Multi-Class Classification Fusion using Boosting for Identifying Steganography Methods

Benjamin M. Rodriguez*[a] and Gilbert L. Peterson[a]
[a]Department of Electrical and Computer Engineering
Graduate School of Engineering and Management, Air Force Institute of Technology

ABSTRACT

There are over 250 image steganography methods available on the Internet. In digital image steganalysis an analyst has three goals, first determine if an embedded message exists, next determine the embedding method used to create the stego image and finally extract the hidden message. The objective of this paper lies on the second goal, that is, to identify the embedding technique used to create the steganography image. Several detection systems currently exist, so the identification problem becomes one of determining which detection system has correctly identified the embedding method. In this work, the individual detection systems are fused using boosting. Boosting is a powerful technique for combining an ensemble of base classifiers to produce a form of committee with improved performance over any of the single classifiers in the ensemble. The results in this paper show that boosting takes advantage of the individual strengths from each detection systems and classification performance is increased by 10%.

Keywords: Boosting, Multi-class Classification, Steganalysis, System Fusion, System Identification

1. INTRODUCTION

In digital image forensics, it is important to determine if an image contains a hidden file. The problem of digital image steganalysis has moved from simply determining if an image contains hidden information to extracting the hidden message. In order for a steganalyst to extract the hidden information from an image an intermediate step that identifies the method used to create the steganography image is required. With over 250 steganography methods available on the Internet, it is important to develop multi-class steganalysis systems capable of properly labeling the image as clean or containing steganography. If an image is identified as being a cover stego file the embedding method must be determined. This intermediate step of identifying the steganography method enables the steganalyst to then target the steganography method and extract the hidden information.

There are several multi-class detection systems available. Each system has certain advantages and disadvantages in comparison with the others. A steganalyst should use as many of these tools as possible when analyzing a set of images. A problem arises when an image is identified to contain hidden information. Each detection system used can potentially return different classification labels representing different embedding techniques. In the event each of the detection systems identifies a different steganography method, the analyst must then properly determine the correct method from the different set of identified steganography methods. The solution described in this paper fuses the results of each detection systems to get better detection accuracy and alleviate the steganalyst from having to make this assessment.

In this paper, a method for fusing multi-class detection systems is presented. The fusion system under consideration is boosting which is a powerful technique for combining an ensemble of base classifiers to produce a committee whose performance can increase over any of the single classifiers in the ensemble. In our algorithm, the classification results from each multi-class classifier are weighted to generate the final class label. This fusion system is applied to the steganography fingerprint domain, in which the classifier identifies the statistical patterns in an image, which distinguish one steganography algorithm from another. The embedding methods targeted are F5,[38] JP Hide and Seek,[18] JSteg,[36] Model-based,[31] OutGuess[26] and StegHide.[13]

To generate the steganography-fingerprinting tool, four detection systems are fused. Three of the systems are individual multi-class methods differing in the way features are generated.[9,20,29] In feature generation, any number of steganography methods may be undertaken as long as the images used are part of the training process. The fourth detection system is

StegDetect.[26] This method is capable of detecting F5, JP Hide and Seek, JSteg, and OutGuess without additional training. By combining the multiple classifiers together through boosting the testing results of the system shows improved classification accuracy in the multi-class domain over individual multi-class classifiers. Results show that through the novel addition of the classifier fusion step to the multi-class steganalysis system, the classification accuracy is improved by up to 10%.

The following section presents related work in the areas of steganography methods, feature generation methods, and steganalysis systems. Section 3 describes two systems; first is the multi-class steganalysis system that includes generated features, feature selection and a multi-class support vector machine (SVM), second is the boosting fusion system in which the four multi-class steganalysis systems are combined. Section 4 presents the results, including individual system results and results from the fusion system of all four methods. We end with a brief conclusion in Section 5.

2. RELATED WORK

This section discusses the JPEG image format, six JPEG image data embedding methods, two feature generation methods, and the detection systems. The embedding methods are briefly introduced to give the reader an insight of the characteristics when data is hidden within the JPEG images. The three feature generation methods used in this paper are described. The importance of describing the feature generation methods is to provide the reader with an understanding of how the multi-class steganalysis systems introduced in Section 3 can be changed based on various feature generation methods.

