NEURAL NETWORK-BASED DOUBLE ENCRYPTION FOR JPEG2000 IMAGES

Qurban Ali Memon
College of Engineering
United Arab Emirates University

qurban.memon@uaeu.ac.ae

ABSTRACT

The JPEG2000 is the more efficient next generation coding standard than the current JPEG standard. It can code files with less visual loss, and the file format is less likely to be affected by system file or bit errors. On the encryption side, the current 128-bit image encryption schemes are reported to be vulnerable to brute force. So there is a need for stronger schemes that not only utilize the efficient coding structure of the JPEG2000, but also apply stronger encryption with better key management. This research investigated a two-layer 256-bit encryption technique proposed for the JPEG2000 compatible images. In the first step, the technique used a multilayer neural network with a 128-bit key to generate single layer encrypted sequences. The second step used a cellular neural network with a different 128-bit key to finally generate a two-layer encrypted image. The projected advantages were compatible with the JPEG2000, 256-bit long key, managing each 128-bit key at separate physical locations, and flexible to opt for a single or a two-layer encryption. In order to test the proposed encryption technique for robustness, randomness tests on random sequences, correlation and histogram tests on encrypted images were conducted. The results show that random sequences pass the NIST statistical tests and the 0/1 balancedness test; the bit sequences are decorrelated, and the histogram of the resulting encrypted images is fairly uniform with the statistical properties of those of the white noise.

Keywords: JPEG2000 image, neural network, random sequence, cellular neural network, block cipher.

Received: 9 March 2016 Accepted: 20 October 2016

INTRODUCTION

In the current multimedia environment, security and protection of data is essential to fulfil vendor rights and client requirements. As Internet is evolving so are the tools, applications and threats. People are fascinated by recent applications to simplify their professional work (Memon & Khoja, 2009), and secure their data access (Memon, Akhtar, & Aly, 2007). With respect to the storage or transfer of images and videos, organizations prefer to use the encryption of image/video data as an alternative to other approaches. Nowadays, a great deal of money is being invested to increase the security level of the image data transmitted or stored over public channels, and a lot of research is are being reported in this field.

Since the encryption process is a one way function, the artificial neural networks are claimed to be best suited for this purpose as they possess features like high security, no distortion and their ability to perform nonlinear input-output characteristics. Thus, the need for key exchange can be eliminated, which otherwise is a perquisite for most of the algorithms used today. As an example, Lian (2007) investigated the neural network properties to propose a low-cost authentication for images or videos. The author claimed that the approach has the embedded ability to detect whether the data is modified maliciously. The author finally highlighted several open issues in this field like: which property of neural networks has to be exploited for data protection; which neural network models are suitable for data protection; and the learning ability of neural networks. In another work Munukur and Gnanam, (2009), used neural network in the receiver for the purpose of decryption by exploiting back propagation algorithm in the receiver to train it with a 12-bit cipher text as an input, and a 8-bit plain text being the target output. The plain text at the input also included some impurity based on some pre-determined key to mislead any possible eavesdropper.

Chaos has also been investigated in combination with neural networks. As an example, a neural network was proposed by Lian (2009), which was composed of a chaotic neuron layer and a linear neuron layer. The network was then used to construct a block cipher that encrypted the plaintext into a cipher text using a key in order to construct a chaotic neural network-based block cipher with good computing security. In that, the block cipher involved two processes: a diffusion process implemented by a chaotic neuron layer and a confusion process implemented by a linear neuron layer. These processes were iterated a number of times to improve encryption complexity. In another work that used chaos and neural network together, Lian (2011) exploited the neural

network structure to process much media contents in a parallel manner. The scheme combined encryption and watermarking together. The encryption part used random sequences generated from the chaos system with the help of an encryption key. This key was then used to encrypt the media contents with a neural network structure. However, the apparent disadvantage in this scheme was that more sub-keys needed to be transmitted to the receiver. Similarly, the work of Joshi, Udupi, and Joshi (2012) targeted the securing of image data transmission using a randomness algorithm by introducing confusion in the data, and the addition of impurities to misguide the cryptanalyst. In another research Bigdeli, Farid, & Afshar (2012a), proposed an image encryption/ decryption algorithm based on chaotic neural network. The employed network comprised two layers: chaotic neuron layer (CNL) and permutation neuron layer (PNL), each with three layers. The approach used a 160-bit-long authentication code to generate initial conditions and the parameters of both layers. The overall process was repeated several times to make it robust and increase complexity. The proposed method used two more keys where a slight mismatch in one of them will fail in successfully decrypting an image. In another work, the same authors (Bigdeli et al., 2012b) proposed an encryption method based on a hybrid chaos-based encryption algorithm. The algorithm employed permutation–diffusion architecture that used chaotic control parameters for permutation. These control parameters for the permutation stage were generated by a logistic map. In the diffusion stage, another chaotic logistic map with different initial conditions and parameters was used to generate initial conditions for a hyper-chaotic Hopfield neural network to generate a key stream for the image homogenization of the shuffled image. Zirra (2011) tried techniques different from the chaotic neural networks, where scrambling was used to transform the information into a set of linear equations and deciphering was achieved by solving the systems of the linear equations together with principles of the delta encoding scheme, a formula and a lookup table. For further reading on this subject, readers are encouraged to refer to Memon (2014 and 2006).

