International Journal of Signal Processing, Image Processing and Pattern Recognition
Vol. 8, No. 1 (2015), pp. 161-192
http://dx.doi.org/10.14257/ijsip.2015.8.1.16

Secured Image Transmission Using a Novel Neural Network Approach and Secret Image Sharing Technique

Rakesh Kumar[1] and Meenu Dhiman[2]

[1]Department of Computer Science & Engineering
NITTTR, Chandigarh, India
[2]Department of Information Technology
M. M. University, Mullana, Ambala, India
raakeshdhiman@gmail.com, meenur194@gmail.com

Abstract

In this paper we have combined both cryptography and steganography techniques. This provides the higher level of secure system in which the secret information can be transferred over any unsecured communication channel and to overcome the threat of intrusion. The presented work aims at secure image transmission where a random encryption algorithm is used to encrypt different shares of stego image, which is created when a secret image and the cover image are embedded together, and produces shares using a secret sharing technique. At the receiving end, decryption of the encrypted shares are done using an artificial neural network and hence eliminating the need of key exchange before the transmission of data which is a prerequisite for most of the general encryption algorithm. Artificial neural network is used to provide high security of data and to produce distortion less decrypted images. The reversible process is used to reconstruct the secret image from the decrypted images. Peak signal to noise ratio, structure similarity and mean square error are used to analyze the quality of stego images. The simulation results show that the secret is reconstructed without loss and the time taken for encryption and decryption is very less.

Keywords: *Security, Image processing, Secret sharing, Encryption, Decryption, Artificial neural network.*

1. Introduction

We propose a system architecture that can provide a potential solution for non-key, secret sharing based crypto steganography approach, thus facilitating optimum solution for secured data communication. The strength of our system resides in adding multiple layers of security. Goal of this paper is to develop a new system [1] that can be highly secured, as secret cannot be retrieved easily from the image by interacting hackers [2] in the communication process.

Image steganography [3, 4] is a technology where a secret image is embedded in a cover image in such a way that the existence of secret information is hidden. Once the cover image has information embedded in it, it is called a stego image. The important requirements of image steganography algorithm are invisibility, payload capacity, and robustness against imagemanipulation attacks [5, 6]. Cryptography [7] is a science where content of message is kept secret. It is a collection of Encryption/decryption algorithm, Integrity check functions and Digital signature schemes. Traditionally the data encryption techniques are used for information security [8]. To perform encryption, different Data Encryption Standards (DES) have been developed since 1977. However, after progressing towards 1998 these standards were vulnerable to brute force attacks, differential and linear cryptanalysis. Hence, the need for new encryption standard, several ciphers [9] have been proposed such as AES, 3DES, RSA, Serpent, two fish, blowfish and IDEA. Even these were not sufficient for the voluminous data so the other methodologies are developed such as affine transform, the chaotic system [10] and the frequency domain algorithm [11].

However, keeping communication private [12] over the network is very difficult even after having numerous methods and techniques. So here, we propose a random algorithm for encryption where the pattern is not specific and the decryption scheme is based on a Multilayer Feed forward (MLFF) neural network, which uses a backpropagation learning method. Thus, we provide a solution where the information is not disclosed [13] to other persons who are not authorized to access it. Hence, the secrecy or the confidentiality is not compromised under any circumstances and only the intended person [14] is able to intercept and retrieve the secret.

Whether it is image processing or otherwise, most of the encryption /decryption algorithms used are generic, as they use key. The strength of an algorithm lies on the length of key because of which a key exchange has become a prerequisite prior to the data transmission. In the present work, encryption process uses random substitutions. After that, a second level encryption is carried out by addition of impurity or doping which creates more confusion and misguide the cryptanalyst to obtain the cipher. At the receiving end, it uses MLFF neural network to obtain the original image. The elimination of the key exchange and the usage of ANNs [15] for high security are the major strengths of the proposed work.

We have also amplified the proposed algorithm by adding another technique named as secret sharing technique. In this secret image is sent on the unsecured communication channel [16] is first embedded with a cover image and then the stego image is divided into shares. Each share is encrypted using the proposed random encryption algorithm and sent over the different channels like email, pen drive, CD’s, DVD’s, sockets etc. The concept lies here are even if a cryptanalyst or steganalyst [17] is able to decrypt a code or find the existence of secret, which is quite difficult, will not have other shares to retrieve the secret. Once all the shares are received, the secret data can be retrieved through reversible process.

The essentials of the secret image sharing mechanism [18] are as follows-

1. The *t* shares are required to reveal the secret data.
2. No (t - 1) participants are able to learn anything about the secret data.
3. The shadow [19] must be meaningful.
4. The quality of the stego images must be satisfactory.
5. The retrieved secret image must be distortion-free or a lossless image.

Our image-sharing algorithm inhibits all the above defined essential requirements. The reversibility of the new scheme allows authorized participants to reconstruct the distorted stego image to the original, without distortion. After retrieving the secret data, the genuine participants were able to retrieve the distortion less secret, which is major necessity for the bank secret codes, medical, military and other important artistic images too.

Hence, we can say that crypto steganography [20,21] is a powerful tool, which enables people to communicate over the unsecured channel without possible eavesdropper ever knowing that there is a form of any communication in the first place, even if it happens it is too difficult to break the code at the second place, say even this happens all the shares will never be there to retrieve the secret. Finally we can conclude that the proposed technique is very effective and highly secured for data communication [22,23].

The field of ANN is interdisciplinary which is explained in the section 2. The concept of the secret sharing is elaborated in section 3, followed by the System architecture of the proposed algorithm in section 4 and section 5 incorporates the experimental results and simulations and finally section 6 presents conclusion and future scope.

2. Artificial Neural Network

It is a powerful data-modelling tool, which is made up of parallel-distributed processor [24]. It means that multi simple processing units or nodes are interconnected with one another by means of communication link to perform a task. Neural network [25] acquires knowledge through learning [26] where the knowledge is stored within inter-neuron connection strengths known as synaptic weights.

Each neuron has an internal state known as activation, which is a function of the inputs it has received. The use of neural network [27] offers the Input-Output Mapping property and the learning methods, which are as follows-

1. Supervised or Associative learning.
2. Unsupervised or Self-organized learning.

The supervised learning is based on the output that is already defined and neural network knows the target. While training, the network matches its output with the predefined target outputs.

In the self-organised learning, the output or the target value is unknown to the neural network. It learns the distribution of patterns and makes a classification of those patterns. Then the decision is taken according to the similar patterns that are assigned to get the same output cluster.

We have used the supervised learning method for training of neural network. This learning method is used as a mapping function. From the various neural nets we have used a backpropagation neural net. There are different perceptron and one-layer networks are available but they are limited in their capabilities. We have used a multilayer Feedforward backpropagation neural net which has a nonlinear node characteristics. These nonlinear characteristics enable MLFF to overcome the limitations of one-layer networks. There are three steps to train a neural network

1. Feedforward of the input training pattern.
2. Calculation performed and the backpropagation of the associated error.
3. Adjustment of the weights.

However, the computation of the Feedforward net and the training process is slow but once the neural network is trained properly, the accuracy of the output is cent per cent and it produces result very fast.

Neural network is an information processing system, which plays a major role for information security. In this field many have used neural networks for encryption/decryption of either text or image.

Network security is explained by Khalil Shihab [28]. The decryption algorithm is based on MLFF neural network and trained by backpropagation learning algorithm. Encryption scheme and creation of a private key is based on Boolean algebra. It is shown that guessing of keys is not possible. No memory complexities are present in this proposed scheme.

Munukur R. K, Gnanam V. [29] proposed a scheme that eliminates the need of key decryption process. They have used a cipher text that can be decoded using artificial neural network. To confuse the eavesdropper they introduced lies in the information making it more difficult to decode the cipher text.

Liew Pol Yee, De Silva L.C. [30] explains application of multilayer perceptron network. In this, a hashing algorithm that is resistant to birthday attacks and collision free is explored. It is shown that it is very difficult to recover an input from a hashed output.

3. Secret Sharing

Secret sharing [31] sometimes also known as secret splitting refers to a method where a secret image is split into different shares and distributed among a group of participants. Each participant receives a share. The secret can be reconstructed only when a sufficient number of shares are combined together. In this the individual shares are of no use on their own.

A (*t, n*) threshold scheme is used in this approach where there are only one dealer and multi players or participants are present. The dealer distributes share to *n* players and all players can together reconstruct the secret but no group of fewer than *t* players can retrieve the secret.

Secret sharing schemes [32,33] are ideal and very important for storing highly sensitive information. For example, in military the missile launch codes, fingerprint images used as a password or bank account numbers etc. are highly confidential. It could be disastrous if they are revealed [34]. Therefore, secret sharing appeared as we cannot keep the entire thing at one place because it demands the higher level of secrecy and cannot keep the multiple copies at different locations, as it demands the higher level of reliability and hence results in more confidentiality and reliability.

Steganography [35, 36] is a method of embedding the secret message into a cover image where the existence of the secret is hidden. The images, which hide the secret in it, are known as stego-image. Few of the already proposed techniques (Thien and Lin [37,38], Lin and Tsai [39], Chang and Hwang

[40], Feng et al. [41]) on image processing are based on threshold (*t*,*n*) polynomial and produces meaningful as well as meaningless stego images.

Shamir's polynomial-based [42] secret sharing scheme performs very well where the secret is image and the stego image is divided into *n* shares and *t* shares are required to reconstruct a secret. Visual secret sharing scheme proposed by Naor and Shamir [43], Lukac and Plataniotis[44] is another image secret sharing technique where, only human visual sight is needed to decode and share the secret pixel with very little calculation but results in extremely poor quality of secret image which is not beneficial in the applications where slight change can result disastrous.

In other proposed approaches like Thien and Lin produce small shadow images with size 1/*t* times that of the original image which is more suitable for faster transmission but the shrunken version results into the insufficient quality of the recovered secret for the high-end applications. In the algorithm proposed by Chang and Hwang suggests a vector quantization codebook of the secret image, which is good in sharing process, but produces a noise-like shadow images hence very susceptible for the censors. Few other schemes proposed by Feng et al., Chang and Lin[45], Lin and Tsai, [46,47] uses the steganography technique for the color secret image with authentication, where every four LSB bits in red, green and blue plane are reserved. Say nine bits for embedding of secret data and three bits for the authentication purpose. This technique is good for color image but not suitable for gray-color image as it distorts the quality of stego images.

All the above weaknesses can be overcome by a novel image secret sharing technique proposed in this paper which uses (*t*,*n*) threshold scheme. Authorized participants are allowed to reconstruct the secret from stego. In the existing methods, the distortions are large. The proposed threshold image sharing approach and reversible scheme [48] for images reveals the secret image without loss [49] with a good quality of stego images [50].

