Multi-Class Detector of Current Steganographic Methods for JPEG Format

Tomáš Pevný and Jessica Fridrich, IEEE member

***Abstract*—The aim of this paper is to construct a practical forensic steganalysis tool for JPEG images that can properly analyze both single- and double-compressed stego images and classify them to selected current steganographic methods. Although some of the individual modules of the steganalyzer were previously published by the authors, they were never tested as a complete system. The fusion of the modules brings its own challenges and problems whose analysis and solution is one of the goals of this paper.**

By determining the stego algorithm, this tool provides the first step needed for extracting the secret message. Given a JPEG image, the detector assigns it to 6 popular steganographic algorithms. The detection is based on feature extraction and supervised training of two banks of multi-classifiers realized using support vector machines. For accurate classification of single-compressed images, a separate multi-classifier is trained for each JPEG quality factor from a certain range. Another bank of multi-classifiers is trained for double-compressed images for the same range of primary quality factors. The image under investigation is first analyzed using a pre-classifier that detects selected cases of double-compression and estimates the primary quantization table. It then sends the image to the appropriate single- or double-compression multi-classifier. The error is estimated from more than 2.6 million images. The steganalyzer is also tested on two previously unseen methods to examine its ability to generalize.

I. Introduction

Steganography is the act of covert communications, meaning that only the communicating parties are *aware* of the secret communication. To accomplish this, the message is typically hidden within innocent-looking cover objects. The main objective is to create an algorithm for embedding secret messages so that their very presence in the stego objects cannot be proved. Thus, the most important attribute of steganography is *undetectability*, which, stated informally, means that no algorithm exists that can determine whether an object contains a hidden message. While there were several attempts to formalize the concept of undetectability in steganography [17], [36], [12], the most frequently used definition was given by Cachin [5] using an information-theoretic framework.

The counterpart of steganography is steganalysis whose goal is discovering the presence of hidden messages. Even though steganography is considered broken when the mere presence of the secret message is discovered, we often desire to recover some attributes of the message, such as its length, or the message content itself. The first necessary step towards this goal is to determine the data hiding algorithm. In some special situations, this information may be readily available, such as when a suspect's computer is seized and stego software is installed on the hard disk. Perhaps a more typical situation occurs when steganographic content is detected while monitoring network traffic. In this case, the task of recovering the message is much more challenging and must start with determining the stego algorithm used to embed the message, which is the focus of this paper.

We constrain ourselves to the JPEG format because it is the most ubiquitous image format in use today. There are several challenges that need to be overcome to obtain accurate multi-classification of JPEG images. First, the JPEG format accepts as a parameter the quantization table(s). The tables drive the quantization of DCT coefficients and thus change their statistical properties. This effectively enlarges the space of covers and further complicates steganalysis because a classifier trained on one quality factor may give less accurate results on images with a different quality factor (see, e.g., Table 3 and 4 in [25]). Second, multiple JPEG compression may dramatically change the statistics of DCT coefficients and thus cause some steganalysis methods to fail [7]. Ste-

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.

Tomáš Pevný is a PhD. student at the Department of Computer Science, Binghamton University, NY 13902 USA (e-mail: pevnak@gmail.com)

Jessica Fridrich is a Professor at the Department of Electrical and Computer Engineering, Binghamton University, NY 13902 USA (607-777-6177; fax: 607-777-4464; e-mail: fridrich@binghamton.edu)

ganalyzers that extract features in the spatial domain [2], [3], [6], [20], [21], [35], [1] are somewhat less sensitive to the above two effects than steganalyzers with features calculated from quantized DCT coefficients [27], [7], [32], [24], [26], especially those that use calibration for estimating the cover image [27], [7]. On the other hand, recent comparisons provided in [27], [32], [35], [24] indicate that features computed directly from DCT coefficients provide more accurate detection results.

The classifier proposed in this paper uses such features obtained by merging the extended DCT feature set with a reduced Markov feature set. This set was recently shown [27] to achieve a significantly improved performance over prior art. We build upon our previously published partial results [27], [24], [26] to obtain a practical solution to the problem of multi-classification of a larger set of steganographic algorithms that produce both single- and double-compressed images. Even though parts of the individual modules of this steganalyzer were previously analyzed by the same authors [27], [24], [26], the fusion of the modules into one general steganalysis system brings its own challenges. For example, it is not clear how the errors of each individual module accumulate when the modules interact with each other.

This paper is organized as follows. To reduce the complexity and storage needed to construct the steganalyzer, in the next section we make some simplifying assumptions about the cover images used by the steganographer. Section III describes the structure of the proposed multi-classifier and explains some of our design choices. Subsequent sections describe the individual modules from which the multiclassifier is built, such as the *double-compression detector* and the *primary quality factor estimator* in Section IV, the *multi-classifier for single-compressed images* in Section V, and the *multi-classifier for double-compressed images* in Section VI. Experimental results appear in Section VII. Finally, Section VIII concludes the paper.

II. BASIC ASSUMPTIONS

In this section, we explain the assumptions based on which we devise our steganalyzer. These assumptions define its scope and guide our design choices. First, we introduce notation used throughout the paper.

A. Notation

JPEG compression [22] starts by dividing the image into disjoint 8×8 pixel blocks B and transforming each block using the Discrete Cosine Transformation (DCT), obtaining thus an 8×8 block of real numbers d_{ij}, $i, j = 0, \ldots, 7$. Next, the DCT coefficients, d_{ij}, are divided by quantization steps from the quantization matrix[1] Q_{ij} and rounded to integers

$$D_{ij} = \text{round}\left(\frac{d_{ij}}{Q_{ij}}\right),\ i, j \in \{0, \ldots, 7\}.$$

Let n_B be the total number of all 8×8 blocks in the image. The i, j-th quantized DCT coefficient in the k-th block is denoted as $D_{ij}^k,\ k \in \{1, \ldots, n_B\}$. The pair $(i, j) \in \{0, \ldots, 7\} \times \{0, \ldots, 7\}$ indexes horizontal and vertical *spatial frequency* (or *mode*) of the DCT coefficient.

During decompression, each block of quantized DCT coefficients is multiplied by the quantization matrix Q, $\hat{d}_{ij} = Q_{ij} \cdot D_{ij}$, and the Inverse Discrete Cosine Transformation (IDCT) is applied to $\hat{d}_{ij}$. The resulting values are rounded to integers and truncated to a finite dynamic range. The block of decompressed pixel values $\hat{B}$ is thus

$$\hat{B} = \text{trunc}(\text{round}(\text{IDCT}(Q_{ij} \cdot D_{ij}))),\ i, j \in \{0, \ldots, 7\}.$$

B. Assumptions

The first assumption concerns the selection of stego algorithms for multi-classification. The classifier will be trained to recognize 6 current popular steganographic programs: F5 [34], Model Based Steganography without [30] (MBS1) and with [31] deblocking (MBS2), JP Hide&Seek (http://linux01.gwdg.de/%7Ealatham/stego.html), OutGuess [29] ver. 0.2 with histogram correction, and Steghide [14]. We intentionally left out the very first publicly available steganographic program Jsteg (http://zooid.org/~paul/crypto/jsteg/) and the state-of-the-art MMx algorithm [19] so that we can test how well the multi-classifier handles stego images produced by previously unseen stego methods. The 6 stego programs were carefully selected to cover essentially all types of stego algorithms available today. The F5 uses a different embedding operation than LSB flipping and incorporates matrix embedding to minimize the number of embedding changes. MBS1 and MBS2 use LSB flipping and are designed to preserve a model of DCT coefficients. OutGuess and Steghide preserve the first order statistics. OutGuess does so by making additional changes (statistical restoration) while Steghide exchanges pairs of

[1]The widely spread IJPEG library (publicly available ftp://ftp.uu.net/graphics/jpeg/jpegsrc.v6b.tar.gz) uses a set of "standard" quantization matrices $\mathcal{T}$ indexed by a *quality factor* from the set $\{1, 2, \ldots, 100\}$. The algorithm for calculating the quantization matrices is implemented in the function "jpeg_quality_scaling" in the file "jcparam.c".

coefficients. JP Hide&Seek, due to Allan Latham is a heuristically improved modification of Jsteg.

All algorithms above, with the exception of OutGuess and F5, accept as input JPEG images and embed by directly manipulating their DCT coefficients. Thus, when a single-compressed image is used as cover, the stego image is never recompressed (double-compressed). F5 and OutGuess, when presented with a JPEG cover image, first decompress it to the spatial domain and then recompress prior to embedding a message (the default quality factor is 75 for OutGuess and 80 for F5) . Thus, the resulting stego image is double-compressed.

The second assumption concerns the source of cover images. We assume that the steganographer uses as covers either single compressed JPEG images or images that were never JPEG compressed (images in some raw format). We also assume that F5 and OutGuess are only run with two quality factors – 75 and 80. Under these assumptions, the stego image is either single-compressed or double-compressed with secondary quality factor 75 or 80. This assumption was accepted to reduce the storage and computational complexity required to construct the steganalyzers (see more detailed discussion in Section VI-B).

Based on the assumptions above, we pronounce a JPEG image *double-compressed* if the JPEG compression was applied twice, each time with a different quantization matrix but with the same alignment of the 8×8 grid. The first matrix $Q^{(1)}$ is called the *primary quantization matrix* and the second matrix $Q^{(2)}$ the *secondary quantization matrix*. We say that a specific DCT *coefficient* D_{ij} was double-quantized if and only if $Q_{ij}^{(1)} \neq Q_{ij}^{(2)}$.

Note that the assumptions exclude the case when the cover image is spatially shifted (cropped) after decompression but prior to the second compression. We do not consider such images double-compressed even though the previous compression will undoubtedly have some effect on steganalysis.

III. The multi-classifier

Blind steganalyzers based on feature extraction and machine learning are well suited for multi-classification because images embedded using different steganographic algorithms form distinct clusters in the feature space [24]. This is because different steganographic algorithms introduce different artifacts into images and thus shift the feature vector to a specific portion of the feature space. For instance, the F5 algorithm [34] increases the number of zeros and slightly decreases the number of all non-zero DCT coefficients. By using a large number of features, we increase the chances that two different embedding programs will produce feature vectors located in different parts of the feature space. The differences between distributions of feature vectors for cover and stego images can be captured using machine-learning methods.

The features are usually constructed using heuristic principles that in one way or another aim to capture small modifications due to steganographic embedding. The first known application of blind classifiers to steganalysis used features computed from quality metrics [2]. The same work also presented a classifier of selected watermarking methods based on linear regression. The same authors later proposed a different set of features based on binary similarity measures [3], [1]. Farid [6], [20] constructed the features from higher-order moments of distribution of wavelet coefficients from several high-frequency sub-bands and their local linear prediction errors. Other choices include the center of gravity of the histogram characteristic function [13], absolute moments of the histogram characteristic function constructed in the wavelet domain [35], higher-order absolute moments of the image noise residual in the wavelet domain [11], and the statistics of full-frame DFT coefficients [33].

