Towards Multi-class Blind Steganalyzer for JPEG Images

Tomáš Pevný[a], Jessica Fridrich[b]

[a]Department of Computer Science, [b]Department of Electrical and Computer Engineering, SUNY Binghamton, Binghamton, NY 13902–6000

Abstract. In this paper, we use the previously proposed calibrated DCT features [9] to construct a Support Vector Machine classifier for JPEG images capable of recognizing which steganographic algorithm was used for embedding. This work also constitutes a more detailed evaluation of the performance of DCT features as in [9] only a linear classifier was used. The DCT features transformed using Principal Component Analysis enable an interesting visualization of different stego programs in a three-dimensional space. This paper demonstrates that, at least under some simplifying assumptions in which the effects of double compression are ignored, it is possible to reliably classify stego images to their embedding techniques. The classifier is capable of generalizing to previously unseen techniques.

1 Introduction

The goal of steganography is to hide the very presence of communication by hiding messages in innocuous looking objects, such as digital media files. The original object, also called the cover object, is slightly modified to obtain the stego object that carries the secret message. In a symmetric communication scheme, the embedding process depends on a secret (stego key) shared between both communicating parties. The main requirement of steganalysis is undetectability of the hidden data by an unauthorized party who knows all details of the embedding mechanism and the source of cover objects but does not have the stego key (Kerckhoffs' principle). The concept of steganographic security (undetectability) was formalized, for example, in [2,5,25,14].

Methods for discovering the presence of hidden messages and determining their attributes belong to steganalysis. In practice, a steganographic scheme is considered secure if no existing steganalytic attack can be used to distinguish between cover and stego objects with success better than random guessing [6].

Steganalytic methods can be roughly divided into two categories. The first category is formed by methods targeted to a specific embedding technique, capitalizing on the assumption that we know the embedding algorithm [10]. The second category is formed by blind approaches in which the knowledge of the embedding algorithm is not assumed [9,8,3,4]. Instead, most blind approaches assume that one can somehow characterize all "natural images" using an appropriate set of features that should be as sensitive to steganographic modifications as possible. A classifier is then built to distinguish in the feature space between natural images and stego images.

As one can expect, targeted approaches should provide better reliability and accuracy than blind approaches. Targeted methods range from very specific simple ideas that pertain to a specific implementation to more general methods that address a general embedding paradigm, such as LSB embedding [10,7], and finally to methods that can be easily adjusted to address a very large spectrum of data hiding methods (e.g., detection of additive signals, such as ±1 embedding [22,16,15]). The disadvantage of targeted methods is that their design cannot be automatized and new techniques might have to be developed each time a new steganographic methodology appears. This problem of extensibility is removed by blind approaches.

From a certain point of view, however, there is no difference between targeted and blind approaches as they both benefit from progress in the other. In fact, in each targeted method one or more quantities (distinguishing statistics [10]) are calculated, some with a definite meaning, e.g., an estimate of the message length, and then thresholded to reach a decision about the presence of hidden message. It is certainly possible to add such distinguishing statistics to blind steganalyzers and further improve their performance. Because distinguishing statistics are designed to be sensitive to embedding changes of certain kind, they also provide guiding principles for constructing good features. This was the case with features designed for DCT coefficients of JPEG files [9]. The idea of calibration, which was originally invented for targeted attacks against F5 [10] and OutGuess [10], was adopted for construction of calibrated features that are sensitive to embedding modifications but exhibit less variation from image to image. Based on the results quoted in [9,17] and the results shown in Section 3, classifiers based on these features currently achieve the most reliable and accurate performance for blind steganalysis of JPEG images. This is why they were chosen for this study.

The goal of this paper is to construct a classifier for JPEG images capable of not only distinguishing cover images from stego images but also assigning stego images to known JPEG steganographic techniques. Such a tool is essential for steganalysis in the wide sense (forensic steganalysis) whose main goal is to recover the hidden data. Obviously, the first step is to identify the stego algorithm used to embed the data. In this paper, we use the calibrated DCT based features [9] calculated directly in the DCT domain to construct a Support Vector Machine (SVM) classifier. We focus on steganalysis of JPEG images because the JPEG format is by far the most common image format in use today. As our goal is to investigate the fundamental issues associated with building such classifier rather than constructing a ready-to-use application, we constrained ourselves to a database of test images with known processing history and origin. This gives us the possibility to better understand the influence of processing, analyze the outliers, and identify the limitations of the proposed approach. Also, in this study we chose to ignore the difficult issue of double compression by presenting the cover images compressed with the same quality factor as the one used for the stego images.

