MACHINE LEARNING BASED ADAPTIVE WATERMARK DECODING IN VIEW OF ANTICIPATED ATTACK

Asifullah Khan^{a, *}, Syed Fahad Tahir^c, Abdul Majid^b, and T. S. Choi^a

^aDepartment of Mechatronics, Gwangju Institute of Science and technology, 1 Oryong-Dong, Buk-Gu, Gwangju 500-712, Republic of Korea, Email: {asifullah, tschoi}@gist.ac.kr

^bDepartment of Information and Computer Sciences, Pakistan Institute of Engineering and Applied Sciences, Nilore, Islamabad, Pakistan, Email: <u>ab_majid@pieas.edu.pk</u>:

^cFaculty of Computer Science & Engineering, Ghulam Ishaq Khan (GIK) Institute of Engineering Science & Technology, Swabi, Pakistan, Email: fahad_290@yahoo.com

Abstract

We present an innovative scheme of blindly extracting message bits when a watermarked image is distorted. In this scheme, we have exploited the capabilities Machine Learning (ML) approaches for nonlinearly classifying the embedded bits. The proposed technique adaptively modifies the decoding strategy in view of the anticipated attack. The extraction of bits is considered as a binary classification problem. Conventionally, a hard decoder is used with the assumption that the underlying distribution of the Discrete Cosine Transform coefficients do not change appreciably. However, in case of attacks related to real world applications of watermarking, such as JPEG compression in case of shared medical image warehouses, these coefficients are heavily altered. The sufficient statistics corresponding to the maximum likelihood based decoding process, which are considered as features in the proposed scheme, overlap at the receiving end, and a simple hard decoder fails to classify them properly. In contrast, our proposed ML decoding model has attained highest accuracy on the test data. Experimental results show that through its training phase, our proposed decoding scheme is able to cope with the alterations in features introduced by a new attack. Consequently, it achieves promising improvement in terms of Bit Correct Ratio in comparison to the existing decoding scheme.

Keywords: Watermarking, Support Vector Machines (SVM), Artificial Neural Networks (ANN), Discrete Cosine Transform (DCT), Bit Correct Ratio (BCR), Decoding.

1. Introduction

Watermarking, closely related to the fields of cryptography and steganography, is the art of imperceptibly altering a digital medium to embed a message about that digital medium. With the wide spread and complex use of digital medium, a question about the security of the digital medium arises immediately. This concern is effectively addressed by watermarking, owing to its three nice characteristics; imperceptibility, inseparability from the cover content, and its inherent ability to undergo the same transformations as experienced by the cover content. In addition, it can be employed on many digital medium like text, audio, images, graphics, movies and 3D models. Its purpose is largely to counter problems like unauthorized handling, copying and reuse of information. Pertinent applications of watermarking include ownership assertion, authentication, broadcast monitoring and integrity control [1-2].

Watermarking scheme is mostly designed in view of its applications. Decoding of a watermark from the watermarked image is an important phase of a watermarking system, especially, when an attack on the watermarked image is highly probable. There is no such watermark decoding scheme that can perform well under all hostile attacks. However, with the growing need of sophisticated watermarking applications, we need a decoding scheme that should perform well under a specific set of conceivable attacks. Generally, channel noise and JPEG compression are the two most common attacks. They can appreciably change the underlying distribution of Discrete Cosine Transform (DCT) coefficients. The traditional decoders assume that the distribution of DCT coefficients is not heavily altered and thus are not able to retain performance under such attacks. In contrast, the proposed Machine Learning (ML) decoding models are able to learn the distribution of the altered coefficients and achieve a significant margin of improvement. However, the trained ML decoding model has to be provided at the decoding side for blindly extracting the message bits. Depending upon the nature of the watermarking application, we can make the trained ML decoding model public. On the other hand, the trained model can be provided through a private channel or encrypted along with other secret knowledge of the watermarking system. The encryption layer that overlays above the watermarking layer could be very effective in enhancing the overall security of the watermarking system [3]. In essence, the constraint of making the trained model private on one-side limits its applications, but on the other side, enhances the security of the watermarking system [4], especially, if we consider the complex security requirement of unauthorized decoding [3]. The potential applications of such adaptive ML decoding techniques could be like device control, and broadcast monitoring, as in both cases the ML decoding scheme can be employed in hardware form with the capability of adaptively modifying in view of the new attacks. Piracy detection and copyright demonstration when associated with a copyright authentication center could also be potential applications.

Recently, Support Vector Machines (SVM) and Artificial Neural Networks (ANN) based ML techniques have been applied to improve watermarking systems [5-12]. Especially, some researchers have concentrated on developing strategies for watermark detection/decoding by exploiting the learning capabilities of ML techniques. However, in the present work, we present a novel idea of adaptively modifying the decoding strategy in view of a specific application of the watermarking system. For example, consider a watermarking application where it is highly probable to JPEG compress the watermarked images before transmission or storing in an image warehouse, such as shared medical image warehouses utilized for remote diagnostic aid applications and telesurgery. In such a scenario, it is judicious to exploit the learning capabilities of ML systems by providing it information about the distortion caused by the JPEG compression during its training phase. Once the ML Decoding model is developed, it can be effectively employed for blindly extracting the embedded message from the distorted image. Other pertinent examples of such specific applications consist of a channel characterized by Gaussian noise, intentional or unintentional filtering, valuematric distortion, etc. Our main contributions in this regard are as follows:

• Bit extraction is considered as a binary classification problem in view of hostile attacks.

- Exploitation of the fact that distortion caused by a single attack might have incurred varyingly on different frequency bands.
- Employment of SVM and ANN based ML models for adaptively developing high performance watermark decoding in view of its intended application.

The remaining part of this paper is organized such that section 2 describes the related research work. Section 3 provides a brief overview of ANN and SVM techniques. Our proposed watermark decoding scheme is described in section 4. This includes dataset generation as well as the development of ML based decoding models. Results and discussion are explained in section 5, followed by conclusion in the last section.

We use many abbreviations and for reader clarity, we explicitly mention these abbreviations in this section. They include, Support Vector Machines (SVM), Artificial Neural Networks (ANN), Discrete Cosine Transform (DCT), Machine Learning (ML), Threshold based Decoding (TD), and Bit Correct Ratio (BCR).

2. Related Research Work

Some researchers have put sizable effort to develop new decoder structures for increasing decoding performance. For example, Barni et al. [13] have proposed a new decoding algorithm that is powerful in case of non-additive watermarking techniques. Likewise, Hernandez et al. [14], Nikolaidis et al. [15] and Briassouli et al. [16]-[17] have developed nonlinear detection/decoding structures that improve on the correlation-based techniques frequently used in watermarking systems. Their models exploit the properties of probability density functions of the transform domain coefficients of the cover work. Nevertheless, these efforts do not present a generic scheme that could easily develop a decoder useful in case of a new watermarking application and the subsequent attack. As such, there is a strong need of easily generating application-specific decoders. These decoders need to be blind as well as effective against the conceivable attack.