In this paper, we use calibrated features computed directly from the quantized DCT coefficients as described in detail in our previous work [27]. Because the embedding changes are "lumped" in the DCT domain, some features will be very sensitive to embedding changes while other features may not change at all. This is a desirable property for multi-class steganalysis. The purpose of calibration is to make the features sensitive to embedding changes and relatively insensitive to image content by taking differences between quantities calculated from the stego image and an estimate of the cover image[2]. The main reason for choosing these features is their excellent performance documented by comparisons to prior art [32], [27], [24]. The disadvantage is that they are more sensitive to the quality factor of the stego image and the combinations of quality factors in double-compressed images [8]. In this paper, we resolve this issue by first detecting selected cases of double-compressed images and then sending the stego image to an appropriate multi-classifier, depending on the stego image quality factor and the fact whether or not it was double-compressed.

Figure 1 shows a high-level description of the proposed steganalyzer. Its structure logically follows from the assumptions made in Section II-B. The steganalyzer

[2]More detailed description of the features appears in Section V and in the original publications [7], [32], [27].

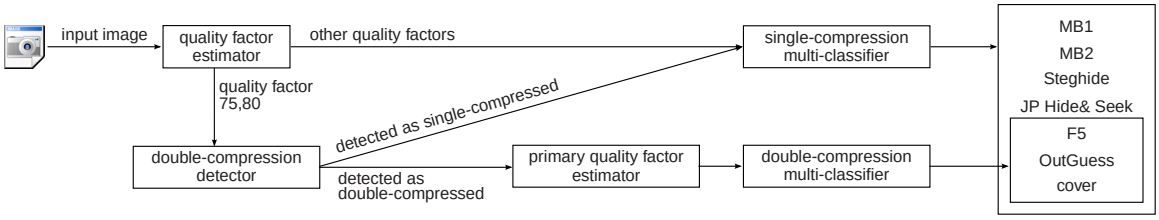


Fig. 1. Structure of the multi-class steganalyzer.

consists of two multi-classifiers preceded by a detector of double-compression that preclassifies the image under inspection and then sends it to the appropriate multi-classifier. The single-compression multi-classifier is in fact a collection of 34 multi-classifiers $\mathcal{S}_i$, indexed by the quality factor $i \in \mathcal{Q}_{34}$,

$$\mathcal{Q}_{34} = \{63, 64, \ldots, 93, 94, 96, 98\}.$$

This set of quality factors was a compromise between our desire to create as general classifier as possible and our limited computational and storage resources. Each multi-classifier $\mathcal{S}_i$ is trained to assign JPEG images with quality factor i to 7 categories—covers and 6 stego algorithms described in Section II-B. The second multi-classifier consists of two multi-classifiers $\mathcal{D}_{75}$ and $\mathcal{D}_{80}$ that assign double-compressed JPEG images with secondary quality factors 75 and 80 to cover, F5, or OutGuess, because only these two algorithms can create double-compressed images during embedding under the assumptions of Section II-B.

The whole process of analyzing an image starts by inspecting its quality factor[3] i. If $i \notin \{75, 80\}$, the image is sent directly to the *multi-classifier for single-compressed images* $\mathcal{S}_i$. If $i \in \{75, 80\}$, the image is forwarded to the *double-compression detector*. If the double-compression detector detects the image as single-compressed, the image is sent to $\mathcal{S}_i$. If the image is detected as double-compressed, the primary quality factor of the image is estimated, added as an additional 275-th feature, and sent with the image to the *multi-classifier for double-compressed images* $\mathcal{D}_i$, which classifies it as either cover, F5, or OutGuess.

The accuracy of the double-compression detector has a major impact on the accuracy of classification of images with quality factors 75 and 80. If the double-compression detector deems a single-compressed image as double-compressed, it can be classified only as cover or embedded by F5/OutGuess, even though the image was embedded by a different algorithm. One of the goals of this paper is to evaluate the performance of the complete system when all its modules are cascaded as explained above. We estimate the accuracy of the blind steganalyzer from more than 2.6 million images under conditions similar to real use (no side knowledge about compression history of inspected image is provided to the classifier). Results shows that the error of the blind steganalyzer is acceptable.

[3]If the image has a non-standard quantization table, the closest standard quantization table (and thus a quality factor) is found by matching low-frequency quantization steps (see Section IV-A).

IV. Detection of Double-Compression

In this section, we describe the preclassifier whose function is to detect selected cases of double-compression and estimate the primary quality factor for images detected as double-compressed. Due to the impact of the double-compression detector on the accuracy of the overall solution (explained in the previous section), the double-compression detector has to be tuned to a low false positive rate (incorrectly detecting a single-compressed image as double-compressed). Since the construction of the double-compression detector and the primary quality factor estimator was described in detail in a separate paper [23], we only provide a brief description of both.

A. Quality factor estimator

The JPEG standard allows using arbitrary quantization matrices that do not necessarily have to correspond to any standard quantization matrix for any quality factor. Because the multi-classifiers from which our steganalyzer is constructed are indexed by the quality factor, if the secondary JPEG quantization matrix $Q^{(2)}$ is non-standard, we find the closest standard quantization matrix $Q \in \mathcal{T}$ so that we can send the image to the proper multi-class steganalyzer. This is achieved using the following formula

$$Q = \arg\min_{Q \in \mathcal{T}} \sum_{(i,j) \in \mathcal{L}} |Q_{ij} - Q^{(2)}_{ij}|,$$

where the sum is taken over a selected band of spatial frequencies $\mathcal{L}$. Since most non-zero DCT coefficients in natural images are in the low-frequency band, it is important to match these modes rather than modes corresponding to high spatial frequencies. For the range of quality factors from $\mathcal{Q}_{34}$, we used the band $\mathcal{L}_3 = \{(i,j)|i+j \leq 3\}$.

B. Double-Compression detector

The detector of double-compression returns a binary output—whether or not a JPEG image was double-compressed. Since double-compressed images may also be modified by steganographic embedding, the detector must not be confused by the embedding changes. In other words, the double-compression detection and primary quantization table estimation must be *robust* to steganographic modifications. Additionally, for the reasons explained at the end of Section III, the detector needs to be tuned to a *low false positive rate* (incorrect detection of a single-compressed image as double-compressed).

One possible approach to detection of double-compression would be to accept a parametric model for the distribution of DCT coefficients for each DCT mode (e.g., a generalized Gaussian distribution), further accept a model of the noise introduced by stego embedding, and then formulate the problem of estimation of primary quantization steps using Maximum Likelihood principle with the parameters of the cover DCT coefficients as nuisance parameters. The problem with this approach is that the character of the embedding changes varies across steganographic methods and is not possible to model the stego channel using a simple stochastic process. Optimal detectors derived under the wrong model may thus provide highly unstable results.

Prior art on double-compression detection and estimation includes [10], [9], [28], [8]. In this paper, we selected the detector designed for use in steganalysis [23] because it is robust and exhibits the required low false positive rate. We briefly describe the general ideas of its construction here, while referring to the original publication for full details.

The detector is a binary classifier that decides whether or not a given image is double-compressed. The features for classification are histograms of selected low-frequency DCT modes. We denote by h_{ij} the histogram of absolute values of DCT coefficients for spatial frequency (i,j)

$$h_{ij}(m) = \sum_{k=1}^{n_B} \delta\left(\left|D_{ij}^k\right| - m \cdot Q_{ij}^{(2)}\right), \qquad (1)$$

where $m \geq 0$ is an integer and δ is the indicator function, $\delta(x) = 1$ if $x = 0$ and $\delta(x) = 0$ when $x \neq 0$. In a single-compressed image, the histograms h_{ij} follow the quantized Laplacian (or generalized Gaussian) distribution [16]. After the image is decompressed to the spatial domain and compressed again with a different (secondary) quantization matrix $Q^{(2)}$, the histograms may no longer follow the same distribution. Depending on the combination of the primary and secondary quantization steps, the histograms may exhibit different artifacts of double-compression. We refer to [9] for an in-depth description of different types of double-compression artifacts. The same work also discusses why tools of pattern recognition are well suited to detect these artifacts.

Our implementation of the double-compression detector uses soft-margin support vector machines ($C-$SVM) [4] with a Gaussian kernel. The feature vector $\mathbf{x}$ of the $C-$SVM consists of normalized histograms of absolute values of DCT coefficients for 9 lowest spatial AC frequencies

$$\mathcal{L}_3 = \{(i,j)|i+j \leq 3\}, \qquad (2)$$

$$\mathbf{x} = \left\{ \frac{1}{C_{ij}} \left(h_{ij}(0), h_{ij}(1), \ldots, h_{ij}(15)\right) \middle| \ (i,j) \in \mathcal{L}_3 \right\},$$

where C_{ij} are normalization constants $\left(C_{ij} = \sum_{m=0}^{15} h_{ij}(m)\right)$. The dimension of the feature vector is $16 \times |\mathcal{L}_3| = 144$. To speed-up the training of the double-compression detector, a separate detector is trained for each value of the secondary quality factor.

C. Primary quality factor estimator

Images detected as double-compressed are further sent to an estimator of the primary quality factor. This estimator works in two steps. First, it estimates the primary quantization steps of DCT modes from $\mathcal{L}_3$ and then from these estimated quantization steps it finds the primary quality factor using the maximum likelihood principle.

The primary quantization steps for modes from $\mathcal{L}_3$ 2 are estimated by a collection of SVM-based multi-classifiers $\mathcal{F}_{Q_{ij}^{(2)}}$ indexed by the secondary quantization steps $Q_{ij}^{(2)}$. The number of classes for each multi-classifier is determined by the range of quantization steps Q_{ij} for quality factors from $\mathcal{Q}_{34}$. The feature vector $\mathbf{x}$ for the multi-classifier $\mathcal{F}_{Q_{ij}^{(2)}}$ is formed by the normalized histogram of absolute values of the first 16 multiples of

$Q_{ij}^{(2)}$ of all DCT coefficients $|D_{ij}^k|$ for all $k = 1, \ldots, n_B$

$$\mathbf{x} = \frac{1}{C}(h_{ij}(0), h_{ij}(1), \ldots, h_{ij}(15)), \tag{3}$$

where C is a normalization constant chosen so that $\sum_{m=0}^{15} x(m) = 1$.

Note that the feature vector $\mathbf{x}$ cannot distinguish between the following three cases: $Q_{ij}^{(1)}$ is a divisor of $Q_{ij}^{(2)}$, $Q_{ij}^{(1)} = 1$, and $Q_{ij}^{(1)} = Q_{ij}^{(2)}$. Thus, all these cases are classified into one common class where $Q_{ij}^{(1)} = Q_{ij}^{(2)}$. This phenomenon imposes a fundamental limitation on the performance of the detector. Fortunately, the double-compressed image in these three cases does not exhibit any discernible traces of double-compression, hence influences steganalysis in a negligible manner. In other words, the failure to distinguish between these cases is not essential for steganalysis.