Cellular neural networks (CNNs) provide both continuous time and local interconnection features. The basic circuit is called a cell, which contains linear and non-linear elements and sources. Cells can be characterized as multiple input-single output nonlinear processors all described by one, or one among several different, parametric functionals (Chua & Yang, 1988). A state variable characterizes a cell itself, and the notion of distance implies that the network is intrinsically defined in space; and typically a 1-, 2- or 3- dimensional space is considered. The neighborhood adjacent cells only connect directly to each other, but are indirectly affected by other cells due

to propagation effects caused by the continuous time dynamics of the system. The cell topology can be considered as rectangular, triangular, hexagonal or a 3-dimensional array realized as a stack of 2-dimensional layers. Cells may be identical or different, otherwise with typically a small neighborhood. Though cells are characterized by adjacent neighbors due to local nature, they are assigned some global properties due to continuous time features.

Because of afore-mentioned properties, cellular neural networks have received greater attention by researchers. For example, Xu et al. (2005) explored the criteria for the existence of a unique equilibrium point and its global asymptotic stability of continuous delayed CNNs to offset oscillations caused by the existence of time delays in CNNs. Likewise, Yi et al. (2015) proposed two kinds of cellular neural networks based on mem-elements -- MC-CNN lets a mem-capacitor replace the conventional linear capacitor of a cellular neural network cell, while EM-CNN is economical in fabricating cost for better implementation of CNN. In another work, a new model of CNNs with transient chaos was proposed by Wang et al. (2007), who proposed adding negative self-feedback once dynamic equations have been transformed into discrete time, resulting in transient chaos. Peng, Zhang, Liao, (2009) showed that as the number of cells increases beyond four, a hyper chaotic behavior is observed in the cellular neural network, and thus requires more keys to describe the state of the system.

Due to the dynamics in CNNs, cellular neural networks have found applications in image processing, pattern recognition, classification, and combinatorial optimization amongst others. CNN has typically one type of unit processor in one layer, however, some applications require the collaboration of distinct dynamics. Ayhan and Yalcin (2011) proposed the randomly reconfigurable cellular neural network to mimic the joint effort of distinct types of neurons. In the biomedicine area, recently a new algorithm using fuzzy cellular neural network has been proposed Shitong and Min, (2006) to automatically detect white blood cells by developing a complete contour around cell.

To summarize, much research has appeared in literature to address the encryption of data either before transmission or for storage. The issues that are still being investigated are complexity, the robustness in the presence of malicious attack, as well as compatibility with current standards. In this research, neural network structures are examined in combination with wavelet transform for image encryption and decryption. The motivation behind the use of wavelets is that current image transmission and storage was mostly preferred using the JPEG2000, which is a new evolving standard for image transmission and coding. This is motivated by the fact that the JPEG2000 is

better at compressing images (up to 20 per cent plus), and that it can allow an image to be retained without any distortion or loss (Nguyen & Marpe, 2014). The paper is structured as follows. In the next section, background information about some of the steps in the proposed approach is briefly discussed. Following that, the proposed approach is presented that describes the key parts of the solution. The following section analyzes the performance of the approach with regard to key space, NIST statistical test, histogram, correlation coefficient and the 0/1 balancedness test. In the end, conclusions are presented followed by references.

BACKGROUND

This section presents background information about some needed steps, which are required to be executed in the proposed approach. Each of these is briefly discussed below.

Bit Plane Decomposition: By bit plane decomposition, it is meant that an image $p(x, y)$ of size NxN with 256 gray levels is decomposed into eight (~ $\log_2$ (number of gray levels)) binary images. Similarly, a 16-bit data will have sixteen binary images. Each of these binary images is called a bit plane with sets of bits corresponding to a given bit position in each of the binary numbers representing the gray level. The first bit plane will contain the set of most significant bit of each gray level value; likewise the last bit plane will contain the set of least significant bits. Thus, the first bit plane gives the roughest but most critical approximation of the pixel, whereas each later bit plane improves this approximation as we continue to add on successive bit planes.

XOR operation: This is one of the Boolean operations that can be done on binary images. Typically, these operations are known as masking functions, where $p(x, y)$ is a binary image and a masking binary pattern is chosen to mask its bits. In the case of the XOR operator, it inverts bits. It produces output value of logical one, whenever $p(x, y)$ and the masking bit are different. In other words, it is used to highlight differences in a binary image. Likewise, XNOR is used to highlight similarities in a binary image. The XOR operation is commutative, associative and self-inverse. For example, we want to XOR 8-bit gray level values of 166 and 210 together. 166 is 10100110 in binary and 210 is 11010010 in binary. The result of XOR in bitwise operation is 01110100 in binary or 116 in decimal. Interestingly, if this result is XORed with 210 again, we get back the original value of 166. This reversible property of XOR is used in encryption, where masking is used in the transmitter and unmasking is done on the receiver side.