2.1 Embedding Methods

Testing is conducted against six embedding methods. The embedding methods range from simple embedding techniques that alter the AC coefficients (JP Hide and JSteg) in the JPEG compression system to more complicated embedding techniques that maintain natural characteristics in AC coefficients (F5, Model-based, OutGuess and StegHide). The embedding methods are described in detail from a detection point of view by the following references, Kharrazi, Sencar and Memon, [17] Pevny and Fridrich,[23] Provos and Honeyman,[27] and Rodriguez and Peterson.[28] The JPEG image encoder is shown in the simplified block diagram of Figure 1, the compression itself is performed in the following sequential steps: Preprocessing block (8 by 8 sub-image extraction), forward DCT block, quantization, separation of the coefficients and Huffman encoder (variable-length code assignment).

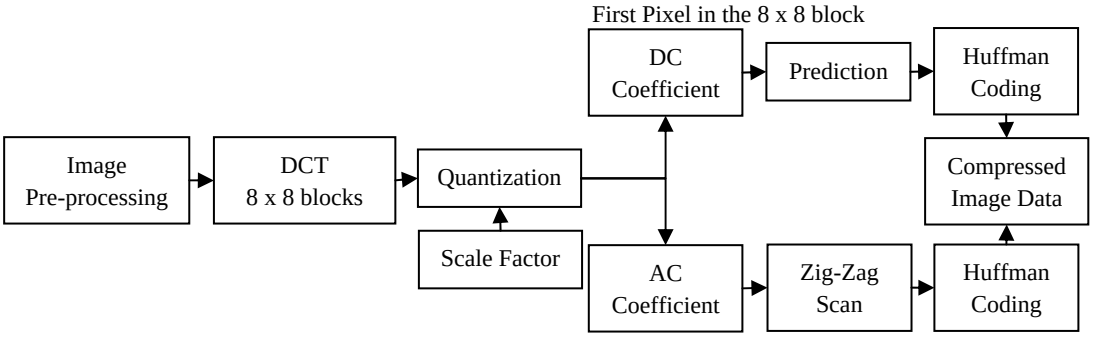

Fig. 1. Block Diagrams of Sequential JPEG Encoder

2.2 Feature Generation Methods

Several steganalysis feature generation methods have been developed.[10,19,33,39,40] However, in this section the three feature generation methods used in the fusion are discussed. The first method developed by Lyu and Farid[20,21] is a wavelet based method. The second method is a DCT based feature generation method in which the features are developed with the use of functions for both the input image and the predicted image.[23,24] In the third method DCT features are developed by means of calculating statistics from the DCT decomposition of the coefficient frequencies and directions for each of the 8 by 8 blocks.[29]

2.2.1 Wavelet Statistics

Lyu and Farid[20,21] proposed a higher-order statistical model constructed from multi-scale wavelet decomposition of an image. This approach relies on building higher-order statistical models for natural images and looking for deviations from these models. Lyu and Farid use the statistical regularities that are obtained from the decomposition of images using wavelets in order to differentiate clean images from stego images.

First, a mapping from the spatial domain to the wavelet transform domain is performed using a decomposition that splits the frequency space into multiple orientations and scales. The orientations are represented as vertical, V, horizontal, H, and diagonal, D, subbands at scale i applied independently to each color channel, $c \in \{r, g, b\}$, denoted as $V_i^c(x, y)$, $H_i^c(x, y)$, and $D_i^c(x; y)$. Second, given the decomposed image, the statistical model is composed of the mean μ, variance σ^2, skewness γ_1, and kurtosis γ_2, of the subband coefficients at each orientation (V,H,D), scale i and color channel c. This method captures higher-order statistical correlations using a second set of statistics collected based on the errors in a linear predictor of coefficient magnitude. The multi-scale decomposition contains scales $i = 1,\ldots,n$, the total number of basic coefficient statistics (features) is 36, and the total number of error statistics (features) is also 36, yielding a total number of 72 statistic (features). These statistics form the feature vectors used to discriminate between clean images and stego images.

2.2.2 DCT Based Feature Generation

Two types of features are calculated over an image for this method: first order features and second order features. The features in the DCT and spatial domains are calculated from a vector functional F applied to the stego JPEG image J_1. This functional could be the global DCT coefficient histogram, a co-occurrence matrix, spatial blockiness, or a number of other functions.[9] The stego image J_1 is decompressed to the spatial domain, cropped by four pixels in each direction and recompressed with the same quantization table used in decompressing J_1 to obtain J_2, as shown in Figure 2. The vector functional F is then applied to J_2. The final feature f is obtained from the difference in the vector functional between the original and modified image as follows:

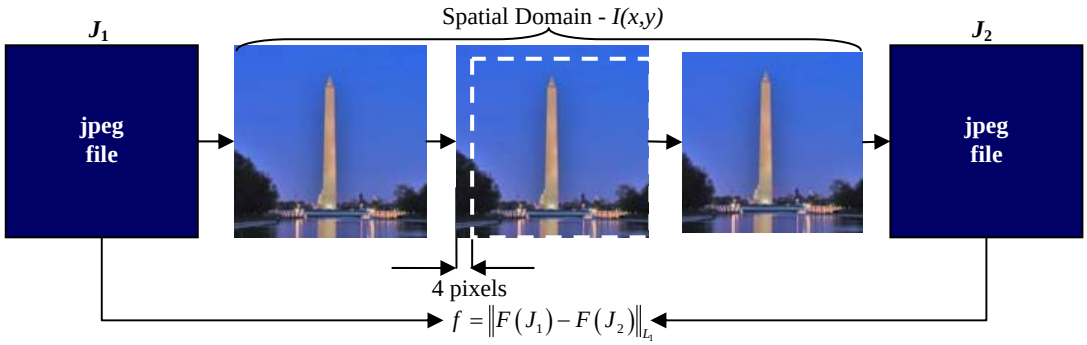

Fig. 2. Feature Generating Structure

This set of feature is referred to as extended DCT features resulting in 193 features.[24] Another 81 features are created using a calibrated Markov process in which the differences between absolute values of neighboring DCT coefficients are calculated for both J_1 and J_2 images. The calibrated Markov features take differences between DCT coefficients along four directions: horizontal, vertical, diagonal, and minor diagonal resulting in a total of 274 DCT features.[24]

2.2.3 Feature Generation: DCT Combined Directional and Frequency Band Distance Measure Features

Here we introduce a feature generation method that calculates statistics that exploit the 8 by 8 block DCT coefficients decomposition.[29] The general structure in Figure 4 shows the basic feature design of the method in this section. The shifted coefficients block in Figure 3 looks for the identification of blockiness from neighboring 8 by 8 blocks caused by embedding methods. The coefficients block uses a select number of DCT coefficients to generate the features while the predictors block calculates an estimate of the altered coefficients from an embedding method.

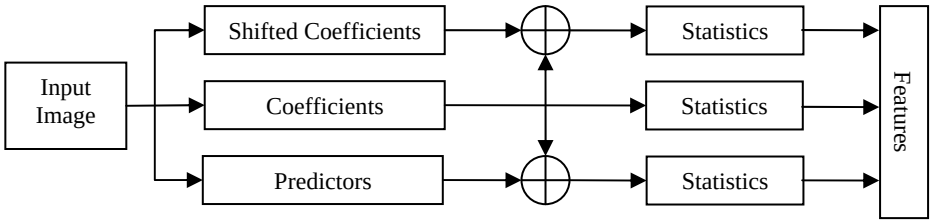

Fig. 3. General Feature Generation System

Both the coefficient and predictor feature generation methods separate the DCT coefficients into low, medium and high frequencies as well as vertical, diagonal and horizontal directions. This is referred to as DCT decomposition. The features are generated by calculating the higher order statistics, 1st, 2nd, 3rd, and 4th moments, 2nd, 3rd, and 4th central moments and entropy, over the sets of selected coefficients, $D(k)$, and the predictors, $P(k)$. $D(k)$ represents a vector containing the coefficients to be analyzed and $P(k)$ represents the predictors to be analyzed. The number of features generated is dependent on the DCT decomposition and the selected coefficients. This produces 234 features from the 72 shifted coefficients, 72 raw coefficients, 72 predictors and 18 histogramming features. The features provide information about the embedding method as variations exist in embedding modifications in the vertical, horizontal and diagonal directions and also in the low, medium and high frequency bands.

2.3 Existing Steganography Detection Systems

The two feature sets mentioned in Section 2.2 are used to identify a variety of embedding methods used for JPEG images. However, in certain applications a steganalyst may need a set of automated tools. For the forensics practitioner, several steganalysis tools exist:

- ILook Investigator © toolsets (http://www.ilook-forensics.org/)
- Inforenz Forager® (http://www.inforenz.com/software/index.html)
- SecureStego (Air Force Research Laboratory, Rome, NY)
- StegDetect[26]
- WetStone Stego Suite™ (http://www.wetstonetech.com)

Other systems developed for research purposes are StegSpy2.1,[6] an individual set of analyzing tools by Guillermito[12] and XStegSecret.[22] These tools currently assist the digital forensics examiner; however, there are improvements that can be made, specifically in the area of targeting specific embedding methods not available in these methods. In order to demonstrate the capability of these available tools, we include StegDetect as one of the methods in the fusion classifier. StegDetect is a method capable of detecting several different steganographic methods which have been used to embed hidden information in JPEG images. Of the six embedding methods used in this paper, StegDetect is able to detect the following: F5 (header analysis), JP Hide and Seek, JSteg and OutGuess.