If we denote the estimated and the true primary quantization steps as $\hat{Q}_{ij}^{(1)}$ and $Q_{ij}^{(1)}$, respectively, the closest standard quantization matrix is calculated using the maximum likelihood principle as

$$\hat{Q}^{(1)} \quad = \quad \arg\min_{Q \in \mathcal{T}} \prod_{(i,j) \in \mathcal{L}_9} P(\hat{Q}_{ij}^{(1)} | Q_{ij}^{(1)}, Q_{ij}^{(2)}).$$

The value $P(\hat{Q}_{ij}^{(1)} | Q_{ij}^{(1)}, Q_{ij}^{(2)})$ is the probability that the multi-classifier $\mathcal{F}_{Q_{ij}^{(2)}}$ detects the primary quantization step $\hat{Q}_{ij}^{(1)}$ when the correct primary quantization step is $Q_{ij}^{(1)}$ and the secondary quantization step is $Q_{ij}^{(2)}$. These probabilities are obtained from the training set used for training the classifiers $\mathcal{F}$.

V. Single-Compression Multi-Classifier

In this section, we provide the details of the multi-classifier for single-compressed JPEG images, including the description of the feature set and the training and testing methodology.

A. Features

The feature set is comprised of 274 features—193 extended calibrated DCT features (the original 23 DCT features were introduced in [7]) and a reduced set of 81 Markov features (the original 324 Markov features were proposed by Shi et al. [32]). We merged both feature sets because each feature set captures a different type of dependencies between DCT coefficients. While the extended DCT features model *inter-block* dependencies between DCT coefficients, the Markov features capture *intra-block* dependency among DCT coefficients of similar spatial frequencies within the same 8×8 block. Another reason for merging the sets is that the classifiers employing each feature set individually have complementary performance [27].

All features in the merged feature set are calibrated (see Figure 2). Calibration is a process through which one can estimate macroscopic properties of the cover image from the stego image. We briefly review it here, because it is an essential part of the feature calculation and because it has to be done properly for double-compressed images. More detailed description of calibration can be found in [7], [24].

Calibration starts by decompressing the stego JPEG image J_1 to the spatial domain, desynchronizing the spatial 8×8 grid (e.g., by cropping a few pixels in both directions), and compressing again with the same quantization matrix as the stego image J_1. The newly obtained JPEG image J_2 has most macroscopic features similar to the original cover image because the cropped image is visually similar to the original image. The cropping brings the 8×8 DCT grid "out of sync" with the previous compression, which effectively suppresses the influence of the previous JPEG compression *and* the embedding changes. The calibrated feature is obtained as the difference between a functional calculated for J_1 and J_2. The net effect of calibration is that the calibrated feature will be less sensitive to image content and more sensitive to embedding changes.

1) Extended DCT feature set: The *original* DCT features [7] were constructed from 23 functionals $\mathbf{F}$, which is a mapping assigning a scalar, vector, or a matrix to a JPEG image. The calibrated features are obtained as the difference $\mathbf{F}(J_1) - \mathbf{F}(J_2)$.

The first functional is the normalized histogram $\mathbf{H}$ of all $64 \times n_B$ luminance DCT coefficients

$$\mathbf{H} = (H(L), \ldots, H(R)), \tag{4}$$

where $L = \min_{i,j,k} D_{ij}^k$, $R = \max_{i,j,k} D_{ij}^k$. Here, and in the rest of this section, by normalized we understand dividing the absolute counts in the histogram by their sum so that $\sum_{m=L}^{R} H(m) = 1$.

The next 5 functionals are the normalized histograms

$$\mathbf{h}_{ij} = (h_{ij}(L), \ldots, h_{ij}(R)), \tag{5}$$

of DCT coefficients for the following 5 AC DCT modes $(i,j) \in \mathcal{L}_2 = \{(i,j) | i + j \leq 2\}$.

The next 11 functionals are dual normalized histograms represented with 8×8 matrices $\mathbf{g}_d(i,j)$, $i,j = 0, \ldots, 7$, for each $d \in \{-5, \ldots, 5\}$

$$\mathbf{g}_d(i,j) = \sum_{k=1}^{n_B} \delta(d, D_{ij}^k). \tag{6}$$

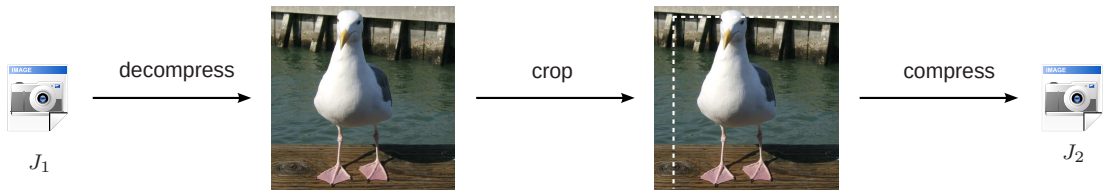


Fig. 2. Calibration for single-compressed images. The calibrated feature is obtained as the difference $\mathbf{F}(J_1) - \mathbf{F}(J_2)$ constrained to a limited range.

where δ is again the indicator function. The normalization here means that $\sum_{i,j=0}^{7} \mathbf{g}_d(i,j) = 1$.

The next 6 functionals capture inter-block dependency among DCT coefficients. The first functional is the variation V

$$V = \frac{1}{|\mathbf{I}_r| + |\mathbf{I}_c|} \left[\sum_{i,j=0}^{7} \sum_{k=1}^{|\mathbf{I}_r|-1} \left| D_{ij}^{\mathbf{I}_r(k)} - D_{ij}^{\mathbf{I}_r(k+1)} \right| + \sum_{i,j=0}^{7} \sum_{k=1}^{|\mathbf{I}_c|-1} \left| D_{ij}^{\mathbf{I}_c(k)} - D_{ij}^{\mathbf{I}_c(k+1)} \right| \right], \quad (7)$$

where $\mathbf{I}_r$ and $\mathbf{I}_c$ denote the vectors of block indices $1, \ldots, n_B$ while scanning the image by rows and by columns, respectively.

Two blockiness functionals B_α (8) are scalars calculated from the decompressed JPEG image representing an integral measure of inter-block dependency over all DCT modes over the whole image. In (8), M and N are image height and width in pixels, and $\mathbf{c}_{r,s}$ are greyscale values of the decompressed JPEG image, $\alpha = 1, 2$.

The remaining functional (9) is the co-occurrence matrix $\mathbf{C}(u,v)$.

The original DCT features [7] were obtained as the L_1 norm of the difference between the functionals $\|\mathbf{F}(J_1) - \mathbf{F}(J_2)\|_{L_1}$. This, however, removed too much information potentially useful for steganalysis. Thus, in the extended DCT feature set, we replaced the L_1 norm by the difference constrained to a small finite range. For the global histogram functional $\mathbf{H}$ and the 5 normalized histograms of individual DCT modes $\mathbf{h}_{ij}$, $(i,j) \in \mathcal{L}_2$, the features are formed by the differences of the normalized histogram vectors in the range $m = -5, \ldots, 5$. The normalized histogram features are thus

$$\mathbf{H}^{(J_1)}(m) - \mathbf{H}^{(J_2)}(m), \ m \in \{-5, \ldots, 5\},$$

$$\mathbf{h}_{ij}^{(J_1)}(m) - \mathbf{h}_{ij}^{(J_2)}(m), \ m \in \{-5, \ldots, 5\}, \ (i,j) \in \mathcal{L}_2.$$

In the formulas above, and in the rest of this section, the superscript denotes the image to which the functional is applied. For the dual histogram functionals $\mathbf{g}_d$, $d \in \{-5, \ldots, +5\}$, the differences for the 9 lowest AC modes are used

$$\mathbf{g}_d^{(J_1)}(i,j) - \mathbf{g}_d^{(J_2)}(i,j), \ (i,j) \in \mathcal{L}_3.$$

We opted for features defined as the difference between the normalized histograms in a fixed range rather than using shape parameters, such as moments [6], [35], because the changes to individual DCT coefficients are characteristic for many steganographic schemes. For example, Jsteg avoids coefficients 0 and 1 and introduces characteristic step artifacts in the counts of the remaining values. F5 increases the number of zeros and decreases the counts of other coefficients, while OutGuess influences mostly the coefficient pair $-2, -1$. Thus, first order statistics of DCT coefficients will be quite sensitive and useful features for steganalysis.

Because the variation and two blockiness functionals are scalars, they are not extended. For the co-occurrence matrix, we take the differences of central elements in the range $(u,v) \in [-2,+2] \times [-2,+2]$, yielding 25 features

$$\mathbf{C}^{(J_1)}(u,v) - \mathbf{C}^{(J_2)}(u,v), \ (u,v) \in [-2,+2] \times [-2,+2].$$

The rationale behind restricting the range of the differences between functionals to a small interval around zero is that the DCT coefficients follow a generalized Gaussian distribution centered around zero. Thus, the central part of the functionals holds the most useful information for steganalysis.

The total number of extended DCT features is thus $1 \times 11 + 5 \times 11 + 9 \times 11 + 1 + 2 + 25 = 193$ (see Table I).

B. Original, calibrated, and reduced Markov features

The Markov feature set as proposed in [32] models the differences between absolute values of neighboring DCT

$$B_\alpha = \frac{\sum_{r=1}^{\lfloor (M-1)/8 \rfloor} \sum_{s=1}^{N} |\mathbf{c}_{8r,s} - \mathbf{c}_{8r+1,s}|^\alpha + \sum_{s=1}^{\lfloor (N-1)/8 \rfloor} \sum_{r=1}^{M} |\mathbf{c}_{r,8s} - \mathbf{c}_{r,8s+1}|^\alpha}{N \lfloor (M-1)/8 \rfloor + M \lfloor (N-1)/8 \rfloor} \tag{8}$$

$$\mathbf{C}(u,v) = \frac{\sum_{i,j=0}^{7} \sum_{k=1}^{|\mathbf{I}_r|-1} \delta\left(u, D_{ij}^{\mathbf{I}_r(k)}\right) \delta\left(v, D_{ij}^{\mathbf{I}_r(k+1)}\right) + \sum_{i,j=0}^{7} \sum_{k=1}^{|\mathbf{I}_c|-1} \delta\left(u, D_{ij}^{\mathbf{I}_c(k)}\right) \delta\left(v, D_{ij}^{\mathbf{I}_c(k+1)}\right)}{|\mathbf{I}_r| + |\mathbf{I}_c|}. \tag{9}$$

Functional	Dimensionality
Global histogram $\mathbf{H}$	11
5 AC histograms $\mathbf{h}$	5×11
11 Dual histograms $\mathbf{g}$	11×9
Variation V	1
2 Blockiness B	2
Co-occurrence matrix $\mathbf{C}$	25

TABLE I
EXTENDED DCT FEATURE SET WITH 193 FEATURES.