4. System Architecture

The overall system architecture proposed in this research work is shown in Fig. 1. The mentioned system architecture represents the overall functionalities of the proposed research work. Here two input images have been taken one is for cover image and another is for secret image. In the proposed architecture, we have taken both images in gray scale format. In case of color RGB images, these are required to be converted into gray format first and then feed them as input for further processing. At the transmitter side, we have applied the two levels of encryption using stego image generation, permutation, substitution and impurity addition. This double encrypted image is then transmitted. The size of secret image is taken as half of the size of cover image. i.e. $(H * W) / 2$. Once the images have been selected the two parameters called prime number (m) and number of shares (t) have to be assigned, where the limitations of prime number under consideration is $m \geq 7$ and shares $t= 2$ it generates number of shares as defined by variable t. At the receiver side decryption is done by removing the impurity in first step and then using ANN.

4.1 Shadow Derivation

Secret sharing is a process of shadow generation and has two main parts: sharing and recovery. During sharing, a secret image is embedded into a cover image and we generate n shadow images. During recovery the reverse procedure is followed and the secret image is regenerated from the shadow images given any t out of n shadows images. Shamir [42] proposed a (t, n) threshold secret sharing scheme based on polynomial interpolation. Given t distinct points, (x_1,y_1), (x_2,y_2),…,(x_t,y_t) in 2-dimensional space, we can generate a unique polynomial function F(x) of degree $(t – 1)$. Without the loss of generality, S is the shared secret and is assumed to be a number. To divide S into n shadows, we randomly select a prime number m and $(t – 1)$ digits, $C_1, C_2, ... , C_{t – 1}$, that range from 0 to $(m – 1)$, and we construct $F(x) = S + C_1x^1 + ... + C_{t–1}x^{–1}$ mod m. After $F(x)$ is defined, n shadows, $z_1, z_2,…,z_n$ can be generated using the following equations

$z_1 = F(1), z_2 = F(2), ... ,z_i = F(i), ... , z_n = F(n)$

Given any t out of n pairs of (x_i,y_i), the polynomial $F(x)$ can be reconstructed by Lagrange interpolation and we can find the original S. If we have less than t pairs of (x_i,y_i), $F(x)$ cannot be reconstructed successfully because of insufficient data.

Fig. 2 shows the entire process of stego image generation (first level encryption) using shadow generation and quantization. In this process, we have taken two images cover image C and secret image S. We convert the secret image into m-ary notational system. An array of coefficients C_1, C_2, C_3 ....C_{t-1} is defined to implement polynomial function $F(x)$. Variables p, m and t are used for shadow generation where, p is the pixel value of cover image, m is the prime number and t is the number of shares generated. Obtain the number of coefficients for $(t – 1)$ digits. The polynomial function is framed for channel. The $(t – 1)$ digits of a channel are the inputs of polynomial function. Participants generate their shares. The following modules with examples show the overall functionalities like shadow derivation, quantization, stego image generation and secret image reconstruction.

Example:

1. Shadow Generation

$C_1= 2$, $t = 2$, $m = 7$, $S = 3$ where C_1 is coefficient, t is number of shares, m is prime number, S is secret.

$F(x) = (S + C_1x^1 + C_2x^2 +...+C_{t-1}x^{t-1})$ mod m

Therefore, for two shares the equation will be

$F(x) = (S + C_1x)$ mod m

For each x=1, 2

$F(1) = (S + C_1 * 1)$ mod m

⇨ $(3 + 2 * 1)$ mod 7

⇨ 5 *mod* 7

⇨ 5

$F(2) = (S + C_1 * 2)$ mod m

⇨ $(3 + 2 * 2)$ mod 7

⇨ $7 \textit{ mod } 7$
⇨ 0

To share the secret image S, it can be converted to m-ary notatational system. We can embed one secret pixel into polynomial $F(x)$. If $m = 7$ and two secret pixel values are 92 and 113, then the converted digits are $(1, 6, 2)_7$ and $(2, 1, 3)_7$. One secret pixel to be embedded into polynomial $F(x)$ is [6, 9].

To increase the capacity of embedded secret data we can embed $t – 1$ secret digit into $F(x)$. E.g. embedding $t – 1$ secret digits into $F(x)$, assume $s_1, s_2, \ldots, s_{t–1}$ are the secret digits of S and C is the cover image having pixel value p. Invertible sharing is also capable of preserving the value of p. For this we compute the value of d as

$$d = p \bmod m$$

An invertible polynomial $F(x)$ can be written as

$$F(x) = s_1 + s_2x^1 + \ldots + s_{t-1}x^{t-2} + dx^{t-1} \bmod m$$

For example in (3, 3) threshold system, we can share two digits $(s_1, s_2) = (2, 5)_7$ of S into cover pixel having pixel value 243.

$$d = 241 \bmod 7 = 3$$

Thus, $F(x)$ can be written as

$$F(x) = s_1 + s2x^1 + dx^2 \bmod m$$
$$= 2 + 5x^1 + 3x^2 \bmod 7$$

We can assign unique key K_i for each participants where $i = 1, 2, \ldots, n$. The n shadows z_i can be generated by feeding the secret key K_i into $F(x)$.

$z_1=F(K_1), z_2=F(K_2) \ldots z_i=F(K_i) \ldots z_n=F(K_n)$

In this way, shadows are generated.

4.2 Quantization Process

Quantization process $Q = \mathit{floor}\ (p / m) *m$ preserves the cover image pixel value in order to retain the actual quality of cover image. Quantized pixel value is computed with the help of two operations division and multiplication. Divide the cover image pixel by the prime number m and take the floor value, perform multiplication by prime number on the floor value, which gives the quantized value of cover image pixel.

Example:
Quantization

48	50
51	48

42	49
49	42

$Q = \mathit{floor}\ (p / m) *m$

$Q = \mathit{floor}\ (48 / 7) * 7$

$Q = 6 * 7$

$Q = 42$

4.3 Stego Image Generation

The cover image is used for hiding the generated shadow images. To generate the stego images, we embed the pixels of shadow image into cover image. The quantization value generated from cover image is added to the pixel from shadow image to get the stego image.
$Stego = F + Q$ --- (1)
Where, F represents shadow value while Q is for quantized value.

Example:
Share Generation or Creation of Stego Images
From equation (1), we get
($Sh1 = P_q + F(1)$)
⇨ 42 + 5
⇨ 53

($Sh2 = P_q + F(2)$)

⇨ 42 + 0
⇨ 42

By repeating the shadow derivation and quantization process, we can generate all secret shares into the cover pixel in order to obtain n stego images. Subsequently, we can distribute meaningful stego images and the keys to the involved participants.

4.4 Encryption Module

Each stego image, which is to be encrypted, is read pixel by pixel and the transformation is done on these pixels using permutation and substitution at the first level. Second level of encryption involves the impurity addition. Two level of encryption are used to obtain high level of image encryption. It is required, because for some of the images, first level of encryption is not sufficient, although an intensity of the pixel has changed drastically but the picture is still in an understandable form. The reason behind this is that after first level of encryption all, the pixels with same original value will have the same encrypted value, because of which intensity changes but image can be still visible. To overcome this problem, the second level encryption is used in which the impurity changes with respect to the pixel position, it means that the pixel with same original value will have two different values after the second level of encryption. The transformation process for encryption (level 1 and level 2) is shown in Fig. 3. The generated stego images after the first round of encryption is further encrypted either by using permutation or without permutation.

Encryption Algorithm

First Level Encryption:

Step 1: Obtain the pixel value of the image file and convert it to binary using de2bi function, say the value of the pixel is 165; its binary equivalent is {10100101}.

Step 2: Split the pixel byte into two parts nibbles i.e. lower bits from 1:4 and upper bits from 5:8. The result will be {1010} and {0101}.

Step 3: Exchange the nibbles and concatenate them to form a resultant byte, {01011010}.

Step 4: EX-OR the original msnibbles and lsnibbles to get the doping byte, {1111}.

Step 5: Append an extra bit by shifting the bits of the impurity by 5 bits to the right. Now we get 9 bit number, {111100000}.

Step 6: EX-OR the results of Step 3 and Step 5, {110111010}.

Step 7: Convert the result obtained at step 6 into decimal and then add impurity 117 to it.

Step 8: Continue Step 1 to Step 7 for all pixels of the image file.

Second level Encryption:
Step 9: For the normalization of the matrix, we need to add two columns, the value of 117 is added to the first new column, and 627 is added to the second column.

Step 10: Add another level of impurity with resultant matrix shown in the table 4 for second level encryption such that impurity changes with respect to the position of the pixel.
Formula: **-** (100 + (rowno – 1) * 57)

4.5 Decryption Module

The decryption process is accomplished by Artificial Neural Networks (ANNs) [51]. An artificial neural network consists of layers of interconnected “artificial neurons” or “computational units”. Fig. 4(a) shows a simple artificial neural network in which there is one neuron, which converts the inputs to output. A “neuron” in a neural network is called a “node” or “unit”. An ANN represents a highly parallelized dynamic system with a directed graph topology that can receive the output information by means of a reaction of its state on the input actions. The ensembles of interconnected artificial neurons generally organized into layers or fields include neural networks. Artificial neural networks are adaptive networks of simple non-linear computing elements called neurons which are intended to abstract and model some of the functionality of human nervous system in an attempt to partially capture some of its computational strengths. The environments in which neural networks can operate could be deterministic (noiseless) or stochastic (noisy). Artificial neural networks are composed of interconnecting artificial neurons (programming constructs that mimic the properties of biological neurons). ANN 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.

4.5.1 Architecture of Artificial Neural Network

The basic architecture consists of three types of neuron layers: input, hidden, and output. In feed-forward networks, the signal flow is from input to output units, strictly in a feed-forward direction. The data processing can extend over multiple layers of units, but no feedback connections are present. Recurrent networks contain feedback connections. Contrary to feed-forward networks, the dynamical properties of the network are important. In some cases, the activation values of the units undergo a relaxation process such that the network will evolve to a stable state in which these activations do not change anymore.

4.5.2 Feed Forward Neural Network

Feed-forward ANNs allow signals to travel in only one way from input to output. There is no feedback (loops) i.e. the output of any layer does not affect the same layer.

Feed-forward ANNs are straight forward networks that associate inputs with outputs. They are widely used in the field of pattern recognition. This type of organisation is also referred to as bottom-up or top-down. Single-layer perceptron, multilayer perceptron and radial basis function are types of feed forward neural networks.

