. For instance a function $y = x^2$ is taken for generation of a set of 1000 input-output data. Then the generated data is used to train a neural network. But before proceeding to the training part, a learning method is illustrated.

3.1 LEARNING ALGORITHM:

The learning algorithm used here is the Backpropagation method [1] in feedforward network architecture. The algorithm may be given as:

STEP 1: Initialise the weights w and v , and a learning parameter (η). For the problem considered in this project, we take $\eta = 1$. Choose maximum error $E_{\max}$ and take initial error $E = 0$.

STEP 2: Taking sigmoid function as the activation function, we train the network. Hence, $f(t) = \frac{1}{1+e^{-t}}$ and the initial input is taken as z. Therefore, the output for the first layer is $y \leftarrow f(v_j^t z)$, for $j = 1, 2, 3, \ldots, m$. Output of the hidden layer will be $o_k \leftarrow f(w_k^t y)$ for $k = 1, 2, 3, \ldots, n$, where v_j is j'th row of v for $j = 1, 2, 3, \ldots, m$ and w_k is k'th row of w for $k = 1, 2, 3, \ldots, n$.

STEP 3: The loss function, that is, the error value is calculated for each output. $E \leftarrow \frac{1}{2}(d_k - o_k)^2 + E$ where d_k is the desired output for $k = 1, 2, 3, \ldots, n$.

STEP 4: The error signal terms of the output layer in this step are

$$\delta_0 = [(d_k - o_k)(1 - o_k)o_k]$$
$$\delta_{y_j} = [(1 - y_j)y_j]\delta_0 w_{kj}$$

STEP 5: The weights of the output layer are adjusted as $w_{kj} \leftarrow w_{kj} + \eta\delta_{ok}v_j$.

STEP 6: The weights of the hidden layer are adjusted as $v_{ji} \leftarrow v_{ji} + \eta \delta_{yj} z_i$ for $j = 1, 2, 3, \ldots, m$ and $i = 1, 2, 3, \ldots, n$.

STEP 7: If $E < E_{\max}$, terminate the training session, otherwise go to step 2 with $E \leftarrow 0$ and initiate the new training.

3.2 LEVENBERG MARQUARDT BACKPROPAGATION METHOD

The Levenberg Marquardt method [7] may be used in conjunction with the backpropagation method to train a neural network. It has been designed to approach the second order training speed without computing the Hessian matrix in a way similar to that of quasi Newton methods. When the performance function is of the form of sum of squares, then we may approximate the Hessian matrix as $H = J^T J$ and the gradient as $g = J^T e$, where e is the vector of network errors and J is the Jacobian matrix containing the first derivatives of the network errors with respect to the weights and biases. The Jacobian matrix may be computed through a standard backpropagation technique.

The Levenberg Marquardt algorithm uses the approximation to obtain the Hessian matrix, from the following Newton's method:

$$x_{k+1} = x_k - [J^T J + \mu I]^{-1} J^T e.$$

1. When $\mu = 0$, the above equation approximates to Hessian matrix.
2. When the value of μ is large, the above equation becomes gradient descent with a small step size. Since Newton's method is faster and more accurate for minimum error, so the aim is to shift the method towards Newton's method as quickly as possible.

As such the value of μ is decreased after each successful step performance and increased only when the given step increases the performance function. Hence, the performance function has been minimised in each step.

3.3 TRAINING OF THE FUNCTION $y = x^2$

Let us construct a neural network consisting of a single input layer with one node having 1000 patterns, a hidden layer with 15 neurons and an output layer with one node having 1000 patterns. The trained network is tested with sample values to check the efficiency of training. The parameters chosen for the training is illustrated and the different test values are shown in Table 1.

Table 1 Testing for the function $y = x^2$ $(\mu = 0.01, \mu\_dec = 0.001, \mu\_inc = 10)$

Test Points	Output	Target output
2	4.79	4
10	102.87	100
24	575	576

Here, μ_dec is the μ decrease factor and μ_inc is the μ increase factor.

It may be seen from Table 1 that the test results are approximately equal to the target values. Hence, the neural network has been trained successfully.

3.4 TRAINING OF SHIFT CIPHERS

A shift cipher is a substitution cipher where we substitute each letter by another. The plain text may be encrypted by using the relation $C \equiv P + k \pmod{26}$ where C is the cipher text, P is the plain text and k is the shift factor, $(0 \le k \le 25)$. For training of the neural network, initially a sentence is selected. The following sentence has been used for training:

"SILENCE IS GOLDEN. THE MAN IS WALKING IN THE RAIN. THE DOGS ARE BARKING. THE BAKERY WAS CLOSED TILL NINE TO PROTEST AGAINST THE NEW TAX LAWS. THIS ENRAGED THE CUSTOMERS." (1)

Then the letters are assembled into blocks of two and the corresponding number to each letter is written. In this case, we use a shift factor $(k) = 2$ and the generated input-output data is given in Table 2. This data set is used to train a neural network taking input as P and target output as Normalised C.