coefficients as a Markov process. The features are then obtained as the empirical transition probability matrices. Their calculation starts by forming the matrix $F(u,v)$ of absolute values of DCT coefficients in the image by simply arranging the DCT coefficients in the same way as pixels in the image by replacing each 8×8 block of pixels with the corresponding block of DCT coefficients. Next, four difference arrays are calculated along four directions: horizontal, vertical, diagonal, and minor diagonal (further denoted as $F_h(u,v)$, $F_v(u,v)$, $F_d(u,v)$, and $F_m(u,v)$, respectively)

$$\begin{aligned} F_h(u,v) &= F(u,v) - F(u+1,v), \\ F_v(u,v) &= F(u,v) - F(u,v+1), \\ F_d(u,v) &= F(u,v) - F(u+1,v+1), \\ F_m(u,v) &= F(u+1,v) - F(u,v+1). \end{aligned}$$

From these difference arrays, four empirical transition probability matrices $\mathbf{M}_h, \mathbf{M}_v, \mathbf{M}_d, \mathbf{M}_m$ are constructed as (10)

where S_u and S_v denote the dimensions of the matrix $F(u,v)$. If the matrices $\mathbf{M}_h, \mathbf{M}_v, \mathbf{M}_d, \mathbf{M}_m$ were taken directly as features, the dimensionality of the feature set would be too large, because the range of differences between absolute values of neighboring DCT coefficients could be quite large. To alleviate this problem, the authors of [32] proposed to use only the central $-4, \ldots, 4$ portion of the matrices with the caveat that the values in the difference arrays $F_h(u,v)$, $F_v(u,v)$, $F_d(u,v)$, and $F_m(u,v)$ larger than 4 were set to 4 and values smaller than -4 were set to -4 prior to calculating $\mathbf{M}_h, \mathbf{M}_v, \mathbf{M}_d, \mathbf{M}_m$. Thus, all four matrices have the same dimensions 9×9 and the number of features is $4 \times 81 = 324$.

As the calibration is known to improve features' sensitivity to embedding, we incorporated it into the calculation of the Markov features. The calibrated Markov features are formed by the differences $\mathbf{M}^{(c)} = \mathbf{M}^{(J_1)} - \mathbf{M}^{(J_2)}$. The dimensionality of the calibrated Markov features is the same as is the dimensionality of the original Markov features. In [27], an observation was made that the accuracy of the steganalyzer with the averaged calibrated Markov features $\overline{\mathbf{M}} = (\mathbf{M}_h^{(c)} + \mathbf{M}_v^{(c)} + \mathbf{M}_d^{(c)} + \mathbf{M}_m^{(c)})/4$ (dimension 81) is similar to the accuracy of the steganalyzer based on full calibrated Markov features (dimension 324) combined with extended DCT features. Thus, in this paper, we use the feature set obtained by merging 193 extended DCT features and 81 averaged Markov features, obtaining thus a 274-dimensional feature set.

C. Multi-Classifier

The multi-classifier for single-compressed JPEG images detects cover images and images embedded by 6 steganographic algorithms listed in Section II-B. It consists of a bank of SVM-based multi-classifiers $\mathcal{S}_i$ trained for each value of the quality factor $i \in \mathcal{Q}_{34}$. This modular approach has several advantages over the monolithic design with one big multi-classifier for all quality factors. First, the training of the modular multi-classifier is by the order of 34^2 faster. This is because the complexity of SVM training is approximately cubic in the number of training examples. Second, it is possible to extend the multi-classifier to other quality factors without having to change the classifiers that were already trained. Although it is possible to use a multi-classifier to classify images with quality factors different from the one the multi-classifier was trained for, the accuracy of classification decreases [25].

The multi-classifiers $\mathcal{S}_i$ are implemented as a set of $\binom{n_{cl}}{2}$ binary classifiers for every pair of classes, where

$$
\begin{aligned}
\mathbf{M}_h(i,j) &= \frac{\sum_{u=1}^{S_u-2}\sum_{v=1}^{S_v}\delta(F_h(u,v)=i,F_h(u+1,v)=j)}{\sum_{u=1}^{S_u-1}\sum_{v=1}^{S_v}\delta(F_h(u,v)=i)}\\
\mathbf{M}_v(i,j) &= \frac{\sum_{u=1}^{S_u}\sum_{v=1}^{S_v-2}\delta(F_v(u,v)=i,F_v(u,v+1)=j)}{\sum_{u=1}^{S_u}\sum_{v=1}^{S_v-1}\delta(F_v(u,v)=i)}\\
\mathbf{M}_d(i,j) &= \frac{\sum_{u=1}^{S_u-2}\sum_{v=1}^{S_v-2}\delta(F_d(u,v)=i,F_d(u+1,v+1)=j)}{\sum_{u=1}^{S_u-1}\sum_{v=1}^{S_v-1}\delta(F_d(u,v)=i)}\\
\mathbf{M}_m(i,j) &= \frac{\sum_{u=1}^{S_u-2}\sum_{v=1}^{S_v-2}\delta(F_m(u+1,v)=i,F_m(u,v+1)=j)}{\sum_{u=1}^{S_u-1}\sum_{v=1}^{S_v-1}\delta(F_m(u,v)=i)}
\end{aligned}
\tag{10}
$$

n_{cl} is the number of classes into which we wish to classify ($n_{cl}=7$ in our case). During classification, the feature vector is presented to all $\binom{n_{cl}}{2}$ binary classifiers and the histogram of their answers is created. The class corresponding to the maximum value of the histogram is selected as the final target class. If there are two or more classes with the same number of votes, one of the classes is randomly chosen. Since we classify into 7 classes, we need $\binom{7}{2}=21$ binary classifiers. This approach to multi-classification is known as the *max-wins* or *one to one* extension. According Hsu et al. [15], it is one of the most versatile frameworks for classification into more than two classes.

The binary classifiers used in each multi-classifier $\mathcal{S}_i$ were soft-margin SVMs with the Gaussian kernel $\exp(-\gamma\|x-y\|^2)$. Soft-margin SVMs can be trained on non-separable data by penalizing incorrectly classified images with the factor $C\cdot d$, where d is the L_1 distance from the separating hyperplane and C is a constant. The penalization parameter C can be different for the positive and the negative classes, which allows to lower the false positive rate at the expense of a higher false negative rate.

The hyper-parameters (C,γ) of the C-SVMs were determined as

$$\arg\min_{(C,\gamma)\in\mathcal{G}}\hat{E}(C,\gamma),$$

where $\hat{E}(C,\gamma)$ denotes the error estimated by $5-$fold cross-validation for hyper-parameters (C,γ), and $\mathcal{G}$ is a multiplicative grid

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

To overcome the problem that the set $\mathcal{G}$ is unbounded, we exploit the fact that for most practical problems, the error surface of SVMs estimated using cross-validation is convex. The grid-search for a particular SVM started by estimating the error on all points in a set common to all SVMs. After this initial search, we checked if the point with the least estimated 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.

If one prefers to use different costs for the penalization parameter C for the positive and negative classes, for example to lower the false positive rate, the grid-search has to be performed for the triplet $(C_{\text{pos}},C_{\text{neg}},\gamma)$, instead of the pair (C,γ). Because the search for a suitable triplet is very computationally expensive, we did not use this approach in our classifiers.

Prior to training, all elements of the feature vector were scaled to the interval $[-1,+1]$. The scaling coefficients were always derived from the training set. For the n_{cv}-fold cross-validation, the scaling coefficients were calculated from the remaining $n_{cv}-1$ subsets.

VI. Double-Compression Multi-Classifier

To motivate our effort in this section, we first explain why double-compression presents a problem for reliable steganalysis using features calculated in the DCT domain and why it is necessary to detect double-compression and modify the classifier. To this end, we created stego images using F5 and OutGuess from 2500 covers unseen by the classifier whose layout was outlined in the previous section. Details of the construction and the training procedure are in Section VII. Each raw image was first compressed with 34 quality factors from $\mathcal{Q}_{34}$ and subsequently embedded with 100%, 50%, and 25% of relative payload. The quality factor for F5 and OutGuess was set to 75. The stego images were thus double-compressed during embedding. Out of these $2500\times 34\times 3$ images, we randomly selected 10000

images and presented them to the multi-classifier $\mathcal{S}_{75}$. The resulting confusion matrix is shown in Table II and should be contrasted to Table III, which demonstrates classification accuracy that can be achieved using the same classifier on single-compressed images embedded using F5 and OutGuess. We clearly see that the classifier trained on single-compressed images does not handle double-compressed images well. The reason for the poor performance on double-compressed images is that the calibration did not properly estimate the properties of the cover image. We explain this in more detail in the next section.

A. Calibration of double-compressed images

If the stego image has been double-compressed before embedding (as it is the case for F5 and OutGuess), the calibration will output an approximation to the single-compressed cover image instead of the double-compressed one. Failure to calibrate properly may lead to very inaccurate steganalysis results [7]. The proper way to calibrate is to estimate the primary quantization matrix and mimic the double compression when calibrating. In other words, the decompressed stego image after cropping should be first compressed with the primary (cover) quantization matrix $Q^{(1)}$, decompressed, and finally compressed again with the secondary quantization matrix $Q^{(2)}$. The primary quantization matrix $Q^{(1)}$ is estimated using the primary quantization matrix estimator described in Section IV. Figure 3 shows the calibration process properly modified to compensate for double-compression.

B. Multi-classifier

The multi-classifier for double-compressed JPEG images classifies into three classes: cover images, images embedded by F5 and OutGuess algorithms, because under the assumptions from Section II-B, only these two algorithms can produce double-compressed images. The construction of the multi-classifier for double-compressed images is very similar to the construction of the multi-classifier for single-compressed images described in Section V. It consists of two max-wins SVM-based multi-classifiers, $\mathcal{D}_{75}, \mathcal{D}_{80}$, designed for secondary quality factors 75 and 80. We remark that the feature vector for these classifiers has dimensionality 275 (one more than for the single-compression classifiers) because we add the estimated primary quality factor as another feature.

The development of the multi-classifier for double-compressed JPEG images requires considerable computation power, because we have to prepare examples of all combinations of primary and secondary quality factors we want to detect. This is why we only created the multi-classifier for two secondary quality factors chosen as the default quality factors of OutGuess and F5, respectively. The primary quality factors were all taken from $\mathcal{Q}_{34}$.

VII. Experimental Results

Here, we first describe the image database used to create the multi-classifier and the training and testing sets. This section also contains interpretation of all experimental results.

A. Image database

The image database was created from 6006 images of natural scenes taken under varying conditions (exterior and interior images, images taken with and without flash and at various ambient temperatures) with the following digital cameras: Nikon D100, Canon G2, Olympus Camedia 765, Kodak DC 290, Canon PowerShot S40, images from Nikon D100 downsampled by a factor of 2.9 and 3.76, Sigma SD9, Canon EOS D30, Canon EOS D60, Canon PowerShot G3, Canon PowerShot G5, Canon PowerShot Pro 90IS, Canon PowerShot S100, Canon PowerShot S50, Nikon CoolPix 5700, Nikon CoolPix 990, Nikon CoolPix SQ, Nikon D10, Nikon D1X, Sony CyberShot DSC F505V, Sony CyberShot DSC F707V, Sony CyberShot DSC S75, and Sony CyberShot DSC S85. All images were taken either in the raw TIFF format or in a proprietary manufacturer raw data format, such as NEF (Nikon) or CRW (Canon) converted to the 24-bit TIFF format. The image resolution ranged from 800×631 for the scaled images to 3008×2000. We have included scaled images into the database to increase its diversity.