4.5.3. Single Layer Perceptron

The simplest kind of neural network is a single-layer perceptron network, which consists of a single layer of output nodes; the inputs are fed directly to the outputs via a series of weights. In this way it can be considered the simplest kind of feed-forward network. The sum of the products of the weights and the inputs is calculated in each node, and if the value is above some threshold (typically 0) the neuron fires and takes the activated value (typically 1); otherwise it takes the deactivated value (typically -1). Neurons with this kind of activation function are also called artificial neurons or linear threshold units. A perceptron can be created using any values for the activated and deactivated states as long as the threshold value lies between the two. Most perceptrons have outputs of 1 or -1 with a threshold of 0 and there are some evidences that such networks can be trained more quickly than networks created from nodes with different activation and deactivation values. Perceptrons can be trained by a simple learning algorithm that is usually called the delta rule. It calculates the errors between calculated output and sample output data, and uses this to create an adjustment to the weights, thus implementing a form of gradient descent. Single-unit perceptrons are only capable of learning linearly separable patterns.

4.5.4 Delta Rule

The delta rule is a gradient descent learning rule for updating the weights of the artificial neurons in a single-layer perceptron. It is a special case of the more general back propagation algorithm. For a neuron *j* with activation function *g*(*x*), the delta rule for *j*'s, i^{th} weight is given by

$$\Delta W_{ij} = (t_j - y_j) g'(h_j) x_i$$

The delta rule is commonly stated in simplified form for a perceptron with a linear activation function as $\Delta W_{ij} = \alpha (t_j - y_j) x_i$ where α is known as the learning rate parameter.

4.5.5 Multilayer Neural Networks

This class of networks consists of multiple layers of computational units, usually interconnected in a feed-forward way. Each neuron in one layer has directed connections to the neurons of the subsequent layer. In many applications the units of these networks apply a sigmoid function as an activation function. The universal approximation theorem for neural networks states that every continuous function that maps intervals of real numbers to some output interval of real numbers can be approximated arbitrarily closely by a multi-layer perceptron with just one hidden layer. This result holds only for restricted classes of activation functions, *e.g.*, for the sigmoidal functions. Multi-layer networks use a variety of learning techniques, the most popular being back-propagation. Here, the output values are compared with the correct answer to compute the value of some predefined error-function. By various techniques, the error is then fed back through the network. Using this information, the algorithm adjusts the weights of each connection in order to reduce the value of the error function by some small amount. After repeating this process for a sufficiently large number of training cycles, the network will usually converge to some state where the error of the calculations is very small. In this case, one would say that the network has learned a certain target function. To adjust weights properly, one applies a general method for non-linear optimization that is called gradient descent. For this, the derivative of the error function with respect to the network weights is calculated, and the weights are then changed such that the error decreases (thus going downhill on the surface of the error function). For this reason, back-propagation can only be applied on networks with differentiable activation functions.

4.5.6. Back Propagation Algorithm

Backpropagation neural nets shown in Figure 4(b) are those Feedforward networks, which use backpropagation learning method for their training. Backpropagation algorithm uses gradient descent for the updation of the weights. The weights are the connection strengths between nodes in adjacent layers. The updation is necessary as it helps in minimizing the squared error between network output values and the target output values. Error can be reduced if weights are changed according to the derivative of the error function with respect to the network weights calculated. In other words, weights are adjusted according to its contribution to the error by using gradient descent. It is a supervised learning method, and is a generalization of the delta rule. It requires a dataset of the desired output for many inputs, making up the training set. ANN performs a highly adaptive nonlinear decision function from training examples. Establishing a relationship among coefficients of the image by using Back-Propagation Neural Networks (BPNN) model is done. For a selected coefficient, the network is trained with its 3×3 neighbors as input vector and the value of the coefficient as output. Construct three layers BPNN with 8, 10 and 1 neurons in the input, hidden and output layer respectively. The tangent sigmoid, purelin transfer function are used for recognition Input Layer Hidden layer Output Layer.

Phase 1: Propagation- Each propagation involves the following steps:

1. Forward propagation of a training pattern's input through the neural network in order to generate the propagation's output activations.
2. Backward propagation of the propagation's output activations through the neural network using the training pattern's target in order to generate the deltas of all output and hidden neurons.

Phase 2: Weight update for each weight-synapse follow the following steps:

1. Multiply its output delta and input activation to get the gradient of the weight.
2. Bring the weight in the opposite direction of the gradient by subtracting a ratio of it from the weight.

This ratio influences the speed and quality of learning; it is called the learning rate. The sign of the gradient of a weight indicates where the error is increasing. This is why the weight must be updated in the opposite direction. Repeat phase 1 and phase 2 until the performance of the network is satisfactory.

To properly train the neural network, we feed the model a variety of real life examples, called training sets. The data sets normally contain input and output data. The neural network creates connections and learns patterns based on this input and output data sets.

Training of neural network is an iterative process. It produces the correct output for the given input by comparing the actual output with the desired or the target output. To generate better output the weights in the network are adjusted using backpropagation algorithm in such a manner that the output, which is produced, should be closer to the correct answer. Figure 4(c) illustrates how the training sets are generated. First we compute the average frame of the image. Then the resulting frame is divided into non overlapping 3×3 blocks. The center of each block is the output while the neighbor’s coefficients are the input. Finally we proceed to neural network training until the specified goal or the maximum number of iteration is reached. Once the process converges, the final weights are stored in a file. This is also known as save ANN. After training, application of network involves only the computations of feed forward phase. The overall mechanism for decryption has been presented in Fig. 5 which shows that first the impurity is removed from encrypted image and then further decryption is done using artificial neural network approach.

The Neural Network is trained for the standard mapping value and weights, and biases are stored before applying the input to it. Here in this research work we have implemented MATLAB's [54] neural network toolbox for implementing and training the neural network. The system is designed for three layers *i.e.*, input layer, hidden layer and output layer. The input layer of a neural network is determined from the characteristics of the application inputs. There are 320x240 (*i.e.*, 76800) pixels in each frame coming from an image. Each pixel contains three elements (red, green, and blue components). Thus, the total number of elements in a frame is 3x76800 (*i.e.*, 230400). If all these elements are directly put into the neural network, it will be almost impossible to process the image in real-time with a standard PC. Therefore, a preprocessing stage is incorporated to reduce the size and dimensionality of the input pattern. Moreover, we have taken grey scale image for implementation. Otherwise we can convert the color frame into a gray level image. Finally, the resulting image matrix is reshaped to form a standard pattern (column-vector).

Hidden layer automatically extracts the features of the input pattern, and reduces its dimensionality further. There is no definite formula to determine the number of hidden neurons. In this research, a hit-and-trial method was used to identify the number of neurons in the single hidden layer. It was found that with one input node, one output node and 695 hidden neurons, we accomplish the best performance. The tangent hyperbolic activation function was chosen for the hidden layer after comparing its converging results with those of the logistic sigmoid function. The tangent hyperbolic function and its fast approximation are given as: -

$$a_{i1} = \tanh(n_{i1}) = \frac{e^{n_{i1}} - e^{-n_{i1}}}{e^{n_{i1}} + e^{-n_{i1}}} \cong \frac{2}{1 + e^{-2n_{i1}}} - 1$$

where a_{i1} is i^{th} element of a_1 vector containing the outputs from the hidden neurons, and $n_{i1}$is i^{th} element of n_1 vector containing net-inputs going into the hidden neurons.

The decryption process is achieved by loading the data matrix first. Secondly, removal of the impurity and extra columns should be deleted, results in second level decryption. First level decryption is achieved by the simulation of network using ANN. The output of this stage is the decrypted image. In this way, all the shares are to be decrypted and then next process of secret image reconstruction can be done from these decrypted images.

Training of Neural Network

Step 1-Store all the pixel values of first level encryption.

Step 2-Convert indices to vector using ind2vec function.

Step 3-Detect minimum and maximum values.

Step 4-Initialize multilayer Feedforward backpropagation network function newff with the given parameters.

a) Network with hidden layer = 695 neurons
b) Transfer function = logsigmoid
c) Training function trainrp = resilient backpropagation

```
    d)  Learning function learngdm = gradient descent
        withmomentum weight and bias

Step 5-Set the parameters for the network using
       net.trainParam

Step 6-Train the neural network.

Step 7-Save the network.

Step 8-Testing the ANN

Step 9-Simulate the neural network

Step 10-Find the maximum value and its position.

If accuracy is cent percent then
  Neural network is trained properly
Else
  Train the neural network again
```

4.6 Secret Image Reconstruction

Secrets can be reconstructed only with minimum of *t* stego images, less than *t* is of no use. In this process, the decrypted images are taken into consideration and then after combining shares they do form the original secret image. Thus, retrieving the original secret image through a robust steganography technique [52,53]. The secret retrieval mechanism using multiple decrypted shares has been presented in Figure 6.

Example for Reconstruction:

1. Get the equations of shadow values.

$F(1) = (S + C_1 * 1) \bmod m$ ---(2)
$F(2) = (S + C_1 * 2) \bmod m$ ---(3)
From equation (2) and equation (3), we get
$(S + C_1 * 1)=5$
$(S + C_1 * 2)=7$
Hence, the Secret = 3.

Secure transmission of data is needed in the worldwide computer network environment. The effective and secure protection of sensitive information are primary concerns where it is a risk if a set of secret data is held by only one person, because the secret data set may be lost incidentally or modified intentionally. That is why in this algorithm we first transform the secret pixels into *m*-ary notational system and then transformed secret data are shared using (*t*,*n*) threshold sharing scheme. Once the stego images are created, secure transmission of data is done by encrypting them using a random algorithm. Decrypting of the shares can be obtained by multilayer Feedforward neural network and reconstruction of the secret can be obtained by reversible process and hence retrieve the lossless secret image and allows us to combine both the technologies to send the vital information in a secret way.

4.7 Features of the Proposed Work

1. Implementation of (*t*,*n*) threshold approach for secret sharing and reversible scheme for reconstruction of the shares.
2. The quality of stego images for the two shares are calculated, where average PSNR=40.5dB.
3. Retrieval of the lossless secret image whose Correlation Coefficient=1.
4. Usage of MLFF Backpropagation neural network for decryption leads to key exchange prior to data transmission has been eliminated.
5. Execution time for encryption and decryption is very less as compared to other algorithms.
6. Highly secured random algorithm has been used for encryption level hence making it difficult for an eavesdropper to decrypt and thus provide a high-level security.
7. Faster training has been achieved by using trainrp function of MATLAB neural network toolkit.
8. Cent percent accuracy is achieved using ANN, where error is zero.