PROPOSED APPROACH

In this section, we present the proposed approach. Consider plain image $p(x, y)$ of size NxN. The first step in JPEG2000 is to apply n-level wavelet transform to the image. For the purpose of simplicity, we assume n=2. In wavelet transform decomposition, each level produces four frequency subbands of the input image, where each one is the quarter-size of the original. In the proposed approach, we do not apply encryption on these subbands directly; rather these subbands undergo bit plane decomposition to generate eight binary images for each subband. Depending upon complexity need, a set of these binary images is transformed into encrypted bit plane images. If desired, these encrypted subbands can then undergo the next step of the JPEG2000 encoder, or otherwise inverse wavelet transform is applied to generate the encrypted image. There are three variables that add complexity to the encryption process executed through random chaotic sequences: one is the number of levels the image undergoes wavelet decomposition; another is the number of subbands that are bit plane decomposed; and the third is the set of bit planes for encryption. The proposed approach is shown in Figure 1, where we have the XOR operation between the pseudo-random sequence generated by 8-4-2-1 chaotic neural network and the binary bits from the subband image pixels. We call this first step single encryption.

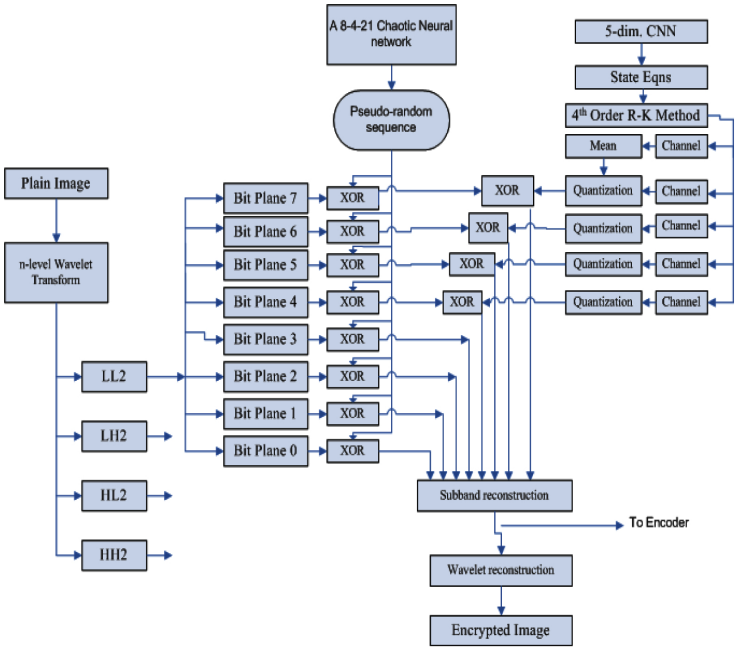

Figure 1. Proposed approach.

In order to generate the pseudo-random sequence, an 8-4-2-1 neural network, as shown in Figure 2, is employed to introduce non-linearity in generating a sequence. A 64-bit input key i.e. A= [A_1, A_2, A_3, ……, A_{64}] is applied at the input layer such that 8-bits enter at each node of the layer. The output of this layer can be written as:

$$B = f^{n0}(Aw_o + A_o, K_0) \quad (1)$$

where w_0 is the matrix of size 8x8 *i.e.*, w_0=[$w_{0,0}$, $w_{0,1}$ $w_{0,2}$ $w_{0,3}$ $w_{0,4}$ $w_{0,5}$ $w_{0,6}$ $w_{0,7;}$ $w_{1,0,\ldots\ldots;}$ $w_{7,7}$], A is the input vector, the bias is A_0 = [a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7], K_0 is the control parameter [k_0, k_1, $k_{2,\ldots\ldots,}$ k_7] and $n0$ is the random number generated by the key generator in the range $1 \leq n0 \leq 10$. The function f is the transfer function based on the piecewise linear chaotic map (PWLCM) (El Assad, et al., 2008) and is given by:

$$x(n) = f\big(x(n-1)\big) = \begin{cases} \dfrac{x(n-1)}{k} & if\ x(n-1) \in [0, k[\\ \dfrac{x(n-1)-k}{0.5-k} & if\ x(n-1) \in [k, 0.5[\\ f\big(1-x(n-1)\big) & if\ x(n-1) \in [0.5, 1] \end{cases}$$

where k ε [0, 0.5[and x(n) ε [0, 1]. x(0) and k are used as secret keys. For a dynamical system to generate the highest Lyapunov exponent, k is typically chosen to be 0.5.

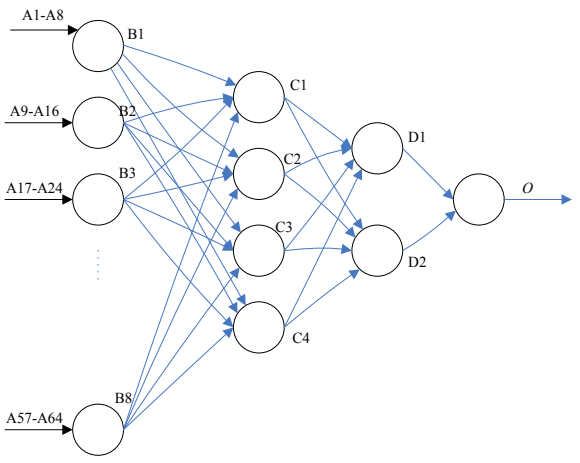

Figure 2. Generation of N-bit pseudorandom sequence for chaos.

