Statistically Undetectable JPEG Steganography: Dead Ends, Challenges, and Opportunities

Jessica Fridrich
SUNY Binghamton
Department of ECE
Binghamton, NY, 13902-6000
001 607 777 2577
fridrich@binghamton.edu

Tomáš Pevný
SUNY Binghamton
Department of CS
Binghamton, NY, 13902-6000
001 607 777 5689
pevnak@gmail.com

Jan Kodovský
SUNY Binghamton
Department of ECE
Binghamton, NY, 13902-6000
001 607 777 5689
jan@kodovsky.com

ABSTRACT

The goal of this paper is to determine the steganographic capacity of JPEG images (the largest payload that can be undetectably embedded) with respect to current best steganalytic methods. Additionally, by testing selected steganographic algorithms we evaluate the influence of specific design elements and principles, such as the choice of the JPEG compressor, matrix embedding, adaptive content-dependent selection channels, and minimal distortion steganography using side information at the sender. From our experiments, we conclude that the average steganographic capacity of grayscale JPEG images with quality factor 70 is approximately 0.05 bits per non-zero AC DCT coefficient.

Categories and Subject Descriptors

I.4 [**Image Processing and computer vision**]

General Terms

Algorithms, Security, Theory

Keywords

Steganalysis, steganography, capacity, blind steganalysis, JPEG.

1. INTRODUCTION

Steganography is the art of undetectable communication. As opposed to cryptography that conceals the content of a message by encrypting and then communicating it in an overt manner, steganography achieves privacy by hiding the very existence of the message in an innocent looking cover object. Steganography is considered broken if a mere presence of secret message is detected. The concept of steganographic security (statistical undetectability) has been formalized by Cachin [6].

MM&Sec'07, September 20–21, 2007, Dallas, Texas, USA.

Most steganographic systems today hide messages by slightly modifying an existing cover object, such as a digital image. The JPEG, being the most common image format in use today, received by far the most attention from the stego community. Practical stego systems were designed either using heuristic principles or constructed secure with respect to a given model. One of the first JPEG stego schemes was JSteg[1]. It employed the widely-used embedding paradigm called Least Significant Bit (LSB) embedding applied to quantized DCT coefficients. The message bits were simply embedded as LSBs in DCT coefficients different from 0 and 1. This exclusion seemed necessary because the values 0 and 1 constitute an LSB pair (values that only differ in their LSBs) and by allowing changes in coefficients equal to 0 a large distortion would be introduced. The embedding path through the image was originally sequential and in later implementations extended to a pseudo-random path. This type of embedding is easily detectable even at very low embedding rates using the histogram attack [38] (for sequential embedding) and the attacks described in [40, 23, 41]. In particular, it was reported in [22] that embedding rates as low as 0.05 bits per non-zero *usable* coefficient (coefficient not equal to 0 or 1) can be reliably detected using the generalized category attack.

The high detectability of JSteg is due to the fact that it introduces characteristic artifacts into the first order statistics (histogram) of DCT coefficients. The next generation of stego methods thus focused on preserving statistical properties of cover images. Such methods were later termed as containing "statistical restoration" [35]. The idea is to divide the cover object into two disjoint parts, embed the message in one, and use the other part to perform "corrections" in order to preserve selected statistical quantities, most frequently the histogram of DCT coefficients [30, 28, 19, 8, 35]. A related strategy is the basis of Model Based Steganography [31], where a *model* of the DCT coefficients is preserved. This had the advantage that relatively high capacity schemes could be obtained that were able to preserve not only the model of the global histogram of DCT coefficients but also all 64 histograms of individual DCT modes or even a selected higher-order statistics [32].

Among the most notable stego methods that were designed from heuristic principles, we mention the F5 algorithm [37]. The author correctly attributed the high detectability of methods that use LSB embedding to the embedding operation itself—the flipping of LSBs. This idem-

[1] `http://zooid.org/~paul/crypto/jsteg/`

potent embedding operation is indeed known to be highly detectable in the DCT domain as well as the spatial domain [20]. The embedding operation in F5 decreases the absolute value of the coefficients by 1, which, in combination with a few other tricks, freed the histogram from any obvious artifacts. F5 incorporated another important design element called Matrix Embedding, which is a coding scheme that increases the embedding efficiency (the number of bits embedded per unit distortion).

Another general direction in steganography, that recently received considerable attention, can be loosely termed "minimal distortion" embedding. Each coefficient is assigned a scalar value expressing the contribution of making an embedding change at that coefficient to overall detectability. If the raw, uncompressed cover image is available to the sender rather than just its JPEG compressed form, the sender can use the knowledge of the unquantized DCT coefficients to jointly minimize the overall distortion due to quantization and embedding. This type of embedding is called Perturbed Quantization [16] and was also utilized in the MMx stego system [21]. The guiding design principle of these methods is the belief that the smallest the embedding distortion, the harder it is to detect the embedding changes. In other words, the methods are focusing on increasing the embedding efficiency.

For completeness, we mention two other methods designed for the JPEG format, which is the JPEG-compatibility-steganalysis resistant method [27] and the recently proposed YASS [34]. Both methods essentially embed data in the spatial domain in a robust manner and then distribute the image as JPEG. The embedded message thus must be robust to JPEG compression, which can be arranged using error correction. YASS has been shown to be undetectable using current best blind steganalysis classifiers with payload of approximately 0.05 bits per non-zero DCT coefficient.

In this paper, we aim at finding the steganographic embedding capacity for covers formed by JPEG files given the current state of the art in steganography and steganalysis. Steganographic capacity is the largest payload that can be undetectably embedded in an image. We are also interested how various design elements, such as matrix embedding, adaptive content-dependent selection channels, or improved embedding efficiency influence statistical detectability.

Due to the lack of accurate statistical models for natural images, we do not approach the problem from a theoretical model for DCTs. We believe that in order to obtain concrete results, the model would have to be simplified to a degree that would make the results less relevant. Instead, we turned our attention to the latest blind steganalysis classifiers. Since the feature sets employed in classification are capable of detecting a very wide range of stego systems, the features are obviously mapping out the space of JPEG images very well. It thus makes sense to use the features extracted from a large number of cover images as an empirical model for JPEG images and evaluate security of stego schemes with respect to this large-dimensional model. In fact, this approach can be viewed from the point of view of Cachin's definition of steganographic security. The cluster of cover image features maps out the pdf of cover images as in Monte Carlo simulations. The security of a concrete stego system can then be evaluated by the Kullback-Leibler distance between the empirical pdfs of cover and stego images in the feature space. In this paper, instead of the KL distance, we measure the separation of feature clusters by training a support vector machine and use the Bayesian point on the receiver operating characteristic curve (ROC) as the measure of detectability. By using this measure instead of the KL distance, we can actually attribute a very specific meaning to the detectability score, which is the smallest overall error that a specific classifier can achieve.

To make this paper self-contained, in the next section we briefly summarize the necessary background from steganography and steganalysis and give a more precise meaning to some of the concepts discussed above. The tested steganographic methods are described in Section 3. The setup of all experiments including the testing methodology used in this paper and all experimental results and their interpretation are in Section 4. The paper is concluded in Section 5. Everywhere in the paper we use the calligraphic font for sets and boldface for vectors or matrices. Whenever important concepts are defined, they are highlighted in italics.