Identification of embedding methods is essential in attempts to extract the hidden information. Knowing the embedding method dictates the extraction method that will be required. The variations in each of the embedding methods leave a fingerprint on the JPEG image. The problem is how to classify the different results from different steganalysis tools. In the next section two feature generation methods are described in which the features are used to train a multi-class classification system.

3. METHOD

In this section, two major portions of the overall fused detection system are described. The first is the multi-class classification system. Within this system are the following components: the feature generation, preprocessing, classification and labeling steps of the multi-class detection system shown in Figure 3 are described in detail. The second is the fusion technique, AdaBoost,[7,8] used to develop the complete system that identifies steganography embedding methods.

3.1 Multi-class Detection System

The multi-class detection system consists of five steps (refer to Figure 4):

1) Define a training data set in which the stego classes has been assigned. In this paper the data set consists of clean and stego images. The stego images are created using six embedding methods (F5, JP Hide and Seek, JSteg, Model-base, OutGuess and StegHide).
2) The second step is the generation of features from the available data set. This step allows any desired feature generation method to be used.[9,10,19,28,33,39,40] In Section 4 three feature generation methods are used to develop the three multi-class detection systems.[20,21,23,24,29]
3) The third step preprocesses features based on standardizing the individual training features. Standardizing the raw features (generated features) shifts the centroid of the data to the origin, and stretches or compressed the data range according to the unit variance. This is done by subtracting the mean and dividing by the standard deviation of each feature separately. The mean and standard deviation are calculated for each feature from all of the available instances. This is followed by the ranking and selection of a subset of features using Fisher's discriminant ratio to quantify the separability of individual features. FDR is a straightforward technique which measures the discrimination of sets of real numbers. Other methods of preprocessing are outlier removal, data normalization, feature selection using other criterions, and feature extraction using principal component analysis.[11]
4) The classification step is to train a classifier using a training data set. In this paper a subset of the features generated are used to train the classifier. The subset is selected using 5-fold cross validation.[30] In a multi-class classification system an ideal classifier would be to use a multi-class classifier.[1,5,11,16,35] In this paper a set of two-class SVM classifiers are used in which individual classes are trained against each other (one-vs.-one).
5) In this step, the class labels are assigned. The class labels are the individual steganographic method under investigation. A majority vote is used to assign the class labels.

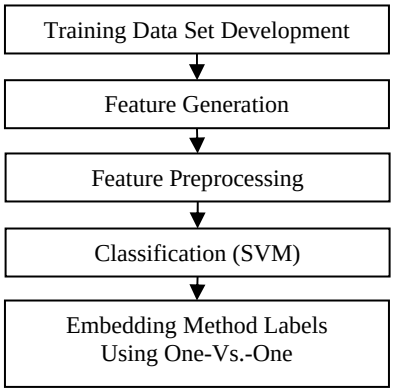

Fig. 4. Block Diagrams of Multi-Class Detection System

3.1.1 Support Vector Machine

The support vector machine (SVM) is a classification algorithm that provides state-of-the-art performance in a wide variety of application domains. The goal of the SVM is to produce a model that predicts target value of data instances in the testing set given only the attribute values. A SVM performs pattern recognition for two-class problems by determining the separating hyperplane that has maximum distance to the closest points of the training.[2,15,25,32,37] These closest points are called support vectors. In order to accomplish this, the SVM performs a nonlinear separation in the input space by using a nonlinear transformation $\phi(\cdot)$ that maps the data points $\mathbf{x}$ of the input space, $\mathbb{R}^n$, into a higher dimensional space, called kernel space $\mathbb{R}^p$ ($p > n$). The mapping $\phi(\cdot)$ is performed in the SVM classifier by a kernel function $K(\cdot,\cdot)$. Given ℓ samples $\left\{\left(\mathrm{x}_i, y_i\right)\right\}_{i=1}^{\ell}$, the decision function of the SVM is linear in the kernel space although not in the feature. In this paper the SVM method used is LibSVM[3] which uses a Sequential Minimal Optimization (SMO) for binary SVM with L_1-soft margin.[4]

3.1.2 One-Against-One

In several multi-class classification methods two-class classifiers are combined using one-against- one.[1,5,11,34,35] Learning architectures are used to combine several two-class classifiers in order to create a multi-class classifier. In this method training is accomplished by comparing one class against each of the other classes. This produces several classifiers in which a winner-take-all approach is used. The winner-take-all assigns the class label based on a majority vote. The goal is to train the multi-class rule based on the majority vote strategy. This is a reliable method assuming that the feature space is separable from one class to the other.