ML models both in watermarking and Steganography are effectively employed at the embedding, detection and decoding stages [5-12]. For instance, very recently, Fridrich et al. [5] have shown improved blind steganalysis by Merging Markov and DCT features and utilizing SVM. Fu et al. [6] have applied SVM for logo detection. The difference in the intensity level of pixels' blue components is used for training the SVM. Lyu et al. [7] utilize the learning capabilities of SVM to classify watermarked images based on the high-order statistical models of natural images. Shen et al. [8] employ Support Vector Regression at the embedding stage for adaptively embedding the watermark in the blue channel in view of the human visual system. On the other hand, Sang et al [11] have proposed a zero-watermark scheme that employs ANN for feature extraction. Recently, Wang et al. [9] have proposed an ANN controller for selecting the strength of the embedded data in view of the human audio system. Similarly, Li et al. [12] have used Independent Component Analysis to extract watermark blindly. Yet, the performance of their system is dependent on the statistical independence between the original cover work and the watermark. Nonetheless, none of these ML based approaches take into consideration the intended application and consequently are not adaptive towards a new anticipated attack while detecting/decoding the watermark.

As far as the applications of Genetic Programming (GP) in watermarking are concerned, Khan et al. [19-21] have used GP for perceptually shaping watermark with respect to both the conceivable attacks and cover image at the embedding stage. In a recent work, Khan [18] has proposed the modification of decoder structure using GP in accordance to both the cover image and conceivable attacks. However, in his proposed scheme, the sufficient statistics of the maximum likelihood based information decoding process are modified using the genetically evolved nonlinear mapping function. The modified sufficient statistics are then presented to a

threshold-based decoder. In contrast, in the present work, this nonlinear mapping in view of the anticipated attack is achieved inherently through the kernel functions in case of SVMs, and hidden layers in case of ANN. As for as distortions suffered by a watermarked data are concerned, various researchers, e.g. Cox et al. [1], and Piva et al. [2] have studied and categorized these distortions. For example, addition of different types of noise, signal processing attacks such as D/A conversion, color reduction, linear filtering attacks like high pass and low pass filtering, lossy compression, geometric distortions etc. Keeping in view these distortions, researchers have also investigated various approaches to make watermark system more reliable. They have proposed redundant embedding, selection of perceptually significant coefficients, spread spectrum modulation and inverting distortion in the detection phase [1]. Efforts are put in to theoretically evaluate the performance of a watermarking sachem in presence of a specific distortion. For instance, the performance of spread-transform dither modulation watermarking system is theoretically evaluated assuming non additive attacks [22]. In another recent work, Cox et al [23], incorporate the idea of perceptual shaping into spread transform dither modulation based watermarking scheme for improving imperceptibility as well as robustness against JPEG compression. In contrast, we exploit the learning capabilities of both SVM and ANN models for adaptively modifying the decoding mechanism in view of a specific attack. This is accomplished by training the proposed ML decoding model in view of the specific attack. Our present work is an extension of our previous work [24], where we have analyzed the performance of SVM based decoding only against Gaussian noise attack. However, in the present work, we not only analyze and compare the performance of different ML decoding models using both selfconsistency and cross validation techniques, but also study their performance against diverse benchmark attacks.

3. Machine Learning Techniques

3.1 ANN Models

ANN based ML techniques are extensively used in pattern recognition. They are mostly categorized in terms of supervised and unsupervised learning algorithms. The ANN networks present a distinct way to analyze data, and to recognize patterns within the data [25-26]. A network is characterized by its architecture, learning method and activation function. Architecture of a neural network describes the pattern in which the neurons are interconnected. In this work, we are using supervised ANN models in which n input training pairs (x_i, y_i) ,

where $x_i \in \mathbb{R}^N$ and $y_i \in [-1, 1]$, are presented to the ANN network.

In order to develop ANN decoding scheme, back-propagation learning algorithm is employed during training phase. This algorithm computes the error e, for output neuron j, as follows:

$$e_i(t) = z_i(t) - y_i(t) \tag{1}$$

where z_j and y_j are the actual and target output for neuron j for iteration t. The average squared error of the network is obtained as:

$$\xi_{avg}(t) = \frac{1}{n} \sum_{t=1}^{n} \xi(t) \tag{2}$$

where, $\xi(t) = \frac{1}{2} \sum_{j \in P} e_j^2(t)$, represents instantaneous sum of squared errors and *P* indicates number of neurons in

the output layer. The average error ξ_{avg} is a cost function of the network, which helps the network learn using the training samples. The objective of the learning process is to adjust the free parameters (weights, learning rate, and steepness of the activation function) of the network to minimize ξ_{avg} .

The weight vector w and the bias b are updated according to Levenberg-*Marquardt* Algorithm [27]. This algorithm works iteratively in search of weights and biases that minimize the cost function; mostly the sum-squared of the difference between the target and network output responses. While training moderate-sized networks, this algorithm trains neural networks at a rate 10-100 times faster than the usual gradient descent based back-propagation method. In this algorithm, the updated weights w_{n+1} are computed as:

$$w_{n+1} = w_n - [\mathbf{J}^T \mathbf{J} + \mu \mathbf{I}]^{-1} \mathbf{J}^T \mathbf{e}$$
(3)

where, μ and **e** denotes a constant scalar and a vector of network error respectively. **J** represents the Jacobean matrix, which contains first derivatives of the network errors with respect to the weights and biases.

3.2 SVM Models

SVM is a margin-based classifier having excellent generalization capabilities [28-29]. Such models try to find an optimal separating hyper-plane between data points of different classes in a high dimensional space. The error in SVM models occur if the data points appear on the wrong side of the boundary. In case of a linearly separable data, a hyper-plane is determined by maximizing the distance between the support vectors. Consider *n* training pairs (x_i, y_i) , where $x_i \in \mathbb{R}^N$ and $y_i \in [-1, 1]$, the linear decision surface is defined as:

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i x_i^T \cdot x + b$$
⁽⁴⁾

where, $\alpha_i > 0$. In order to find an optimal hyper-plane, the solution of the following optimization problem is sought.

$$\Phi(w,\xi) = \frac{1}{2}w^T w + C \sum_{i=1}^{N} \xi_i , \qquad (5)$$

subject to the condition $y_i \left(w^T \Phi(x_i) + b \right) \ge 1 - \xi_i, \quad \xi_i \ge 0.$

where C > 0 is the penalty parameter of the error term $\sum_{i=1}^{N} \xi_i$ and $\Phi(x)$ is nonlinear mapping. The weight vector

W minimizes the cost function term $w^T w$. Each point $\Phi(x)$ in the new space is subject to Mercer's theorem [29], in which kernel functions are defined as:

$$f(x) = \sum_{i=1}^{N_s} \alpha_i y_i K(x_i, x) + b = \sum_{i=1}^{N_s} \alpha_i y_i \Phi(x_i) \cdot \Phi(x_j) + b$$
(6)

In order to obtain different SVM classification models, we have investigated performance of the following three popular kernel functions:

- $K(x_i, x_j) = x_i^T \cdot x_j$ (Linear kernel with parameter C)
- $K(x_i, x_j) = \exp(-\gamma \| x_i x_j \|^2)$ (RBF with kernel parameters γ , C)
- $K(x_i, x_j) = [\gamma < x_i, x_j > +r]^d$ (Polynomial kernel with parameters γ , r, d and C)