2. BACKGROUND

A steganographic system is a mechanism that embeds a *secret message* $m \in \mathcal{M}$ in a *cover object* $x \in \mathcal{C}$ using a secret shared *stego key* $k \in \mathcal{K}$, obtaining the *stego object* $y \in \mathcal{C}$ that carries m. The set $\mathcal{M}$ is the set of all messages that can be communicated, $\mathcal{K}$ is the set of all stego keys, and $\mathcal{C}$ is the set of all available cover objects. The embedding mechanism is formally captured using the *embedding mapping*

$$Emb : \mathcal{C} \times \mathcal{M} \times \mathcal{K} \to \mathcal{C},\ y = Emb(x, m, k).$$

This mapping has to be supplied with the corresponding *extraction mapping* that extracts the hidden message from the stego object if the correct stego key is provided

$$Ext : \mathcal{C} \times \mathcal{K} \to \mathcal{M},\ Ext(y, k) = m.$$

The *embedding capacity* of the stego scheme is $\log_2 |\mathcal{M}|$ bits. Defining a distance d on $\mathcal{C}$, $d : \mathcal{C} \times \mathcal{C} \to [0, +\infty)$, the *embedding efficiency* is the number of embedded bits per unit distortion

$$e = \log_2 |\mathcal{M}| / E\{d(x, y)\}, \tag{1}$$

where $E\{d(x, y)\}$ is the expected value of the embedding distortion taken over uniformly distributed keys, messages, and covers.

If x and y are vectors, the places where embedding changes occur, $x_i \neq y_i$, can be chosen pseudo-randomly or using a *selection rule*, for example, as in adaptive steganography. If the selection rule uses information calculated from some local neighborhood, we speak of a *content-adaptive* selection rule.

A steganographic system is considered *undetectable* if no statistical test can distinguish between the set of cover objects and stego objects. An information-theoretical definition of steganographic security has been given by Cachin [6]. The set of all cover objects $\mathcal{C}$ is endowed with a probability distribution function (pdf) $P_c(x)$ describing the probability of encountering a given cover object $x \in \mathcal{C}$. If the covers are selected with pdf $P_c(x)$ and embedded with a message and secret key both randomly (uniformly) chosen from their corresponding sets, we obtain the pdf of stego objects $P_s(x)$. The Kullback–Leibler distance

$$D(P_c||P_s) = \sum_{x \in \mathcal{C}} P_c(x) \log \frac{P_c(x)}{P_s(x)} \tag{2}$$

is taken as the measure of security. Stego systems with $D(P_c||P_s) < \epsilon$ are called ϵ-secure.

Because it is impossible to work with the pdf on the set of all possible covers, in practice a simplified model is accepted for covers, which enables us to make concrete claims about the stego system and measure its security. For example, modeling the cover as a sequence of realizations of an independent identically distributed (iid) random variable, any steganographic method that preserves the pdf of cover elements will be undetectable because the pdf is its complete statistical description. A more realistic model for natural images assumes that the individual cover elements form a Markov chain with some probability transition matrix [36]. While other choices are certainly possible, in this paper, we accept the following model for cover images.

Inspired by the success of recent blind steganalysis methods, we represent the cover JPEG image using a feature vector $\mathbf{f} \in \mathbb{R}^n$ in some high dimensional Euclidean space derived from its quantized DCT coefficients and its spatial representation. We select the 274-dimensional feature vector consisting of 193 extended DCT features and 81 Markov features described in [12, 33, 29]. Based on the comparison in [29], blind steganalysis constructed using this feature vector achieves performance that is one of the best compared to currently available blind steganalysis classifiers capable of detecting stego content in JPEG images [3, 2, 33, 1, 25, 10, 39, 9, 29, 12]. Since the classifier that uses these features is very sensitive to steganographic embedding while providing low false alarm ratio, the feature vector is mapping out the space of JPEG images very well. By generating the feature vector for a large number of training images, we essentially sample the distribution of covers P_c . Embedding the images with messages using a given stego technique then allows us to sample the stego image distribution P_s. As explained in the introduction, we do not measure the impact of embedding using the KL distance $D(P_c||P_s)$ and instead train a support vector machine classifier for the clusters of cover and stego images and report the minimal total average probability of error

$$P = \frac{P_{FA} + P_{MD}}{2} \tag{3}$$

on a testing set. Here, P_{FA} and P_{MD} are the probability of false alarm and missed detection, respectively. The quantity P is equivalent to the minimal total Bayes cost under equal probability of encountering a cover or stego image and equal costs of false alarms and missed detections. As opposed to the KL distance, the probability P has a specific meaning and is easily interpretable as the minimal average probability of false decision using the employed classifier. We remark that this approach to evaluating steganographic security is consistent with the concept of defining security with respect to a steganalyzer as proposed in [7].

3. TESTED STEGANOGRAPHIC METHODS

In this paper, we selected several steganographic methods that we consider the best candidates for secure steganography based on the steganalysis results reported in current literature [33, 25, 10, 39, 26, 29]. The candidates are F5 [37], Steghide [19], JP Hide&Seek[2], Model Based Steganography without deblocking [31], MMx [21], and Perturbed Quantization while double compressing [16]. We also study some modifications of these algorithms, which are described in the text. In the following text, we outline the most important and relevant embedding principles of these schemes referring the reader to the corresponding original publications for more details.

3.1 Statistics-preserving steganography (Steghide, Model Based Steganography)

We did not include in our list OutGuess [30] and other methods that use statistical restoration [35] as they are typically highly detectable. The idea behind statistical restoration could only work if we were able to restore *all* essential statistics that determine the cover model. Preserving the first order global histogram of DCT coefficients is not enough as the DCT coefficients exhibit many complex inter-block and intra-block dependencies. The additional distortion due to restoration of the coefficient histogram further disturbs these dependencies and in the end paradoxically may cause the methods to be more detectable[3]. These claims are supported by the results reported in [33] and independently in [29].

As a representative example of schemes designed to preserve the global DCT histogram [28, 19, 8], we chose the Steghide algorithm based on its steganalysis reported previously [29]. Steghide embeds by swapping DCT coefficients and thus avoids changing the histogram.

Another related class of methods includes steganographic techniques designed to preserve a *model* of the DCT coefficients rather than their statistics. Model Based Steganography (MBS) starts by dividing the cover AC DCT coefficients into two parts. The first one is not changed during embedding (the 7 most significant bits) and is used to construct 63 models for the 63 histograms of individual AC DCT modes. The model is formed by the probability distribution of LSBs in each LSB pair of coefficient values. The second part (the coefficient's LSBs) is overwritten with message bits prebiased to comply with the model. The prebiasing, which is achieved via arithmetic decompression of the message bits, optimizes the embedding efficiency. An important advantage of this approach is that it is possible to preserve not only the global histogram of all DCT coefficients but also the histograms of individual DCT modes without much increase on complexity or penalty in embedding payload. A more advanced version of this embedding principle reserves one half of embedding capacity for embedding and uses the second half to restore another important higher order statistics—the blockiness [32]. As with other schemes that utilize statistical restoration, this step actually increases the detectability of the method [33, 29], which is the main reason why we do not test MBS with deblocking in this paper.