The one-against-one approach constructs $k(k\text{-}1)/2$ classifiers from two different classes for each one of the training data sets. In this paper seven classes (1 clean + 6 stego), $k = 7$, are used requiring 21 classifiers to be trained. In most classification systems a voting strategy is used. In binary classification as in the SVM the voting strategy votes are cast for all data points where the majority number of votes for a class wins, "max wins". This may lead to a situation where two classes have the same number of votes. The approach used in this paper is to select the class with the smallest index.[14]

3.2 Boosting

The previous steps described allow the multi-class classification structure to be outlined. This section focuses on the fusion of an ensemble of classification methods to improve the final steganalysis multi-class classification system. The fusion method described in this section is the AdaBoost method.[7]

Boosting is a powerful technique for combining an ensemble of base classifiers to produce a form of committee whose performance can be significantly increased over any of the single classifier. The most widely used form of boosting is AdaBoost, developed by Freund and Shapire.[7] Boosting provides good results even if the base classifiers, are weak learners, and have a performance that is only slightly better than random.[8]

The primary difference between boosting and bagging is that the base classifiers are trained in sequence, and each base classifier is trained using a weighted form of the data set in which the weighting coefficient associated with each data point depends on the performance of the previous classifiers. In particular, points that are misclassified by one of the base classifiers are given greater weight when used to train the next classifier in the next sequence. Once all the classifiers have been trained, their predictions are then combined through a weighted majority voting scheme. AdaBoost calls a given weak or base learning algorithm repeatedly in a series of rounds, $t_n = 1,\ldots,T$. The precise form of the AdaBoost algorithm is given below:

AdaBoost Algorithm[1]

1. Initialize the data weighting coefficients $\{w_n\}$, by setting $w_n^{(1)} = 1/N$ for $n = 1,\ldots,N$.
2. For $m = 1,\ldots,M$:

   (a) Fit a classifier $y_m(\mathbf{x})$ to the training data by minimizing the weighted error function

$$J_m = \sum_{n=1}^{N} w_n^{(m)} I\left(y_m(\mathbf{x}_n) \neq t_n\right) \tag{1}$$

   where $I\left(y_m(\mathbf{x}_n) \neq t_n\right)$ is the indicator function and equals 1 when $y_m(\mathbf{x}_n) \neq t_n$ and 0 otherwise.

   (b) Evaluate the quantities

$$\varepsilon_m = \frac{\sum_{n=1}^{N} w_n^{(m)} I\left(y_m(\mathbf{x}_n) \neq t_n\right)}{\sum_{n=1}^{N} w_n^{(m)}} \tag{2}$$

   and then use these to evaluate

$$\alpha_m = \ln\left\{\frac{1-\varepsilon_m}{\varepsilon_m}\right\} \tag{3}$$

(c) Update the data weighting coefficients

$$w_n^{(m+1)} = w_n^{(m)} e^{\alpha_m I\left(y_m(\mathbf{x}_n) \neq t_n\right)} \tag{4}$$

3. Make predictions using the final model, which is given by

$$Y_M(\mathbf{x}) = \text{sign}\left(\sum_{m=1}^{M} \alpha_m y_m(\mathbf{x})\right) \tag{5}$$

The first base classifier $y_1(\mathbf{x})$ is trained using weighting coefficients $w_n^{(1)}$ that are all equal, which corresponds to the usual procedure for training a single classifier. In Step 2(c), subsequent iterations in the weighting coefficients $w_n^{(m)}$ are increased for data points that are misclassified and decreased for data points that are correctly classified. Successive classifiers are forced to place greater emphasis on points that have been misclassified by previous classifiers, and data points that continue to be misclassified by successive classifiers receive even greater weight. The quantities ε_m represents the weighted measures of the error weights of each of the base classifiers on the data set. Therefore, in Step 2(b) the weighing coefficients α_m give greater weights to the more accurate classifiers when computing the overall output given by Step 3.[1]

Next we describe the general process used in the boosting averaging algorithm for steganalysis. In boosting, each of the multi-class classifiers including StegDetect is weighted based upon their individual performance.

General Process

1. Generate features
2. Select relevant features
3. Create classification Model-based on one-vs.-one training
4. Use majority vote strategy to populate the confusion matrix containing actual and predicted classified values for clean, F5, JP Hide and Seek, JSteg, Model Base, OutGuess and StegHide trained image sets.
5. Repeat Step 1 through 4 for each of the feature generation methods.20,21,23,24,29
6. Use the four systems to train the fusion model by assigning weights to each of the systems.