4. Proposed Watermark Decoding Scheme

For comparative analysis, we have implemented the watermark scheme proposed by Hernandez et al. [14]. This watermarking technique is oblivious and embeds message into the low and mid frequency coefficients of 8x8 DCT blocks of a cover image. In their scheme, they model the DCT coefficients of each frequency band using generalized Gaussian distribution. One of the reasons for using this watermarking scheme in our comparative analysis is that it employs DCT in blocks of 8x8 pixels, in a manner similar to the widely used JPEG compression algorithm. Secondly, this watermarking scheme has strong theoretical foundations [14]. They have employed a TD scheme assuming that the probability density function of the original coefficients remains the same even after embedding. However, this assumption may not be valid when attack is performed on the watermarked image. In their maximum likelihood based watermark extraction scheme, first, sufficient statistics corresponding to each embedded bit is computed and then it is compared with a threshold. In the absence of an attack, two non-overlapping distributions of the sufficient statistics, corresponding to +/-1 bits are generated as shown in figure1-a. Under such circumstances, a simple TD model is sufficient to decode +/-1 bits from the watermarked image. However, we have observed that in case of an attack on the watermarked image, these distributions overlap as shown in figure1-b. Consequently, simple TD model is unable to decode the message bits efficiently.

To address such message decoding problems in watermarking, we propose a novel idea of decoding the message bits using SVM and ANN based ML techniques. We assume that a non-separable message in lower dimensional space might be separable if it is mapped to higher dimensional space. This mapping to higher dimensional space is what the hidden layers in case of ANN and the kernel functions in case of SVM are achieving.

(a)

Figure 1: Distribution of sufficient statistics of the maximum likelihood based decoding system corresponding to +/-1 bits; (a) before and (b) after Gaussian Attack (σ =10)

(b)

In our proposed scheme, a generalized dataset is created (section 4.1) for developing SVM and ANN models. Their performance is evaluated by using *self-consistency* and *cross-validation* techniques. In *self-consistency*, the performance of classification models is reported using the training dataset. However, in *cross-validation*, entirely different test dataset is used. In analogy to pattern recognition, a bit corresponds to a sample and a message corresponds to a particular pattern of these samples.

SVM classification models are developed by using linear, polynomial and RBF kernel functions. The kernels functions in SVM model are optimized by using grid search technique. In grid search, optimal values of kernel parameters are obtained by selecting various values of grid range and step size. ANN models based on *backpropagation algorithm* [27] are also developed. Finally, a comparative analysis of SVMs, ANN and TD models is carried out. Our decoding scheme mainly consists of following two modules, as shown in figure 2:

(1) Dataset Generation module, and

(2) Machine Learning based Decoding Module.

It should be noted that in the current work, we are focusing only on the message retrieval using intelligent techniques and thus have employed the simple but commonly used error correction technique, i.e. repetition coding. Employment of advanced error correction strategies [30-31], for example, trellis [32], low-density parity-check [33], and turbo [34] coding, would certainly improve the overall message retrieval performance in all of the cases.

Figure 2: Basic block diagram of the proposed ML decoding system

4.1 Proposed Dataset Generation

In order to analyze the performance of our proposed scheme, we have generated a dataset of 16000 bits. For this purpose, five different images, each of size 256×256 , are used. Next, a message of size 128 bits is embedded in each image. The whole process is repeated 25 times by changing the secret key used to generate the spread spectrum sequence. In this way, it produces 125 different messages and consequently, 125 different watermarked images. Gaussian noise attack with σ =10 is applied on each image. Finally, sufficient statistics corresponding to each embedded bit for every watermarked image is computed. This produces a data set of $16000 = 125 \times 128$ bits. In this way, we form a data set representing 125 different messages, being embedded in five different types of images generating 16000 bits. This dataset is assumed a generalized one, as we have taken into account the effect of both the cover image distribution, and the secret key on the sufficient statistics. Table 1 shows the different parameters of our data set. A similar approach is taken in case of Wiener and JPEG compression attacks.

Table 1: Parameters of input dataset

4.1.1 Generating Attacked Watermarked Images

The underlying watermarking technique that we have used to analyze ML based decoding is the spread spectrum based watermarking approach proposed by Hernandez et al. [14]. In this approach, the product of the spread-spectrum sequence and expanded message bits is multiplied with a perceptual mask $\alpha[\mathbf{k}]$ to obtain the watermark. Let us denote the 2-D discrete indices in DCT domain by $[\mathbf{k}]$. The 2-D watermark signal $W[\mathbf{k}]$ is given as:

$$W[\mathbf{k}] = S[\mathbf{k}] \cdot \mathbf{b}[\mathbf{k}] \cdot \boldsymbol{\alpha}[\mathbf{k}]$$
(7)

where $S[\mathbf{k}]$ is a pseudo random sequence and $b[\mathbf{k}]$ is the repetition-based expanded code vector, corresponding to the message to be embedded. Adding this watermark to the original image in transformed domain, represented by $X[\mathbf{k}]$ performs the embedding:

$$Y[\mathbf{k}] = X[\mathbf{k}] + W[\mathbf{k}] \tag{8}$$

where $y[\mathbf{k}]$ represents the watermarked image.

In analogy to communications, the watermark $W[\mathbf{k}]$ is our desired signal, while the cover image $X[\mathbf{k}]$ acts as an additive noise. Algorithm 1 contains a high-level description for generating attacked watermarked images using different images and secret keys.

Algorithm 1: Generating watermarked image Database

//We omit the 2-d vector indices [k] for elaboration purpose

//x, y: original and watermarked images respectively in spatial domain

//fa: attack function, z: attacked watermarked image

//X, Y: original and watermarked images respectively in DCT domain

//W: watermark, S: spread spectrum sequence, b: expanded message vector

// α : perceptual mask, I_{max} : No. of images, Q_{max} : No. of secret keys

1: Encode the message of size 128 bits using an error correction technique and expand it to form a vector b

for $i \leftarrow 1$ to I_{max} do	//select different standard images
X=DCT2(x)	//compute 8x8 block DCT of the image
$X^i \leftarrow X$	
$\alpha^i \leftarrow \alpha$	//compute perceptual mask of the image
for $q \leftarrow 1$ to Q_{max} , do	//select different secret keys
generate S _q	//generate the spread spectrum sequence
$W = \alpha^{i}.S_{q}.b$	// compute the watermark
$W^{i}_{q} \leftarrow W^{i}$	
$Y_{q}^{i} = X^{i} + W_{q}^{i}$	//perform watermark embedding
$y_q^i = invDCT2(Y_q^i)$	//perform inverse DCT
$z_{q}^{i} = f_{a}(y_{q}^{i})$	//perform attack on the watermarked image
	for $i \leftarrow 1$ to I_{max} do X=DCT2(x) $X^{i} \leftarrow X$ $\alpha^{i} \leftarrow \alpha$ for $q \leftarrow 1$ to Q_{max} , do generate S_q $W=\alpha^{i}.S_q.b$ $W^{i}_{q} \leftarrow W$ $Y^{i}_{q}=X^{i}+W^{i}_{q}$ $y^{i}_{q}=invDCT2(Y^{i}_{q})$ $z^{i}_{q}=f_{a}(y^{i}_{q})$