3.2 Heuristic algorithms (F5, –F5, nsF5, JP Hide&Seek)

The F5 algorithm contains two important design principles. The first one is the character of its embedding modifications chosen in such a way that the absolute value of the DCT coefficient is always decreased by one. F5 only embeds into non-zero AC DCT coefficients. If a coefficient becomes

[2] `http://linux01.gwdg.de/~alatham/stego.html`

[3] Statistical restoration also substantially decreases embedding efficiency.

zero after embedding, which can only happen for coefficients equal to 1 or −1, so called *shrinkage* occurs and the same bit is reembedded at the next coefficient. This is necessary because the decoder reads message bits only from non-zero coefficients and will thus skip over the zeroed coefficients. In an attempt to minimize the impact of embedding on the DCT histogram, F5 skips over 50% of all DCT coefficients equal to 1 or −1. This measure was incorporated into F5 later after successful steganalysis methods for F5 appeared.

The second important element of F5 is its incorporation of *matrix embedding* (also called *syndrome coding*) using binary Hamming codes. Matrix embedding enables embedding more bits per one embedding change and thus increases embedding efficiency. For a general treatment on matrix embedding, see [4] or [13]. We explain the concept on a simple example. Let $\mathbf{x}$ denote the column of LSBs of 7 DCT coefficients and $\mathbf{m}$ the column of 3 message bits. We now show how to embed these 3 bits in 7 DCT coefficients by making at most one embedding change. We form a 3×7 binary matrix $\mathbf{H}$ whose columns are all non-zero vectors of length 3

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

During embedding, we first check if $\mathbf{Hx} = \mathbf{m}$. If equality holds, no embedding changes are necessary ($\mathbf{y} = \mathbf{x}$) and the message bits can be extracted as the syndrome of $\mathbf{y}$: $\mathbf{m} = \mathbf{Hy}$. If $\mathbf{Hx} \neq \mathbf{m}$, we first find the difference $\mathbf{Hx} - \mathbf{m}$ as a column in $\mathbf{H}$, say the j-th column. Then, we embed $\mathbf{m}$ in $\mathbf{x}$ by changing the j-th bit of $\mathbf{x}$, obtaining the modified vector of stego LSBs $\mathbf{y}$. It is easy to verify that now the recipient recovers the correct message from the stego image by forming the same matrix-vector multiplication $\mathbf{m} = \mathbf{Hy}$. Thus, we are able to embed 3 bits in 7 pixels by making on average $1 - 1/2^3$ changes (remember that no change was necessary when $\mathbf{Hx} = \mathbf{m}$, which happens with probability $1/2^3$).

This process can be obviously generalized to allow embedding p bits in $2^p - 1$ pixels by making at most 1 embedding change or using $1 - 2^{-p}$ changes on average. The relative payload we are embedding is

$$\alpha_p = p/(2^p - 1) \tag{4}$$

and the embedding efficiency is (in bits per non-zero AC DCT coefficient)

$$e_p = p/(1 - 2^{-p}). \tag{5}$$

The embedding efficiency of F5, however, is lower than (5) because of the additional changes introduced by shrinkage. Also, shrinkage increases the impact of embedding on the histogram of DCT coefficients. The negative effects of shrinkage can be eliminated in several ways. The first and obvious choice is to modify the embedding operation and *increase* the absolute value of the coefficient instead of decreasing. In the absence of shrinkage, we can embed with efficiency (5). We call this algorithm –F5 and include it in our set of tested algorithms.

The second choice to alleviate the impact of shrinkage is to apply *wet paper codes* [17]. Wet paper codes were designed to allow the sender to use side information unavailable to the decoder. In F5, the decoder reads the message from non-zero DCT coefficients. Thus, when a DCT coefficient is zeroed out, the decoder does not know that it was originally non-zero. This problem can be overcome using syndrome coding similar to matrix embedding. Let us assume that the sender wishes to embed p bits $\mathbf{m} \in \{0,1\}^p$ in n AC coefficients with LSBs $\mathbf{x} \in \{0,1\}^n$, out of which only k coefficients $x_i, i \in \mathcal{I}$, $|\mathcal{I}| = k$, are non-zero[4]. The sender modifies some $x_i, i \in \mathcal{I}$, so that the vector of modified LSBs $\mathbf{y} \in \{0,1\}^n$ satisfies

$$\mathbf{Dy} = \mathbf{m}. \tag{6}$$

The binary $p \times n$ matrix $\mathbf{D}$ is shared between both the sender and the recipient. Thus, the sender's task is to find the solution of (6) so that, $x_i = y_i$ for $i \notin \mathcal{I}$ and the Hamming weight of $\mathbf{x} - \mathbf{y}$ is as small as possible. It was shown in [17] that by choosing $\mathbf{D}$ as a random binary matrix with $p < 20$, there exist computationally efficient methods for embedding the relative payload α with embedding efficiency slightly better than the one for matrix embedding with binary Hamming codes (5). An additional advantage of using random matrices is that we now have a *continuous* family of codes whose parameters can be better adjusted to each specific payload instead of a discrete and quite sparse set offered by Hamming codes. Thus, wet paper codes can elegantly eliminate the problem of shrinkage in F5. We included in our tests a routine that simulates the embedding changes as they would be carried out in the F5 algorithm coupled with wet paper codes. We call this method nsF5 (no-shrinkage F5).

The last heuristic method we tested is the JP Hide&Seek algorithm because its statistical detectability is among the lowest based on the results reported in [29]. The details of the embedding mechanism have not been published but the source code is available from the author. We have experimentally determined that the embedding changes are mostly constrained to LSB flippings and some second LSB flippings.

3.3 Minimal distortion steganography (PQ, MMx)

Perturbed Quantization (PQ) and MMx algorithms try to minimize the embedding distortion using side information only available to the sender. PQ does so by using wet paper codes [17], while MMx uses a modified matrix embedding using binary Hamming codes. We first explain PQ.

Imagine that each DCT coefficient x_i is assigned a scalar value ρ_i expressing the embedding distortion if x_i needs to be modified during embedding. Here, we could, for example, take into account that changes of different quantized DCT coefficients by 1 may have various impact because the coefficients are multiplied by different quantization steps during decompression. Alternatively, if the uncompressed cover image is available, ρ_i can express the additional distortion caused by embedding when compared to the uncompressed (unquantized) image. Let x_i' be the value of the i-th DCT coefficient after dividing it by the quantization step but before rounding to an integer. The value of x_i' will be between two integer values $a \leq x_i' < a + 1$. We can round it either to a or to $a + 1$ in order to encode a different bit. Denoting by $r_i = x_i' - [x_i'] \in [-1/2, 1/2]$ the rounding error, then the difference in quantization errors due to rounding to a or $a + 1$ is $\rho_i = 1 - 2|r_i|$. In order to embed m bits, we select m DCT coefficients with the smallest ρ_i or, equivalently, coefficients closest to the middle of the quantization inter-

[4]In other words, we are embedding at relative payload $\alpha = p/k$ bpac.

vals ($r_i \approx 0.5$). This is the general idea behind Perturbed Quantization steganography.