The output of each layer becomes input to the next layer, apart from becoming input to that neuron itself. Continuing in the same fashion, the output of the remaining layers is calculated as follows:

$$C = f^{n1}(Bw_1 + B_o, K_1) \quad (2)$$

$$D = f^{n2}(Cw_2 + C_o, K_2) \quad (3)$$

$$O = f^{n3}(Dw_3 + D_o, K_3) \quad (4)$$

where the matrices w_1, w_2, w_3 have sizes equivalent to 4x8, 2x4 and 1x2; B_0, C_0, D_0 with sizes 4x1, 2x1, and 1x1; K_1, K_2, K_3 with sizes 4x1, 2x1, and 1x1, respectively. During iterations at each layer, the control parameters are also adjusted using the respective layer outputs in such a way that the respective range lies in [0.4, 0.6]; for example K_0=0.2xB+0.4 to get chaotic behavior. Like n_0, the values of n_1, n_2, and n_3 are obtained through key generation. Once the value of the output is obtained between 0 and 1, this value is normalized in the range 0-255. In order to enforce randomness, this normalized value is then compared with a threshold of 127 to obtain 0 or 1 in the sequence.

Key Generator: Many chaotic key generators exist but the one used in this research involves the 1-D cubic map (Djellit Ilhem and Kara Amel, 2006). It takes a 64-bit random key Key = [Key_1, Key_2, Key_3, Key_4] to calculate the initial conditions based on its 16-bit component (Key_i) such that $y(0) = \left(\sum \frac{Key_i}{2^{16}}\right) mod(1)$ of the 1-D cubic map and returns values of the map using iterations. The states of the cubic map are written as (Gao and Chen, 2008):

$$y(n+1) = \lambda y(n)\big(1 - y(n).y(n)\big) \quad (5)$$

where λ is typically set at 2.59 as a control parameter, and the state of equation is satisfied by $0 \leq y(n) \leq 1$. In order to generate initial conditions for the neural network, Equation (5) is first iterated 50 times and the values are discarded, and then iterated again to initialize w_0, w_1, w_2, w_3, A_0, B_0, C_0, D_0, K_0, K_1, K_2, K_3, n_0, n_1, n_2, n_3. In order for Equation (5) to provide randomness and reproducibility of the same initial conditions each time it is run even on different computing machines, it should be ensured that values like Key, λ set at 2.59, and initial iteration of 50 to discard values are used with the same precision arithmetic. Further test analysis for robustness of sequences is discussed in the performance analysis section.

Reproducibility: Reproducibility is the ability of an entire program to yield the same results each time it is run, either by the same person or a different one working independently. This means that, every time the code is run, it will produce the same results with a high degree of precision. Reproducibility is important for debugging and building confidence. Sometimes, a lack of reproducibility is termed as a shortcoming of any random generator. Thus, there are general conditions for the random generator for reproducibility:

1. One should use the same random number generator seed.
2. The model of the generating code shall not change.
3. The same initial values or conditions are used.
4. The precision of computation shall remain the same as originally used.
5. Avoid running macros or custom visual basic for application (VBA) functions, which do not exactly generate the same results from simulation to simulation.

There are some technical concerns with reproducibility like the portability of the code generator from one operating system to another or the use of parallelization involving the number of threads on the platform. The first one relates to precision, and the second one to the number of threads running for the same code. The only caution that should be exercised is that the same precision is to be used across platforms, and that one iteration of a random number generation model shall not refer directly or indirectly to another value in an independent thread. This same concern holds for multiple CPUs, that is, if the code is run on multiple CPUs or a CPU with multiple threads, the order of generating random numbers shall remain the same as that generated by using serial computation. With these constraints in mind, 100,000 bits of code sequences were generated on the same machine twice and once on another machine, and were found to be the same each time.

For the interest of the reader, the single level encryption achieved in step 1 is exemplified in Figure 3, where the original 'woman' image is shown followed by the wavelet decomposition to generate four subband images, each the quarter size of the original. In order to enter the next step, each of the subbands or the desired set of subbands are bit level decomposed to generate eight binary images. In Figure 3, eight binary images of only one subband image are shown. Once the pseudo-random sequence is generated, Figure 3 shows the XOR operation between one binary image and the pseudo-random sequence. Continued in this way, once all binary images are encrypted by the pseudo-random sequence, the binary images enter the reverse process to generate the desired encrypted subband(s) followed by inverse wavelet transform to generate the encrypted image.