4.1.2 Proposed Feature Extraction

When a watermarked image is attacked, the message within the image is also corrupted. We have computed features corresponding to each bit of a message, in two different ways. In the first method, the sufficient statistics r_i corresponding to each bit of the maximum likelihood based decoding system across all the frequency bands is computed [14]. Therefore, we obtain a random value r_i corresponding to each bit as given below:

$$r_{i} \triangleq \sum_{\mathbf{k} \in G_{i}} \frac{\left| Y[\mathbf{k}] + \alpha[\mathbf{k}] S[\mathbf{k}] \right|^{c[\mathbf{k}]} - \left| Y[\mathbf{k}] - \alpha[\mathbf{k}] S[\mathbf{k}] \right|^{c[\mathbf{k}]}}{\sigma[\mathbf{k}]^{c[\mathbf{k}]}}$$
(9)

where, G_i denotes the sample vector of all DCT coefficients in different 8×8 blocks that correspond to a single bit *i*, σ represents the standard deviation of the distribution, while *c* dictates the shape of generalized Gaussian distribution. The values of *c* and σ are estimated from the received watermarked image at the decoding stage. For bipolar signal, $b[\mathbf{k}] \in [-1,1]$, the estimated bit \hat{b}_i in TD model is computed as follows:

$$\hat{b}_{i} = \operatorname{sgn}(r_{i}) \ \forall \ i \in \{1, 2, \cdots, N\}$$
 (10)

In the second method, we do not compute sufficient statistics across all the frequency bands; rather we compute it across the same channel. In this case, the sample vector G_i used in equation 9 changes to G_i^j , which is defined as the sample vector of all DCT coefficients in different 8×8 blocks that correspond to a single bit *i* and the *jth* frequency band. This allows us to keep the sufficient statistics across each frequency band as a feature itself. Therefore, corresponding to a single bit, the number of features is equal to the number of selected frequency bands. As described in detail in section 6, mostly, we have selected 22 frequency bands and consequently, 22 features. This is because ML models, as against the TD model, have the capability to exploit the different frequency bands by learning their corresponding level of distortion incurred by the attack. Each sample in the training dataset consists of a pair of input pattern of 22 features and the corresponding target value. The target value consists of the original bit embedded in the image. These target values of training dataset are used to make the SVM model learn the behavior of bits when distorted by an attack. Algorithm 2 describes the main steps involved in extracting features for the proposed ML decoding scheme from the attacked watermarked images.

Algorithm 2: Feature Extraction for the proposed ML decoding model

//We or	We omit the 2-d vector indices [k] for elaboration purpose							
//x, y: o	//x, y: original and watermarked images in spatial domain respectively							
//X, Y:	original and watermark	ed images in DCT domain respectively, ri: //sufficient statistics						
//W: wa	atermark, S: spread spec	trum sequence, b: expanded message vector						
//C, σ: s	shape parameter, and sta	ndard deviation of the Generalized Gaussian //distribution respectively						
// α: pe	creeptual mask, Imax: No.	of images, Q _{max} : No. of secret keys						
1: for	i←1 to I _{max} do	//select different standard images						
2:	X=DCT2(x)	//compute 8x8 block DCT of the image						
3:	$X^i \leftarrow X$							
4:	$\alpha^{i} \leftarrow \alpha$	//compute perceptual mask for the image						
5:	5: for $q \leftarrow 1$ to Q_{max} , do //select different secret keys							
6:	generate S _q //generate the spread spectrum sequence							
7:	C : compute C, and σ for each frequency band using maximum likelihood estimation [14]							
8:	compute feature r _i o	corresponding to each bit using equation 9						

4.1.3 Data Sampling Techniques Utilized

To investigate the performance analysis of our proposed decoding scheme, we have employed both the *self*consistency and cross-validation data sampling techniques. The aim of using self-consistency technique is to check the performance of trained SVM and ANN decoding models on the training dataset. However, crossvalidation technique is used to develop a generalized decoding model that can perform well even on the novel data samples. In this technique, the whole dataset is divided into four equal parts. One part is selected for training and the remaining three are kept for test purpose. This process is repeated four times so that whole dataset can be used in 4-fold. Finally, average BCR of the classification models are computed.

4.2 Developing Machine Learning based Decoding Models

4.2.1 Performance Measure

In this work, we have compared the performance of classification models in terms of BCR. It is computed as follows:

$$BCR(M,M') = \frac{\sum_{i=1}^{L_m} \overline{\left(m_i \oplus m'_i\right)}}{L_m}$$
(11)

where *M* represents the original, while *M*' represents the decoded message, L_m is the length of the message and \oplus represents exclusive-OR operation. It should be noted that (1 - BCR) represents the bit incorrect ratio. Even a small margin of improvement in BCR can heavily affect the performance of a watermarking system.

4.2.2 Developing SVM models

In order to develop Linear, RBF, and Poly-SVM models, we have used '*LIBSVM*' toolbox [35]. This toolbox has all the basic functions for creation and training of SVM models.

4.2.2.1 Parameter optimization of SVM models:

The performance of SVM models can be optimized by using various optimization techniques [29], [36-38]. However, we have employed the most simple but efficient *grid search* technique, as described in [29]. This technique helps to find the optimal values of SVM kernel parameters by selecting appropriate grid range and step size. Poly-SVM has four adjustable parameters *d*, *r*, γ and *C*. On the other hand, for simplicity, the value of degree, *d* and coefficient, *r* are priori fixed at *d* = 3, *r* = 1. By using grid search, the optimum value of C is computed in the range of $[2^{-2}, 2^{2}]$ with step size of ΔC equal to 0.4. Similarly, the optimal value of γ is computed in the range of $[2^{-2}, 2^{8}]$ with $\Delta C = 0.4$. In case of RBF-SVM, the estimated range and step size of C and γ are given as; $C = [2^{-2}, 2^{2}]$ with $\Delta C = 0.4$, $\gamma = [2^{-2}, 2^{8}]$ with $\Delta \gamma = 0.4$. The optimal value of C, for linear kernel is obtained by adjusting the grid range of $C = [2^{-2}, 2^{5}]$ with step size of $\Delta C = 0.4$. The overall behavior of RBF-SVM model during the optimization of the values of C and γ is shown in figure 3. This figure shows the relatively high dependency of RBF-SVM model on the parameter γ than that of C.

Figure 3: BCR performance dependency of RBF-SVM based decoding model on parameters C and γ .

4.2.3 Developing ANN models

The implementation of back-propagation algorithm is carried out by using Neural Network Toolbox of MATLAB 7 [25]. In order to use this toolbox, first, we initialize the hidden and output layer units, activation functions of hidden and output layers, stopping criterion and the training algorithm. First and second hidden layers are configured with 25 and 15 neurons respectively. All the necessary parameters setting of ANN model for a dataset of 16000 bits and exploiting 22 features are shown in Table 2.

Table 2: Parameter settings for ANN based decoding method effective against Gaussian noise attack

The activation functions of ANN models describe the behavior of neurons. These functions may be linear and nonlinear. A list of the used linear and nonlinear activation functions is given in Table 3.