This results in a multi-class model that receives four inputs, three from each of the trained detection system and one from StegDetect, from an individual image to be classified.

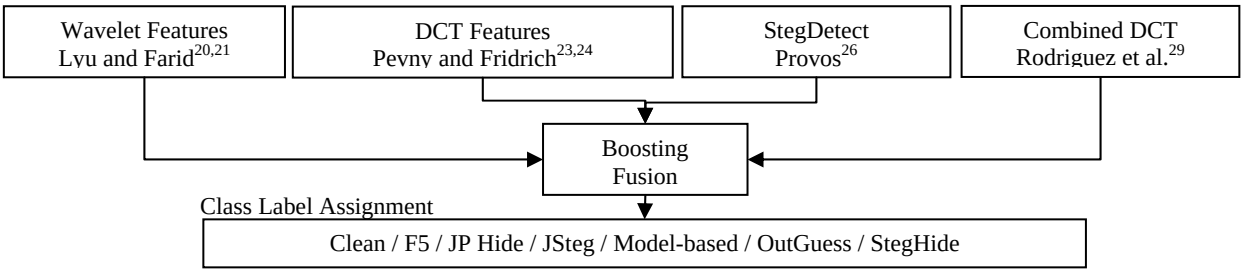

Fig. 5. Block Diagrams of AdaBoost Fusion Structure

4. RESULTS

Testing results are based on an image data set of 1000 512x512 RGB JPEG images consisting of clean and stego images. It should be noted that the presented results are not benchmarking any of the individual classification system against each other; rather the results are used to show how the fusion takes advantage of the individual system's performance. The results are only based on one image size and one JPEG compression in order to simplify the demonstration of the proposed concept.

The stego images are generated from the six embedding methods used. The clean images in this test set did not overlap the same images as the stego images along with any of the images from one stego type to another; e.g., none of the F5 images were the same as the JSteg images. The amount of hidden information embedded within each of the files is a

maximum of 4000 characters which is equivalent to one page of text. The percentage of altered coefficients varies based on the embedding method as follows:

- F5 has an average of 0.3% of the coefficients altered
- JP Hide and Seek has an average of 2.8% of the coefficients altered
- JSteg has an average of 6.7% of the coefficients altered
- Model-based has an average of 7.8% of the coefficients altered
- OutGuess has an average of 1% of the coefficients altered
- StegHide has an average of 1% of the coefficients altered

The training set of images consisted of 200 clean images and 100 stego images of each category. The image test set used in Tables 1 and 2 consists of 50 clean and 25 stego images in each category. The training and test sets are created using 5-fold cross validation.

Table 1. Test Set Classification Accuracy for Individual Detection Systems

Image Type	Wavelet Features	DCT Features	StegDetect	Combined DCT
Clean	46.2 ± 0.837	42.4 ± 1.673	41.8 ± 1.304	45.0 ± 0.707
F5	21.0 ± 0.707	23.2 ± 1.483	25.0 ± 0.0	18.4 ± 0.548
JP Hide	23.4 ± 0.548	22.2 ± 1.924	18.0 ± 1.0	20.8 ± 0.447
JSteg	22.8 ± 0.447	22.8 ± 1.643	20.8 ± 1.924	22.4 ± 0.548
Model-based	13.8 ± 2.588	16.6 ± 0.548	0.0 ± 0.0	17.8 ± 0.837
OutGuess	17.0 ± 1.414	14.8 ± 0.837	18.2 ± 2.168	18.0 ± 0.707
StegHide	17.4 ± 1.140	17.8 ± 0.447	0.0 ± 0.0	18.6 ± 0.548

The results for the individual systems are shown in Table 1 and the combined results are shown in Table 2. The columns of Table 1 show the classification accuracy of correctly identifying each of the embedding methods. By examining each column in Table 1, none of the individual multi-class classification algorithm outperforms the others. For example, StegDetect detects all of the F5 images but fails to detect Model-based and StegHide. The Wavelet Statistics[20,21] correctly labels most of the clean images but only detects and average of 13.8 out of 25 Model-based images. DCT Based Feature Generation[23,24] is strong at identifying F5, JP Hide and JSteg images. In addition, the DCT Combined Directional and Frequency Band Distance Measure Features identify the most OutGuess and Model-based images. The results in Table 2 show the improvement of combining all of the multi-class systems.