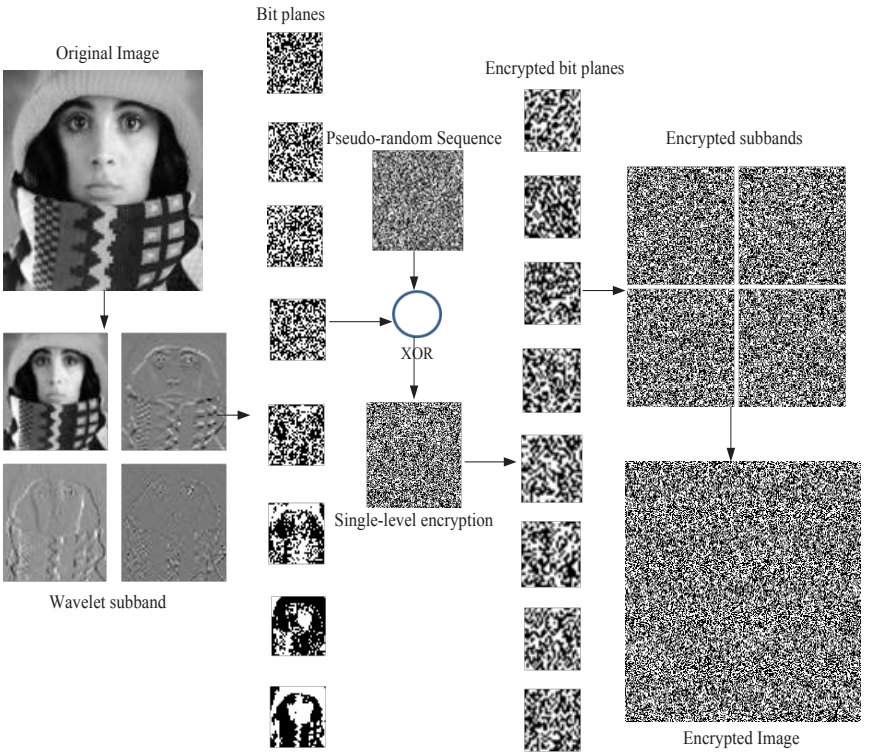

Figure 3. Sample view of single level encryption.

For the second step, a 5th order CNN model is used and its state equations are described as (Chua, Yang, 1988):

$$\frac{dx_j}{dt} = -x_j + a_j f(x_j) + \sum_{\substack{k=1 \\ k \neq j}}^{5} A_{jk} f(x_k) + \sum_{k=1}^{5} S_{jk} x_k + \tilde{I_j} \quad (6)$$

In Equation (6), the parameters are set as follows:

$\tilde{I}$ is a threshold and is generally set to 0; $a_4 = 202$, which means that output of the cell 4 affects the 4th cell and its influence is 202; $a_j = 0$ (for j=1,2,3,5); A_{jk}=0 means the output of the cell and its adjacent cells has no influence on the state of the cell except the fourth cell; S_{jk} represents the influence weight that k cell has on cell j: $S_{11}=S_{23}=S_{33}=1$; $S_{13}=S_{14}=S_{45}=$ S55=-1; $S_{12}=S_{15}=S_{21}=S_{24}=S_{25}=S_{34}=S_{35}=S_{51}=S_{52}=S_{54}$=-1; S_{22}=3; S_{31}=11; S_{32}=-12; S_{41}=92; S_{44}=-94; S_{53}=15

Based on these parameters, the state equations can be generated as follows:

$$\frac{dx_1}{dt} = -x_3 - x_4 \quad (7)$$

$$\frac{dx_2}{dt} = 2x_2 + x_3$$

$$\frac{dx_3}{dt} = 11x_1 - 12x_2$$

$$\frac{dx_4}{dt} = 92x_1 - 95x_4 - x_5 + 202f(x_4)$$

$$\frac{dx_5}{dt} = 15x_3 - 2x_5$$

The Lyapunov exponents (being important characteristics of a chaotic system) of Equation (7) are: 0.2953, 0.5285, 0.1264, -3.9205, -17.4382, which proves that the system is hyper chaotic. In order to solve these state equations, the classical fourth-order Runga-Kutta method (Kumar, et al., 1977) is used, as:

$$y_{n+1=}y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \quad (8)$$

where

$$k_1 = f(t_n, y_n);\ k_2 = f\left(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_1\right), k_3 = f\left(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_2\right), k_4 = f(t_n + h, y_n + hk_3)$$

In Equation (8), the initial values are set as: step size h=0.005, $x_1(0)$=0.1, $x_2(0)$=$x_3(0)$=$x_4(0)$=$x_5(0)$=0.2, and the number of iterations μ is set at 16384 to generate μ hyper chaotic values. Thus five channels of hyper chaotic sequences are generated with each channel having 16384 values. The mean 'm' of channel x_1 is calculated as follows:

$$x_{ij} = 0, \quad if\ x_{ij} \leq m$$

$$x_{ij} = 1, \quad if\ x_{ij} \geq m$$

where i=1,2,3,4,5 and j=1,2,3,4,….,16384. Thus four channels (x_1, x_2, x_3, and x_4) are quantified into four binary sequences (X_1, X_2, X_3, X_4). These four binary sequences are XORed with four most significant sequences from step 1 (single encryption) to generate four most significant doubly-encrypted sequences. For simulation purposes, the various parameters set to run the proposed system are shown in Table 1.

Table 1

Parameters for Simulation of the Proposed System

Parameter values
k=0.5
λ=2.59
$\tilde{I}$ =0
a_1=0; a_2=0; a_3=0; a_4=202; a_5=0
S_{11}=S_{23}=S_{33}=1; S_{13}=S_{14}=S_{45}= S_{55}=-1; S_{12}=S_{15}=S_{21}= S_{24}=-S_{25}=S_{34}=S_{35}= S_{51}=S_{52}=S_{54}=-1; S_{22}=3; S_{31}=11; S_{32}=-12; S_{41}=92; S_{44}=-94; S_{53}=15
h=0.005
$x_1(0)$=0.1; $x_2(0)$=$x_3(0)$=$x_4(0)$=$x_5(0)$=0.2
μ=16384

PERFORMANCE ANALYSIS

In this section, we report the performance of the proposed approach. Specifically, different measurements and tests are explained to demonstrate complexity, robustness and effectiveness.