In the beginning, we divided all 6006 raw images into two disjoint groups. The first group, which consisted of 3500 images, was used to create the training examples from the first 7 cameras on the list (including the down-sampled images). The second group containing the remaining 2500 images was used for testing. Thus, no image or its different form were simultaneously used for testing and training. This strict division of images enabled us to estimate the performance on never seen images from completely different sources. Images from both groups were processed in exactly the same way.

We created single-compressed stego images with 34 different quality factors from the set $\mathcal{Q}_{34}$ embedded by 6 steganographic algorithms: F5, MBS1, MBS2, JP Hide&Seek, OutGuess, and Steghide. Using all algorithms, except MBS2, we embedded messages of

target	Cover	F5	JPHS	MBS1	MBS2	OutGuess	Steghide
F5	13.32%	83.45%	0.24%	0.02%	0.07%	2.82%	0.08%
OutGuess	11.73%	3.69%	0.05%	0.19%	0.31%	83.82%	0.21%
Cover	81.55%	11.96%	0.37%	0.02%	0.09%	5.96%	0.04%

TABLE II
CONFUSION MATRIX ON DOUBLE-COMPRESSED IMAGES OBTAINED USING THE SINGLE-COMPRESSION MULTI-CLASSIFIER DESCRIBED IN SECTION V AND VII. THESE POOR RESULTS MOTIVATE THE CONSTRUCTION DESCRIBED IN THIS SECTION.

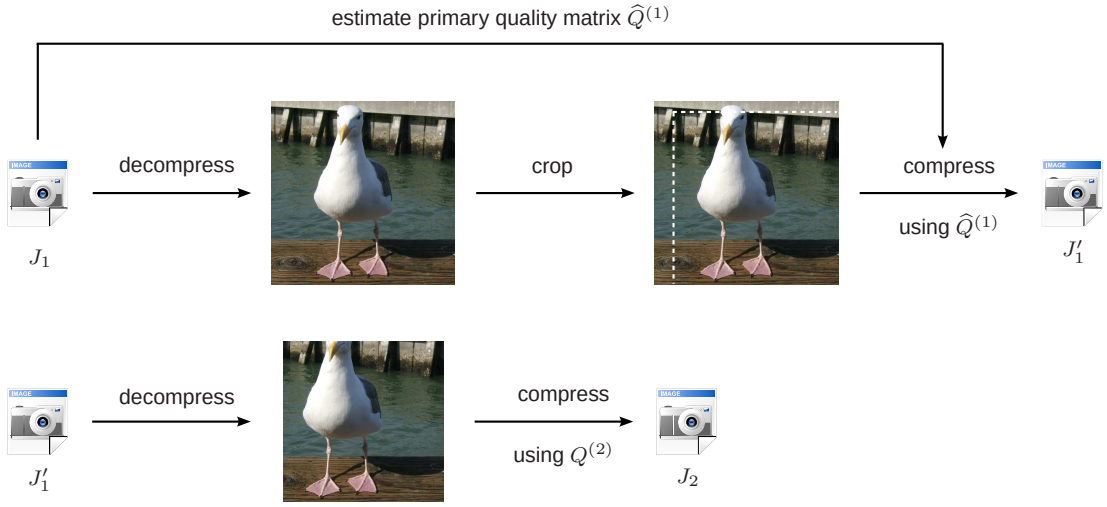


Fig. 3. Calibration for double-compressed images.

three different lengths: 100%, 50%, and 25% of the embedding capacity for each algorithm. All MBS2 images were embedded only with 30% of the capacity of MBS1 because during embedding of longer messages the deblocking part of MBS2 often fails. The capacity of JP Hide&Seek was estimated as 10% of the size of the JPEG file, as recommended by its author. The implementation of OutGuess had to be modified to produce images with quality factor smaller than 75.

The double-compressed stego images were created by OutGuess and F5 only. We embedded message lengths 100%, 50%, and 25% of embedding capacity for each algorithm and image. The double-compressed images were prepared with 34 different primary quality factors from $\mathcal{Q}_{34}$ and with secondary quality factors 75 and 80—the default of OutGuess and F5.

The database contains single and double-compressed cover images with the same combinations of quality factors as the stego images. The total number of images in the database was $|\mathcal{Q}_{34}| \times 17 \times 6006 + |\mathcal{Q}_{34}| \times 2 \times 7 \times 6006 \approx 6,330,000$.

B. Training the multi-classifier for single-compressed JPEG images

The max-wins multi-classifier trained for each quality factor out of $\mathcal{Q}_{34}$ employs $\binom{n_{class}}{2} = 21$ binary classifiers for every pair out of $n_{class} = 7$ classes. The construction of the training set for multi-classifiers for quality factor 75 and 80 differs from multi-classifiers for other quality factors $\mathcal{Q}_{34}\backslash\{75, 80\}$. This difference is caused by the fact that the multi-classifiers on quality factors 75 and 80 should also be trained on double-compressed images detected as single-compressed.

For quality factors $\mathcal{Q}_{34}\backslash\{75, 80\}$, the training set of each binary classifier consisted of 3400 examples from each class (6800 examples in total). If, for a given class, more than one message length was available (all algorithms except MBS2 and cover), the training set had an equal number of stego images embedded with message lengths corresponding to 100%, 50%, and 25% of the algorithm embedding capacity.

For the quality factors 75 and 80, the training set should also contain double-compressed images incorrectly detected by the double-compression detector (Section IV-B) as single-compressed. There are two main

benefits of including these double-compressed images in the training set for the single-compression multi-classifier. First, the multi-classifier needs to handle such images properly, because the double-compression detector is tuned to low false positive rate, which comes at the expense of a higher false negative rate (double-compressed images detected as single-compressed). Second, these misclassified images increase the number of cover, F5, and OutGuess examples for training, which enables us to have a larger training set for all binary classifiers except for the binary classifiers detecting MBS2. The training set for binary classifiers detecting the MBS2 algorithm consisted of 3400 examples from each class (6800 examples in total). The training sets for all other binary classifiers consisted of approximately 3×3400 examples from each class (20400 examples in total).

C. Training the multi-classifier for double-compressed JPEG images

Both max-wins multi-classifiers (for secondary quality factors 75 and 80) employ only 3 binary classifiers. All images available for training were pre-classified by the double-compression detector (Section IV-B) in order to train the binary classifiers on double-compressed images *detected as* double-compressed. The training set for all 3 binary classifiers consisted of 10000 examples (20000 examples total). The distribution of primary quality factors of images in the training sets followed the distribution determined by the double-compression detector. The training set did not contain examples of single-compressed images detected as double-compressed.

D. Accuracy of the entire steganalyzer

The accuracy of the entire steganalyzer, depicted in Figure 1, was obtained on a testing set containing approximately $|\mathcal{Q}_{34}| \times 17 \times 2506 + |\mathcal{Q}_{34}| \times 2 \times 7 \times 2506 \approx 2,640,000$ images never seen by the classifier. The actual running time needed to classify one image depends its type (single- or double-compressed image) and size. On average, for a five megapixel image, the running time is about 10 sec. on a 64bits 2.2GHz AMD Opteron machine.

Figure 4 shows the detection accuracy (image is correctly assigned to the embedding algorithm) of the multi-classifier on stego images. The accuracy on images containing longer messages (50% and more) is most of the time better than 97% and remains practically the same through the entire range of quality factors. As the embedded message becomes shorter, the accuracy of correct detection decreases, which is to be expected. The accuracy on images embedded with MBS1, MBS2, JP Hide&Seek, and Steghide exhibits drops for quality factors 75 and 80 caused by incorrect detection of single-compressed images as double-compressed (recall that the double-compression detector is used only for secondary quality factors 75 and 80). This phenomenon underlines the importance of a low false positive rate of the double-compression detector, as already discussed in Section IV-B.

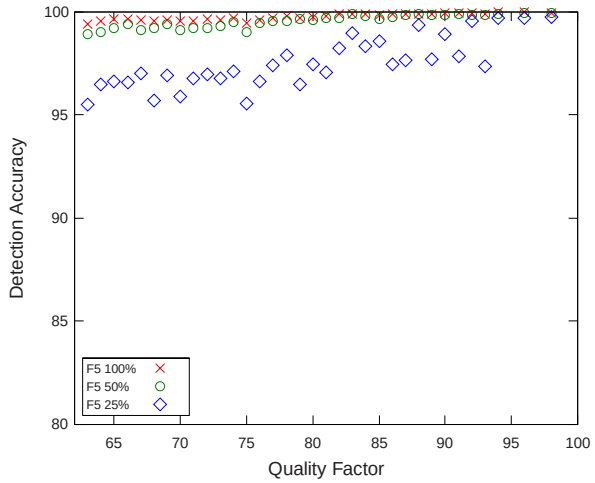
The overall false positive rate on single-compressed cover images is 1.2%, which is relatively low, considering the fact that during training we did not impose any bias to lower the false positive rate. The detection accuracy on cover images with different quality factors is shown in Figure 5.

Comparing the detection accuracy on double-compressed images shown in Figure 6 with the detection accuracy on single-compressed images (Figures 4, 5), we can see that the accuracy is not much affected by double-compression. Moreover, it changes only little with varying primary quality factor, which confirms our claim made in Section IV-C that the failure to correctly estimate the primary quality factor has negligible effect on steganalysis. The only exception are images embedded with a 25% message by the F5 algorithm. In this case, the performance is worse for double-compressed images. We attribute this loss in accuracy to the combined distortion caused by double compression and the small number of embedding changes due to matrix embedding.

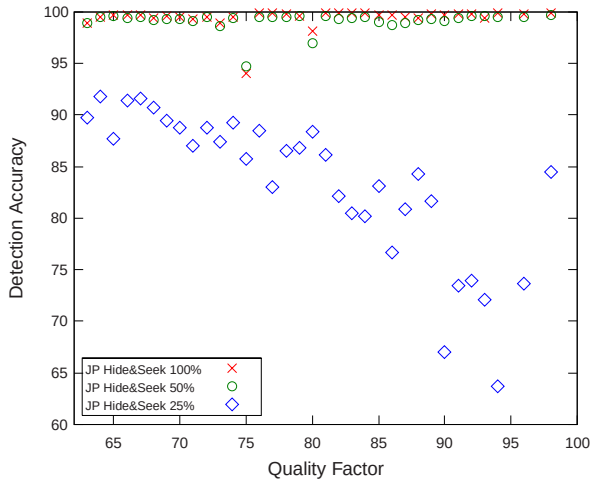
Tables III,IV show the confusion matrices on single and double-compressed images with secondary quality factors 75 and 80. The classification accuracy decreases as the embedded message gets shorter. At this point, we would like to point out certain fundamental limitations that cannot be overcome. In particular, it is not possible to distinguish between two algorithms that employ the same embedding mechanism by inspecting the statistics of DCT coefficients. For example, two algorithms that use LSB embedding along a pseudo-random path will be indistinguishable in the feature space. This phenomenon might be responsible for "merging" of the MBS1, MBS2, and Steghide classes.