In the next section, first we briefly discuss previous art in blind steganalysis and the DCT-based features. In Section 3, we give the implementation details of SVMs used in this paper, we discuss various issues associated with training and testing procedures, and describe the image database. Section 4 starts with building two-class SVMs for individual steganographic techniques. We also include a comparison of the performance of DCT features with wavelet-based features [8] on the most popular JPEG stego programs. The section continues with an attempt to visualize the stego programs in the feature space transformed using the Principal Component Transformation. Then, we give experimental results obtained from a universal steganalyzer designed to distinguish between two classes of cover and stego images. Finally, at the end of Section 4 we present and analyze the results of experiments with the multi-class steganalyzer. The paper is concluded in Section 5.

2 Blind JPEG Steganalysis

The idea to use a trained classifier to detect data hiding was first introduced in a paper by Avcibas et al. [3]. In this paper, image quality metrics were proposed as features and the method was tested on several robust watermarking algorithms as well as LSB embedding. Avcibas et al. [4] later proposed a different set of features based on binary similarity measures between the LSB plane and the second LSB plane capitalizing on the fact that most steganographic schemes use the LSB of image elements as the information-carrying entity. Farid [8] constructed the features from higher-order moments of distribution of wavelet coefficients and their linear prediction errors from several high-frequency sub-bands. The same authors also showed that SVMs generally provide better performance as classifiers compared to linear classifiers. Other authors have investigated the problem of blind steganalysis using trained classifiers [23,11].

The steganalyzer described in this paper is based on features obtained from the DCT coefficients as described in [9]. Calculating the features directly in the JPEG domain provides certain attractive features. First, we expect the biggest sensitivity for features calculated in a domain in which the embedding changes are lumped — the DCT domain. Second, targeted analysis showed us the benefit of calibration, which is the process of estimating the macroscopic properties of the cover image from a slightly geometrically deformed decompressed stego image recompressed with the same quantization matrix.

The DCT features are constructed from 23 vector functionals f of three types — 17 first order functionals, 4 second order functionals, and 2 blockiness functionals. The first order functionals are histograms of 5 lowest-frequency AC DCT modes, the global DCT histogram, and 11 dual histograms (distribution of a certain value d among all 64 DCT modes for $d = \{-5, \ldots, 5\}$). The higher order functionals capture the inter-block dependency of DCT coefficients. They include the variation of coefficients (sum of absolute values of differences of DCT coefficients from neighboring blocks) and 3 quantities derived from co-occurrence matrices. The two blockiness functionals are the sum of discontinuities along 8×8 block boundaries and they are also the only functionals calculated in the spatial domain.

The values of the functionals $f_1, f_2, \ldots, f_{23}$ for the cover image are estimated from a slightly geometrically deformed (e.g., cropped by a few pixels) stego image recompressed using the same quantization table. Denoting the estimated functionals as $\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_{23}$, the final features are calculated as the L1 norm $\|f_i - \hat{f}_i\|$ between the functional f calculated from the stego image and the same functional calculated from the cropped and recompressed stego image. The logic behind this choice for features is the following. The cropping and recompression produce a "calibrated" image with most macroscopic features similar to the original cover image. This is because the cropped stego image is perceptually similar to the cover image and thus its DCT coefficients should have approximately the same statistical properties as the cover image. The cropping is important because the 8×8 grid of recompression "does not see" the previous JPEG compression and thus the obtained DCT coefficients are not as influenced by previous quantization (and embedding) in the DCT domain. One can think of the cropped / recompressed image as an approximation to the cover image or as a side-information.

3 Constructing Classifiers

3.1 2-class Support Vector Machines

Support Vector Machines are the tool of choice for steganography classifiers [8,17]. However, in most papers dealing with steganalysis, the authors rarely provide implementation details. We strongly believe that providing the details is necessary to enable fair independent verification of the reported results by peers. Thus, in this section, we describe all important elements of our realization of classifiers using SVMs.

Despite the advantages and simplicity of linear Support Vector Machines (SVM), in most applications, they are not sufficient, since we usually deal with noisy data in non linearly separable regions. SVMs with nonlinear kernels and the penalty parameter C can deal both with nonlinearity and outliers. The price of this extension is that before the training can start, we have to determine the penalty parameter C and the kernel and its parameters. There exist many different kernels that can even be combined together. In our preliminary experiments, we tried the linear, Gaussian, polynomial, and exponential kernels. As the Gaussian kernel ($\exp(-\gamma\|x - y\|)$) gave us the best overall results, we used it in all experiments described in this paper. The Gaussian kernel has one parameter γ controlling its width. The extended SVM capable of dealing with outliers has an additional penalty parameter C. These parameters affect the overall performance of the classifier and are highly data/problem dependent. Following the guide [12], the parameters were determined through a search on a multiplicative grid $(\gamma, C) \in \left\{(2^i, 2^j) | i \in \{-5, \ldots, 3\},\ j \in \{-2, 9\}\right\}$ with 5-fold cross-validation.