5. Research Work Results and Simulation

This research work is implemented on MATLAB 7.10.0 simulation tool with gray scale cover image shown in Figure 7(a) and secret image shown in Figure 7(b). Secret image Figure 7(b) is the image that we want to hide inside the cover image Figure 7(a). In this process, number of shares is generated using threshold based polynomial function and a multiple level of encryption and decryption was carried out. The decryption has been done with MLFF neural network with Backpropagation learning. Our work (ANN enabled decryption) is a robust and reliable solution for secret share based steganography. We used three measurement metrics in our experiments to evaluate the quality of the decrypted secret image. The peak signal-to-noise ratio (PSNR) is a popular method for evaluating the image quality. PSNR measures the difference between two images. When two images are similar, the PSNR value is high, and vice versa. According to literature [2], the acceptable quality of a lossy image ranges from 30 dB to 50 dB, where a higher value is better. Structure Similarity [55] (SSIM) is second metric that we have used to evaluate the image quality. This method divides an image into several blocks that are the same size and that are non-overlapping. We then calculate the similarity of every single block in the two images. The values will range between -1 and 1, where 1 means that two images are identical. By contrast, the value is -1 when two images are completely different. Third metric is mean square error (MSE), which is defined as square of error between original cover image and stego image. The error indicates the distortion in an image.

The peak signal to noise ratio (PSNR) derivations of Stego images has also been achieved. In this work, we have gained higher PSNR, illustrating better image quality and perceptual originality. To estimate the quality of the stego images, the PSNR is used.

$$PSNR = 10 \log_{10}\left(\frac{255^{2}}{MSE}\right) dB$$

Mean Square Error between the original cover image and the stego image is calculated using the formula given below.

$$MSE = \frac{(Max\ (double\ (x, y))^{2}}{d}$$

The simulation results obtained for the creation of Stego images have been shown in Fig. 8. Multiple stego images are created after applying the shadow generation and quantization as shown in the figure. It illustrates the creation of stego images of the steganography. It can be clearly seen from the image that the secret image is now embedded into the cover image and is not visible. Here we have taken two parameters prime number m = 7 and number of shares t = 2 and then compute it for generation of shares.

The above steps represent share generation. As the number of shares (t) provided here are two, therefore two shares have been generated. Once the two shares have been generated these are processed for encryption process. In Figure 9, the first level encryption is shown without using keys. Although using this technique secret image is encrypted but still it is not completely hidden. An eavesdropper may steal some or more portion of the secret image. Hence, to enhance the security to the secret image second level encryption is applied as shown in Fig. 10. Now, we can see that after the application of second level of encryption neither the cover image nor the secret image is visible. So, eavesdropper can’t have any idea about the secret. Once the shares have been encrypted, they are saved and then transmitted through the different communication channel for further enhancing the physical security. The transmission can be done via different media i.e, one share can be sent through email and the other share can be sent through CD. Even if an intruder receives a share it is of no use since to reconstruct a secret, one needs all the shares.

After receiving the doubly encrypted multiple shares at the receiver terminal, multi layer feedforward neural network is applied for decrypting each share, thus results in decrypted image shown in Figure 11. But with ANN we cannot regenerate the secret image and have to apply first level decryption. Figure 12 shows the decrypted cover image after applying first level of decryption. Once all the shares are decrypted (second and first levels) then the secret image can be retrieved by combining the decrypted shares as shown in Figure 13. It can be clearly shown from the Figure 13 that by using our technique the quality of secret image is not at all degraded.

The encryption and decryption process is shown step by step. First the stego image share 1 undergoes the process of 1st level encryption where the image is visible to some extent because the encrypted value will be same if original pixels have got the same value and hence the need arises for 2nd level of encryption. Similarly, the decryption process follows first 2nd level decryption and then 1st level decryption is done.

We have also facilitated a permutation and de-permutation based steganography scheme, where at encryption we can provide encryption key and the same key would be utilized to de-permute the 1st level decrypted image to retrieve data. The usage of key is to show the difference between using key and non-key approach. The results achieved after using a key is shown in Figure 14. Figure 14 clearly shows that the image generated by first level encryption using key is more difficult to interpret by the intruder as compared to the image generated by first level encryption without using key (Figure 9).

Table 1 represents the PSNR values of two stego images where the value of prime number m is chosen as 7, 11, 13, 17, 19. The distortions present in the stego images are calculated using PSNR. The PSNR value ranges from 40.93 dB to 31.17 dB for different values of prime number m. For m = 7 we get the least distortion and the PSNR value reaches near to 41 dB which is quite good.

Table 2 represents PSNR values for different Stego images and it depicts the encryption and decryption time for different images and for different shares. A higher PSNR value is always required which means that the quality of the stego image is more similar to that of original cover image. PSNR value less than 30 dB is considered as an unacceptable quality. PSNR value less than 35 dB means that some of the very important signal characteristics are lost. Good quality is considered, if PSNR value is greater than 35 dB. In our case we get the PSNR value more than 39 dB. We get the encryption time ranging from 0.42 sec to 0.61 sec. But in most of the cases the encryption time for any image is generally 0.5 sec. Decryption ranges from 9.73 sec to 10.67 sec. However, the decryption is more than the encryption time but it is still less than the earlier techniques proposed in the literature with the achieved image quality.

Table 3 depicts the pixel value after 1st level of encryption. We have used a random substitution algorithm for transforming the original pixel values to the encrypted values. It can easily be seen from the table that the pixel values are transformed from the original value to a greater extent. More the difference in the original and encrypted pixel values more will be the encryption. We have achieved the difference of upto 250 which is very difficult for the intruder to understand.

Table 4 depicts that the pixel value for original and encrypted (1st and 2nd level) pixels. Table shows the pixel value (51) changes with respect to its position during the second level encryption. In this the pixel value (51) has different value in every row if image due to the addition of impurities in it. This further increases the level of security and makes difficult for the intruder to analyse the secret data.

As disscussed ealier that more is the variation in the original and encrypted pixel values, more it will be difficult for intruder to steal the secret information. The graph shown in Fig. 15 shows the pixel transformation of pixel value 30 for the first and second level encryption from one to tenth row. The graph clearly demonstrates that there is huge variation in original and encrypted pixel values ranging from 450 to 960.

Our scheme is not limited to the images of only limited sizes. We have implemented the our approach for the images of different sizes as shon in Table 5. The table represents the encryption and decryption time for different shares generated for the different sizes of images. These values signify the response and execution speed of developed system with image of different resolution. It can be seen that size of image has very less impact on our scheme and the proposed scheme is applicable for all types and sizes of images.

We have compared the performance of our scheme with the existing schemes as shown in Table 6. The comparasion is done on the basis of three parameters PSNR, SSIM and MSE. The results show that the proposed algorithm achieves the PSNR value upto 40.6 dB for the prime number m=7 and number of shares t=2 which is quite high as compared to the other five techniques. The structure similarity (SSIM) value is also very high in the proposed scheme i.e. 0.993. The MSE is quite low in our work as compared to the other existing works.

The training of neural network is shown in Fig. 16. The training window will appear during training and shows that the data has been divided using the dividerand function, and the Levenberg-Marquardt (trainlm) training method has been used with the mean square error performance function. Recall that these are the default settings for feedforwardnet. During training, the progress is constantly updated in the training window. The magnitude of the gradient of performance and the number of validation checks are more important. The magnitude of the gradient and the number of validation checks are used to terminate the training. The gradient will become very small as the training reaches a minimum of the performance. If the magnitude of the gradient is less than 1e-5, the training will stop. This limit can be adjusted by setting the parameter net.trainParam.min_grad. The number of validation checks represents the number of successive iterations that the validation performance fails to decrease. If this number reaches 6 (the default value), the training will stop.

During the neural networking training process, firstly we assign the four weights. W_1, W_2, b_1 and b_2. These are the weights assign between the input and hidden neurons and between the hidden and output neurons.

$W_1 = a + (b\text{-}a) * rand(S1,R);$
$W_2 = a + (b\text{-}a) * rand(S2,S1);$
$b_1 = a + (b\text{-}a) * rand(S1,1);$
$b_2 = a + (b\text{-}a) * rand(S2,1);$

Then, we calculate the values and plot the different graphs of performance, training state and regression. From the training window, we can access four plots: performance, training state, error histogram and regression. It shows the various training states of neural networks at different epochs. The training state plot shows the progress of other training variables, such as the gradient magnitude, the number of validation checks, *etc*.

Figure 18 shows the neural network performance. The generated graph shows the achieved performance as 9.9786e005 for the goal of 0.0001, the accuracy is cent per cent. Hence, the retrieved secret image at the receiver side is exactly same as the original secret image. Figure 19 shows the graph between epoch values and sum squared error. Sum squared error measure the performance according to the sum of squared errors. The performance plot shows the value of the performance function versus the iteration number. It shows that the best training performance is 35.71 and is achieved at 283 epochs. Finally Figure 20 represents the regression plot which shows a regression between network outputs and network targets.

6. Conclusion and Future Scope

The system has been developed using threshold polynomial function for secret share development. Once the shares are generated, the transmission of shares can be done directly. However, in order to enhance the security and effectiveness a two level encryption has been implemented on the shares to protect them from any modifications during transmission. During encryption process, impurity is added with each level and thus transforms the pixel values with high randomness. A MLFF backpropagation neural network has been applied at decryption level so that the highly randomized stego shares can be decrypted in less time with higher efficiency. A need to key exchange prior to the transmission of data has been eliminated. Faster training has been achieved by using 'trainrp' function of MATLAB neural network tool kit. The accuracy of the system is found to be very good. Once the images have been decrypted they are combined together using reversible scheme to retrieve the original secret image, and thus the proposal has been done successfully.

The results for PSNR, encryption and decryption time, pixel values at different encryption level depicts that the developed system can play a vital role in image processing. Hence, the clubbing together of cryptography and steganography concept would be highly effective and provide an optimum solution for secure data communication over network. The presented work thus provides a means of robust, fast, accurate and secure data transmission and reception.

The future work will try to develop an enhanced steganography approach that can have the authentication module along with encryption and decryption modules. Meanwhile, the work can be enhanced for other data files like video, audio, text. Similarly, the steganography modules can be developed for 3D images.

References

[1] G. A. Francia, M. Yang and M. Trifas, "Applied image processing to multimedia information security", Proc. Int. Conf. Image Analysis and Signal Processing, **(2009)**.

[2] G. J. Cancelli, B. M. Doerr and I. J. Cox, "A comparative study of steganalyzers", Proceedings of IEEE 10th Workshop on Multi- media Signal Processing, MMSP, **(2008)**.

[3] J. Fridrich, D. Soukal and M. Goljan, "Maximum likelihood estimation of length of secret message embedded using PMK steganography in spatial domain", Proceedings of IST/SPIE Electronic Imaging: Security, Steganography and Watermarking of Multimedia Contents, **(2005)**.