E. Novelty detection (Jsteg and MMx)

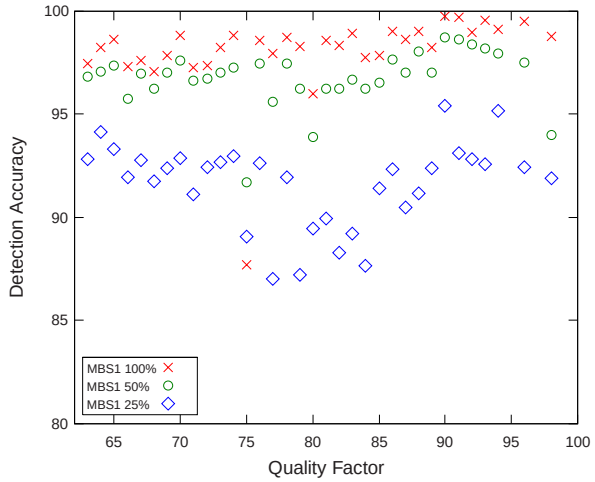
To asses the ability of the multi-classifier to generalize to previously unseen stego methods, we present to it stego images created by two methods on which the classifier was not trained: Jsteg (http://zooid.org/~paul/crypto/jsteg/) and the recently proposed MMx [19]. Jsteg



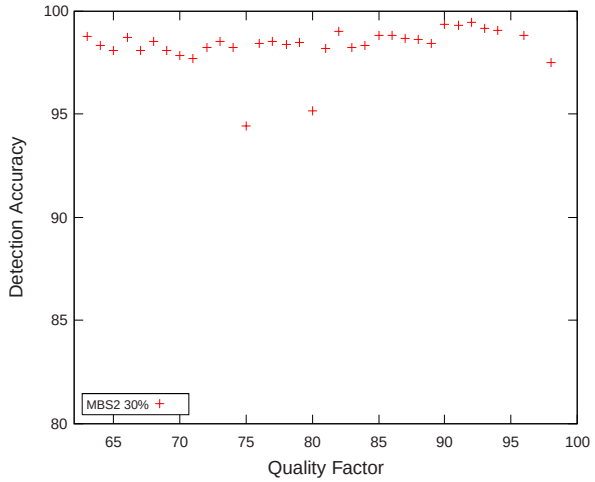
(a) F5



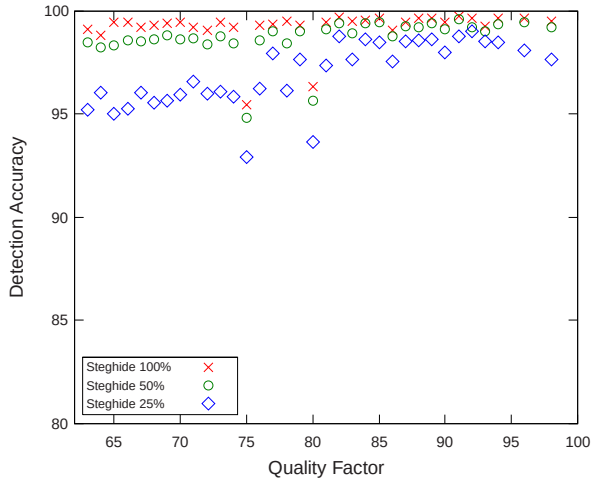
(b) JP Hide&Seek



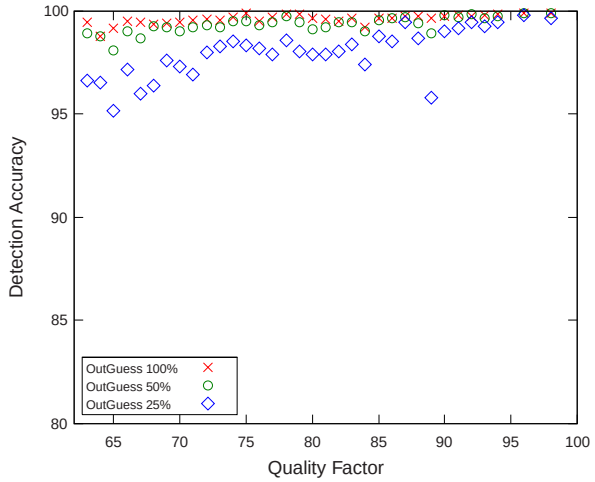
(c) MBS1



(d) MBS2

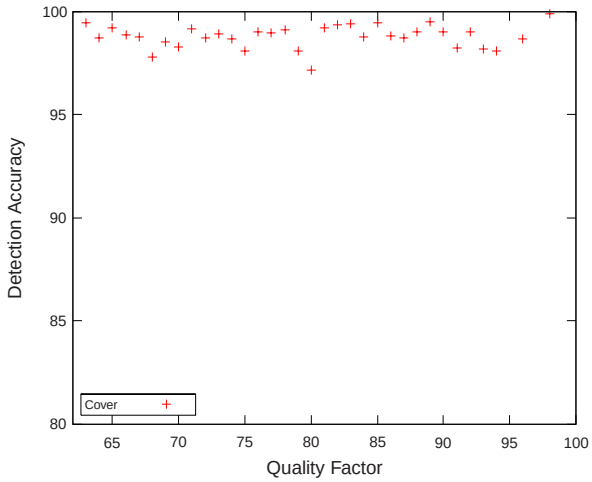


(e) Steghide



(f) OutGuess

Fig. 4. Detection accuracy of the multi-classifier on single-compressed JPEG images. Note the different y-axis scale for JP Hide&Seek.



(a) Cover

Fig. 5. Detection accuracy of the multi-classifier on single-compressed cover images from the testing set.

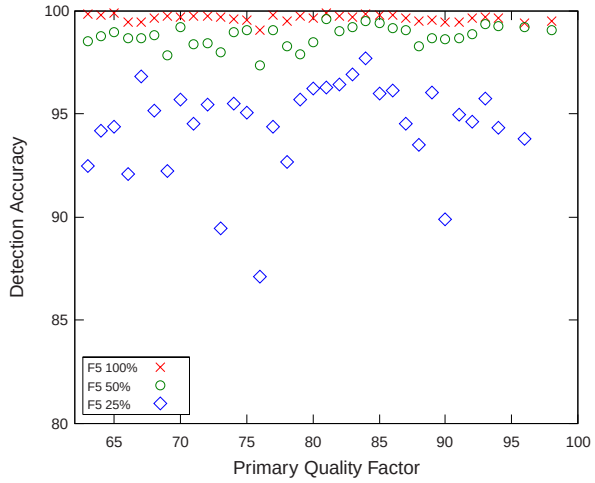
		Classified as					
Embedding algorithm	Cover	F5	JP Hide&Seek	MBS1	MBS2	OutGuess	Steghide
F5 100%	0.24%	99.65%	0.00%	0.00%	0.00%	0.10%	0.00%
JP Hide&Seek 100%	2.88%	2.52%	94.05%	0.00%	0.00%	0.56%	0.00%
MBS1 100%	0.16%	0.40%	0.00%	87.72%	1.44%	10.04%	0.24%
OutGuess 100%	0.04%	0.35%	0.00%	0.02%	0.01%	99.57%	0.01%
Steghide 100%	0.04%	0.32%	0.00%	0.08%	0.08%	4.03%	95.45%
F5 50%	0.81%	98.79%	0.00%	0.01%	0.01%	0.36%	0.02%
JP Hide&Seek 50%	2.48%	2.28%	94.69%	0.00%	0.00%	0.56%	0.00%
MBS1 50%	0.36%	0.20%	0.00%	91.69%	1.56%	5.64%	0.56%
OutGuess 50%	0.25%	0.42%	0.00%	0.02%	0.02%	99.26%	0.03%
Steghide 50%	0.44%	0.36%	0.00%	0.28%	0.04%	4.03%	94.85%
MBS2 30%	0.76%	0.24%	0.00%	0.76%	94.45%	3.67%	0.12%
F5 25%	4.60%	94.47%	0.04%	0.01%	0.01%	0.78%	0.10%
JP Hide&Seek 25%	10.98%	2.68%	85.78%	0.00%	0.00%	0.56%	0.00%
MBS1 25%	3.47%	0.60%	0.00%	89.06%	1.16%	2.56%	3.15%
OutGuess 25%	1.81%	0.61%	0.00%	0.02%	0.03%	97.40%	0.13%
Steghide 25%	3.04%	0.96%	0.00%	0.56%	0.20%	2.32%	92.93%
Cover	98.40%	0.95%	0.05%	0.01%	0.00%	0.59%	0.01%

TABLE III

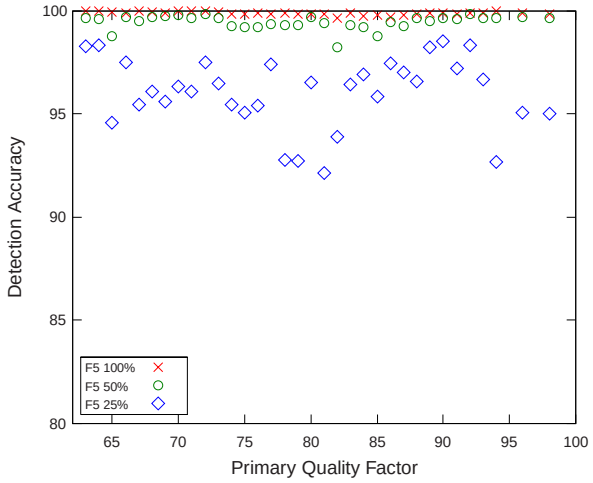
CONFUSION MATRIX FOR THE MULTI-CLASSIFIER TRAINED FOR QUALITY FACTOR 75 TESTED ON SINGLE AND DOUBLE-COMPRESSED 75-QUALITY JPEG IMAGES. THE FIRST COLUMN CONTAINS THE EMBEDDING ALGORITHM AND THE RELATIVE MESSAGE LENGTH. THE REMAINING COLUMNS SHOW THE CLASSIFICATION RESULTS.

is a simple adaptation of LSB embedding to the quantized DCT coefficients (coefficients 0 and 1 are skipped), while the MMx method is a sophisticated algorithm that utilizes the knowledge of the uncompressed image as side information to minimize the overall embedding distortion. The embedding mechanism also uses a modified matrix embedding to further minimize the impact of steganographic changes.