This means that for each pair (γ, C) the training set was divided into 5 subsets. Four of them were used for training and the remaining fifth subset was used to calculate the validation error. This was repeated five times for each subset. The validation errors from each subset were averaged, to obtain an estimate of the performance on unknown data. The final values of the parameters (C, γ) were determined by the least average validation error. After determining the parameters, we used the whole training set to train the SVM. We note that we have implemented the SVM ourselves and did not use any publicly available library.

Data preprocessing has a major influence on the performance of the SVM. We tested two different preprocessings — one consisting only of scaling and the second one of Principle Component Transformation (PCT) and subsequent scaling. Our experiments were inconclusive as to which preprocessing was better. Although the error on the training set was usually lower when the PCT was used, the error on the testing set was higher. Therefore, we chose scaling as the only preprocessing step in all experiments described in this paper. As shown in Section 4.2, the PCT is useful for visualizing the features.

By scaling, we understand that all elements of the feature vector were linearly scaled to the interval $[-1, +1]$. The scaling coefficients are always derived from the training set. When the n-fold cross-validation is employed, coefficients are computed on $n - 1$ subsets used for training to estimate the validation error of the remaining subset.

3.2 Multi-class Support Vector Machines

Support vector machines are naturally able to classify only 2 classes. There exist various extensions to enable the SVMs to handle more then two classes. They can be roughly divided into two groups $-$"all-together" methods and methods based on binary classifiers. A good survey with comparisons is the paper by Hsu [13] where the authors conclude that methods based on binary classifiers are better for practical applications. We tested the "Max Wins" and Directed Acyclic Graph (DAG) SVMs [19]. Both methods employ $\frac{n(n-1)}{2}$ binary classifiers for every pair of classes (n is the number of classes into which we wish to classify). Since both approaches had very similar performance in our tests, we only present the results from the "Max Wins" classifier.

In the Max Wins method, the sample that we want to classify is presented to all classifers and the histogram of their answers is created. The class corresponding to the highest peak is selected as the target class.

3.3 Database of Images

For our experiments, we created a database containing more than 35000 images obtained from 3436 different source images taken by various digital cameras all originally stored in the raw or lossless TIFF format (Nikon D100, Canon G2, Olympus Camedia 765, Kodak DC 290, Canon PowerShot S40, and images from Nikon D100 scaled by a factor of 2.9 and 3.76). For each image, we embedded

		OutGuess	F5	MB1	MB2
Machine	cover	100% /50% /25%	100% / 50% / 25%	100% / 50% / 25%	30%
cover×F5	2700	–	900 / 900 / 900	–	–
cover×MB1	2700	–	–	900 / 900 / 900	–
cover×MB2	2700	–	–	–	2700
cover×OutGuess	2700	900 / 900 / 900	–	–	–
cover×stego	2700	225 / 225 / 225	225 / 225 / 225	225 / 225 / 225	675
MB1×F5	–	–	900 / 900 / 900	900 / 900 / 900	–
MB1×MB2	–	–	–	900 / 900 / 900	2700
MB1×OutGuess	–	900 / 900 / 900	–	900 / 900 / 900	–
MB2×F5	–	–	900 / 900 / 900	–	2700
MB2×OutGuess	–	900 / 900 / 900	–	–	2700
OutGuess×F5	–	900 / 900 / 900	900 / 900 / 900	–	–

Table 1. Training set for the 2-class SVMs. The leftmost column denotes the particular SVM, the remaining columns contain the number of randomly chosen images in the training set.

a random binary stream of different lengths using five different algorithms — OutGuess ver 0.2 [20], F5 [24], MB1, MB2 [21], and JP Hide&Seek [1]. For F5, MB1, and OutGuess we created three different versions of each image with different lengths of the embedded message — 100%, 50%, 25% of the maximal capacity for a given image and embedding algorithm. For MB2, we embedded only one message of length equivalent to 30% of the capacity of MB1 to minimize the cases when the deblocking algorithm fails. For JP Hide&Seek, in compliance with the directions provided by its author we inserted messages with length equal to 10% of the image size.

If we summarize our database, each raw image is present in 12 different forms — MB1 100%, MB1 50%, MB1 25%, MB2 30%, F5 100%, F5 50%, F5 25%, OutGuess 100%, OutGuess 50%, OutGuess 25%, JP Hide, and cover JPEG. All cover and stego images used the same quality factor of 75.

In the beginning of our experiments, we divided the images into two disjoint sets — the training and testing sets. The training subset contained 2900 images (referring to the unique source images). From this set we have randomly chosen the training sets (Table 1) used for determining the parameters of each SVM and for its training as described in Section 3.1. The testing subset consisted of the remaining 534 source images. By dividing the source images into two disjoint sets, we made sure that during training no images from the testing set were used in any of their forms.