[4] K. B. Raja, C. R. Chowdary, K. R. Venugopal and L. M. Patnaik, "A secure image steganography using LSB, DCT and compression techniques on raw images", Proceedings of IEEE 3rd International Conference on Intelligent Sensing and Information Processing, ICISIP, **(2005)**; Bangalore, India.

[5] V. M. Potdar, S. Han and E. Chang, "Fingerprinted secret sharing steganography for robustness against image cropping attacks", Proceedings of IEEE Third International Conference on Industrial Informatics (INDIN), **(2005)**; Perth, Australia.

[6] X. X. Ping and T. Zhang, "Improved LSB matching steganography resisting histogram attacks", Proceedings of the IEEE International Conference on Computer Science and Information Technology, **(2010)**.

[7] S. B. Sadkhan, "Cryptography: current status and future trends", Proceedings of IEEE International Conference on Information & Communication Technologies: From Theory to Applications, **(2004)**; Damascus, Syria.

[8] Gollmann, Computer security, John Wiley & Sons Publishers, **(1999)**.

[9] J. Fridrich, M. Goljan and D. Hogeg, "Steganalysis of JPEG images: breaking the F5 algorithm", Proceedings of Information Hiding: Fifth International Workshop, IH Noordwijkerhout, The Netherlands, Lecture Notes in Computer Science, **(2002)**; Springer.

[10] S. A. Su, A. Lin and J. C. Yen, "Design and realization of a new chaotic neural encryption/decryption network", Proc. IEEE Asia-Pacific Conf, Circuits and Systems, APCCAS, **(2000)**.

[11] S. Liu, H. Yao and W. Gao, "Steganalysis of data hiding techniques in wavelet domain", Proc. of Int. Conf. on Information Technology: Coding and Computing, **(2004)**.

[12] D. Lou and J. Liu, "Steganographic method for secure communication", Computer and Security, **(2002)**.

[13] F. A. P Petitcolas, R. J. Anderson and M. G. Kuhn, "Information hiding A survey", Proceedings of the IEEE Special Issue on Protection of Multimedia Content, **(1999)**.

[14] Li, "A survey on image steganography and steganalysis", Journal of Information Hiding and Multimedia Signal Processing, vol. 2, no. 2, **(2011)**, pp. 142-172.

[15] H. B. Burk, "Comparing artificial neural networks to other statistical methods for medical outcome prediction", IEEE International conference, **(1994)**.

[16] M. Owens, "A discussion of covert channels and steganography", SANS Institute, **(2002)**.

[17] X. Kong, Z. Wang and X. You, X, "Steganalysis of palette images: attack optimal parity assignment algorithm", Proceedings of Fifth IEEE International Conference on Information, Communications and Signal Processing, **(2005)**.

[18] C. C. Chang, C. Y. Lin and C. S. Tseng, "Secret image hiding and sharing based on the (t,n) threshold", Fundamental Informaticae, vol. 76, **(2007)**, pp. 399-411.

[19] R. Z. Wang and C. H. Su, "Secret image sharing with smaller shadow images", Pattern Recognition, Lett., vol. 27, no. 6, **(2006)**, pp. 55l–555.

[20] A. T. Jamil, "Steganography: The art of hiding information is plain sight", IEEE Potentials, vol. 18, no. 1, **(1999)**.

[21] H. Wang and S. Wang, "Cyber warfare: Steganography vs. steganalysis", Communications of the ACM, vol. 47, no. 10, **(2004)**, pp. 76-82, 127.

[22] M. E. Whitman and H. M. Mattord, "Principles of information security", Thomson Course Technology Publishers, **(2003)**.

[23] N. F. Johnson and S. Jajodia, "Steganalysis: The investigation of hidden information", Proceedings of the IEEE Information Technology Conference, **(1998)**, pp. 113-116.

[24] A. Shou and L. Wi, Analysis of Contrasting Neural Network with Small-World Network, **(2008)**, pp. 57 – 60.

[25] A. T. Kondo, Feedback GMDH-type neural network using prediction error criterion and its application to 3-dimensional medical image recognition, **(2008)**, pp. 1050 – 1055.

[26] X. Hau, Research of model of Quantum Learning Vector Quantization Neural Network, vol. 8, **(2011)**, pp. 3893 – 3896.

[27] Y. M. Zing, Study on Characteristic of Fractional Master-Slave Neural Network, vol. 2, **(2012)**, pp. 498 – 501.

[28] K. Shihab, "A backpropagation neural network for computer network security", in Proc. Journal of Computer Science, vol. 2, **(2006)**, pp. 710-715.

[29] R. K. Munukur and V. Gnanam, "Neural network based decryption for random encryption algorithms", Proc. 3rd Int. Conf. Anti-counterfeiting, Security, and identification in Communication, ASID, **(2009)**.

[30] L. P. Yee and L. C. De Silva, "Application of multilayer perceptron network as a one-way hash function", Proc. Int. Conf. Neural Networks, IJCNN, **(2002)**.

[31] P. Y. Lin and C. S. Chan, "Invertible secret image sharing with steganography", Pattern Recognition Letters, vol. 31, **(2010)**, pp. 1887–1893.

[32] C. C. Chang, C. C. Lin, C. H. Lin and Y. H. Chen, "A novel secret image sharing scheme in color images using small shadow images", Inform. Sci., vol. l78, no. 11, **(2008)**, pp. 2433– 2447.

[33] D. Wang, L. Zhang, N. Ma and X. Li, "Two secret sharing schemes based on Boolean operations", Pattern Recognition, vol. 40, no. l0, **(2007)**, pp. 2776–2785.

[34] C. C. Chang, C. Y. Lin and Y. Z. Wang, "New image steganographic methods using run-length approach", Information Sciences, vol. 176, no. 22, **(2006)**, pp. 3393–3408.

[35] D. Artz, "Digital steganography: Hiding data within data", IEEE Internet Computing Journal, vol. 5, no. 3, **(2001)**, pp. 75-80.

[36] E. T. Lin and E. J. Delp, “A review of data hiding in digital images”, Proceedings of the Image Processing, Image Quality, Image Capture Systems Conference, PICS, the Society for Imaging Science and Technology, **(1999)**.

[37] C. C. Thien and J. C. Lin, “An image-sharing method with user-friendly shadow images”, IEEE Transactions on Circuits and Systems for Video Technology, vol. 3, no. 12, **(2003)**, pp. 1161–1169.

[38] C. C. Thien and J. C. Lin, “Secret image sharing”, Computer Graphics, vol. 26, no. l, **(2002)**, pp. 765–770.

[39] C. C. Lin and W. H. Tsai, “Secret image sharing with steganography and authentication”, Journal of Systems & Software, vol. 73, no. 3, **(2004)**, pp. 405–414.

[40] C. C. Chang and R. J. Hwang, “Sharing secret images using shadow codebooks”, Information Sciences, vol. 111, no. 1–4, **(1998)**, pp. 335–345.

[41] J. B. Feng, H. C. Wu, C. S. Tsai and Y. P. Chu, “A new multi-secret images sharing scheme using Largrange’s interpolation”, The Journal of Systems & Software, vol. 76, no. 3, **(2005)**, pp. 327–339.

[42] A. Shamir, “How to share a secret”, Communications of the Association for Computing Machinery, **(1979)**, pp. 612–613.

[43] M. Naor and A. Shamir, “Visual cryptography. Advances in Cryptology- EuroCrypt”, In: LNCS, vol. 950, **(1995)**, pp. 1–2.

[44] R. Lukac and K. N. Plataniotis, “Digital image indexing using secret sharing schemes: a unified framework for single-sensor consumer electronics”, IEEE Transactions on Consumer Electronics, vol. 51, no. 3, **(2005)**, pp. 908–916,

[45] C. C. Chang and I. C. Lin, “A new (t, n) threshold image hiding scheme for sharing a secret colour image”, Proceedings of International Conference on Communication Technology, **(2003)**.

[46] C. C. Chang, Y. P. Hsieh and C. H. Lin, “Sharing secrets in stego images with authentication”, Pattern Recognition, vol. 4l, no. l0, **(2008)**, pp. 3l30–3l37.

[47] C. N. Yang, T. S. Chen, K. H. Yu and C. C. Wang, “Improvements of image sharing with steganography and authentication”, J. Syst. Software, vol. 80, no. 7, **(2007)**,pp. l070–l076.

[48] Y. C. Tsai, H. L. Hu Yeh, “Reversible image hiding scheme using predictive coding and histogram shifting”, Signal Processing, vol. 89, no. 6, **(2009)**, pp. 1129–1143.

[49] Z. Li, X. Chen, X. Pan and X. Zeng, “Lossless data hiding scheme based on adjacent pixel difference”, Proceedings of the International Conference on Computer Engineering and Technology, **(2009)**.

[50] Y. Y. Lin and R. Z. Wang, “Improved invertible secret image sharing with steganography”, IID-MSP, **(2011)**, pp. 93-96.

[51] K. C. Lee, “Hybrid neural network-driven reasoning approach to bankruptcy prediction: comparison with MDA, ACLS, and neural network”, vol. 3, **(1994)**, pp. 1787 – 1792.

[52] Y. S. Wu, C. C. Thien and J. C. Lin, “Sharing and hiding secret images with size constraint”, Pattern Recognition, vol. 37, no. 7, **(2004)**, pp. l377–l385.

[53] K. H. Jung and K. Y. Yoo, “Data hiding method using image interpolation”, Computer Standards and Interfaces vol. 31, no. 2, **(2009)**, pp. 465–470.

[54] A. Knight, “Basics of MATLAB and beyond”, CRC-press-LLC, **(2000)**,

[55] W. Zhou, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity”, Image Processing, IEEE Transactions on, vol. 13, **(2004)**, pp. 600-612.