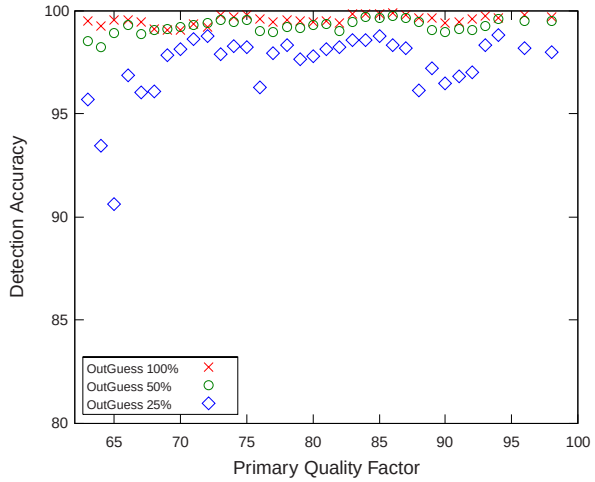
From our testing image database, we prepared stego images embedded with 100%, 50%, and 25% of the embedding capacity of Jsteg. The stego images for MMx were embedded with random messages of relative length 2/3, 3/7, 4/15 bpac (bits per non-zero DCT coefficient). These payloads were selected to match the capacities determined by the codimension of the Hamming codes used for matrix embedding. MM2, and MM3 stand for the version of the algorithm that allows up to two or three modifications per embedding block. The security improves with the number of allowed changes. The "MM1" algorithm would be essentially identical to F5. The quality factor for all images was set to 75. All stego images were single-compressed.



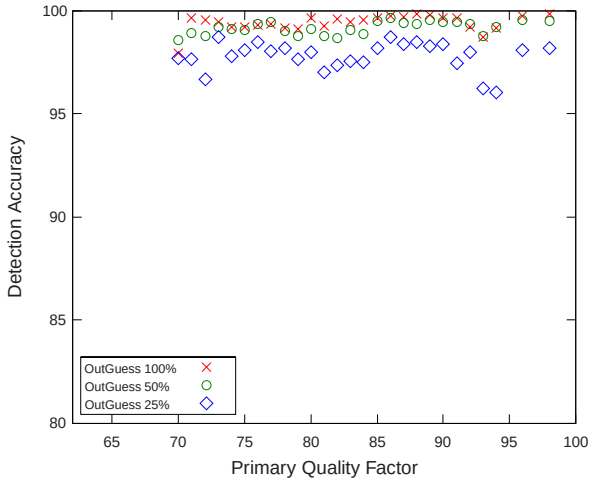
(a) F5, secondary quality factor 75



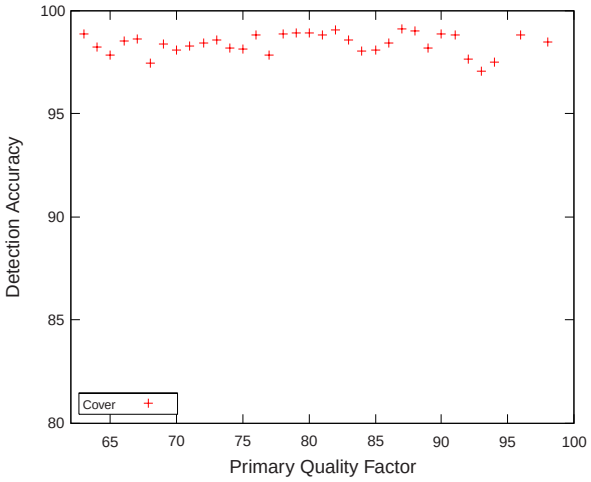
(b) F5, secondary quality factor 80



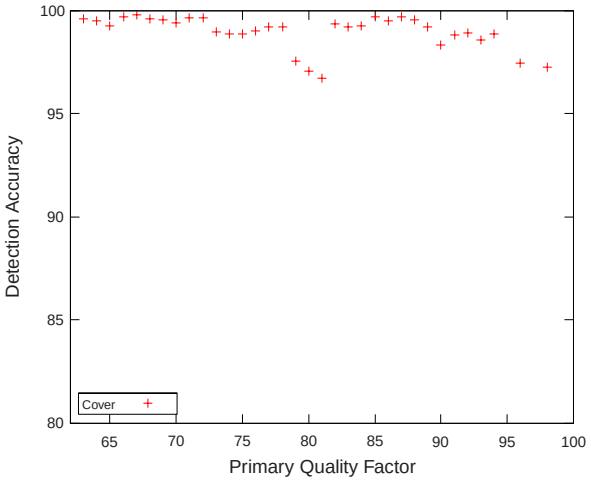
(c) OutGuess, secondary quality factor 75



(d) OutGuess, secondary quality factor 80



(e) Cover, secondary quality factor 75



(f) Cover, secondary quality factor 80

Fig. 6. Accuracy of the multi-classifier on double-compressed JPEG images with secondary quality factors 75 and 80. The graph showing OutGuess images with secondary quality factor 80 starts from the primary quality factor 70 because OutGuess fails to embed message into images with combination of primary quality factors $63, \ldots, 69$ and secondary quality factor 80.

Embedding algorithm	Cover	F5	JP Hide&Seek	MBS1	MBS2	OutGuess	Steghide
	Classified as						
F5 100%	0.07%	99.89%	0.00%	0.00%	0.00%	0.03%	0.00%
JP Hide&Seek 100%	0.84%	1.04%	98.12%	0.00%	0.00%	0.00%	0.00%
MBS1 100%	0.08%	0.44%	0.00%	96.00%	1.12%	1.88%	0.48%
OutGuess 100%	0.04%	0.46%	0.00%	0.01%	0.00%	99.47%	0.01%
Steghide 100%	0.04%	0.64%	0.00%	0.04%	0.04%	2.92%	96.33%
F5 50%	0.41%	99.47%	0.00%	0.00%	0.01%	0.10%	0.00%
JP Hide&Seek 50%	1.40%	1.56%	97.00%	0.00%	0.00%	0.04%	0.00%
MBS1 50%	0.36%	0.72%	0.00%	93.89%	1.96%	2.04%	1.04%
OutGuess 50%	0.23%	0.60%	0.00%	0.02%	0.02%	99.11%	0.02%
Steghide 50%	0.28%	0.80%	0.00%	0.08%	0.12%	3.08%	95.65%
MBS2 30%	1.36%	0.28%	0.00%	0.84%	95.16%	2.12%	0.24%
F5 25%	3.49%	96.07%	0.03%	0.00%	0.00%	0.39%	0.02%
JP Hide&Seek 25%	9.19%	2.32%	88.38%	0.00%	0.00%	0.12%	0.00%
MBS1 25%	2.08%	0.60%	0.00%	89.46%	1.48%	1.92%	4.47%
OutGuess 25%	1.33%	0.82%	0.02%	0.01%	0.03%	97.75%	0.04%
Steghide 25%	2.44%	1.74%	0.00%	0.22%	0.09%	1.87%	93.64%
Cover	98.93%	0.72%	0.18%	0.00%	0.00%	0.18%	0.00%

TABLE IV

CONFUSION MATRIX FOR THE MULTI-CLASSIFIER TRAINED FOR QUALITY FACTOR 80 TESTED ON SINGLE AND DOUBLE-COMPRESSED 80-QUALITY JPEG IMAGES. THE FIRST COLUMN CONTAINS THE EMBEDDING ALGORITHM AND THE RELATIVE MESSAGE LENGTH. THE REMAINING COLUMNS SHOW THE CLASSIFICATION RESULTS.

	Cover	F5	JP Hide&Seek	MBS1	MBS2	OutGuess	Steghide
Jsteg 100%	0.20%	57.91%	0.00%	0.00%	0.04%	41.81%	0.04%
Jsteg 50%	0.20%	57.59%	0.00%	0.04%	2.40%	39.58%	0.20%
Jsteg 25%	1.04%	57.63%	0.00%	5.47%	3.67%	30.99%	1.20%

TABLE V

CONFUSION MATRIX FOR 2504 IMAGES EMBEDDED WITH JSTEG AS DETECTED WITH THE MULTI-CLASSIFIER (NOT TRAINED ON JSTEG).

	Cover	F5	JP Hide&Seek	MBS1	MBS2	OutGuess	Steghide
MM2-(1,3,2)	0.56%	0.92%	0.00%	0.80%	91.29%	1.92%	4.51%
MM2-(1,7,3)	0.92%	0.20%	0.00%	14.18%	27.08%	1.68%	55.95%
MM2-(1,15,4)	10.34%	0.44%	0.00%	27.52%	1.24%	0.68%	59.78%
MM3-(1,3,2)	0.44%	1.04%	0.00%	0.84%	91.61%	1.84%	4.23%
MM3-(1,7,3)	1.08%	0.20%	0.00%	15.06%	26.40%	1.88%	55.39%
MM3-(1,15,4)	17.05%	0.44%	0.04%	27.84%	1.16%	0.56%	52.92%

TABLE VI

CONFUSION MATRIX FOR 2504 IMAGES EMBEDDED WITH MMX AS DETECTED WITH THE MULTI-CLASSIFIER (NOT TRAINED ON MMX).

Table V shows that Jsteg is very reliably detected using the multi-classifier even though Jsteg embedded images were not used for the classifier construction. It is interesting to note that Jsteg was mostly detected as F5 and OutGuess. Images embedded with the MMx algorithm were also reliably detected as stego and were assigned mostly to Model Based Steganography and Steghide (see Table VI). The missed detection rate for MMx quickly increases with decreasing message length due to the improved matrix coding scheme. We note that it is possible that the multi-classifier will not be able to detect steganographic methods with entirely different types of embedding changes. This interesting and relevant issue will be subject of our future research.

F. Steganalysis of images with custom quantization matrix

The purpose of the experiment presented in this section is to evaluate the accuracy of the steganalyzer on images with custom (non-standard) quantization matrices. Such matrices are used in some consumer digital cameras.

We collected 300 JPEG images from a 6-megapixel camera Fuji E550 with JPEG compression setting HQ. We chose this camera because it uses a wide range of custom quantization tables depending on the scene.

Among the 300 images, there were total of 165 different quantization matrices. The average quality factor across all 300 images, determined as described in Section IV-A, was approximately 90. As before, we embedded 3 relative payloads, 25%, 50%, and 100% of the image capacity, using F5, JP Hide&Seek, MBS1, Steghide, and OutGuess, and 30% relative message length using MBS2. The stego images produced by JP Hide&Seek, Steghide, MBS1, and MBS2 retained their original custom quantization matrices, while images from F5 and OutGuess produced double-compressed images with their default quality factors.

The classification accuracy of the proposed blind steganalyzer is presented in Table VII. The detection accuracy for stego images is quite good, but the detection suffers from an increased false positive rate. It is to be expected that the steganalyzer may not classify images with very high quality factors correctly. This limitation occurs due to the fact that at higher JPEG qualities more non-zero AC coefficients start appearing in medium and high frequencies. And if the match between the custom quantization matrix and the closest standard matrix is poor in those bands, the features extracted by the steganalyzer will start statistically deviating from the training set, which will result in less reliable detection.