In our experiments, we built various SVMs all of which were trained on randomly chosen images from the training set. Thus, it could happen that one image was present in the training set in several different forms (the same source image with different message lengths, embedded with different stego-algorithm, or of different sizes).

Table 1 summarizes the number of examples in the training sets used in our experiments.

4 Experimental Results

In this section, we present experimental results from our classifiers. Unless stated otherwise, all results were derived on samples from the testing set that were not used in any form during training.

4.1 Two-class SVMs

We started our experiments by first constructing a set of two-class SVMs for distinguishing cover JPEG images from stego images embedded with a specific steganographic software. We construct such classifiers for both the DCT features and wavelet features [8] to obtain some performance comparison. The Matlab program for calculating the wavelet features was obtained from the authors' web site (http://www.cs.dartmouth.edu/˜farid/research/steganography.html). Since the authors of [8] do not describe the configuration of their SVM, we determined the parameters ourselves as described in Section 3.1. The features were compared on four different tasks — distinguishing between cover images and one specific steganographic algorithm (F5, MB1, and OutGuess 0.2 embedded with 100%, 50%, and 25% messages, and MB2 embedded with 30% message of the MB1 capacity).

Although the wavelet features were originally proposed to be calculated from the luminance component only, it has later been shown that they benefit from considering the chrominance channels as well, which is especially true for detection of the F5 algorithm [24,18]. As we are interested in performance comparison also for grayscale images, which in general appears to be the worst case for steganalysis both in the DCT and spatial domains, in our experiments we calculated both feature sets only from the luminance part of JPEG images. Thus, there were total of 23 DCT features and 72 wavelet features. The composition of training examples for each particular SVM is given in Table 1.

The SVM parameters together with errors on the training and testing sets are shown in Table 2. We conclude that for grayscale JPEG images, the DCT features perform better than wavelet features. This is not surprising as the DCT features were built specifically for JPEG files while the wavelet features are more universal and can be used for steganographic methods that embed in any domain. The results are also compatible with the previously published evaluation of blind steganalyzers in [17] and the work [9]. Another conclusion we can draw is that the least detectable stego program among the tested algorithms is the MB2 algorithm (Model Based Steganography with deblocking). At 30% of capacity of MB1, the MB2 algorithm is detected in 98.2% cases with 3.8% false alarms. Comparing this with the results for MB2 reported for the same features with a linear classifier in [9], we see that the SVM classifier has markedly better performance. Another advantage of DCT features is that the training is faster

Classifier	γ	C	Misclassification on train. set	test. set	False positives on train. set	test. set
DCT — cover×F5	0.5	64	1.29%	1.8%	0.59%	1.8%
DCT — cover×MB1	0.25	128	1.26%	1.6%	1.07%	1.4%
DCT — cover×MB2	0.5	64	1%	1.8%	1.3%	3.8%
DCT — cover×OutGuess	0.25	32	0.07%	1%	0.1%	0.2%
Wavelet — cover×F5	0.03125	64	12%	21.9%	17.7%	28.1%
Wavelet — cover×MB1	0.03125	64	9.5%	20%	12%	23.75%
Wavelet — cover×MB2	0.03125	64	11.2%	20.7%%	23.7%	37.3%
Wavelet — cover×OutGuess	0.0625	256	0.25%	6.9%	0.75%	15.6%

Table 2. Error on training and testing sets for wavelet and DCT features (default zero threshold for all machines). Each set includes an equal number of examples of cover and stego images. Due to the high computational load associated with training on the wavelet features, in this submission we only provide results for a smaller database cosisting of 800 training and 320 testing images. Results for the full database will appear in the final version.

because the features have lower dimension and better separability compared to the wavelet features.

4.2 Principal Component Analysis

We used the PCT to analyze the effective dimensionality of the DCT-based feature space. The features were computed from only 1073 images embedded with the maximal message length using OutGuess 0.2, F5, and MB1. Since there were only three eigenvalues with magnitude above 0.02, we could plot the features in a three-dimensional space and nicely visualize the clouds of feature points corresponding to different steganographic algorithms (Figure). This representation enables visual inspection and interpretation, which could be a useful forensic tool by itself.

4.3 Universal Machine

By universal, we mean a classifier able to classify images into two classes — cover and stego images. The training set for this machine is described in Table 1 in the row "cover×stego". The SVM was trained with parameters $\gamma = 0.25$, and $C = 512$ determined using the multiplicative grid search. The error on the training set was 1.44% (1.96% misclassification, 1.09% false positives). Table 3 covers the performance of the universal machine on all images from the testing set in our database. The number of images varies between algorithms because some algorithms fail on some images (e.g., blue sky images). The leftmost column contains the class (algorithm and relative size of the message) to which the examples belong. The remaining columns show the number and percentage of images recognized by the universal machine as cover and stego images for two choices of the threshold.