In this paper, we test three versions of PQ applied to double-compressed JPEG images as described in [15]. For a fixed pair of quality factors Q_1 and Q_2, the DCT coefficient x_i can be used for embedding only when its first and second quantization steps $q_i^{(1)}$, $q_i^{(2)}$, satisfy $kq_i^{(1)} = lq_i^{(2)} + q_i^{(2)}/2$ for some integers k and l. Such coefficients are called *contributing coefficients*. For high embedding capacity, we need to have a large number of contributing coefficients. Thus, following the recommendation in [15], we selected the quality factors $Q_1 = 85$ and $Q_2 = 70$. If we now apply the minimal embedding distortion strategy described in the previous paragraph to all contributing coefficients, we obtain the first PQ method involved in our tests, which we simply denote PQ.

Looking at this approach from an even more general view, we can abstract from the concept of distortion and let ρ_i capture the *impact* of making an embedding change at coefficient x_i. In general, the distortion may not always relate to statistical detectability of embedding changes in a simple manner. For example, we may wish to assign smaller ρ_i to coefficients in textured blocks and higher values to coefficients from smooth blocks and embed a given payload with the *minimal expected impact*. This problem has been studied theoretically in [11] and the first steps toward near-optimal practical embedding schemes were presented in [14].

In this paper, we proposed two adaptive versions of PQ: texture-adaptive PQ (PQt) and energy-adaptive PQ (PQe) that differ only in the choice of the measure of local block content ρ_i. Both approaches are examples of a content-dependent adaptive choice of the selection rule. The coefficients used for embedding are selected among the contributing coefficients based on the block content captured using ρ_i rather than the rounding distortion r_i as in PQ. In both methods, ρ_i will be the same for all DCT coefficients from that block.

The texture measure in PQt is calculated from the single-compressed cover JPEG image with quality factor Q_1 decompressed to the spatial domain. Each 8×8 pixel block is divided into disjoint 2×2 blocks. For each 2×2 block, we calculate the difference between the highest and the lowest pixel value. The texture measure ρ_i is obtained as the sum of these differences over the whole 8×8 block. After obtaining the local texture values ρ_i for all contributing coefficients, in PQt we select for embedding the coefficients with the highest ρ_i.

The energy-adaptive PQe uses a different definition of block content. The values ρ_i represent the *energy* of the DCT block calculated as a sum of squares of all quantized DCT coefficients in the appropriate block. The coefficients are obtained from the single compressed image with JPEG quality factor Q_1. As in PQt, we embed into the contributing coefficients with the highest values of ρ_i.

In contrast to the original distortion-based PQ method where we embed into contributing DCT coefficients one by one starting with the coefficient with the smallest rounding error, in PQt and PQe we select the blocks starting with the block with the smallest ρ_i and always embed into all contributing coefficients from that block. This embedding strategy follows from the fact that ρ_i is a block-based concept rather than tied to a specific coefficient.

The MMx algorithm [21] provides a simple and practical (even though suboptimal) approach by allowing more than one embedding change in matrix embedding using Hamming codes. Again, we explain the principle on a simple example of embedding 3 bits into 7 coefficients $\mathbf{x}$.

The sender first uses the non-rounded value of the i-th DCT coefficient and marks down the rounding distortion $\rho_i = |r_i|$. Then, the sender tries to embed the bits using at most one change as in classical matrix embedding, registering the embedding impact as d_1. Say, if the j-th coefficient x_j had to be rounded to the "other" side, then $d_1 = 1 - \rho_j$. The sender now allows two embedding changes (two coefficients to be rounded to the other side) and lists all pairs of columns $\mathbf{c}_{j'}, \mathbf{c}_{j''}$ from $\mathbf{H}$ such that $\mathbf{c}_{j'} + \mathbf{c}_{j''} = \mathbf{c}_j$. In our example, there will always be exactly 3 such pairs. For each pair, the sender calculates the embedding impact $1 - \rho_{j'} + 1 - \rho_{j''}$. If one of these combined impacts is smaller than d_1, the sender makes embedding changes at that coefficient pair instead, obtaining $d_2 = 1 - \rho_{j'} + 1 - \rho_{j''}$. The only exception to this rule is when a coefficient is to be rounded to zero. Since the decoder reads message bits only from non-zero coefficients, instead of rounding to zero, the coefficient is rounded to 2 (if $x_i > 0$) or -2 (if $x_i < 0$). In this case the embedding distortion will be increased to $1 + \rho_j$. This choice removes the problem with shrinking a non-zero coefficient to zero and thus increases embedding efficiency.

This method is called MMe in [21] and we use this notation in this paper as well. The sender could obviously check for triples of columns or even four-tuples of columns to see if a lower embedding distortion is obtained by modifying 3 or 4 coefficients (methods MM3 and MM4).

4. EXPERIMENTS AND THEIR INTERPRETATION

In this section, we report the results of the experiments and give their interpretation. We start by describing the procedure used to generate the stego images, which was the same for each tested steganographic method.

4.1 Image database

We used a database of 6006 images in the raw format and divided it into two disjoint subsets—the subset used for training contained 3500 raw images, and subset used for testing contained 2506 raw images. The testing set contained images with completely different scenes taken by different cameras and photographers. The images in both sets contained full-resolution images (the largest was a 6 megapixel image) as well as their smaller versions obtained by resizing (the smallest image had dimensions 800×631).

4.2 Cover images

For each tested method, we prepared the cover *grayscale* JPEG image and several stego grayscale JPEG images embedded with different payloads. We opted for grayscale images because it is harder to detect hidden data in them compared to color images where steganalysis can utilize dependencies between color channels. The cover JPEG images for all methods were of quality 70 (the pair 85 and 70 was used for PQ). By a cover image, we always understand an image into which no message is embedded. JP Hide&Seek, Steghide, and MBS directly manipulate the quantized DCT coefficients of the cover JPEG file presented to them. For these three methods, the cover JPEG was obtained using

the JPEG compressor used in F5.

In all versions of PQ, the message was embedded during repetitive JPEG compression with two different quality factors. Thus, we take as cover images the double compressed images and evaluate them against the double compressed and embedded images. Indeed, this is the only reasonable option as comparing stego images with single compressed images would be highly detectable because the classifier would essentially learn to distinguish the impact of repeated compression instead of the embedding changes. The JPEG compressor used in our implementation of PQ methods was the compressor "imwrite" used in Matlab.

We present both F5 and MMx methods with the raw form of the cover in order to prevent double compression in F5 and because MMx requires the raw image. The JPEG cover for these two methods is obtained using their internal JPEG compressor to make sure that the steganalyzers are not detecting subtle differences between different JPEG compressors (which differ mostly by the implementation of the DCT transform). At this point, we would like to emphasize that despite the fact that the influence of the JPEG compressor on steganalysis can be far from negligible, this issue has so far been ignored or overlooked in current steganalysis studies. This is why we decided to study the impact of different JPEG compressors on cover images and on steganalysis in general and report the results in Section 4.6.