VIII. Conclusion

The goal of this paper is to demonstrate that it is possible to reliably classify both single and double-compressed JPEG images to current popular steganographic algorithms using a multi-classifier based on supervised training. We expect this tool to be useful for law enforcement and forensic analysts, because identification of the stego program is the first necessary step towards extracting the secret message.

Even though some of the individual modules of the proposed steganalysis system were previously analyzed by the authors, they were never tested when incorporated in a large system. The fusion of the modules called for appropriate architecture that scales well with the number of classified stego systems and the number of images in the training and testing database while satisfying specific design criteria, such as low probability of false alarm. The solution described in this paper required proper internal hierarchical structure of the modules as well as development of new tools and their tuning to meet specified performance criteria.

In designing the classifier, several obstacles had to be overcome. Since the JPEG format accepts as a parameter the quantization matrix, cover images stored with different matrices (quality factors) have different statistical properties and thus the classifier should not be confused by this diversity. Moreover, some stego programs may produce double-compressed images on their output, which further increases the diversity of images to be classified and may lead to very inaccurate steganalysis without appropriate attention. We approached this problem by constructing the classifier from three parts – the double-compression detector, the classifier for single-compressed images, and the classifier for double-compressed images. The task of the double-compression detector is to detect when an image has been double-compressed and then, if it was, estimate its primary quality factor. The image under inspection is then sent either to a single-compression multi-classifier separately designed for each quality factor, or a double-compression multi-classifier, designed for two secondary quality factors–75 and 80, the default quality factors of F5 and OutGuess.

The classifier design is a result of several simplifying assumptions that we imposed in order to make our task manageable. First, we assume that the cover images are always either single-compressed or uncompressed (i.e., we do not consider the case when the cover image has gone through multiple compression prior to embedding even though this scenario is not unlikely). Furthermore, we assume that F5 and OutGuess were both run only with quality factors 75 and 80. These assumptions were necessary to reach a reasonable trade-off between the computational cost of training the classifiers and available storage for storing the stego images and their features. The complete process of training and testing required processing over 6 million images and over 4TB of disk space. All classifiers were implemented as soft-margin support vector machines with the Gaussian kernel. The feature set consisted of 193 extended DCT features and 81 Markov features V.

The accuracy of the presented steganalyzer was evaluated from experiments on more than 2.6 million images under very general settings. The ability of the blind steganalyzer to generalize to previously unseen stego methods was tested on Jsteg and MMx algorithms. The steganalyzer was able to reliably detect the stego content for all tested relative payloads. We note that in general the multi-class steganalyzer may not be able to detect stego methods with an embedding mechanism that is dissimilar to those of the trained methods. In our future work, we would like to investigate the generalization limits of the multi-classifier in more detail and focus on the problem of detection of unknown algorithms.

This work on steganalysis of JPEG images points to novel design principles for steganalysis not limited to

	Cover	F5	JPHS	MBS1	MBS2	OutGuess	Steghide
F5 100%	0.67%	99.33%	0.00%	0.00%	0.00%	0.00%	0.00%
JPHS 100%	1.00%	0.33%	98.67%	0.00%	0.00%	0.00%	0.00%
MBS1 100%	0.00%	0.00%	0.00%	97.31%	2.69%	0.00%	0.00%
OutGuess 100%	0.00%	0.00%	0.00%	0.00%	1.67%	98.33%	0.00%
Steghide 100%	0.00%	0.00%	0.00%	1.67%	1.33%	0.00%	97.00%
F5 50%	2.00%	98.00%	0.00%	0.00%	0.00%	0.00%	0.00%
JPHS 50%	0.33%	0.67%	99.00%	0.00%	0.00%	0.00%	0.00%
MBS1 50%	0.00%	0.00%	0.00%	98.65%	1.01%	0.00%	0.34%
OutGuess 50%	0.67%	0.00%	0.00%	1.67%	4.67%	92.67%	0.33%
Steghide 50%	0.00%	1.33%	0.00%	2.00%	0.33%	0.00%	96.33%
MBS2 30%	0.00%	0.00%	0.00%	15.49%	83.84%	0.00%	0.67%
F5 25%	7.00%	93.00%	0.00%	0.00%	0.00%	0.00%	0.00%
JPHS 25%	5.33%	1.00%	93.67%	0.00%	0.00%	0.00%	0.00%
MBS1 25%	0.67%	0.67%	0.00%	90.91%	0.67%	0.00%	7.07%
OutGuess 25%	17.00%	6.67%	0.33%	0.67%	2.67%	62.67%	10.00%
Steghide 25%	3.00%	3.33%	0.00%	5.33%	0.00%	0.00%	88.33%
Cover 0%	90.67%	3.33%	6.00%	0.00%	0.00%	0.00%	0.00%

TABLE VII
CONFUSION MATRIX FOR STEGO IMAGES WITH NON-STANDARD QUANTIZATION TABLES FROM FUJI E550.

the JPEG domain. In particular, the idea to preclassify the image under investigation based on its origin or processing history and then analyzing it with appropriately trained steganalyzer can boost the performance of the steganalyzer in other embedding domains. For example a preclassifier of the type of cover source (e.g., scan, digital photograph, decompressed JPEG) might be used to select the best steganalysis method for the particular cover class. This idea was recently proposed to improve the accuracy of LSB detectors [18].

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, *Proceedings SPIE, Electronic Imaging, Security, and Watermarking of Multimedia Contents III, San Jose, CA, 2001*, volume 4314, pages 523–531.

[3] I. Avcibas, N.D. Memon, and B. Sankur. Image steganalysis with binary similarity measures. In *Proceedings, International Conference on Image Processing, Rochester, NY, September 22–25, 2002*, volume 3, pages 645–648. IEEE, 2002.

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

[5] 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.

[6] 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, IW 2002, Noordwijkerhout, The Netherlands, October 7–9, 2002*, volume 2578 of *Lecture Notes in Computer Science*, pages 340–354. Springer-Verlag, New York.

[7] 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.

[8] J. Fridrich, M. Goljan, and D. Hogea. Steganalysis of JPEG images: Breaking the F5 algorithm. In *5th International Workshop on Information Hiding, Noordwijkerhout, The Netherlands, October 7–9, 2002*, volume 2578 of *Lecture Notes in Computer Science*, pages 310–323. Springer, New York.

[9] J. Fridrich and J. Lukáš. Estimation of primary quantization matrix in double compressed JPEG images. In *Digital Forensic Research Workshop*, 2003.

[10] D. Fu, Y. Q. Shi, and Q. Su. A generalized Benford's law for JPEG coefficients and its applications in image forensics. In E. Delp and P. W. Wong, editors, *Proceeedings of SPIE Electronic Imaging, Security and Watermarking of Multimedia Contents IX*, volume 6505, pages 1L1–1L11, 2007.

[11] M. Goljan, J. Fridrich, and T. Holotyak. New blind steganalysis and its implications. In E.J. Delp and P.W. Wong, editors, *Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VIII, San Jose, CA, January 16–19*, volume 6072, pages 1–13, January 2006.

[12] J.J. Harmsen and W.A. Pearlman. Capacity of steganalysis channels. In J. Dittmann and J. Fridrich, editors, *Proceedings ACM Meltimedia and Security Workshop, New York, NY, August 1–2, 2005*, pages 11–24.

[13] J.J. Harmsen and W.A. Pearlman. Steganalysis of additive noise modelable information hiding. In E.J. Delp and P.W. Wong, editors, *Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents V, Santa Clara, CA, January 21–24, 2003*, volume 5020, pages 131–142.

[14] 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.

[15] C. Hsu and C. Lin. A comparison of methods for multi-class support vector machines. Technical report, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, 2001.

http://citeseer.ist.psu.edu/hsu01comparison.html.

[16] A. L. Jain. *Fundamentals of Digital Image Processing*. Prentice-Hall, 1989.

[17] S. Katzenbeisser and F.A.P. Petitcolas. On defining security in steganographic systems. In E.J. Delp and P.W. Wong, editors, *Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents IV, San Jose, CA, January 21–24, 2002*, volume 4675, pages 50–56.

[18] A. D. Ker and R. Böhme. Revisiting weighted stego-image steganalysis. In E. J. Delp and P. W. Wong, editors, *Proceedings SPIE, Electronic Imaging, Security, Forensics, Steganography, and Watermarking of Multimedia Contents X*, volume 6819, San Jose, CA, January 27–31, 2008.

[19] 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.

[20] S. Lyu and H. Farid. Steganalysis using color wavelet statistics and one-class support vector machines. In E.J. Delp and P.W. Wang, editors, *Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VI, San Jose, CA, January 19–22, 2004*, volume 5306, pages 35–45.

[21] S. Lyu and H. Farid. Steganalysis using higher-order image statistics. 2006.

[22] W. B. Pennebaker and J. L. Mitchell. *JPEG still image data compression standard*. Van Nostrand Reinhold, 1993. ISBN: 0-442-01272-1.

[23] T. Pevný and J. Fridrich. Detection of double-compression for applications in steganography. *Submitted to IEEE Transactions on Information Forensics and Security*, 2007.

[24] T. Pevný and J. Fridrich. Towards multi-class blind steganalyzer for JPEG images. In *International Workshop on Digital Watermarking*, volume 3710 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 2005.

[25] T. Pevný and J. Fridrich. Determining the Stego Algorithm for JPEG Images. In *Special Issue of IEE Proceedings — Information Security*, volume 153, pages 75–139, 2006.

[26] T. Pevný and J. Fridrich. Multiclass blind steganalyasis for JPEG images. In E.J. Delp and P.W. Wong, editors, *Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VIII, San Jose, CA, January 16–19*, volume 6072, page O0, January 2006.

[27] 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.

[28] A.C. Popescu and H. Farid. Statistical tools for digital forensic. In J. Fridrich, editor, *6th International Workshop on Information Hiding*, volume 3200 of *Lecture Notes in Computer Science*, pages 128–147. Springer-Verlag, Berlin, 2004.

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

[30] 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.

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

[32] 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 *Lecture Notes in Computer Science*. Springer-Verlag, New York, 2006.

[33] Y. Wang and P. Moulin. Statistical modeling and steganalysis of dft-based image steganography. In E.J. Delp and P.W. Wong, editors, *Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Content VIII, San Jose, CA, January 16–19, 2006*, volume 6072, pages 02–1–02–11.

[34] 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.

[35] 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 functions. In M. Barni, J. Herrera, S. Katzenbeisser, and F. Pérez-González, editors, *Proceedings 7th Information Hiding Workshop, Barcelona, Spain, June 6–8, 2005*, volume 3727 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 2005.

[36] J. Zöllner, H. Federrath, H. Klimant, A. Pfitzmann, R. Piotraschke, A. Westfeld, G. Wicke, and G. Wolf. Modeling the security of steganographic systems. In D. Aucsmith, editor, *Information Hiding, 2nd International Workshop*, volume 1525 of *Lecture Notes in Computer Science*, pages 344–354. Springer-Verlag, New York, 1998.