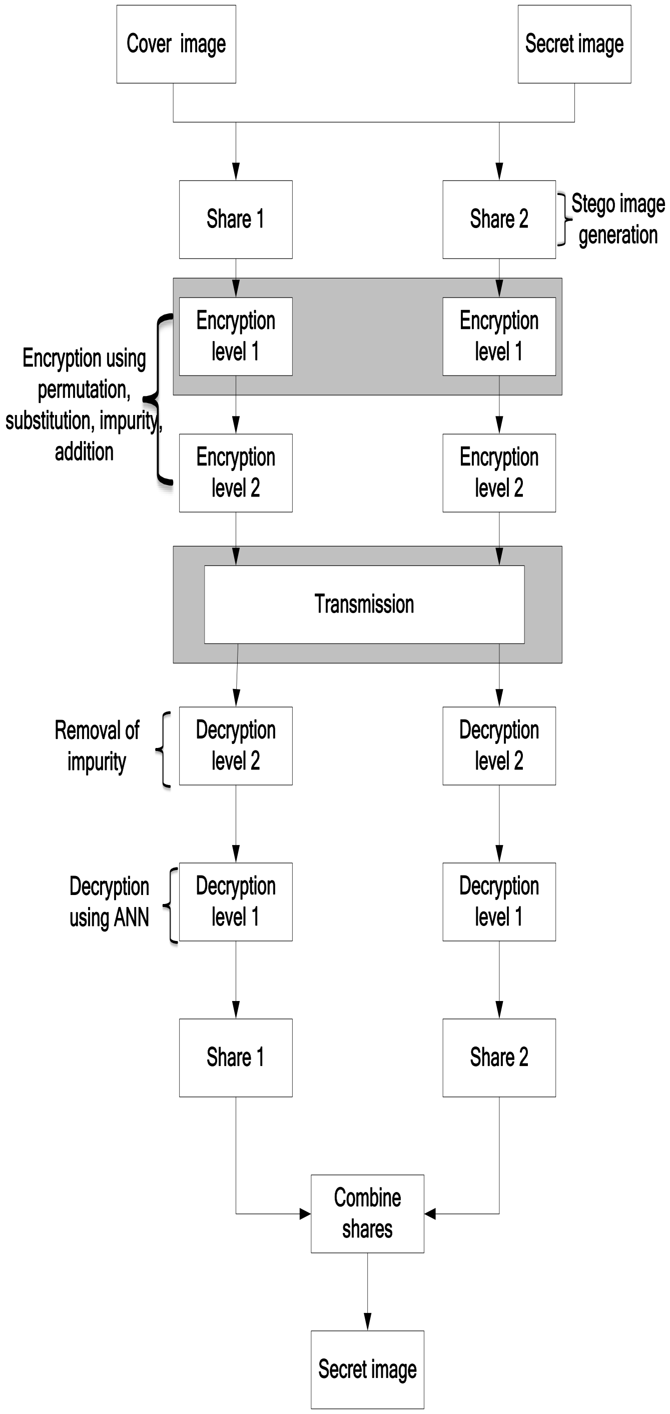


Figure 1. Overall System Architecture

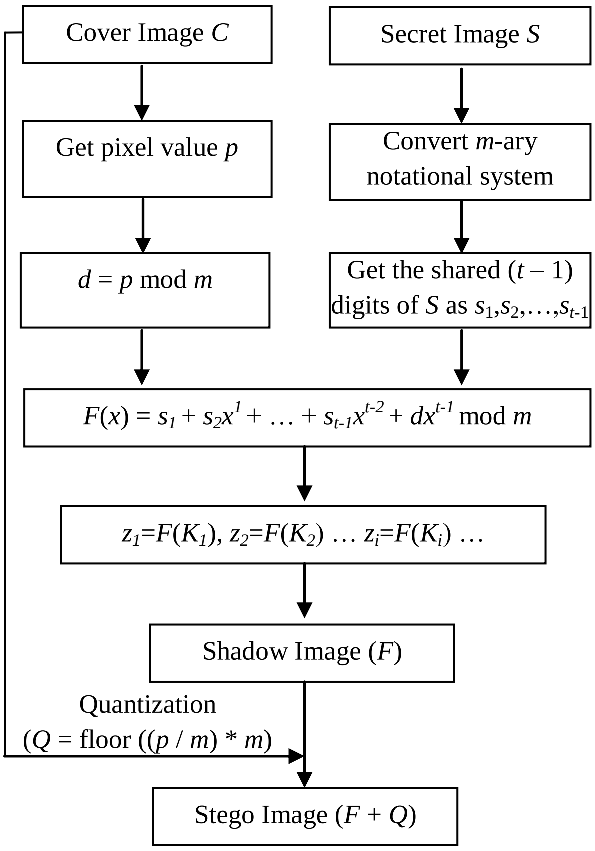

Cover Image C
Secret Image S
Get pixel value p
Convert m-ary notational system
$d = p \bmod m$
Get the shared $(t – 1)$ digits of S as $s_1,s_2,\ldots,s_{t-1}$
$F(x) = s_1 + s_2x^1 + \ldots + s_{t-1}x^{t-2} + dx^{t-1} \bmod m$
$z_1=F(K_1), z_2=F(K_2) \ldots z_i=F(K_i) \ldots$
Shadow Image (F)
Quantization
(Q = floor ((p / m) * m)
Stego Image ($F + Q$)

Figure 2: Stego Image Generation using shadow Image and Quantization

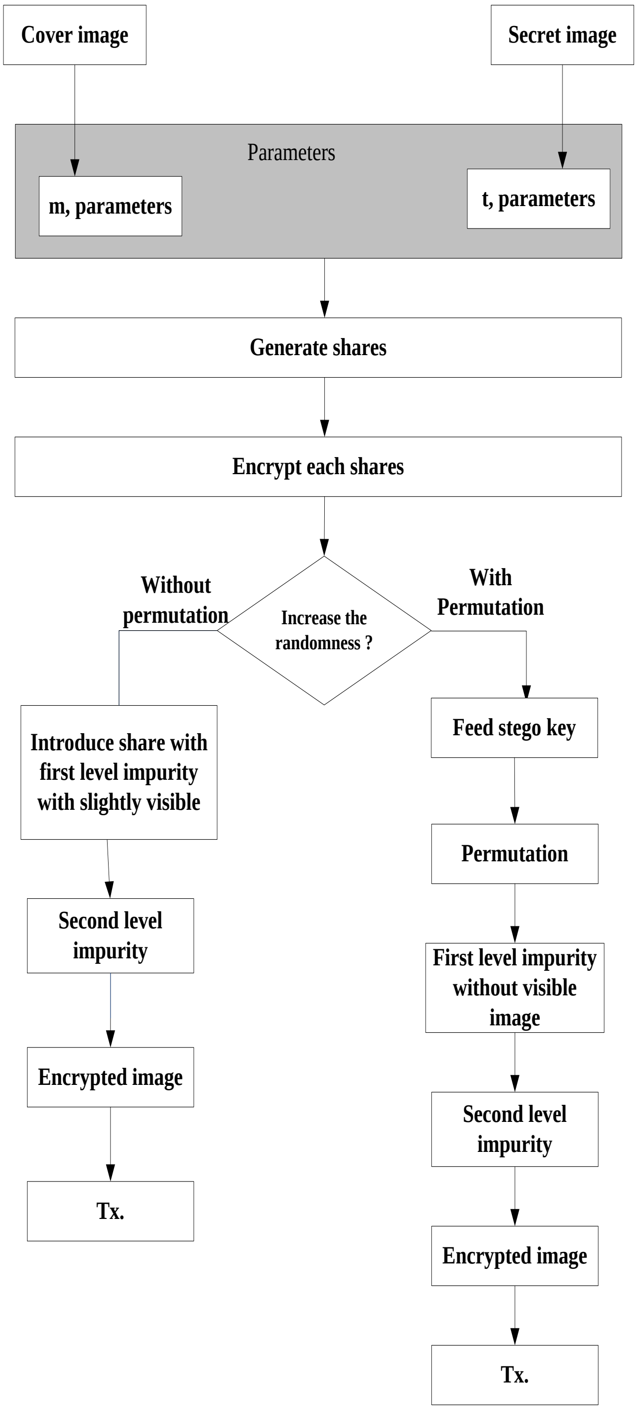


Figure 3. Process of Encryption (level 1 and level 2)

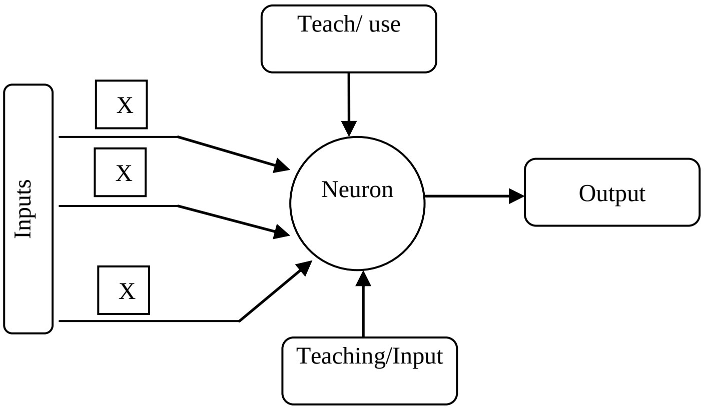


Figure 4(a). Basic Artificial Neural Network

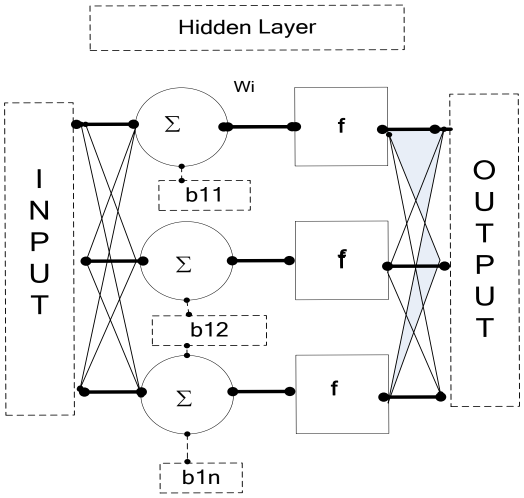


Figure 4(b). Three Layer Backpropagation Neural Net

where, Wi= weight of the ith layer, f=Activation function, bij= bias of the jth neuron in the ith layer