TABLE 2 Data set to train the neural network for shift cipher

TEXT	P	C	NORMALISED C
SI	1808	2010	0.766
LE	1104	1306	0.4977
NE	1302	1504	0.5732
EI	0408	610	0.2325
SG	1806	2008	0.7652
OL	1411	1613	0.6147
DE	0304	506	0.1928
NT	1319	1521	0.5796
HE	0704	906	0.3453
MA	1200	1402	0.5343
NI	1308	1510	0.5755
SW	1822	2024	0.7713
AL	0011	213	0.0812
KI	1008	1210	0.4611
NG	1306	1508	0.5747
IN	0813	1015	0.3868
TH	1907	2109	0.8037
ER	0417	619	0.2359
AI	0008	210	0.08
NT	1319	1521	0.5796

HE	0704	906	0.3453
DO	0314	516	0.1966
GS	0618	820	0.3125
AR	0017	219	0.0835
EB	0401	603	0.2298
AR	0017	219	0.0835
KI	1008	1210	0.4611
NG	1306	1508	0.5747
TH	1507	1709	0.6513
EB	0401	603	0.2298
AK	0010	212	0.0808
ER	0417	619	0.2359
YW	2422	2624	1
AS	0018	220	0.0838
CL	0211	413	0.1574
OS	1418	1620	0.6174
ED	0403	605	0.2306
TI	1908	2110	0.8041
LL	1111	1313	0.5004
NI	1308	1510	0.5755
NE	1304	1506	0.5739
TO	1914	2116	0.8064
PR	1517	1719	0.6551
OT	1419	1621	0.6178
ES	0418	620	0.2363
TA	1900	2102	0.8011
GA	0600	802	0.3056
IN	0813	1015	0.3868
ST	1819	2021	0.7702
TH	1907	2109	0.8037
EN	0413	615	0.2344

EW	0422	624	0.2378
TA	1900	2102	0.8011
XL	2311	2513	0.9577
AW	0022	224	0.0854
SI	1819	2021	0.7702
HI	0708	910	0.3468
SE	1804	2006	0.7645
NR	1317	1519	0.5789
AG	0006	208	0.0793
ED	0403	605	0.2306
TH	1907	2109	0.8037
EC	0402	604	0.2302
US	2018	2220	0.846
TO	1914	2116	0.8064
ME	1204	1406	0.5358
RS	1718	1920	0.7317

The neural network may be trained using the plain text (P) and the normalised C given in Table 2 using MATLAB. The training starts as follows:

- Launch neural network toolbox, *nntool* in MATLAB.
- Import all the input (P) and target output value (Normalised C).
- Create a 2 layer feedforward network, with a known number of neurons in the hidden layer. Also, select a transfer function. For present problem, sigmoid function is selected.
- Simulate the test point values for networks with varying parameters. Number of hidden neurons is taken as 15. For tables 3 to 6, we use different values of training parameters and incorporate the trained ANN results.

TABLE 3 Values obtained from the trained network at different test points
($\mu = 0.001$, $\mu\_dec = 0.01$, $\mu\_inc = 10$)

Input	Target Value	Trained Value
1808	0.7660	0.7666
1204	0.5358	0.5361
2311	0.9577	0.9568
1718	0.7317	0.7319
0403	0.2306	0.2306
2018	0.8460	0.8349

TABLE 4 Values obtained from the trained network at different test points
($\mu = 0.001$, $\mu\_dec = 0.001$, $\mu\_inc = 10$)

Input	Target Value	Trained Value
1808	0.7660	0.7660
1204	0.5358	0.5358
2311	0.9577	0.9577
1718	0.7317	0.7317
0403	0.2306	0.2306
2018	0.8460	0.9050

TABLE 5 Values obtained from the trained network at different test points ($\mu = 0.01$, $\mu\_dec = 0.01$, $\mu\_inc = 10$)

Input	Target Value	Trained Value
1808	0.7660	0.7660
1204	0.5358	0.5358
2311	0.9577	0.9577
1718	0.7317	0.7317
0403	0.2306	0.2306
2018	0.8460	0.8456

TABLE 6 Values obtained from the trained network at different test points ($\mu = 0.01$, $\mu\_dec = 0.001$, $\mu\_inc = 10$)

Input	Target Value	Trained Value
1808	0.7660	0.7649
1204	0.5358	0.5364
2311	0.9577	0.9559
1718	0.7317	0.7315
0403	0.2306	0.2306
2018	0.8460	0.8402

In Tables 3-6, the training results for shift ciphers may be seen at different test points for different values of the training parameters.