Table 3: Activation functions used in the proposed ANN decoding model

A block diagram of the ML based watermark extraction process is shown in figure 4. After computing the features, the trained ANN/SVM models are employed to carry out the message extraction process. Figure 4: watermark extraction

5. Potential Applications of Proposed ML Decoding Scheme

In this section, we discuss the potential applications of our proposed ML based adaptive watermark decoding scheme. We conceive two scenarios. In the first type of applications, only trained ML model has to be utilized. On the other hand, in the second scenario, we might expect change in type of attack. In this 2nd case, the full ML system has to employed in form of a chip.

In case of shared medical information, such as PACS (Picture Archiving and Communication Systems) [39], and DICOM (Digital Imaging and Communications in Medicine) [40], the images are compressed to reduce both memory and bandwidth requirements [41]. In such a scenario, JPEG compression attack is almost inevitable, and therefore, we need to develop an ML based decoding model for extracting the message blindly. The trained ML model could be then deployed as small chip achieving a significant margin of improvement in terms of BCR. As far as health related projects are concerned, a small hardware cost would be of no match to extracting information about the cover data accurately.

Similar approaches needs to be considered in broadcast monitoring [42], and device control [43]. The trained ML decoding models could be deployed in form of a chip in view of the inevitable attacks, such as channel noise in case of broadcasting only, and digital to analog conversion in case of both broadcasting and device control.

Applications related to the second scenario are akin to secure digital camera [44], and those mobiles able to extract hidden information from plain sight [45]. In both of these applications image acquisition and its further processing is involved. However, the user may also be provided with options of focusing, contrast stretching, etc. In such a case of varying attacks, the user may spare few seconds to let the onboard ML model learn the distortion occurred due to the new attack.

6. RESULTS AND DISCUSSION

6.1 Watermark Strength and Imperceptibility Analysis

First, we analyze the strength of the watermark and consequently its affect on the imperceptibility. Generally, it is assumed that higher the strength of the watermark, the higher will be the robustness, although it has been practically shown that this may not always be the case [20]. However, high strength means high distortion of the original image and thus low imperceptibility. Therefore, we show that even though keeping a fair amount of imperceptibility, our proposed ML models are able to retrieve the embedded message from the attacked watermarked image. For this purpose, we first visually analyze the imperceptibility of the watermarked images as explained in [19-20]. Figure 5 and 6 shows the original and watermarked couple image respectively. As obvious, the distortion in the watermarked image is almost impossible to be detected by a human eye. It means that imperceptibility is and consequently low power embedding is performed. This high imperceptibility lays down a limit on the capability of the decoding models, as robustness usually requires high power embedding. In this connection, figure 7 shows the distribution of the watermark for Couple image across both frequency and no. of blocks. It can be easily observed that the amplitude of the watermark varies not only from block to block, but also inside each block. In figure 7, if we pick a selected DCT coefficient and vary the block number, then we can realize the watermark strength variation across a single frequency band. Each of these 22 selected frequency bands have different variation in watermark strength and thus should be dealt with separately.

Figure 5: Original Couple Image Figure 6: Watermarked Couple Image Figure 7: Distribution of the watermark for Couple Image

6.2 Distortion Analysis

In this section, we analyze visually, the amount of distortion introduced by the attack. This is because small distortion incurred by a weak attack may not alter the performance of the traditional TD model. Therefore, we try to compare the performance of the decoding systems in a harsh but same environment. Figure 8 and 9 show the difference in pixel intensities between original and watermarked, and watermarked and attacked images respectively. This difference is 10 times amplified for elaboration purposes. It can be observed from figure 8 that in case of no attack, the difference is not severe and easily visible at less sensitive areas like edges. On the other hand, the difference in case of the attack (Gaussian noise with σ =10) is quite severe. This sort of distortion may easily disturb decoders based on the statistical characterization of the DCT coefficients across the different channels. This is the main reason that we use two types of feature subsets; in one case, we let the ML system exploit the sum of the responses from all the channels, while in the other case, we let it exploit the various frequency channels separately as per their corresponding distortion. Figure 10 shows the Gaussian attacked watermarked image, where the imperceptibility is affected strongly by the resultant distortions.

Figure 8: Pixel Intensity difference between original and watermarked images Figure 9: Pixel Intensity difference between watermarked and attacked watermarked images Figure 10: Gaussian noise attacked watermarked Couple Image

6.3 Cross Validation based Performance Comparison

In order to carry out comparative study, the average BCR of TD model is also computed. The input dataset is divided in the same fashion as it is used for SVM and ANN decoding models. The experimental results are given in Table 4. This table shows that in case of TD model, the average BCR of TD model is approximately the same i.e. 0.984 for both self-consistency and cross validation techniques.

Table 4: BCR performance of TD model using 4-fold cross validation against Gaussian noise attack

The BCR performance of SVM models using the same distribution of input dataset is shown in Table 5. It is observed that Poly-SVM attains maximum value of average BCR, i.e. 100 percent accuracy on the training dataset for the optimized values of $\gamma = 194$ and C = [0.4-2]. On the other hand, the self-consistency performance of RBF-SVM and Linear-SVM is nearly the same, i.e. 0.9877 and 0.9853 respectively.

It is also observed from Table 5 that there is consistency in decoding performance of SVM models even for novel test dataset. These SVM models have maintained their high performance on both the training and test data. However, there is some degradation in the performance of Linear-SVM and RBF-SVM as compared to Poly-SVM. Poly-SVM has achieved 100 percent decoding performance by correctly extracting all the 12000 bits in the test dataset. This is, indeed, a major achievement as far as a distorted watermark is concerned. From Table 4 and 5, the order of overall BCR performance of SVM and TD models is as follows; *Poly-SVM* > *RBF-SVM* > *Linear-SVM* > *TD model*.

Table 6 shows the experimental results of ANN model. Average BCR performance of ANN model is up to 0.999 for the training dataset exploiting 22 features. However, in case of test data, the performance of ANN model degrades up to 0.9746. It is concluded from Table 5 and 6 that average BCR performance of Poly-SVM and ANN models are approximately the same on the training dataset. However, due to its low generalization capability, ANN model could not maintain the same high performance on the test data. Whereas, SVM models, especially, Poly-SVM has maintained high performance on both training and test dataset.

Table 5: BCR performance of SVM models using 4-fold cross validation against Gaussian noise attack

Table 6: BCR performance of ANN model using 4-fold cross validation against Gaussian noise attack

The overall performance comparison in terms of bar chart of three decoding models (SVM, ANN and TD) is shown in figure 11. The order of performance of decoding models on training data is as follows; *Poly-SVM* >

ANN > RBF- SVM > Linear-SVM > TD. On the other hand, in case of test data we observe, Poly-SVM > Linear-SVM > RBF-SVM > TD > ANN.

Figure 11: BCR performance comparision using 4-fold cross-validation and 22 features

6.4 Self-Consistency based Performance Comparison

The comparison of the three decoding models is also analyzed using self-consistency as shown in figure 12 and 13. It can be observed from figure 12 that in this case both *Poly-SVM*, and *RBF-SVM* have the highest performance. The order of performance using self-consistency is as such: *Poly-SVM* = *RBF- SVM* >*ANN* > *Linear-SVM* >*TD*. Analyzing the performance of *ANN*, and *RBF-SVM* on the both self-consistency and cross validation, it can be concluded that the developed *ANN*, and *RBF-SVM* models are not generalized one like the Poly-SVM; achieving high performance both on the self-consistency and cross-validation cases. In context of our decoding problem, this advantage of Poly-SVM might be due to its learning capability based on the inner product of global data points that are selected from the training data points. However, the performance measurement of RBF kernels is based on local Gaussian kernel that might not be suitable for this data.