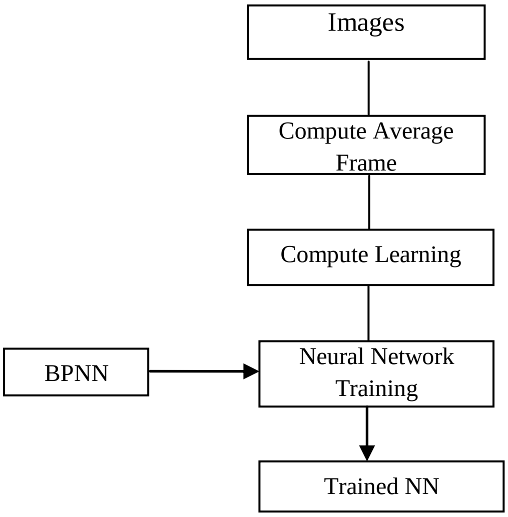


Figure 4(c). Neural Network Training Process

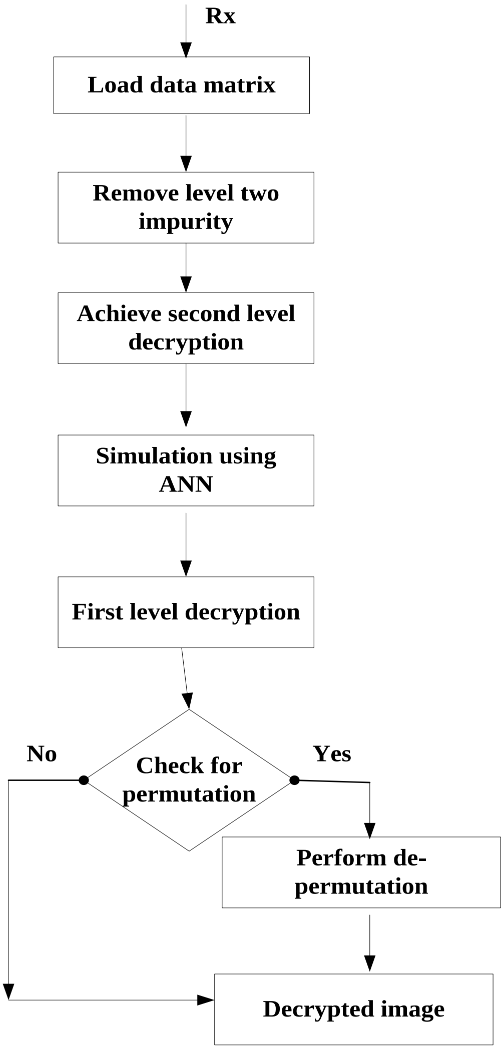


Figure 5. Process of Decryption (level 1 and level 2)

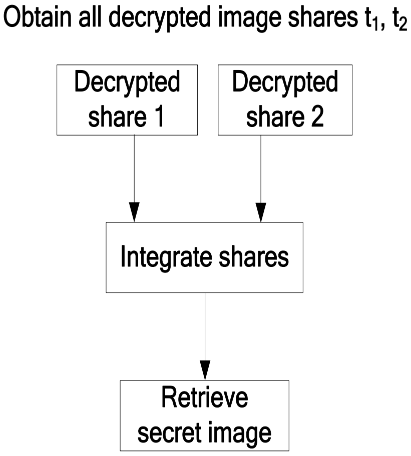
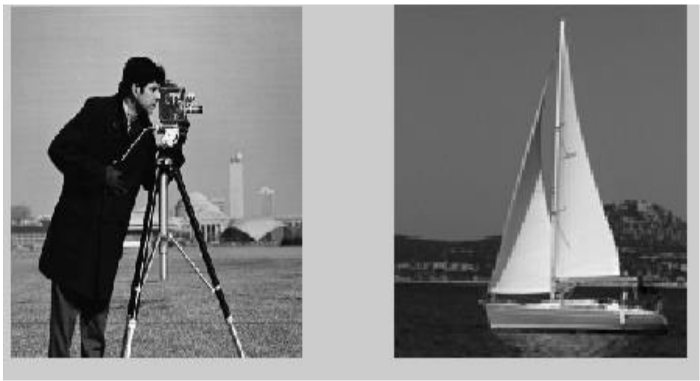


Figure 6. Secret Image Retrieval



(a) Cover Image (b) Secret Image

Figure 7. Input Images

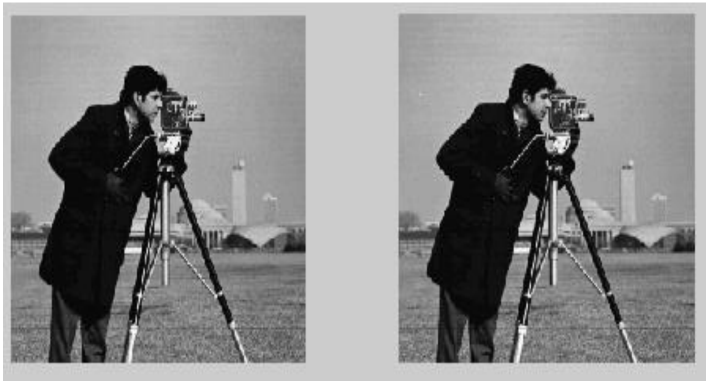


Figure 8. Creation of Stego Images

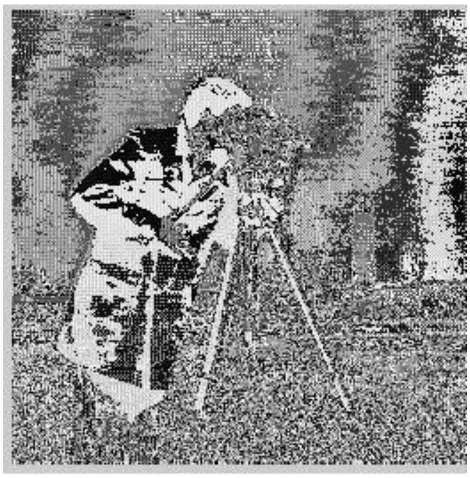


Figure 9. First Level Encryption without using Key

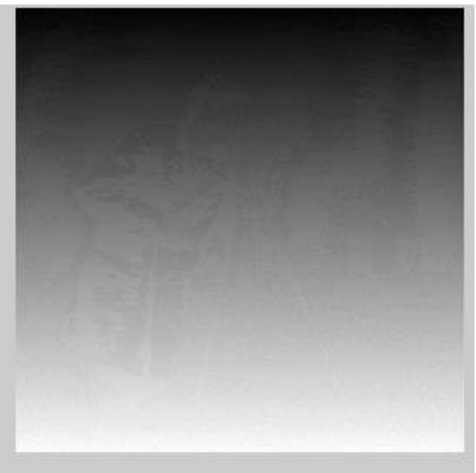


Figure 10. Second Level Encryption

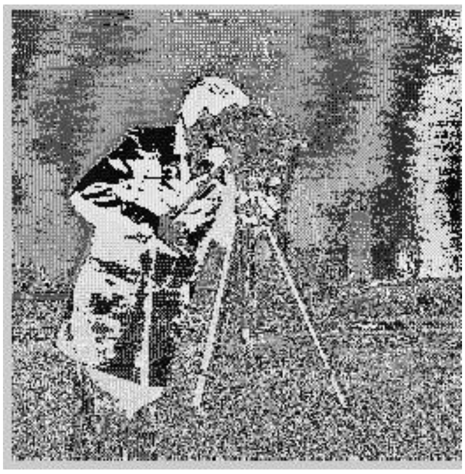


Figure 11. Second Level Decryption

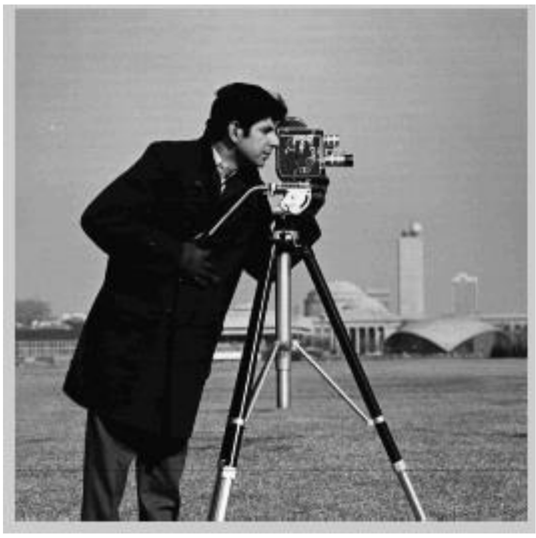


Figure 12. First Level Decryption

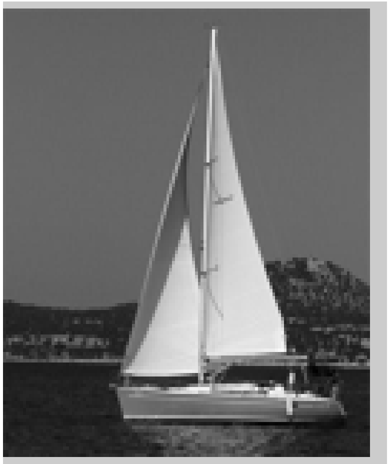


Figure 13. Retrieved Secret Image

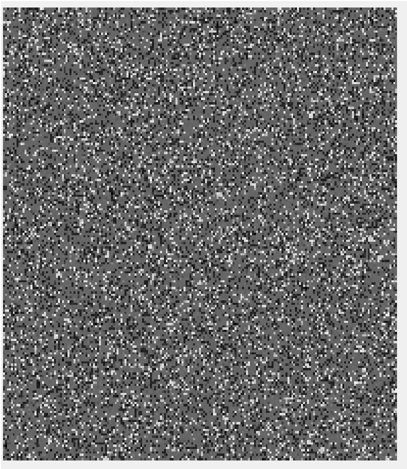


Figure 14. First Level Encryption using Key

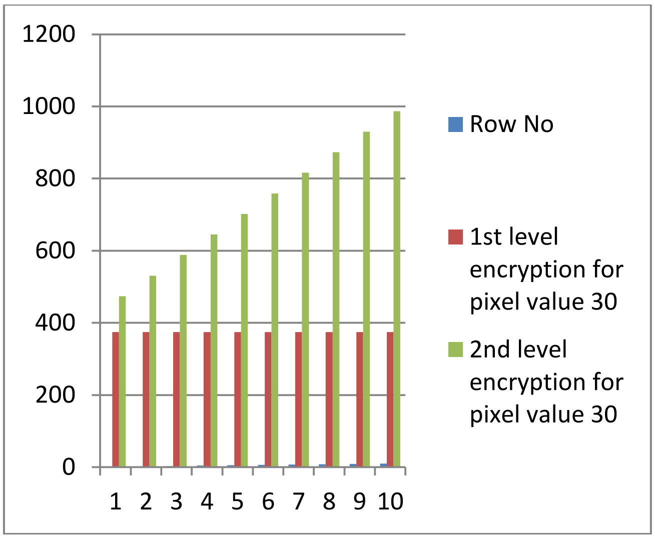
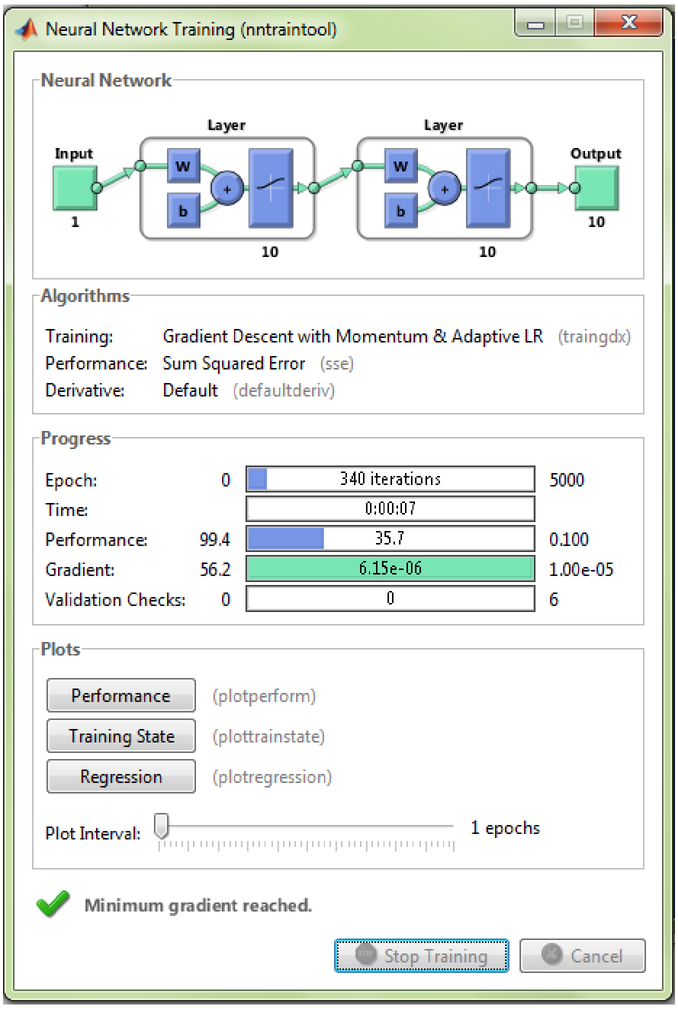