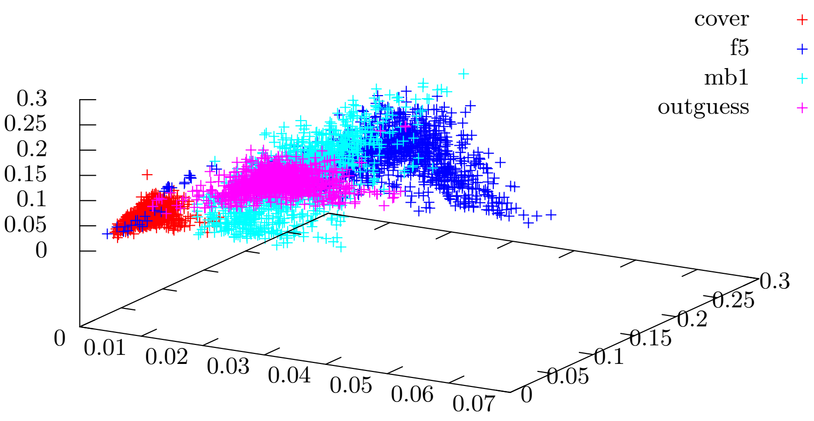

Fig. 1. First 3 coordinates with the highest variance after applying PCT to DCT features.

We see that the universal machine was able to generalize and detect images embedded with JP Hide as stego images even though it was not trained on such images. The second and the third columns of Table 3 show the performance of the universal machine with the default threshold 0. We can see very good detection (above 98%) for all methods for images with messages whose length is greater than 50% of the image capacity with the overall rate of false alarms at 3.5%. After adjusting the threshold from the default value 0 to 0.994593 in order to obtain less than 1% of overall false alarms (fourth and fifth columns), the overall false positive rate lowers to 1.3% and the detection accuracy of stego-images with message length greater than 50% remains at a very good level but the performance on stego images with small messages is worsened. This is especially true for F5, which we attribute to the effects of matrix embedding that improves the embedding efficiency by a large margin for short messages. We also point out the high detection of OutGuess with 25% messages (98.1%). As the capacity of OutGuess is already small, we conclude that OutGuess 0.2 is highly detectable and quite unsafe for steganography.

Some authors report the performance of classifiers using detection accuracy ρ (the area between the ROC curve and the diagonal normalized to 1 for perfect detection) and the false positive rate at 50% detection of stego images. For our universal machine we obtained $\rho = 0.98$ and 0% false positives at 50% stego detection on a database of 480 cover images and the same number of stego images embedded as in Table 3.

	Classified as		Classified as	
Embedding algorithm	cover	stego	cover	stego
F5 100%	2 (0.37%)	532 (99.6%)	5 (0.93%)	529 (99%)
MB1 100%	3 (0.56%)	530 (99.4%)	3 (0.56%)	530 (99.4%)
OutGuess 100%	3 (0.56%)	531 (99.4%)	5 (0.93%)	529 (99%)
F5 50%	2 (0.37%)	532 (99.6%	7 (1.31%)	527 (98.6%)
MB1 50%	2 (0.37%)	531 (99.6%)	4 (0.75%)	529 (99.2%)
OutGuess 50%	3 (0.56%)	531 (99.4%)	4 (0.74%)	530 (99.25)
MB2 30%	27 (5.06%)	506 (94.9%)	73 (13.7%)	460 (86.3%)
F5 25%	54 (10.1%)	480 (89.8%)	149 (27.9%)	385 (72%)
MB1 25%	38 (7.12%)	495 (92.8%)	90 (16.8%)	443 (83.1%)
OutGuess 25%	5 (0.93%)	529 (99%)	10 (1.87%)	524 (98.1%)
JP Hide	6 (1.12%)	528 (98.8%)	10 (1.87%)	524 (98.1%)
cover	515 (96.4%)	19 (3.5%)	527 (98.8%)	7 (1.3%)

Table 3. Performance of the universal machine with the default threshold 0 (second and third columns) and with the adjusted threshold to obtain less than 1% of false alarms (fourth and fifth columns).

SVM	cover×F5	cover×MB1	cover×MB2	cover×OutGuess	MB1×F5
C	64	128	64	32	8
γ	0.5	0.2564	0.5	0.25	1
SVM	MB1×MB2	MB1×OutGuess	MB2×F5	MB2×OutGuess	OutGuess×F5
C	16	64	64	32	64
γ	0.5	0.25	0.5	0.25	0.125

Table 4. Parameters (C, γ) of SVMs with the Gaussian kernel used in the "Max Wins" multi-classifier.