Table 2. AdaBoost Ensemble Fusion Confusion Matrix Results

		Predicted						
		C	F5	JP	JS	MB	OG	SH
Actual	C	**47.8 ± 0.83**	0.8 ± 0.45	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	1.0 ± 0.71	0.4 ± 0.55
	F5	0.0 ± 0.0	**25.0 ± 0.0**	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
	JP	0.0 ± 0.0	0.0 ± 0.0	**23.6 ± 0.55**	1.2 ± 0.45	0.2 ± 0.45	0.0 ± 0.0	0.0 ± 0.0
	JS	0.0 ± 0.0	0.0 ± 0.0	0.8 ± 0.45	**23.8 ± 0.45**	0.0 ± 0.0	0.4 ± 0.55	0.0 ± 0.0
	MB	4.6 ± 1.14	1.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	**18.4 ±0.89**	0.6 ± 0.89	0.4 ± 0.55
	OG	0.6 ± 0.55	0.4 ± 0.55	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	**20.8 ±0.84**	3.2 ± 0.84
	SH	1.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	2.0 ± 0.71	**22.0 ±0.71**

The average classification accuracy in percentage for the individual multi-class systems is; 80.8 ± 3.8 % for the wavelet features, 79.9 ± 4.2 % for the DCT features, 61.9 ± 3.1 % for StegDetect and 80.5 ± 2.1 % for the combined DCT features. The fused system has an overall classification accuracy of 90.7 ± 2.1 % which is an improvement of 10% above the three multi-class feature based systems and almost 30% greater than StegDetect. The presented fusion system takes advantage of the strengths of the individual multi-class systems by properly assigning weights based on classification accuracy.

5. CONCLUSION

The analysis in this paper was conducted using an ensemble of classifiers fused with AdaBoost to determine the performance of the fused system over the individual classifiers. The individual classifiers include DCT features,[23,24] wavelet features,[20,21] combined DCT features[29] and StegDetect.[26] Five separate tests were conducted to show comparisons between the individual system and the fused method. The first four tests show that each of the individual classification methods has certain advantages. The final test shows that the fused system takes advantage of the individual system to improve classification by an increase of 10%. This system can be expanded to incorporate other multi-class steganalysis systems such as ILook Investigator© toolsets, Inforenz Forager®, SecureStego (Air Force Research Laboratory, Rome, NY), and WetStone Stego Suite™.

ACKNOWLEDGEMENTS

The work on this paper was partially supported by the Digital Data Embedding Technologies group of the Air Force Research Laboratory, Information Directorate. The views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Air Force Research Laboratory, the Department of the Air Force, or the U.S. Government. We would additionally like to express our appreciation to June Rodriguez for the contribution of a multitude of digital images for analytical support.

REFERENCES

1. C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, NY, 1995.
2. C. Burgers, "A tutorial on support vector machines for pattern recognition," *In Data Mining and Knowledge Discovery*, 2(2), 121-167, (1998),
3. C.-C. Chang and C.-J. Lin, "LIBSVM: a library for support vector machines," http://www.csie.ntu.edu.tw/~cjlin/libsvm
4. N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, Cambridge University Press, 2000.
5. R. O. Duda, P. E. Hart and D. G. Stork, *Pattern Classification*, 2nd Edition, John Wiley & Sons, 2001.
6. B. Englehardt, StegSpy2.1, http://www.spy-hunter.com/stegspydownload.htm
7. Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *In European Conference on Computational Learning Theory*, 23-37 (1995).
8. Y. Freund and R. E. Schapire, "An introduction to boosting," *In Journal of Japanese Society for Artificial Intelligence*, 14(5), 771-780 (1999)
9. J. Fridrich, "Feature-Based Steganalysis for JPEG Images and its Implications for Future Design of Steganographic Schemes," *In LNCS 6th Information Hiding Workshop*, 3200, 67-81 (2004).
10. D. Fu, Y. Q. Shi, D. Zou and G. Xuan, "JPEG Steganalysis Using Empirical Transition Matrix in Block DCT Domain," *In IEEE 8th Workshop on Multimedia Signal Processing*, 310-313 (2006).
11. K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Second Edition, Academic Press, Inc. San Diego, CA, 1990.
12. Guillermito, Analyzing steganography softwares, http://www.guillermito2.net/stegano/
13. S. Hetzl, StegHide, http://steghide.sourceforge.net/

