Improving Steganalysis by Fusion Techniques: A Case Study with Image Steganography

Mehdi Kharrazi[1], Husrev T. Sencar[2], and Nasir Memon[2]

[1] Department of Electrical and Computer Engineering
[2] Department of Computer and Information Science
Polytechnic University, Brooklyn, NY 11201, USA

Abstract. In the past few years, we have witnessed a number of powerful steganalysis technique proposed in the literature. These techniques could be categorized as either specific or universal. Each category of techniques has a set of advantages and disadvantages. A steganalysis technique specific to a steganographic embedding technique would perform well when tested only on that method and might fail on all others. On the other hand, universal steganalysis methods perform less accurately overall but provide acceptable performance in many cases. In practice, since the steganalyst will not be able to know what steganographic technique is used, it has to deploy a number of techniques on suspected images. In such a setting the most important question that needs to be answered is: What should the steganalyst do when the decisions produced by different steganalysis techniques are in contradiction? In this work, we propose and investigate the use of information fusion methods to aggregate the outputs of multiple steganalysis techniques. We consider several fusion rules that are applicable to steganalysis, and illustrate, through a number of case studies, how *composite* steganalyzers with improved performance can be designed. It is shown that fusion techniques increase detection accuracy and offer scalability, by enabling seamless integration of new steganalysis techniques.

1 Introduction

Steganography refers to the science of "invisible" communication. Unlike cryptography, where the goal is to secure communications from an eavesdropper, steganographic techniques strive to hide the very presence of the message itself from an observer. On the other hand, *steganalysis* techniques are used to detect the presence of hidden messages in an image. The reader is referred to [1] for a review of the field. Essentially there are two approaches to the problem of steganalysis, one is to come up with steganalysis techniques that are specific to a particular steganographic technique. The other is developing universal techniques that are effective over a wide variety of steganographic techniques. [1]

[1] *Universality* can be defined to indicate applicability over all embedding techniques and/or the domains of operation. In this work, we use the notion of *universal* as with respect to embedding techniques.

Specific steganalysis attacks concentrate on image features which are directly modified by the embedding algorithm. For example, F5 [2] embedding algorithm suffers from DCT histogram shrinkage, in which the number of zero DCT coefficients increases after the embedding operation. To exploit this, the specific attack proposed in [3], examines the differences between the histogram of the stego image and it's estimated original. As another example, in the model based embedding technique [4] the crux of the embedding operation lies in fitting a parametric model to the DCT histograms and preserving those models after embedding. The weakness of this approach is that DCT histograms of the cover images do not follow the model precisely. The specific attack proposed in [5] analyzes how well the image's DCT histograms match the fitted model for that image to determine whether the image in question is carrying hidden messages or not. Although such steganalysis techniques would perform well when tested only on the intended embedding method, they are very likely to fail on all other steganographic methods.

Universal steganalysis techniques operate by extracting some inherent *features* of cover images that are likely to be modified when an image undergoes steganographic embedding process. These features are then used to classify the image as either a cover or stego image. There have been a number of universal steganalysis techniques proposed in the literature. These techniques differ in the feature sets they utilize for capturing the characteristics of images. For example, Avcibas et al. [6] calculate several binary similarity measures between the seventh and eighth bit planes of an image. Farid et al. [7,8], obtain a number of statistics from the wavelet transform coefficients of images. On the other hand, Fridrich [9] utilizes DCT coefficient statistics. As observed in [10,6] universal steganalysis techniques do not perform equally over all embedding techniques; Nor are they able to distinguish perfectly between cover and stego images.

Furthermore, the classifier at the heart of each universal steganalyzer needs to be trained using a set of sample cover and stego images. This training process becomes computationally expensive depending on the type of classifier used, sample dataset size, and the separation of cover and stego images in the feature space.

With the availability of different type of steganalyzers (specific and universal) a number of questions would arise:

- What is the performance penalty due to the use of universal (or specific) steganalysis techniques assuming a practical setting of the problem?
- When multiple steganalyzers are used together, how do we deal with contradictory decisions?
- How does detection performance change when multiple embedding techniques are deployed in training the steganalyzer as opposed to using a specific technique, and what is the computational cost for repeating the training process to include new steganographic methods (in the training phase)?
- What is the most efficient strategy to *combine* different steganalyzers?