Key space: The key space of the proposed scheme can be derived from three parts: the 8-4-2-1 neural network key generator, the n-level wavelet signal decomposition, and the cellular neural network. Two keys are used in the 8-4-2-1 neural network: one is the 64-bit seed to neural network and the other is the 64-bit to calculate initial conditions. The number of bits needed for a typical n-level wavelet transform does not exceed three, and that for how many of the bit planes are to be encrypted is also three. A 128-bit key to drive a CNN hyper chaotic system is used to map initial conditions of the CNN and to calculate the CNN system parameters, as shown in Table 1. Thus, the key size for this encryption is 128 (from 8-4-2-1 neural network) + 128 (from CNN) + 3 (wavelet decomposition level) + 3 (no. of bit planes) + 3 (no. of encrypted planes) ~ >256, thus the key space is at least $2^{256} \sim 11.56 \times 10^{76}$.

NIST Statistical Test: Here, in this subsection, the generated pseudo-random sequences are tested by the Statistical Test Suite (STS) by NIST (Rukhin et al., 2001) to quantify and verify the randomness level. The suite includes statistical tests for individual sequences, and in turn generates a *p*-value. The criterion is that if this value is compared to a significance level typically set at 0.01 (for confidence of 99%) is determined to be equal to 1, then the sequence appears to be random; otherwise, non-random. The results of all of these tests run on the first 100,000 bits sequence are shown in Table 2, where we notice that the sequence generated passes the NIST statistical test for individual sequences.

Table 2

NIST Statistical Tests

NIST Statistical Test	*p*-value	Pass rate
Frequency	0.827319	Pass
Block-frequency	0.817450	Pass
Cumulative sums (forward)	0.954010	Pass
Cumulative sums (reverse)	0.808165	Pass
Runs	0.071589	Pass
Longest runs of ones	0.942786	Pass
Rank	0.161278	Pass
FFT	0.417645	Pass
Linear complexity	0.619830	Pass
Serial	0.287165	Pass
Approximate entropy	0.612450	Pass
Lempel-Ziv Compression	0.387341	Pass
Overlapping templates	0.608741	Pass

0/1 Balancedness Test: Golomb (1982) stated that the noise-like sequence should look like an equality distribution. This means that the generated chaotic sequence should have equal distribution between 0 and 1 (i.e. equal number of 1s and 0s). In order to judge the proposed approach by equality distribution, a number of tests were run on the 8-4-2-1 generator to produce sequences of different lengths. These lengths were estimated to be 65536 (based on wavelet subband of size 256x256), 16384 (based on wavelet subband of size 128x128), 4096 (based on wavelet subband of size 64x64), and 1024 (based on wavelet subband of size 32x32). The results are depicted in Table 3 and Figure 4. These results show that the numbers are quite close to 50%.

Table 3

Equality Distribution within the Chaotic Sequence Generated by 8-4-2-1 neural network

Sequence length	Count of 1s	Percentage
1024	515	50.29
4096	2055	50.17
16384	8206	50.08
65536	32775	50.01

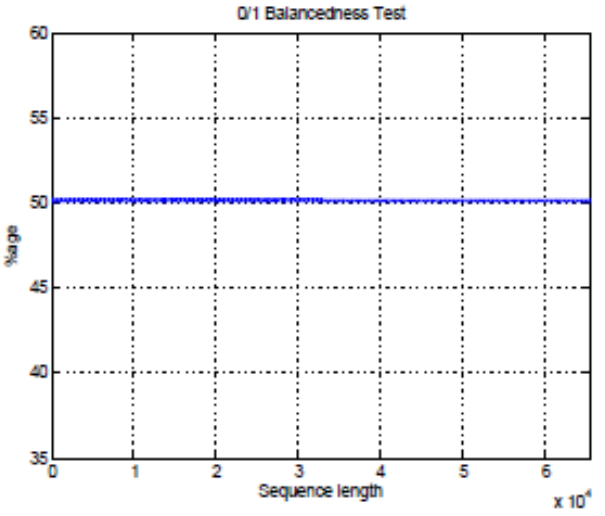

Figure 4. Percentage distribution of 1s in the sequence.

Histogram Analysis: Generally, the histogram of an image depicts the pixel distribution density against the intensity level. Here, we use it to analyze the encrypted image pixel distribution. In order to test the proposed approach in this perspective, the 512x512 "Camera-man" image was encrypted using n=2 with four most significant bit planes. After encrypting these bit planes, the process was run in reverse to construct the encrypted image and the histogram calculated. The results are shown in Figure 5, where x axis shows the gray level, while y axis shows the count at that gray level. It can be clearly seen that the histogram of the encrypted image is fairly uniform with the statistical properties of those of the white noise. To investigate it further, standard deviations were also calculated and were found to be 14.315 and 14.168 respectively, which are lower than that reported in Bigdeli et al. (2012).

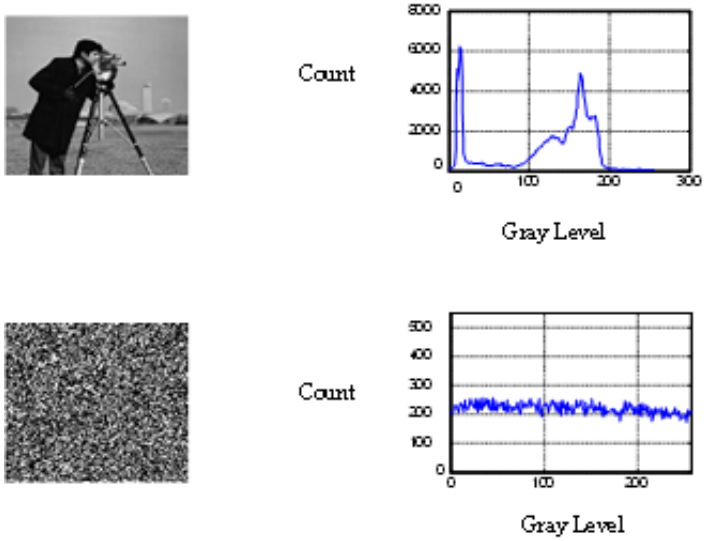

Figure 5. Histogram analysis of proposed approach.