14. C.-W. Hsu, C.-C. Chang and C.-J. Lin, "A comparison on methods for multi-class support vector machines," *In IEEE Transactions on Neural Networks*, 13, 415-425 (2002).
15. C.-W. Hsu, C.-C. Chang and C.-J. Lin, "A practical guide to support vector classification," Department of Computer Science and Information Engineering National Taiwan, http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf, Retrieved March 2006.
16. A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: a review," *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1), 4-37 (2000).
17. M. Kharrazi, H. T. Sencar and N. Memon, "Performance study of common image steganography and steganalysis techniques," *Journal of Electronic Imaging*, 15(4), October-December (2006).
18. A. Latham, Steganography, http://linux01.gwdg.de/~alatham/stego.html, 1999, Retrieved July 20, 2005.
19. W.-N Lie and G.-S. Lin, "A feature-based classification technique for blind image steganalysis," *In IEEE Transactions on Multimedia*, 7(6), 1007-1020 (2005).
20. S. Lyu and H. Farid, "Detecting hidden messages using higher-order statistical models," *In International Conference on Image Processing (ICIP)*, Rochester, NY, 2002.
21. S. Lyu and H. Farid, "Steganalysis using color wavelet statistics and one-class support vector machines," *In SPIE Symposium on Electronic Imaging*, San Jose, CA, 2004.
22. A. Muñoz, XStegSecret, http://stegsecret.sourceforge.net/indexEnglish.htm, Retrieved December 2007.
23. T. Pevny and J. Fridrich, "Determining the stego algorithm for JPEG images," *In Special Issue of IEE Proceedings - Information Security*, 153(3), 75-139 (2006).
24. T. Pevny and J. Fridrich, "Merging markov and DCT features for multi-class JPEG steganalysis," *In Proc. SPIE Electronic Imaging, Photonics West*, 3-4 (2007).
25. J. Platt, "Probabilistic outputs for support vector machines and comparison to regularized likelihood methods," *In Advances in Large Margin Classifiers*, MIT Press, MA, 2000.
26. N. Provos, OutGuess, http://www.outguess.org/, . retrieved July 26, 2006.
27. N. Provos, and P. Honeyman, "Hide and Seek: An Introduction to Steganography," *IEEE Security & Privacy Magazine*, May/June (2003).
28. B. M. Rodriguez and G. L. Peterson, "Steganography detection using multi class classification," *Third Annual IFIP WG 11.9 International Conference on Digital Forensics*, Orlando, Florida, 2007, In Advances in Digital Forensics III, eds. Craiger, P., and Shenoi, S., Springer Science+Business Media, Boston, MA, pp. 193-204.
29. B. M. Rodriguez, G. L. Peterson and K. W. Bauer, "Fusion of multiclass steganalysis systems using Bayesian model averaging," *Fourth Annual IFIP WG 11.9 International Conference on Digital Forensics*, Kyoto, Japan, 2008, In Advances in Digital Forensics IV
30. S. Russell and P. Norvig, *Artificial Intelligence A Modern Approach*, Second Edition, Pearson Education, Inc., Upper Saddle River, NJ, 2003.
31. P. Sallee, "Model-based steganography," *In International Workshop on Digital Watermarking*, Seoul, Korea, 2003.
32. B. Scholkopf and A. J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond, The MIT Press, 2002.
33. Y.Q. Shi, G. Xuan, D. Zou, J. Gao, C. Yang, Z. Zhang, P. Chai, W. Chen and C. Chen, "Image steganalysis based on moments of characteristic functions using wavelet decomposition, prediction-error image, and neural network," *In IEEE International Conference on Multimedia and Expo*, (2005).
34. D. M. J. Tax and R. P. W. Duin, "Using two-class classifiers for multiclass classification," *In Proc. 16th International Conference on Pattern Recognition*, 2, 124-127 (2002).
35. S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, 3rd Edition, Academic Press. San Diego, CA, 2006.
36. D. Upham, JPEG – Jsteg, ftp://ftp.funet.fi/pub/crypt/steganography/, 1993, Retrieved July 20, 2005.
37. V. Vapnik, *Statistical Learning Theory*, Wiley, 1998.
38. A. Westfeld, "F5-A steganographic algorithm high capacity despite better steganalysis," *In Moskowitz, I.S.I, Eds.: Information Hiding 4th International Workshop*, *Lecture Notes in Computer Science*, 2137, 289-302 (2001).
39. Y. Wang and P. Moulin, "Optimized feature extraction for learning-based image steganalysis," *In IEEE Transactions on Information Forensics and Security*, 2(1), 31-45 (2007).
40. 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 Information Hiding Workshop (IHW05)*, 2005.

*benjamin.rodriguez@afit.edu; phone 1 (937) 255-3636 ext. 7543; fax 1 (937) 656-4055