3.5 TRAINING OF THE RSA CIPHERS

In this section RSA ciphers have been trained using neural network. RSA is an asymmetric public key cryptosystem, whose efficiency is based on the practical difficulty of solving factor problems. The algorithm to generate RSA cipher from plain text has been given below:

Algorithm

- Select prime numbers p and q.
- Compute the product $n = p \times q$.
- Compute euler totient function $\phi = (p-1)(q-1)$.
- Select public exponent e, $1 < e < \phi$ such that $\gcd(e, \phi) = 1$.
- Compute the private exponent d by $(d \times e) \bmod \phi = 1$.
- Public key is $\{n, e\}$ and private key is $\{d\}$.

For encryption: $C \equiv (P \wedge e)(\bmod n)$

For decryption: $P \equiv (C \wedge d)(\bmod n)$

where P is plain text and C is cipher text.

The RSA cipher for the sentence (1) as stated above is generated using the following MATLAB code. The value of n=2773 and e=21.

MATLAB code

```
a=input('Enter the value of a');

m=mod(a,2773);

for i=1:20

s=a*m;

m=mod(s,2773);

end

fprintf('Mod value %f',m);
```

Again, arranging the text into blocks of two and writing its corresponding numerical values, the plain text and ciphered text are used for training which are given in Table 7.

TABLE 7 Data set to train the neural network for RSA cipher

TEXT	P	C	NORMALISED C
SI	1808	10	0.0037
LE	1104	325	0.1207
NE	1302	2015	0.7482
EI	0408	2693	1
SG	1806	2113	0.7846
OL	1411	2398	0.8905
DE	0304	2031	0.7542
NT	1319	1760	0.6535
HE	0704	1879	0.6977
MA	1200	366	0.1359
NI	1308	763	0.2833
SW	1822	1182	0.4389
AL	0011	2014	0.7479
KI	1008	316	0.1173
NG	1306	2687	0.9978
IN	0813	852	0.3164
TH	1907	331	0.1229
ER	0417	2192	0.814
AI	0008	976	0.3624
NT	1319	1760	0.6535
HE	0704	1879	0.6977
DO	0314	390	0.1448
GS	0618	1575	0.5848
AR	0017	2330	0.8652
EB	0401	2669	0.9911

AR	0017	2330	0.8652
KI	1008	316	0.1173
NG	1306	2687	0.9978
TH	1507	1055	0.3918
EB	0401	2669	0.9911
AK	0010	1825	0.6777
ER	0417	2192	0.814
YW	2422	2011	0.7468
AS	0018	2049	0.7609
CL	0211	90	0.0334
OS	1418	882	0.3275
ED	0403	2305	0.8559
TI	1908	307	0.114
LL	1111	1007	0.3739
NI	1308	763	0.9543
NE	1304	2522	0.9365
TO	1914	2674	0.9929
PR	1517	2449	0.9094
OT	1419	300	0.1114
ES	0418	2617	0.9718
TA	1900	45	0.0167
GA	0600	2592	0.9625
IN	0813	852	0.3164
ST	1819	417	0.1548
TH	1907	331	0.1229
EN	0413	1888	0.7011
EW	0422	2208	0.8199
TA	1900	45	0.0167
XL	2311	1117	0.4148
AW	0022	2454	0.9113
SI	1819	417	0.1548

HI	0708	2183	0.8106
SE	1804	2096	0.7783
NR	1317	95	0.0353
AG	0006	1991	0.7393
ED	0403	2305	0.8559
TH	1907	331	0.1229
EC	0402	2175	0.8076
US	2018	1107	0.4111
TO	1914	2674	0.9929
ME	1204	160	0.0594
RS	1718	765	0.2841

We train a neural network for 15 hidden neurons and obtain the simulations at different test points, for different values of μ, μ_dec and μ_inc. The following tables show the test results for the different parameters taken.