4.3 Payloads

We tested 5 different payloads determined for each image as a fixed percentage of non-zero AC DCT coefficients from the cover JPEG image. We call this quantity *bpac* (bits per non-zero AC coefficient). This measure of payload will allow us to clearly see the influence of various coding schemes in stego methods that use matrix embedding. We note that two other measures of payload previously used are *bpc*, or bits per non-zero DCT coefficient [12], and bits per *usable* coefficient [23].

For each stego method, we construct separate classifiers for each payload of 0.2, 0.15, 0.1, 0.09, and 0.05 bpac. The reason for including the seemingly redundant payload of 0.09 bpc is as follows. The methods that use matrix embedding using binary Hamming codes provide discontinuous embedding efficiency depending on where the relative payload falls. The values of α_p for $p = 4, \dots, 7$ are $0.267, 0.161, 0.095$, and 0.055. Thus, the payloads 0.2, 0.15, 0.1, 0.09, and 0.05 bpac will be embedded with Hamming codes with $p = 4, 5, 5, 6, 7$. Note that it is not possible to embed the payload 0.1 bpac with $p = 6$ because the relative payload $\alpha_6 = 0.095$ is slightly below 0.1. As a result, without including the payload of 0.09 bpc, we would not see the effect of embedding using the Hamming code with $p = 6$. Moreover, comparing the statistical detectability for payloads 0.095 and 0.1 bpac will allow us to evaluate the contribution of matrix embedding to algorithm undetectability.

To summarize, for each stego method and each payload, we tested the detectability of embedding by constructing a SVM classifier *trained for that specific payload*. We decided to construct a separate classifier for each payload as opposed to one classifier trained for all payloads because, as already explained in Section 2, we are interested in how separable the cluster of cover and stego image features are in the feature space for each payload.

4.4 Classifier

For classification, we used soft-margin support vector machines (C-SVM) with Gaussian kernel [5]. The training set for each classifier consisted of 3400 examples of cover, and 3400 examples of stego images embedded by a given algorithm with a given payload. Because some embedding algorithms fail on singular images (night shots, blue sky, etc.), in order to arrange for the same number of training images for each method, we trained on 3400 images instead of all 3500. The two hyper-parameters of the C-SVMs were the penalty parameter C and the Gaussian kernel width γ. They were determined by estimating the error probability by means of a 5-fold cross-validation on the following multiplicative grid

$$(C, \gamma) \in \{(2^i, 2^j) | i \in \mathcal{Z}, j \in \mathcal{Z}\}.$$

To overcome the problem that this grid is unbounded, we exploit the fact that for most practical problems, the error surface of SVMs is convex. The grid-search for a particular SVM started by evaluating all points in the set

$$(C, \gamma) \in \{(2^i, 2^j) | i \in \{1, \dots 16\}, j \in \{-19, \dots, -3\}\}.$$

After that, we checked if the best point (determined by the smallest cross-validation error) was at the boundary of the grid. If so, we enlarged the grid for this machine in the direction perpendicular to the boundary the best point laid on. We kept doing this until the best point ended up within the explored grid (not on the boundary). This simple algorithm ensured that the distance between the best point and the optimal point was small (within the size of the grid) under the convexity assumption.

Once we found suitable hyper-parameters (C, γ) of the C-SVM, we used the whole training set to train the SVM. We would like to point out, that the error probability $P = 1/2(P_{FA} + P_{MD})$ we used to evaluate the classifiers penalizes false positive and false negative errors equally. Thus, we do need to shift the threshold after the SVM is trained or use weighted Support Vector Machines [24] in order to find the desired point on the ROC curve.

4.5 Results

The experimental results are displayed in Figure 1. Overall, we conclude that for embedding rates between 0.05–0.2 bpac, Steghide, MBS, and JPHS are the most detectable algorithms. The best performance for payloads larger than 0.09 bpac was obtained using the texture and energy-adaptive versions of PQ. The best method for payload 0.05 bpac was the MM3 algorithm. If we declare as statistically undetectable schemes whose error probability $P = 1/2(P_{FA} + P_{MD})$ is above 40, only the MM3 and PQt algorithms at 0.05 bpac could be declared undetectable. We now interpret the results for each algorithm.

PQ: It is interesting that the original non-adaptive version of PQ that strictly minimizes embedding distortion is detectable relatively accurately. This result is also currently the most successful attempt at detecting PQ compared to previously published attacks [18]. The comparison between the original version of PQ and its adaptive versions indicates that incorporating local texture properties into selection rules for steganographic schemes may substantially increase their security with respect to the tested blind steganalyzer. We note that adaptive schemes open door to targeted attacks because the schemes leak information about the location of embedding changes to the attacker.