Figure 12 Self-consistency based BCR performance using different feature subsets

Figure 13 compares the performance of the different ML decoding models on the various images. This analysis is carried out to examine the generalization property as regards the dependency on cover image is concerned. If we imagine a continuous change from a relatively smooth image e.g. Lena towards a relatively textured image e.g. Baboon, then the behavior of the different decoding schemes could be analyzed through the surface plot. It can be easily observed that both *Poly-SVM*, and *RBF-SVM* are independent of the cover image. On the other hand, the performance of *ANN*, *Linear-SVM*, and *TD* models suffers for textured images, especially Baboon image. This is because in highly textured images, the high frequency content is large and it is difficult to survive attacks such as Gaussian noise.

Figure 13 Self-Consistency based Performance Comparison using features extracted from different images

Figure 14 shows the performance comparison of the various decoding models using different feature subsets. Using a single feature, the performance of all the decoding models is approximately the same. However, using 22 features i.e. exploiting the 22 frequency bands separately, the performance of SVM and ANN models improve appreciably. This behavior underpins the fact that nonlinear ML models are able to exploit the high dimensional feature space as compared to linear ML model like *Linear-SVM*.

Figure 14 Performance Comparison using different feature subsets

We also have analyzed the performance comparison at different feature subsets by varying image type. Figure 15 demonstrates that at single feature, the performance of all the models undergoes approximately the same decline, as we move from a relatively smooth image towards a relatively textured image. That is using a single feature, i.e. collective response of the 22 frequency bands, the models cannot learn the distortion incurred by the attack. Consequently, they cannot cope with the increase in the severity of the attack as we move from a relatively textured image. On the other hand, figure 16 demonstrates this very fact of the ML model being able to cope with severity of attack by learning the distortion occurred to the feature space when provided with 22 features. It can be observed that the nonlinear ML models, *Poly-SVM*, *RBF-SVM*, *and ANN* have been able to maintain their performance. Specifically, *Poly-SVM*, *RBF-SVM* give outstanding performance achieving BCR up to 1 across all the different types of images.

Figure 15 Self-Consistency based Performance across increasingly textured images using a single feature

Figure 16 Self-Consistency based Performance across increasingly textured images using 22 features

6.5 Performance Comparison against JPEG Compression

In order to analyze the adaptability of the proposed ML decoding system, we change the conceivable attack and retrain the ML model accordingly. In this case the attack is JPEG compression (QF=80). The sufficient statistics (equation 9) are computed in the same way and the ML model is able to learn the new distortion introduced. Consequently it is able to extract the message accurately but blindly (Table 7). The order of performance is *RBF-SVM* >*Poly-SVM* > *ANN* >*Linear-SVM* > *TD*. The training time for *Poly-SVM* is higher as compared to *RBF-SVM* and *ANN*.

Table 7: Self Consistency based Performance of the decoding models against JPEG Compression Attack

6.6 Performance Comparison against Wiener Attack

We also analyzed the performance of our adaptive decoding scheme by changing the conceivable attack from JPEG compression to Wiener estimation. The ML decoding schemes during their training phase are able to learn the novel distortion being introduced. Table 8 shows the performance comparison of the different decoding schemes against the Wiener attack. It can be observed from Table 8 that *RBF-SVM* is able to cope with such change in distortion and offers highest performance, achieving BCR=1.0 as compared to a BCR=0.8316 for TD model. The order of performance is *RBF-SVM* >*Poly-SVM* >*ANN* >*Linear-SVM* > *TD*.

Table 8: Self Consistency based Performance of the decoding models against Wiener attack

6.7 Security Analysis:

Although in this work, we are mainly emphasizing on the robustness aspects of watermarking schemes, we would also like to analyze the security related aspects. The secret key used to generate the spread spectrum sequence is very important as for as security aspects are concerned. In order to reduce the chances of security leakage, a second key is highly recommended to randomly permute the elements of the matrix of the selected DCT coefficients before embedding. This helps in introducing uncertainty about the correspondence of a codeword element and the selected DCT coefficients. The same key has to be provided at the extraction stage in order to revert the permutation process. In this way, an attacker would have no idea of which coefficients are altered corresponding to a codeword element. As for as the trained ML model used at the extraction phase is concerned, it can be made public or not according to the intended watermarking application. For the trained ML models, unlike correlation based detectors [3], due to their inherent property of transforming the input vector to higher dimensional space, it is very difficult for the attacker to gain knowledge about the key by analyzing the extraction process.

6.8 Temporal Cost based Performance Comparison

Table 9 shows the comparison between the average training and test times of *SVM*, *ANN* and *TD* models. These results are reported for 4000 training data using grid search and 12000 novel data samples respectively. The experimental results are obtained using Pentium IV machine (2.4 GHz, 512 Mb RAM). It is observed that both training and test time of SVM models are lower than that of *ANN* model. This might be due to the inherent, but efficient learning capability of SVM models. As far as the temporal cost of *TD* model is concerned, it has only the test time of 0.07 second.

Table 9: Comparison in terms of temporal cost between various decoding models

From the above discussion of the experimental results, it is observed that our proposed decoding scheme has shown improved performance than that of the *TD* model. In case of *SVM*, we are able to decode 100 percent accurate message from a distorted watermarked image. This shows that in view of conceivable attacks on a watermarked image, which are most common in real world applications of watermarking, it is far better intelligently employing a ML technique for learning the distortion introduced by the attack.

Overall, the SVM based decoding models performs superiorly. However, the parameter optimization of SVM has a great impact on both the accuracy as well training time of the decoding model. Once a high performance model is trained, in view of the intended attacks, the computational cost is comparable to that of *TD* model during the test phase. Although, we have used 22 frequency bands (7-28 in zigzag order) for watermark embedding, the proposed scheme has the potential to be more suitable for effective watermark extraction, if ML based techniques are allowed to exploit the effect of attack on the whole 63 frequency bands. In this case, a feature selection scheme such as Genetic Algorithm in view of the anticipated attack before the ML based decoding approach would be highly desirable.

7 CONCLUSIONS

We have been able to validate the exploitation of machine learning concepts for decoding purpose. The experimental results have demonstrated that both SVM and ANN decoding models are able to adopt according to the hostile environment. Especially, Poly-SVM has shown highest BCR performance. In addition, their adaptability characteristics also reduce the risk of the main security concern; unauthorized decoding. Our proposed intelligent decoding scheme has effectively extracted bits of the distorted watermark and is a generic one—not limited to a specific set of watermarking schemes. The utilization of advanced error correction strategies [30-34] in conjunction with the proposed intelligent decoding would be highly desirable, especially, in view of a cascade of conceivable attacks on the watermarked image. In the current work, we

have employed our proposed decoding scheme for image watermarking, but this technique may also be applied in both audio and video watermarking. Moreover, the proposed technique is easy to implement and has strong potential of being utilized in medical image watermarking, like remote diagnosis aid applications and telesurgery, which utilizes widely distributed sensitive medical information. In such applications, the performance of message recovery could have severe affect on a patient's life.