Correlation Coefficient: Generally, correlation coefficient is considered as a statistical parameter to measure the quality of an encryption process. Theoretically, the autocorrelation function from the generated sequence should be a noise-like impulse at the origin, and almost zero away from the origin. This also means that each sequence bit is decorrelated from the other. The correlation coefficient was calculated using the following equation (El Assad et al., 2008):

$$r_{xy} = \frac{cov(x,y)}{\sqrt{d(x)}\sqrt{d(y)}}\ ;\ \ d(x) = \frac{1}{N}\sum_{i=1}^{N}\left(x_i - \frac{1}{N}\sum_{i=1}^{N} xi\right)^2 \qquad (9)$$

where *cov* (*x*, *y*) stands for the covariance between the two pixels *x* and *y*.

To verify experimentally for the 8-4-2-1 generator, this function was plotted using Equations 1-5 in Figure 6, where *x* axis shows the gray level x_i while y axis shows the autocorrelation function $r_{x,y}$ at that gray level. In this figure, good autocorrelation function can be seen clearly. The maximum value outside the origin is 0.00215. This smallest value outside the origin means that each bit generated is not correlated to the other. Thus, we can easily conclude that the resulting code bits are decorrelated. Table 4 shows the correlation coefficient r_{xy} along the horizontal, vertical, and diagonal directions of batch of (x_i, y_i, for

i=1, 2, 3….N) pairs of gray values of the two adjacent pixels in five original and encrypted images. It is clear from Table 4 that the pixels in the encrypted images have been completely decorrelated due to encryption. Furthermore, statistical values like averages and standard deviations were calculated across five original and encrypted images. Looking at the averages column in the original and the encrypted images, it is clear that the correlation coefficient average for all the cases has dropped from nearly one (~0.9) to insignificant values in the range of 10^{-4}. Likewise, very small values of standard deviation reflect the fact that variation in pixel decorrelation along horizontal, vertical and diagonal directions is very small (of the order of 10^{-4}) in all cases.

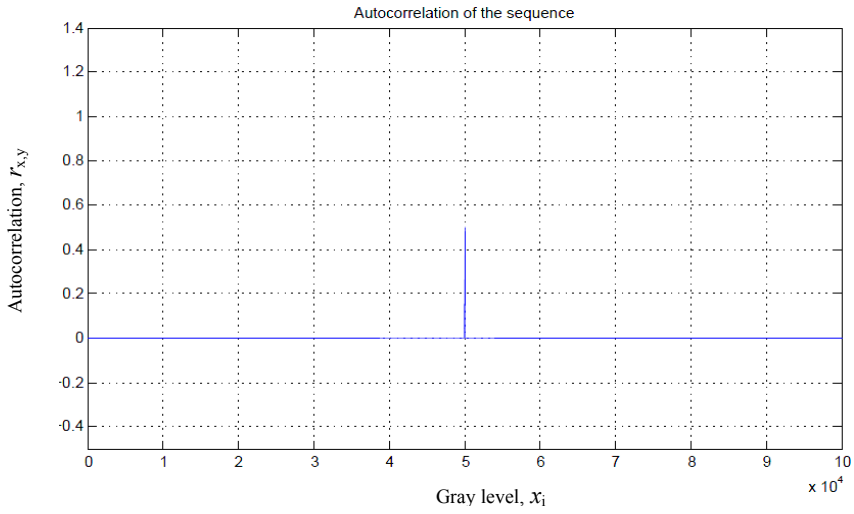

Figure 6. Correlation function of the sequence.

Table 4

Correlation Coefficients of the Original and Encrypted Images

Image	Original					Encrypted				
	Horizontal	Vertical	Diagonal	Average	Standard deviation	Horizontal	Vertical	Diagonal	Average	Standard deviation
Lena	0.9855	0.9881	0.9667	0.9801	0.009534	-0.00072	0.00053	0.00103	0.00028	0.00074
Barbara	0.9785	0.9574	0.9762	0.9707	0.009451	0.00082	-0.00102	0.00063	0.000143	0.00083
Yacht	0.9592	0.9499	0.9487	0.9526	0.004693	-0.000761	0.00052	-0.00049	-0.00024	0.00055
Woman	0.9584	0.9474	0.9462	0.9507	0.00549	0.000541	-0.00061	0.000385	0.000105	0.00051
House	0.9675	0.9486	0.9571	0.9577	0.007729	0.000613	0.00046	-0.00062	0.000151	0.00055

CONCLUSIONS

A JPEG2000 compatible block cipher is proposed in this paper with a two level encryption. In a single level encryption, random sequences are generated through random key generation by the 8-4-2-1 neural network, where hidden layers compute the output using repeated calculations in a cyclic manner to make it robust with increased complexity. During performance analysis, it was demonstrated that the key space for this level is more than 128. For the second level encryption, cellular neural network is used with additional 128-bit keys to generate sequences using the Runga-Kutta method. Thus, the net key size is above 256. Key management can be improved by hiding two keys in two secure physical locations, so that in case of a well-organized code-break in the Internet, the data will remain secure as it is less likely to recover multiple keys. Furthermore, using the 0/1balancedness, histogram and correlation analyses it was experimentally demonstrated that the proposed encryption is effective with robust performance.