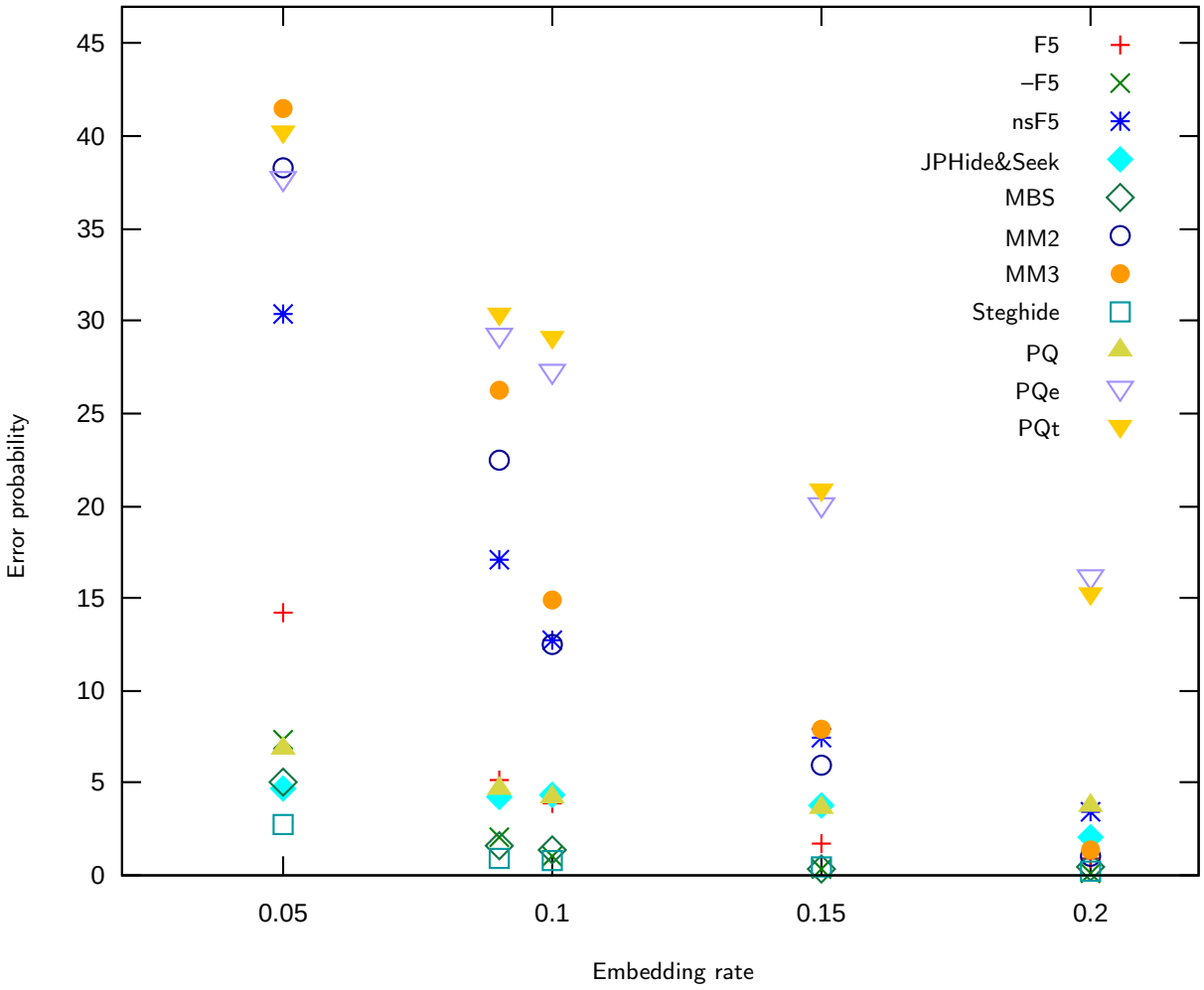

Figure 1: Error probability $P = \frac{1}{2}(P_{FA} + P_{MD})$ of classifiers designed for a specific message length on the testing set.

	Error for bpac				
Algorithm	0.05	0.09	0.10	0.15	0.20
F5	12.94%	4.37%	3.77%	1.48%	0.80%
–F5	7.37%	2.18%	1.16%	0.32%	0.12%
nsF5	28.90%	15.51%	11.18%	6.19%	3.21%
JPHide&Seek	4.01%	3.75%	4.09%	3.65%	2.11%
MBS	4.61%	1.52%	1.12%	0.64%	0.36%
MM2	38.26%	21.88%	12.68%	6.05%	0.82%
MM3	40.86%	25.57%	14.81%	8.11%	1.24%
Steghide	2.58%	0.76%	0.68%	0.38%	0.26%
PQ	6.73%	4.67%	4.49%	4.11%	3.19%
PQe	37.80%	28.79%	27.28%	20.20%	16.15%
PQt	40.06%	31.11%	28.79%	21.74%	15.27%

Table 1: Error probability $P = \frac{1}{2}(P_{FA} + P_{MD})$ of classifiers designed for a specific message length on the testing set.

For the largest payload of 0.2 bpac, both adaptive PQ methods achieved by far the best security when compared to the other methods. The PQ methods could be further improved using matrix embedding by reserving for embedding a larger part of coefficients and embedding into all of them with decreased number of changes as described in [11].

It can be argued, however, that comparing PQ with other methods is not completely fair as PQ produces double compressed JPEG images and relies on side information. Nevertheless, we can correctly state that within the class of double compressed images, the stego images produced by the adaptive versions of PQ are statistically undetectable.

MMx: The MMx methods also rely on side information and produce, as all other methods, only single compressed images. Their undetectability is stemming more from the incorporation of improved matrix embedding, though, as can be clearly seen by comparing the results for payloads 0.09 bpac and 0.1 bpac. While the error rates for schemes not using matrix embedding increased proportionally, schemes that incorporate matrix embedding experience a sudden change, which is most pronounced for the MMx algorithms. The figure also nicely demonstrates the decrease in detectability between MM2 and MM3. The benefit of making more than 1 change starts showing up increasingly more with decreasing payload. This is natural as shorter payloads allow Hamming codes with larger p and thus provide more options to generate the same syndrome, increasing the probability that two or more changes will introduce smaller distortion than a single, uniquely determined change.

The MMx algorithm, too, could be modified in the same manner as nsF5 by incorporating wet paper codes to eliminate the need to always change 1's into 2's. We leave testing such modifications and other advanced versions to our future effort.

F5: The –F5 algorithm is markedly worse that the original F5 despite the fact that it does not have shrinkage. This is an interesting result which validates the choice made by the F5 designer. The F5 without shrinkage (nsF5) performs consistently and significantly better than the original F5. In fact, this is our best tested algorithm that embeds directly into the DCT coefficients without relying on any side information at the sender (the uncompressed cover). It appears that in view of the gain obtained by adaptive PQ schemes, incorporating adaptivity into nsF5 might further improve its security.

It is also interesting to see that for the payload of 0.2 bpac, the error probability of nsF5 is more than twice larger than for both MMx algorithms despite the fact that MMx minimizes distortion and utilizes side information.

Overall, the results clearly show that heuristically designed schemes with improved embedding efficiency using matrix embedding are better than statistics-preserving schemes (Steghide, MBS). It appears that the best approach for embedding in cover JPEG files would be to combine adaptive coefficient selection with matrix embedding (or wet paper codes). This task would require a careful investigation of how to define the context-aware embedding impact at every coefficient.

For convenience, we also report the results from Figure 1 numerically in Table 1.

4.6 Influence of the JPEG compressor

In this section, we study the influence of the JPEG compressor on steganalysis.

By comparing quantized DCT coefficients from JPEG images generated by compressing the same raw image using different compressors, one can obtain surprisingly different DCT planes. We took three different JPEG compressors – the compressor "imwrite" used in Matlab (IMW), the compressor implemented in F5 (F5), and the "convert" routine from ImageMagick package[5] (CON) – and compressed with them over 4000 different raw images to JPEG with quality factor 70. While the number of non-zero DCT coefficients for the same image for all three compressors varied on average only by about 1%, the number of different DCT coefficients was much more volatile. In Table 2, we show the ratio between the number of different DCT coefficients when compressing the same image using three different compressors and the number of all non-zero DCT coefficients in that image, averaged over all 4000 images.

Thus, by generating the covers using a different JPEG compressor than the one used in the stego method might measurably skew the steganalysis results or introduce systematic errors. To better understand the extent of such systematic errors, we performed two types of experiments. In the first test, we investigated the following situation. When testing a steganalyzer on multiple stego methods, one of which is F5, it is too tempting to introduce the following

[5] `http://www.imagemagick.org`

	IMW–F5	IMW–CON	F5–CON
QF 70	5.6%	14.7%	11.3%
QF 85	9.7%	22.6%	16.7%

Table 2: Ratio between the number of different DCT coefficients obtained from the same image using three different compressors and the number of all non-zero DCT coefficients in that image. Results are averaged over 4000 images.

inconsistency. In order to avoid double compressed images, raw images are sent to F5 for embedding, while the cover images are prepared separately. Thus, the JPEG cover images may be inadvertedly obtained using a different compressor than the one used in F5. The differences between incompatible compressors then artificially contribute to detectability of F5. To evaluate how serious this problem might be, we prepared the cover JPEG images using three different compressors, while the stego images were always obtained using nsF5 with the original F5 compressor. In Table 3, we show the detection error for all three situations—cover using IMW vs. stego using nsF5, cover using F5 vs. stego using nsF5, and cover using CON vs. stego using nsF5. One can see that the incompatible JPEG compressor may falsely increase the statistical detectability of nsF5 by up to 15% (!) for the shortest messages. The impact for longer messages is proportionally smaller. Thus, the impact of the compressor can be quite substantial and care needs to be taken to carry out the experiments properly.