REFERENCES

- I. J. Cox, M.L. Miller, J. A. Bloom, Digital Watermarking and Fundamentals, Morgan Kaufmann, San Francisco, 2002.
- [2]. M. Barni, and F. Bartolini, Watermarking systems engineering: Enabling digital assets security and other application, Marcel Dekker, Inc. New York, 2004.
- [3]. Ingemar J. Cox, Gwenaë Doërr, Teddy Furon, Watermarking Is Not Cryptography, Springer, LNCS 4283, (2006) 1–15.
- [4]. L. P. Freire, P. Comesaña, J. R. T. Pastoriza, F. González. Watermarking security: a survey. Transactions on Data Hiding and Multimedia Security I, 4300, (2006) 41-72.
- [5]. J. Fridrich, T. Penvy, Merging Markov and DCT Features for Multi-Class JPEG Steganalysis, *Proc.* SPIE Electronic Imaging, Photonics West, January 2007.
- [6]. Y. Fu, R. Shen H. Lu, Optimal watermark detection based on support vector machines, Proc. of International Symposium on Neural Networks, Dalian, China, August 19-21, 2004, .552-557.
- [7]. S.Lyu, H. Farid, Detecting Hidden Messages Using Higher-Order Statistics and Support Vector Machines, Lecture Notes in Computer Science 2578, (2002) 340-354.
- [8]. R. Shen, Y. Fu, H. Lu, A novel image watermarking scheme based on support vector regression, Journal of Systems and Software 78 (1), (2005) 1-8.
- [9]. Huiqin Wang, Li Mao, Keshan Xiu, New Audio Embedding Technique Based on Neural Network, Proc. of First International Conference on Innovative Computing, Information and Control 3, (2006) 459-462.
- [10]. S. Kırbız, Y. Yaslan, B. Günsel, Robust Audio Watermark Decoding By Nonlinear Classification, European Signal Processing Conference 13, (Sep. 2005) 4-8.
- [11]. J. Sang, X. Liao, M. S. Aslam, Neural-network-based zero-watermark scheme for digital images Optical Engineering.45 (9), 2006.
- [12]. Zhang Li, Sam Kwong, Marian Choy, Wei-wei Xiao, Ji Zhen, Ji-hong Zhang, An Intelligent Watermark Detection Decoder Based on Independent Component Analysis, Springer Berlin LNCS 2939 (2004) 223-234.
- [13]. M. Barni, F. Bartolini, A.D. Rosa A. Piva, A new decoder for the optimum recovery of nonadditive watermarks, IEEE Trans. on Image Processing 10 (5), (2001) 755–766.
- [14]. J.R. Hernandez, M. Amado F. Perez-Gonzalez, DCT Domain watermarking techniques for still images: Detector performance analysis and a new structure, IEEE Trans. On Image Processing, 9 (1), (2000) 55–68.
- [15]. A. Nikolaidis I. Pitas, Optimal detector structure for DCT and subband Domain Watermarking, IEEE ICIP, (2002) 465–467.
- [16]. A. Briassouli, P. Tsakalides A. Stouraitis, Hidden messages in heavy-tails: DCT domain watermark detection using alpha-stable models, IEEE Trans. on Multimedia, 7 (4), (2005) 700–715.
- [17]. A. Briassouli M.G. Strintzis, Locally optimum nonlinearities for DCT watermark detection, IEEE Trans. on Image Processing, 13 (12), (2004) 1604–1607.
- [18]. A. Khan, A Novel Approach to Decoding: Exploiting Anticipated Attack Information Using Genetic Programming, International Journal of Knowledge-Based Intelligent Engineering Systems, 10 (5), (2006) 337-347.
- [19]. A. Khan, Anwar M. Mirza, Genetic perceptual shaping: Utilizing cover image and conceivable attack information during watermark embedding, Journal of Information Fusion, Elsevier Science, 8, (4), (2007) 354-365.

- [20]. A. Khan, Intelligent perceptual shaping of a digital watermark, PhD Thesis, Faculty of Computer Science, GIK institute, Pakistan, May 2006.
- [21]. A. Khan, I. Usman, A. M. Mirza, Genetic Perceptual Shaping of a Digital Watermark: Embedding Watermark in view of a Cascade of Conceivable Attacks, Pattern Recognition, Elsevier Science, 2006 (submitted).
- [22]. F. Bartolini, M. Barni, A. Piva, Performance analysis of ST-DM watermarking in presence of nonadditive attacks, IEEE Trans. on Signal Processing, 52 (10), (2004) 2965-2974.
- [23]. Q. Li, G. Doerr, I. J. Cox, Spread-Transform Dither Modulation using a Perceptual Model,. Proc. of the IEEE International Workshop on Multimedia Signal Processing, 2006.
- [24]. S. Fahad Tahir, A. Khan, A. Majid, A. M. Mirza, Support Vector Machine based Intelligent Watermark Decoding for Anticipated Attack, Proc. of the XV International Enformatika Conference, October 22-24, 2006, Barcelona, Spain.
- [25]. http://www.mathworks.com
- [26]. S. Haykin, Neural Networks A Comprehensive Foundation, 2nd ed., Pearson Education, Canada, 1999.
- [27]. Hagan, M. T. M. Menhaj, Training feedforward networks with the Marquardt algorithm, IEEE Transactions on Neural Networks 5 (6), (1994) 989-993.
- [28]. R. O. Duda, P. E. Hart, D. G. Stork, Pattern Classification, John Wiley & Sons, Inc., New York 2, 2001.
- [29]. C. W. Hsu, C. C. Chang, C. J. Lin, A practical guide to Support Vector Machines, Technical report, Department of Computer Science & Information Engineering, National Taiwan University, 2003.
- [28]. S. Baudry, J. F. Delaigle, B. Sankur, B. Macq, H. Maitre, Analyses of error correction strategies for typical communication channels in watermarking, Signal Processing, Elsevier Science 81, (2001) 1239-1250.
- [29]. B. Ghoraani, S. Krishnan, Chirp-Based Image Watermarking as Error-Control Coding, Proc. of International Conference on Intelligent Information Hiding and Multimedia, (20067) 647-650.
- [30]. E. Esen, A.A.Alatan, Data hiding using trellis coded quantization, Proc. IEEE International Conference on Image Processing, (2004) 59-62.
- [31]. F. I. Koprulu, Application to low-density parity-check codes to watermark channels, Ms. Thesis, Electrical and Electronics Bogaziei University, Turkey, 2001.
- [32]. N. Cvejic, D. Tujkovic T. Seppänen, Increasing robustness of an audio watermark using turbo codes, Proc. IEEE International Conference on Multimedia & Expo, Baltimore, MD, (2003) 1217-1220.
- [33]. C. C. Chang, C. J. Lin, LIBSVM: a library for Support Vector Machines, at http://www.csie.ntu.edu.tw/~cjlin/libsvm/. 2001
- [34]. A. Majid, Optimization and combination of classifiers using Genetic Programming, PhD Thesis, Faculty of Computer Science, GIK institute, Pakistan, May 2006.
- [35]. O. Chapelle, V. Vapnik, O. Bousquet, S. Mukherjee, Choosing Multiple Parameters for Support Vector Machines, Machine Learning, 46 (1-3), (2002) 131-159.
- [36]. C. Staelin, Parameter selection for Support Vector Machines, Technical report, HP Labs, Israel, 2002.
- [37]. Wayne T. Dejarnette, Web Technology and its Relevance to PACS and Teleradiology, Applied Radiology, August 2000.
- [38]. DICOM (Digital Imaging and Communications in Medicine), Part 5(PS3.5-2001): Data Structures and Encoding, Published by National Electrical Manufacturers Association, 1300 N. 17th Street Rosslyn, Virginia 22209 USA, 2001.
- [39]. D. Osborne, Embedded Watermarking for Image Verification in Telemedicine, PhD Thesis, university of Adelaide, Australia, 2005.