4.4 Max Wins Multiclassifier

One of the main goals of this paper is to build a classification machine able to detect not only the presence of secret messages in images, but also recognize steganographic algorithms. For this task, we chose the "Max Wins" algorithm, briefly described in Section 3.2. It consisted of 10 two-class SVMs (all SVMs from Table 1 except for cover×stego) classifying between every pair out of five classes (cover, F5, MB1, MB2, and OutGuess 0.2 classes). The parameters of the binary SVMs are summarized in Table 4. The confusion matrix in Table 5 is used to evaluate the performance.

Similar to the universal machine, the performance significantly improves as the size of messages exceeds 50% of the image capacity. In this case, the "Max Wins" machine is able to correctly identify the algorithm used for embedding with a very good accuracy (over 97%). Comparing its ability to separate cover and stego images with the universal machine, we see that the "Max Wins" has a better performance on images with a low embedding rate. The difference in performance is especially noticeable on images 25% embedded with F5 when the

	Classified as				
Embedding algorithm	cover	F5	MB1	MB2	OutGuess
F5 100%	2 (0.37%)	531 (99.4%)	1 (0.18%)	0 (0%)	0 (0%)
MB1 100%	3 (0.56%)	0 (0%)	526 (98.6%)	1 (0.19%)	3 (0.56%)
OutGuess 100%	2 (0.37%)	0 (0%)	0 (0%)	0 (0%)	532 (99.6%)
F5 50%	4 (0.74%)	522 (97.7%)	7 (1.3%)	1 (0.18)	0 (0%)
MB1 50%	3 (0.56%)	7 (1.3%)	506 (94.9%)	12 (2.26%)	5 (0.93%)
OutGuess 50%	3 (0.56%)	1 (0.18%)	3 (0.56%)	0 (0%)	527 (98.6%)
MB2 30%	8 (1.5%)	14 (2.6%)	17 (3.2%)	492 (92.3%)	2 (0.38%)
F5 25%	17 (3.2%)	463 (86.7%)	27 (5.1%)	26 (5.9%)	1 (0.19%)
MB1 25%	16 (3%)	26 (4.9%)	411 (77.1)	75 (14.1%)	5 (0.93%)
OutGuess 25%	4 (0.75%)	7 (1.31%)	16 (3%)	23 (4.3%)	484 (90.6%)
JP Hide	9 (1.7%)	334 (62.5%)	158 (29.6%)	27 (5.1%)	6 (1.1%)
cover	510 (95.5%)	5 (0.93%)	4 (0.75%)	15 (2.8%)	0 (0%)

Table 5. Confusion matrix for the "Max Wins" multi-classifier with default thresholds. Images are from the testing set only. The left most column contains the algorithm and the embedded message length. The remaining columns show the results of classification.

	cover×F5	cover×MB1	cover×MB2	cover×OutGuess
Threshold	0.748756	1.26615	0.653331	-0.699928
False positives	0.8%	0.8%	0.8%	0.8%
Detection rate	95.6%	89.7%	95.4%	99.3%

Table 6. New thresholds of two-class SVMs used in "Max Wins" classifier.

universal machine has a detection rate of 89.9% vs. 96.8% for the "Max Wins" multi-classifier. The universal machine has a lower overall false positive rate of 3.5% vs. 4.5% for the "Max Wins" classifier. The better performance of the "Max Wins" classifier on images with a lower embedding rate is probably due to the higher total number of examples used during training.

Since a high false positive rate is not desirable, we adjusted the decision threshold for each SVM detecting cover images. For this purpose, we created special training sets intended only for the purpose of adjusting the thresholds. These sets contained the same number of cover and stego images for each embedding algorithm. For example, to adjust the threshold for cover×F5, we prepared a set consisting of 480 cover images, 160 images with 100% message, 160 images with 50% messages, and 160 images with 25% messages. Then we adjusted the threshold to obtain a false positive rate less than 1%.

Table 6 shows the false positive rate and the detection rate for a given machine and threshold. The thresholds were chosen as the smallest values producing the false positive rate below 1%. The thresholds of all remaining SVMs used in the "Max Wins" classifier were set to the default value of 0.

Table 7 shows the performance of the multi-classifier with thresholds adjusted to lower the false positives. We see that the false positive rate was de-