TABLE 7 Values obtained from the trained network at different test points
$(\mu = 0.001,\ \mu\_dec = 0.01,\ \mu\_inc = 10))$

Input	**Target Value**	**Trained Value**
1718	0.2841	0.2264
0813	0.3164	0.3440
0417	0.8140	0.9797
2311	0.4148	0.4362
1419	0.1114	0.2160
0403	0.8559	0.9545

TABLE 8 Values obtained from the trained network at different test points
($\mu = 0.001$, $\mu\_dec = 0.001$, $\mu\_inc = 10$)

Input	Target Value	Trained Value
1718	0.2841	0.2836
0813	0.3164	0.3182
0417	0.8140	0.9068
2311	0.4148	0.4151
1419	0.1114	0.3987
0403	0.8559	0.8964

TABLE 9 Values obtained from the trained network at different test points
($\mu = 0.01$, $\mu\_dec = 0.01$, $\mu\_inc = 10$)

Input	Target Value	Trained Value
1718	0.2841	0.3956
0813	0.3164	0.4462
0417	0.8140	0.8343
2311	0.4148	0.6806
1419	0.1114	0.6514
0403	0.8559	0.8214

TABLE 10 Values obtained from the trained network at different test points
($\mu = 0.01$, $\mu\_dec = 0.001$, $\mu\_inc = 10$)

Input	Target Value	Trained Value
1718	0.2841	0.3305
0813	0.3164	0.3092
0417	0.8140	0.8023

2311	0.4148	0.6821
1419	0.1114	0.4028
0403	0.8559	0.7986

In Tables 7-10, the training results for RSA ciphers may be seen at different test points for different values of the training parameters.

3.6 GENERALISATION PERFORMANCE

The trained results have been analysed for shift and RSA ciphers. The accuracy of different networks obtained by varying the parameters has been observed by evaluating the percentage of trained data. It may be noted from the Tables 3-6 and Tables 7-10 that the training percentage increases as μ_dec is reduced. Smaller values of μ may shift the algorithm to Newton's method, thereby making the error minimisation process faster and more accurate. Table 11 shows the percentage of trained data for RSA ciphers with the parameters chosen in Table 8.

TABLE 11 Results for networks trained for RSA cryptosystem

Topology	**Epochs**	$\lambda_0(\%)$	$\lambda_{\pm 20}(\%)$	$\lambda_{\pm 30}(\%)$	$\lambda_{\pm 40}(\%)$
1 – 15 – 1	1000	31	50.7	61.19	65.67

- $\lambda_0(\%)$ is the complete measure of the training data, where the network computes the exact target value.
- $\lambda_{\pm v}(\%)$ is the near measure of the training data, where the error lies in the interval $(\pm v)$ for $v = 20, 30$ and 40.

4 CONCLUSION AND FUTURE WORK

A neural network based cryptography technique has been implemented to study encryption and decryption techniques. Accuracy is enhanced by proper selection of network topology and parameters in the training algorithm. Related model has been simulated for various example problems. Finally, the accuracy has been demonstrated in form of Tables.

The future work that may be done in this regard includes:

1) Minimisation of the error function by improved methods
2) Implementation of better training algorithms and network architectures
3) Increasing the efficiency of training for the generalised cryptosystems.

REFERENCES

[1] Jacek M. Zurada, Introduction to Artificial Neural Systems, West Publishing Company, St. Paul, 1992.

[2] Thomas Koshy, Elementary Number Theory with Applications, Elsevier, a division of Reed Elsevier India Private Limited, Noida, 2009.

[3] I. Kanter and W.Kinzel, "The Theory of Neural Networks and Cryptography," *Quantum Computers and Computing,* vol. 5, pp. 130-139, 2005.

[4] E.C.Laskari, G.C.Meletiou, D.K.Tasoulis, M.N.Vrahatis, "Studying the performance of artificial neural network networks on problems related to cryptography," *Non linear Analysis: Real World Applications,* vol.7, pp. 937-942, 2006.

[5] G.C.Meletiou, D.K.Tasoulis, M.N.Vrahatis, "A first study of the neural network approach in the RSA cryptography," in *Sixth IASTED International Conference on Artificial Intelligence and Soft Computing (ASC 2002)*, Banff, Alberta, Canada, July 17-19, 2002.

[6] [Online]. Available: http://www.wikipedia.org/.

[7] "Mathworks," The Mathworks, Inc., [Online]. Available: http://in.mathworks.com/help/nnet/ref/trainlm.html;jsessionid=a15fe82129a83dc8a92470543e5c.

LIST OF PUBLICATIONS

- Presented a paper entitled “Cryptography using Neural Network” at the 42nd Annual Conference of Odisha Mathematical Society and a National Seminar on “Uncertain Programming”, 7-8th February 2015, at Vyasanagar Autonomous College, Jajpur Road.