- [40]. http://www.research.ibm.com/trl/projects/RightsManagement/
- [41]. M. L. Miller, I. J. Cox, J.A. Bloom, Watermarking in the Real World: An Application to DVD, Proc. Of Multimedia & Security Workshop, ACM Multimedia, (1998) 71-76.
- [42]. P. Blythe, J. Fridrich, Secure Digital Camera, Department of Electrical and Computer Engineering, SUNY Binghamton, Binghamton, NY 13902-6000. 2004
- [43]. http://news.bbc.co.uk/2/hi/technology/6361891.stm

Tables

Table 2: Parameters of input dataset

Type of Images	Gray Scale
Number of images	5
Name of Images	Baboon, Lena, Trees, Boat & couple.
Image Size	256×256
Message Size	128 bits
Number of keys	25
Type of Attack	Gaussian Attack
Severity of attack	σ=10

Table 2: Parameter settings for ANN based decoding model effective against Gaussian noise Attack

	Neural Network Parameters	Selected Values
1.	Input layer neurons	22
2.	Number of hidden layers	2
	a. Neurons in 1st hidden layer	25
	b. Neurons in 2nd hidden layer	15
3.	Number neurons in the output layer	1
4.	Activation function of two hidden layers	tansig
5.	Activation function of output layer	purelin
6.	Training algorithm	Levenberg-Marquardt
7.	No of epochs	30
8.	Stopping criterion	$\xi_{avg} \leq 0.001$

Table 3: Activation functions used in the proposed ANN decoding model

Function Name	Mathematical Form	Derivatives
Pure Linear	f(x) = x	Constant
Tangent Sigmoid	$f(x) = \frac{2}{[1 + \exp(-2x)] - 1}$	f'(x) = 2f(x)[1-f(x)]

Outsolui noise / hutek							
Training data	BCR Performance	Test data	BCR Performance				
(bits)	(Self-consistency)	(bits)	(Cross validation)				
4000	0.9830	12000	0.98433				
4000	0.9805	12000	0.98517				
4000	0.98775	12000	0.98275				
4000	0.98475	12000	0.98375				

0.984

Avg. BCR

Gaussian noise Attack

0.984

SVM Kernels	С	γ	Training Data (bits)	BCR	Avg. BCR	Test Data (bits)	BCR	Avg. BCR
	48.503	-	4000	0.9852		12000	0.9855	
	48.503	-	4000	0.9832		12000	0.98617	
Linear	111.43	-	4000	0.9868	0.9853	12000	0.9850	0.9855
	111.43	-	4000	0.9860		12000	0.98525	
	[0.4 -2]	194	4000	1.00		12000	1.00	
	[0.4 -2]	194	4000	0.9998		12000	1.00	
Poly	[0.4 -2]	194	4000	1.00	1.00	12000	1.00	1.00
	[0.4 -2]	194	4000	1.00		12000	1.00	
	0.75786	5.2768	4000	0.9850		12000	0.98483	
	1.3195	6.9644	4000	0.9875		12000	0.98475	
RBF	1.000	1.7411	4000	0.9868	0.9877	12000	0.98333	0.9840
	2.2974	9.1896	4000	0.9915		12000	0.98325	

Table 5: BCR performance of SVM models using 4-fold cross validation against Gaussian noise Attack

Training Data (bits)	BCR Performance (Self- consistency)		BCR Performance (Cross validation)
4000	0.9992	12000	0.9762
4000	0.9998	12000	0.9727
4000	0.9998	12000	0.9768
4000	0.9998	12000	0.9727
Avg. BCR	0.9997		0.9746

 Table 6: BCR performance of ANN model using 4-fold cross validation
 against Gaussian noise Attack

Table 7: Self Consistency based Performance decoding models against JPEG Compression Attack

Type Of	Type Of	Data Size	Feature	ML	Parameters		Time	BCR	Hernandez Scheme
Attack	Attack	(bits)	Set	Models	С	γ	(sec.)		(BCR)
				Linear SVM	2	128	40	0.9266	
				Poly SVM	2	128	6895	0.9942	
JPEG	QF=80	16000	22	RBF SVM	2	128	697	0.9998	0.9119
				ANN	Hidden layers=3[8,4,2]	-	160	0.9431	

Type Of Attack	Intensity Of Attack	Data Size (bits)	Feature Set	ML Models	Param	eters	Time (sec.)	BCR	Hernandez Scheme (BCP)
					C	γ			(DCK)
				Linear SVM	2		49	0.9191	
				Poly SVM	2	1.4	573	0.9490	
Weiner	Window size	16000	22	RBF SVM	2	16	713	1.0	0.8316
	=3x 3			ANN	Hidden layers=3 [8,4,2]	-	284	0.9324	

Table 8: Self Consistency based Performance of decoding models against Wiener Attack

Table 9: Comparison in terms of temporal cost between various decoding models against Gaussian noise Attack

	Training Time	Test Time
Decoding Models	Selecting parameters	For 12000 data samples
	for 4000 data samples	1 of 12000 data samples
Linear-SVM	15 min	0.060 sec
Poly-SVM	19 min	0.072 sec
RBF-SVM	17 min	0.075 sec
ANN (30 epochs)	30 min	0.090 sec
TD model	-	0.070 sec



Figure 2: Distribution of sufficient statistics of the maximum likelihood based decoding system corresponding to +/-1 bits; (a) before and (b) after Gaussian Attack (σ =10)



Figure 2: Basic block diagram of the proposed ML decoding system



Figure 3: BCR performance dependency of RBF-SVM based decoding model on parameters C and $\boldsymbol{\gamma}$



Figure 4: Watermark extraction



Figure 5: Original Couple image



Figure 6: Watermarked Couple image



Figure 7: Distribution of the watermark for Couple image



Figure 8: Pixel intensity difference between original and watermarked images



Figure 9: Pixel intensity difference between watermarked and attacked watermarked images



Figure 10: Gaussian noise attacked watermarked Couple image



Figure 11: BCR performance comparision using 4-fold cross-validation and 22 features



Figure 12: Self-consistency based BCR performance using different feature subsets



Figure 13: Self-Consistency based Performance Comparison using features extracted from different images



Figure 14: Performance Comparison using different feature subsets



Figure 15: Self-Consistency based Performance across increasingly textured images using a single feature



Figure 16: Self-Consistency based Performance across increasingly textured images using 22 features