	Classified as				
Embedding algorithm	cover	F5	MB1	MB2	OutGuess
F5 100%	4 (0.75%)	529 (99.1)	1 (0.19%)	0 (0%)	0 (0%)
MB1 100%	5 (0.94%)	0 (0%)	524 (98.3%)	1 (0.19%)	3 (0.56%)
OutGuess 100%	2 (0.37%)	0 (0%)	0 (0%)	0 (0%)	532 (99.63%)
F5 50%	4 (0.75)	521 (97.6%)	7 (1.31%)	1 (0.19%)	1 (0.19%)
MB1 50%	3 (0.56%)	7 (1.31%)	506 (94.9%)	12 (2.6%)	5 (0.94%)
OutGuess 50%	3 (0.56%)	1 (0.19%)	3 (0.56%)	0 (0%)	527 (98.7%)
MB2 30%	29 (5.4%)	11 (2.1%)	14 (2.6%)	477 (89.5%)	2 (0.38%)
F5 25%	64 (12%)	426 (79.8%)	20 (3.8%)	22 (4.1%)	2 (0.37%)
MB1 25%	85 (15.6%)	20 (3.8%)	358 (67.2%)	65 (12.2%)	5 (90,8%)
OutGuess 25%	5 (0.94%)	6 (1.1%)	16 (3%)	22 (4.1%)	485 (90.8%)
JP Hide	10 (1.9%)	332 (62.2%)	159 (29.8%)	27 (5.1%)	6 (1.1%)
cover	525 (98.3%)	1 (0.19%)	3 (0.56%)	5 (0.94%)	0 (0%)

Table 7. Confusion matrix for the “Max Wins” classifier with adjusted thresholds.

creased to 1.69%, while the machine kept its good classification performance on images with larger messages. In comparison with the universal classifier, the false positive rate is now similar (universal — 1.3%×“Max Wins” — 1.7%) but the detection performance of the “Max Wins” classifier images with short messages still outperforms the universal classifier. We note that the training of the “Max Wins” classifier is significantly more time consuming, since it is necessary to train $\frac{n(n-1)}{2}$ more SVMs, while the size of the training set remains the same.

Note that images embedded with the JP Hide algorithm are again correctly identified as stego images and the classifier identifies them mostly as F5 (62%) and MB1 (30%). This suggests a potential similarity between the embedding mechanisms. Obviously, it is possible that different stego programs use the same or very similar embedding mechanisms in which case, their separation by a blind classifier may become impossible. In our future work, we intend to further expand the proposed approach to allow the multiclassifier to recognize a new class (a new embedding mechanism).

Next, we examined the images that were misclassified by the multi-classifier with the intention to learn more about its performance. In particular, we inspected all misclassified cover images and stego images containing a message larger than 50% of the image capacity. Most of the misclassified images were taken by Nikon D100 camera or they were scaled versions of an image taken by this camera. This is surprising, because images taken by Nikon D100 were large (3008×2000) to provide sufficient statistics. We noticed that some of these images were very noisy (images taken at night using 30 second exposures), while others did not give us any visual clues as to why they were misclassified. We note, though, that the capacity of these images was usually below the average capacity of images with the same size.

As the calibration used in calculating the DCT features subjects an image to compression twice, the calibrated image has a lower noise content than the original JPEG image. Thus, we hypothesize that very noisy images might produce outliers. To test this hypothesis, we had blurred the misclassified Nikon D100 cover images (false positives) using a blurring filter with Gaussian kernel with diameter 1 and reclassified them. After this slight blurring, all of them were properly classified as cover images thus confirming our hypothesis.

Most of the misclassified images from the remaining cameras (Canon G2, Olympus Camedia 765, Kodak DC 290, and Canon PowerShot S40) were "flat" images, such as blue sky shots or completely dark images taken with a covered lens (these images were test images used by other members of our research group). The flat images do not provide sufficient statistics for steganalysis. As these images have a very low capacity (in tens of bytes) for most stego schemes, they are not suitable for steganography anyway.

5 Conclusions

In this paper, we build a multi-class steganalytic classifier capable of not only detecting stego images but also classifying them to appropriate stego algorithms. The classifier is a support vector machine with a Gaussian kernel trained on calibrated features calculated directly in the DCT domain [9]. We have trained the classifier on over 35000 images obtained by embedding messages of different sizes using different stego programs in almost 3436 unique source images from several digital cameras.

First, two class machines are built that distinguish between all pairs of image classes (cover, F5, MB1, MB2, OutGuess 0.2). These machines are used to compare the performance with the previously proposed classifier that uses wavelet-based features [8]. The two-class machines are then used to build a multi-class machine using the "Max Wins" approach . The performance is evaluated via confusion matrices. We conclude that it is, indeed, possible to reliably classify stego images to their appropriate stego methods, at least for sufficiently long messages. The multi-class machine is also capable to generalize to previously unseen stego methods (JP Hide&Seek). By analyzing the misclassified images, we conclude that images with a high level of noise are more likely to be misclassified, indicating a possible limitation of the calibration process for calculating features. Including non-calibrated version of the DCT features might help resolve this issue.

In our future work, we plan to extend this multi-classifier to other JPEG steganographic techniques available on the Internet and extend its scope to deal with double compressed images. Also, it is desirable that the classifier can automatically recognize a new embedding algorithm and automatically create a new class of stego images. This is, however, not an easy task to do with support vector machines. Further investigation of this topic is part of our future effort, as well.