	Detection error for bpac				
Compressor	0.05	0.09	0.10	0.15	0.20
cover (CON)	26.8%	14.9%	10.4%	6.7%	2.4%
cover (F5)	28.9%	15.5%	11.2%	6.2%	3.2%
cover (IMW)	14.3%	8.6%	6.3%	4.0%	2.0%

Table 3: The detection error for nsF5 when cover images were prepared using three different compressors while nsF5 always used the F5 compressor. Results are shown for the testing set.

In the second experiment, we implemented nsF5 with all three compressors above, this time always making sure that the cover images were obtained with the same compressor as the one used to generate stego images. As before, we trained a steganalyzer for each payload and compared the error probability for all three JPEG compressors (see Table 4). The differences between detection errors for all three versions of nsF5 are less than 2% and are thus negligible.

	Detection error for bpac				
Compressor	0.05	0.09	0.10	0.15	0.20
nsF5 (CON)	30.7%	15.5%	10.9%	6.0%	2.4%
nsF5 (F5)	28.9%	15.5%	11.2%	6.2%	3.2%
nsF5 (IMW)	28.9%	15.3%	10.8%	6.0%	2.9%

Table 4: Detection error for nsF5 implemented using three different JPEG compressors. Results are shown for the testing set.

5. CONCLUSIONS

In this paper, we investigate statistical detectability of current steganographic methods for JPEG images with several goals in our mind:

- to determine the maximal relative payload at which the methods become statistically undetectable,
- to study the influence of various design elements, such as matrix embedding, adaptive selection rules, minimizing embedding impact using side information at the sender, and the type of the embedding modification,
- to evaluate the influence of different JPEG compressors on steganalysis,
- to present steganalysis results for the most promising candidate methods as well as their modifications.

In summary, the largest payload that can be undetectably embedded in a JPEG file based on the current best blind steganalysis classifiers is about 0.05 bits per non-zero AC DCT coefficient. This result is consistent with the findings reported in [34]. The methods that can achieve this are texture-adaptive Perturbed Quantization (PQt) [16] (while double compressing) and the MM3 method [21]. The PQt method is also the best performer for payloads above 0.05 bpac. The best method among those not utilizing any side information at the sender (the uncompressed image) is the F5 algorithm with shrinkage removed using wet paper codes (nsF5). Overall, we found out that matrix embedding markedly improves security as it dilutes the number of embedding changes. The importance of matrix embedding increases with decreasing payload. Additionally, methods that minimize embedding impact perform very well, including those where the embedding impact takes into account local texture (both adaptive versions of PQ).

If we were to look into the future and provide guidelines for design of the best steganographic method for the JPEG format, we would recommend methods that minimize the embedding impact that takes into account local image texture and that can apply syndrome coding methods (e.g., matrix embedding or wet paper codes) to further decrease the impact of embedding.

6. ACKNOWLEDGMENT

The work on this paper was supported by Air Force Research Laboratory, Air Force Material Command, USAF, under the research grant number FA8750-04-1-0112. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation there on. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of Air Force Research Laboratory, or the U.S. Government.

7. REFERENCES

[1] I. Avcibas, M. Kharrazi, N.D. Memon, and B. Sankur. Image steganalysis with binary similarity measures. *EURASIP Journal on Applied Signal Processing*, 17:2749–2757, 2005.

[2] I. Avcibas, N.D. Memon, and B. Sankur. Steganalysis using image quality metrics. In E.J. Delp and P.W. Wong, editors, *Proceeedings of SPIE Electronic Imaging, Security and Watermarking of Multimedia Contents III*, volume 4314, pages 523–531, 2001.

[3] I. Avcibas, B. Sankur, and N.D. Memon. Image steganalysis with binary similarity measures. In *Proceedings of International Conference on Image Processing*, volume 3, pages 645–648, 2002.

[4] J. Bierbrauer. On Crandall's problem. *Personal communication available from http://www.ws.binghamton.edu/fridrich/covcodes.pdf*, 1998.

[5] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[6] C. Cachin. An information-theoretic model for steganography. In D. Aucsmith, editor, *Information Hiding, 2nd International Workshop*, volume 1525 of *Lecture Notes in Computer Science*, pages 306–318. Springer-Verlag, New York, 1998.

[7] R. Chandramouli, M. Kharrazi, and N.D. Memon. Image steganography and steganalysis: Concepts and practice. In T. Kalker, Y.M. Ro, and I. Cox, editors, *Digital Watermarking, 2nd International Workshop, IWDW 2003, Seoul, Korea, October 20–22, 2003*, volume 2939 of *Lecture Notes in Computer Science*, pages 35–49. Springer-Verlag, New York, 2004.

[8] J. Eggers, R. Bäuml, and B. Girod. A communications approach to steganography. In E.J. Delp and P.W. Wong, editors, *Proceedings SPIE, Electronic Imaging, Security, and Watermarking of Multimedia Contents IV, San Jose, CA, January 21–24, 2002*, volume 4675, pages 26–37.

[9] H. Farid and S. Lyu. Steganalysis using higher-order image statistics. *IEEE Transactions on Information Forensics and Security*, 1(1):111–119, 2006.

[10] H. Farid and L. Siwei. Detecting hidden messages using higher-order statistics and support vector machines. In F.A.P. Petitcolas, editor, *Information Hiding, 5th International Workshop*, volume 2578 of *Lecture Notes in Computer Science*, pages 340–354, 2002.

[11] J. Fridrich. Minimizing the embedding impact in steganography. In J. Dittmann and J. Fridrich, editors, *Proceedings ACM Multimedia and Security Workshop, Geneva, Switzerland, September 26–27, 2006*, pages 2–10. ACM Press, New York.

[12] J. Fridrich. Feature-based steganalysis for JPEG images and its implications for future design of steganographic schemes. In J. Fridrich, editor, *Information Hiding, 6th International Workshop*, volume 3200 of *Lecture Notes in Computer Science*, pages 67–81. Springer-Verlag, New York, 2005.

[13] J. Fridrich, P. Lisoněk, and D. Soukal. On steganographic embedding efficiency. In N. Johnson and J. Camenisch, editors, *Information Hiding, 8th International Workshop*, volume 4437 of *Lecture Notes in Computer Science*. Springer-Verlag, New York, 2006.

[14] J. Fridrich and T. Filler. Practical methods for minimizing embedding impact in steganography. In E.J. Delp and P.W. Wong, editors, *Proceedings SPIE Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents IX, San Jose, CA*, volume 6505, pages 02–03, January 29–February 1 2007.