The advantages of the proposed approach are multifold: firstly the encryption is the JPEG2000 format compliant; secondly the approach is flexible, such that based on the need, either a single level and/or a double encryption can be used. Furthermore, in order to accommodate a longer key, the neural network structure can be expanded to include another hidden layer or the cellular neural network can be adjusted to support a longer key; thirdly is that the watermark techniques can be easily embedded in the proposed scheme due to the availability of subbands and bit planes before encryption.

REFERENCES

Ayhan, T., & Yalcin, M., (2011). Randomly reconfigurable cellular neural network. *Proceedings of 20th European Conference on Circuit Theory and Design*, 604-607.

Bigdeli, N., Farid, Y., & Afshar, K. (2012a). A novel image encryption/ decryption scheme based on chaotic neural network. *Engineering Applications of Artificial Intelligence*, *25*, 753-765.

Bigdeli, N., Farid, Y., & Afshar, K. (2012b). A robust hybrid method for image encryption based on Hopfield Neural Network. *Computers and Electrical Engineering*, 38, 356-369.

Chua, L., & Yang, L., (1988). Cellular neural networks: Theory. In *IEEE Transactions on Circuits and Systems*, *35*(10), 1257-1272.

Djellit, I., & Kara, A. (2006). One-dimensional and two-dimensional dynamics of cubic maps. *Discrete Dynamics in Nature and Society*, Article ID: 15840, doi:10.1155/DDNS/2006/15840

Gao, T., & Chen, Z. (2008). Image encryption based on a new total shuffling algorithm. *Chaos, Solitan, and Fractals*, *38*(1), 213-220.

Golomb, S., (1982). *Shift register sequences*. Revised Edition. Laguna Hills, CA: Aegean Park.

Lian, S. (2007). Image authentication based on neural network. Cornell University, CoRR abs/0707.4524.

Lian, S. (2009). A block chiper based on chaotic neural networks. *Neurocomputing*, *72*, 1296-1301.

Lian S., & Chen, X. (2011). Traceable content protection based on chaos and neural networks. *Applied Soft Computing*, *11*, 4293-4301.

Joshi, S., Udupi, V., & Joshi, D. (2012). A novel neural network approach for digital image data encryption/decryption. *Proceedings of IEEE International Conference on Power, Signals, Controls and Computation*, 1-4.

Memon, Q., & Khoja, S. (2009). Academic program administration via semantic web – a case study. *Proceedings of International Conference on Electrical, Computer, and Systems Science and Engineering*, Dubai, *37*, 695-698.

Memon, Q., Akhtar, S., & Aly, A. (2007). Role management in adhoc networks. *Proceedings of Spring Simulation Multi-conference*, *1*, 131-137, March, Virginia, USA.

Munukur, R., & Gnanam, V. (2009). Neural network based decryption for random encryption algorithms. *3rd International Conference on Anti-counterfeiting, Security and Identification in Communication*, 603-605.

Nguyen, T., & Marpe, D. (2014). Objective performance evaluation of the HEVC main still picture profile. *IEEE Transactions on Circuits and Systems for Video Technology*, 1-8. doi: 10.1109/TCSVT.2014.2358000

Peng, J., Zhang, D., & Liao, X. (2009). A digital image encryption algorithm based on hyper-chaotic cellular neural network. *Fundamenta Informaticae*, *90*, 269-282.

Rukhin, A. et al. (2001). A statistical test suite for random and pseudorandom number generatorsfor cryptographic application. In *National Institute of Standards and Technology Special Publication,* 800-22. Retrieved at http://csrc.nist.gov/rng/

Kumar, A., et al. (1977). Application of Runge-Kutta method for the solution of non-linear partial differential equations. *Applied Mathematical Modelling*, *1*(4), 199-204.

S. El Assad S., Noura H., & Taralova, I. (2008). Design and analyses of efficient chaotic generators for crypto systems. In *Advances in Electrical and Electronics Engineering IAENG Special Edition*, 3–12. doi: 10.1109/WCECS.

Shitong, W., & Min, W. (2006). A new detection algorithm based on fuzzy cellular neural networks for white blood cell detection. *IEEE Transactions on Information Technology in Biomedicine*, *10* (1), 5-10.

Wang, L., et al. (2007). Cellular neural networks with transient chaos. *IEEE Transactions on Circuits and Systems-II: Express Briefs*, *54*(5), 440-444. doi:10.1109/TCS.2007.892399

Xu, S., et al. (2005). Novel global asymptotic stability criteria for delayed cellular neural networks. *IEEE Transactions on Circuits and Systems –II Express Briefs*, *52* (6), 349-353.

Yi, S., et al. (2015). Two novel cellular neural networks based on mem-elements. *Proceedings of the 34th Chinese Control Conference,* 3452-3456. Hangzhou, China.

Zirra, P., et al. (2011). Cryptographic algorithm using matrix inversion as data protection. *Journal of Information and Communication Technology*, *10*, 67-83.