6 Acknowledgements

The work on this paper was supported by Air Force Research Laboratory, Air Force Material Command, USAF, under the research grants number FA8750–04–1–0112 and F30602–02–2–0093. 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. Special thanks belong to Walker Land for help with multiclass support vector machines and to Jozef Sofka for providing us with his collection of images.

References

1. JP Hide&Seek. http://linux01.gwdg.de/˜alatham/stego.html.
2. R.J. Anderson and F.A.P. Petitcolas. On the limits of steganography. *IEEE Journal of Selected Areas in Communications, Special Issue on Copyright and Privacy Protection*, 16(4):474–481, 1998.
3. I. Avcibas, N. Memon, and B. Sankur. Steganalysis using image quality metrics. In *Proceeedings of SPIE Electronic Imaging, Security and Watermarking of Multimedia Contents*, volume 4314, pages 523–531, San Jose, CA, 2001.
4. I. Avcibas, B. Sankur, and N. Memon. Image steganalysis with binary similarity measures. In *Proceedings of International Conference on Image Processing*, volume 3, pages 645–648, 2002.
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, 1998.
6. R. Chandramouli, M. Kharrazi, and N. Memon. Image steganography and steganalysis. In T. Kalker, I. Cox, and Yong Man Ro, editors, *International Workshop on Digital Watermarking*, volume 2939 of *Lecture Notes in Computer Science*, pages 25–49, 2002.
7. S. Dumitrescu, Wu Xiaolin, and Zhe Wang. Detection of LSB steganography via sample pair analysis. In F.A.P. Petitcolas, editor, *Information Hiding. 5th International Workshop*, volume 2578 of *Lecture Notes in Computer Science*, pages 355–374. Springer-Verlag, 2003.
8. 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. Springer-Verlag, 2003.
9. 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, 2005.
10. J. Fridrich, M. Goljan, D. Hogea, and D. Soukal. Quantitative steganalysis: Estimating secret message length. *ACM Multimedia Systems Journal. Special issue on Multimedia Security*, 9(3):288–302, 2003.

11. J.J. Harmsen and W.A. Pearlman. Steganalysis of additive noise modelable information hiding. In *Proceedings of SPIE Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents V*, pages 131–142, Santa Clara, CA, 2003.
12. C. Hsu, C. Chang, and C. Lin. *A practical guide to support vector classification*. Department of Computer Science and Information Engineering, National Taiwan University, Taiwan. http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf.
13. 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.
14. S. Katzenbeisser and F.A.P. Petitcolas. Security in steganographic systems. In *Proceedings of SPIE Electronic Imaging, Security and Watermarking of Multimedia Contents IV*, volume 4675, pages 50–56, San Jose, CA, 2002.
15. A. Ker. Resampling and the detection of LSB matching in colour bitmaps. In *Proceedings of SPIE Electronic Imaging, Security, Steganography and Watermarking of Multimedia Contents VII*, San Jose, CA, 2005. To appear.
16. A. Ker. Steganalysis of LSB matching in grayscale images. *IEEE Sig. Proc. Letters*, 2005. To appear.
17. M. Kharrazi, H. T. Sencar, and N. Memon. Benchmarking steganographic and steganalytic techniques. In *Proceedings of SPIE Electronic Imaging, Security, Steganography and Watermarking of Multimedia Contents VII*, San Jose, CA, 2005. To appear.
18. S. Lyu and H. Farid. Steganalysis using color wavelet statistics and one-class support vector machines. In *Proceedings of SPIE Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents*, pages 35–45, San Jose, CA, 2004.
19. J. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin DAGs for multiclass classification. In S.A. Solla, T.K. Leen, and K.-R. Mueller, editors, *Advances in Neural Information Processing Systems 12*, pages 547–553, 2000.
20. N. Provos. Defending against statistical steganalysis. In *10th USENIX Security Symposium*, Washington DC, 2001.
21. P. Sallee. Model based steganography. In Kalker, I.J. Cox, and Yong Man Ro, editors, *Digital Watermarking. 2nd International Workshop*, volume 2939 of *Lecture Notes in Computer Science*, pages 154–167. Springer-Verlag, 2004.
22. D. Soukal, J. Fridrich, and M. Goljan. Maximum likelihood estimation of secret message length embedded using PMK steganography in spatial domain. In *Proceedings of SPIE Electronic Imaging, Security, Steganography and Watermarking of Multimedia Contents VII*, San Jose, CA, 2005. To appear.
23. R. Tzschoppe, R. Bäuml, J.B. Huber, and A. Kaup. Steganographic system based on higher-order statistics. In *Proceedings of SPIE Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents V*, pages 156–166, Santa Clara, CA, 2003.
24. 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, 2001.
25. 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, 1998.