[15] J. Fridrich, M. Goljan, and D. Soukal. Perturbed quantization steganography using wet paper codes. In J. Dittmann and J. Fridrich, editors, *Proceedings ACM Multimedia and Security Workshop, Magdeburg, Germany, September 20–21, 2004*, pages 4–15. ACM Press, New York.

[16] J. Fridrich, M. Goljan, and D. Soukal. Perturbed quantization steganography. *ACM Multimedia and Security Journal*, 11(2):98–107, 2005.

[17] J. Fridrich, M. Goljan, and D. Soukal. Wet paper codes with improved embedding efficiency. *IEEE Transactions on Information Security and Forensics*, 1(1):102–110, 2006.

[18] G. Gul, A.E. Dirik, and I. Avcibas. Steganalytic features for JPEG compression-based perturbed quantization. *IEEE Signal Processing Letters*, 14(3):205–208, March 2000.

[19] S. Hetzl and P. Mutzel. A graph–theoretic approach to steganography. In J. Dittmann et al., editor, *Communications and Multimedia Security. 9th IFIP TC-6 TC-11 International Conference, CMS 2005*, volume 3677 of *Lecture Notes in Computer Science*, pages 119–128, Salzburg, Austria, September 19–21 2005.

[20] A. Ker. A general framework for structural analysis of LSB replacement. In M. Barni, J. Herrera, S. Katzenbeisser, and F. Pérez-González, editors, *Proceedings, Information Hiding, 7th International Workshop*, volume 3727 of *Lecture Notes in Computer Science*, pages 296–311. Springer-Verlag, Berlin, 2005.

[21] Y. Kim, Z. Duric, and D. Richards. Modified matrix encoding technique for minimal distortion steganography. In N. Johnson and J. Camenisch, editors, *Information Hiding, 8th International Workshop*, volume 4437 of *Lecture Notes in Computer Science*. Springer-Verlag, New York, 2006.

[22] K. Lee and A. Westfeld. Generalized category attack—improving histogram-based attack on JPEG LSB embedding. In *Information Hiding, 9th International Workshop, Saint Malo, France, June 11–13, 2007*, Lecture Notes in Computer Science. Springer-Verlag, 2007.

[23] K. Lee, A. Westfeld, and S. Lee. Category attack for LSB embedding of JPEG images. In *Proceedings of the International Workshop on Digital Watermarking, IWDW 2006, Jeju Island, Korea, November 8–10, 2006*, volume 4283 of *Lecture Notes in Computer Science*, pages 35–48. Springer-Verlag, 2006.

[24] Y. Lin, Y. Lee, and G. Wahba. Support vector machines for classification in nonstandard situations. *Machine Learning*, 46:191–202, 2000.

[25] S. Lyu and H. Farid. Steganalysis using color wavelet statistics and one-class support vector machines. In E.J. Delp and P.W. Wong, editors, *Proceedings of SPIE Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VI*, volume 5306, pages 35–45, 2004.

[26] P. Moulin and Y. Wang. Statistical modeling and steganalysis of DFT-based image steganography. In E.J. Delp and P.W. Wong, editors, *Proceedings of SPIE Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VIII*, volume 6072, pages 607202–1–607202–11, 2006.

[27] R.E. Newman, I.S. Moskowitz, LiWu Chang, and M.M. Brahmadesam. A steganographic embedding undetectable by JPEG-compatibility steganalysis. In F.A.P. Petitcolas, editor, *Information Hiding, 5th International Workshop*, volume 2578 of *Lecture Notes in Computer Science*, pages 258–277. Springer, New York.

[28] H. Noda, M. Niimi, and E. Kawaguchi. Application of QIM with dead zone for histogram preserving JPEG steganography. In *Proceedings ICIP, Genova, Italy, September 2005*.

[29] T. Pevný and J. Fridrich. Merging Markov and DCT features for multi-class JPEG steganalysis. In E.J. Delp and P.W. Wong, editors, *Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents IX, San Jose, CA, January 29–February 1*, volume 6505, pages 03–04, January 2007.

[30] N. Provos. Defending against statistical steganalysis. In *10th USENIX Security Symposium, Washington, DC*, 2001.

[31] P. Sallee. Model-based steganography. In T. Kalker, I.J. Cox, and Y.M. Ro, editors, *Digital Watermarking, 2nd International Workshop, IWDW 2003, Seoul, Korea, October 20–20, 2003*, volume 2939 of *Lecture Notes in Computer Science*, pages 154–167. Springer-Verlag, New York, 2004.

[32] P. Sallee. Model-based methods for steganography and steganalysis. *International Journal of Image Graphics*, 5(1):167–190, 2005.

[33] Y.Q. Shi, C. Chen, and W. Chen. A Markov process based approach to effective attacking JPEG steganography. In N. Johnson and J. Camenisch, editors, *Information Hiding, 8th International Workshop*, volume 4437 of *LNCS*. Springer-Verlag, New York, 2006.

[34] K. Solanki, A. Sarkar, and B.S. Manjunath. YASS: Yet another steganographic scheme that resists blind steganalysis. In T. Furon et al., editor, *Information Hiding, 9th International Workshop, Saint Malo, France, June 11–13, 2007*, to appear in Lecture Notes in Computer Science. Springer-Verlag, New York.

[35] K. Solanki, K. Sullivan, U. Madhow, B.S. Manjunath, and S. Chandrasekaran. Provably secure steganography: Achieving zero K-L divergence using statistical restoration. In *Proceedings ICIP, Atlanta, GA*, October 2006.

[36] K. Sullivan, U. Madhow, B.S. Manjunath, and S. Chandrasekaran. Steganalysis for Markov cover data with applications to images. *IEEE Transactions on Information Security and Forensics*, 1(2):275–287, June 2006.

[37] A. Westfeld. High capacity despite better steganalysis (F5—a steganographic algorithm). In I.S. Moskowitz, editor, *Information Hiding, 4th International Workshop*, volume 2137 of *Lecture Notes in Computer Science*, pages 289–302. Springer-Verlag, New York, 2001.

[38] A. Westfeld and A. Pfitzmann. Attacks on steganographic systems. In A. Pfitzmann, editor, *Information Hiding, 3rd International Workshop*, volume 1768 of *Lecture Notes in Computer Science*, pages 61–75. Springer-Verlag, New York, 2000.

[39] G. Xuan, Y.Q. Shi, J. Gao, D. Zou, C. Yang, Z. Zhang, P. Chai, C. Chen, and W. Chen. Steganalysis based on multiple features formed by statistical moments of wavelet characteristic function. In M. Barni, J. Herrera, S. Katzenbeisser, and F. Pérez-González, editors, *Information Hiding. 7th International Workshop*, volume 3727 of *Lecture Notes in Computer Science*, pages 262–277, 2005.

[40] X. Yu, Y. Wang, and T. Tan. On estimation of secret message length in JSteg-like steganography. In *Proceedings, International Conference on Pattern Recognition, Cambridge, UK, August 23–26, 2004*, volume 4, pages 673–676.

[41] T. Zhang and X. Ping. A fast and effective steganalytic technique against JSteg-like algorithms. In *Proceedings, Symposium on Applied Computing, Melbourne, FL*, pages 307–311, 2003.