Figure 15. Displacement of the Pixel Values



Training: Gradient Descent with Momentum & Adaptive LR (traingdx)
Performance: Sum Squared Error (sse)
Derivative: Default (defaultderiv)

```
Epoch:               0     340 iterations     5000
Time:                      0:00:07
Performance:      99.4     35.7               0.100
Gradient:         56.2     6.15e-06           1.00e-05
Validation Checks:   0     0                  6
```

Plot Interval: 1 epochs

✓ Minimum gradient reached.

Figure 16. Neural Network Training

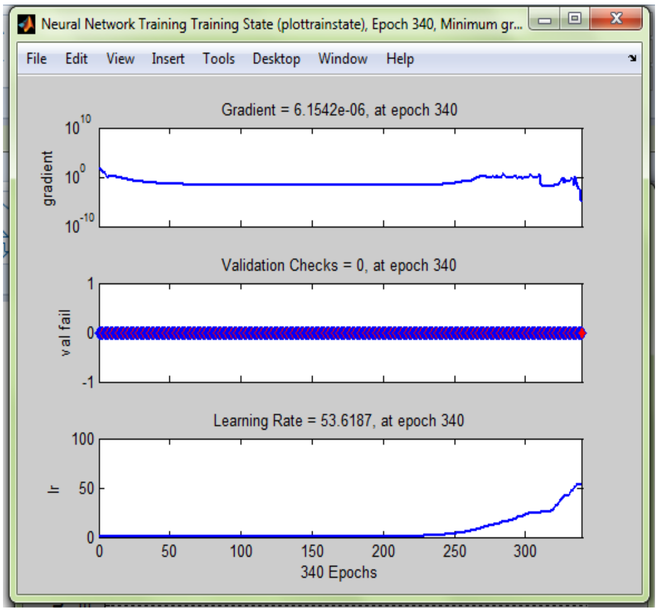

Neural Network Training Training State (plottrainstate), Epoch 340, Minimum gr...
File Edit View Insert Tools Desktop Window Help
Gradient = 6.1542e-06, at epoch 340
gradient
10^{10}
10^{0}
10^{-10}
Validation Checks = 0, at epoch 340
val fail
1
0
-1
Learning Rate = 53.6187, at epoch 340
lr
100
50
0
0 50 100 150 200 250 300
340 Epochs

Figure 17. Training States

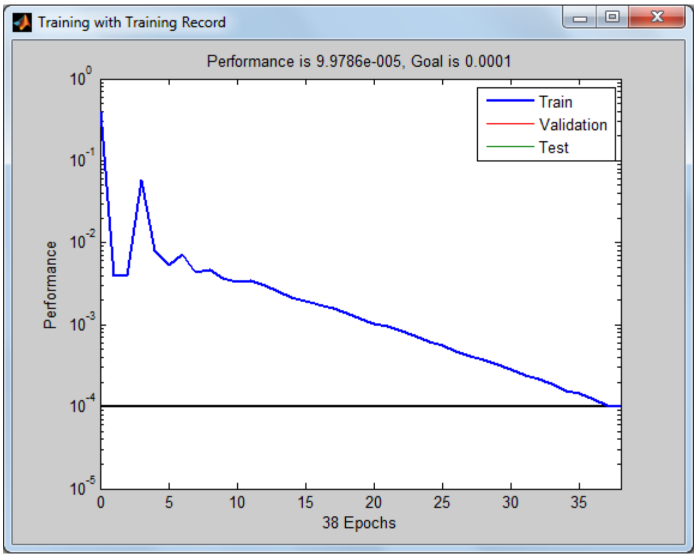


Figure 18. Network Performance

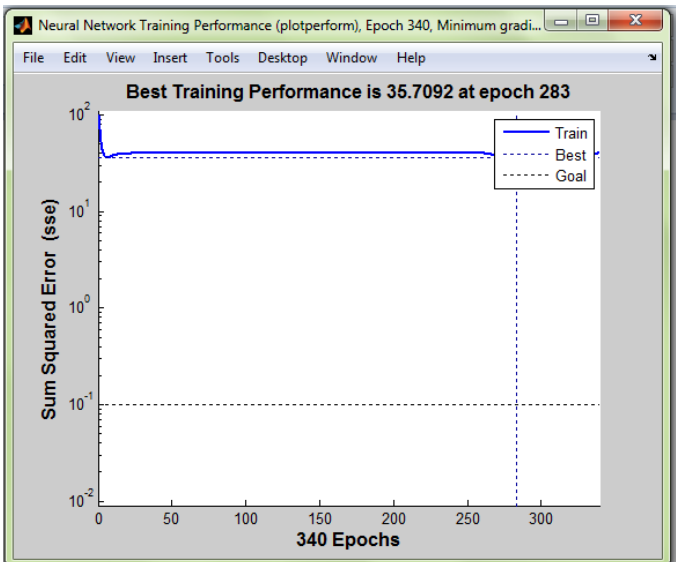

Neural Network Training Performance (plotperform), Epoch 340, Minimum gradi...
File Edit View Insert Tools Desktop Window Help
Best Training Performance is 35.7092 at epoch 283
Sum Squared Error (sse)
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
0
50
100
150
200
250
300
340 Epochs
Train
Best
Goal

Figure 19. Graph between epochs and sum squared error

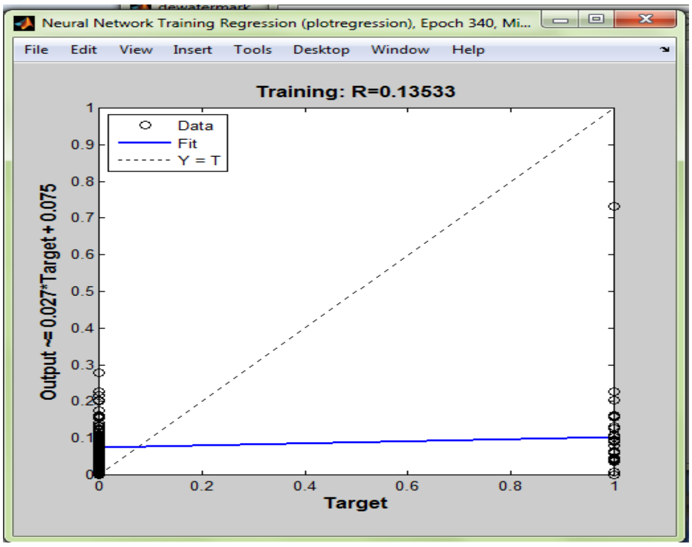

Neural Network Training Regression (plotregression), Epoch 340, Mi...
File Edit View Insert Tools Desktop Window Help
Training: R=0.13533
Output ~= 0.027*Target + 0.075
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0.2
0.4
0.6
0.8
Target
Data
Fit
Y = T

Figure 20. Regression Plot

Table 1. PSNR Values of Stego Images *t* = 2 for Different Prime Numbers

Prime number	PSNR(dB)	
	Share 1	Share 2
7	40.9395	40.1369
11	37.1050	37.8804
13	33.3417	34.3774
17	31.7241	31.4550
19	31.5491	31.1716

Table 2. PSNR with Encryption & Decryption Time for Different Shares

Test images	PSNR (dB)		Time (seconds)		
	Share 1	Share 2	Encryption time		Decryption time
Cameraman.tif	41.1356	40.2022	Share 1	0.5094	9.9686
			Share 2	0.4186	9.7737
Lena.jpg	40.6178	39.7248	Share 1	0.5949	10.6687
			Share 2	0.6147	9.9576
Peppers.png	40.7337	39.5728	Share 1	0.6047	10.2002
			Share 2	0.4708	9.7397
Boat.jpg	40.9395	40.1369	Share 1	0.4983	9.8370
			Share 2	0.4249	9.7527

Table 3. First Level of Encryption

Original pixel value	Pixel value after 1st level encryption
1	165
12	437
54	312
95	554
100	123
144	414
201	177
230	483

Table 4. Second Level Encryption

Row number	Original pixel value	Pixel value after 1st level encryption	Pixel value after 2nd level encryption
1	51	168	268
2	51	168	325
11	51	168	838
18	51	168	1237
30	51	168	1921

Table 5. Encryption/Decryption Time of Image of Different Size

Image file in pixel	Time (seconds)		
	Encryption time		Decryptio n time
256×256×1	Share 1	0.5366	10.6102
	Share 2	0.4633	9.8551
100×100×1	Share 1	0.6569	2.7841
	Share 2	0.5775	2.1826
48×48×1	Share 1	0.6004	1.4192
	Share 2	0.8438	1.1083

Table 6. PSNR, SSIM and MSE Comparisons of Related Secret Image Sharing Mechanisms

Technique	PSNR	SSIM	MSE
Y. S. Wu, C. C. Thien and J. C. Lin [52]	34 dB	0.949	0.0069
K. H. Jung, K.Y. Yoo [53]	35 dB	0.953	0.0058
C. C. Thien, J. C. Lin [38]	37 dB	0.948	0.0052
C. C. Lin, W. H. Tsai [39]	39 dB	0.971	0.0041
C. C. Chang, C. Y Lin, C. S Tse [18]	39 dB	0.970	0.0033
Lin, Yung-Yi., Wang, Ran-Zan [50]	39.2 dB	0.974	0.0028
Proposed Work	40.5 dB	0.993	0.0021