To answer these questions, we propose the use of *information fusion* techniques to incorporate steganalyzers, specific and universal, together. This approach has two potential advantages, in addition to providing a solution to real-life steganalysis problem. First, it improves the accuracy of distinguishing between a set of cover and stego images when multiple steganalyzers are available for use. Second, it reduces the computation cost associated with re-training a steganalyzer built to detect different *types* of stego images when a new steganographic technique has to be added to the training dataset.

The organization of the paper is as follows. In Section2, we review fusion techniques that are applicable to steganalysis problem. In Section3, we study the design of a composite steganalyzer by fusing a number of steganalysis techniques and provide performance comparison results. In Section4, we study how incorporation of a number of steganalysis techniques could be made scalable, by avoiding the cost associated with re-training steganalyzers. Our discussion of the results and conclusions are given in Section5.

2 Fusion Techniques

At the heart of every steganalyzer is a classifier which, given an image feature or feature vector, decides whether the image at hand contains any secret messages. Therefore, the fusion strategies developed for constructing more sophisticated classifiers can be considered for our purposes as well.

Motivated by [11], we review possible fusion strategies for classifiers as related to our work. Figure 1 summarizes different scenarios and classification stages in

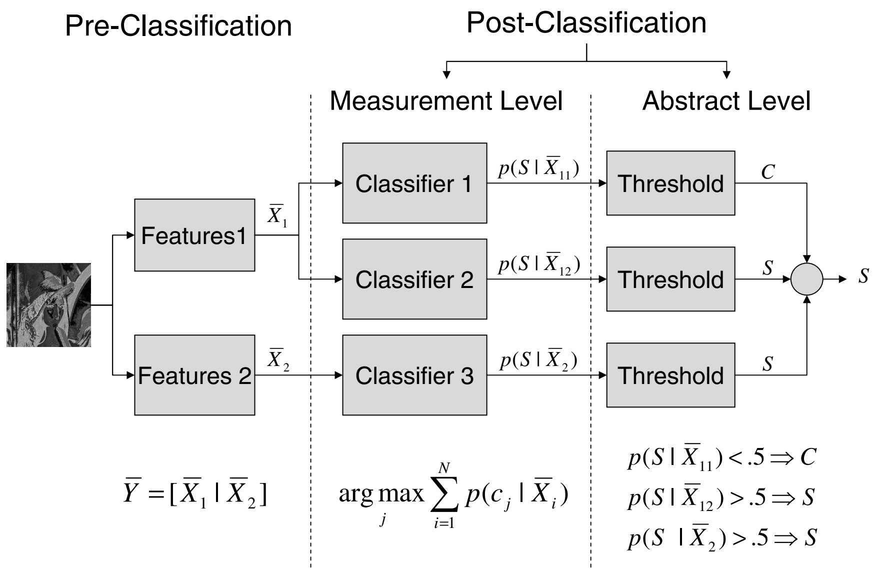

Fig. 1. Different scenarios as well classification stages in which fusion could be applied. For example fusion could be applied among a set of classifier trained using one feature vector, different feature vectors, or a hybrid of the two. Fusion may also be applied at different stages of the classification process (i.e. pre-classification or post-classification).

which fusion could be used. For example, a set of classifiers can be designed using the same feature vector. (I.e., given a feature vector, we could design a linear as well as non-linear classifier and fuse the results together.) On the other hand, fusion could also be applied to a set of classifiers each designed with a separate feature vector. Furthermore the two approaches could be combined into a hybrid approach. But more important is the stage in classification at which fusion is applied. Below, we provide a break up of these stages and discuss how they could be applied to steganalysis techniques.

2.1 Pre-classification

In essence, given an image I, the steganalyst first calculates the feature vector $X_I = [x_1, x_2, x_3, ...]$ from I. The feature vector is then used by a classifier, that was trained on previous observations of X, to output a decision regarding the nature of the image I (i.e., cover or stego). Fusion at this stage could be done by concatenating the feature vectors associated with each steganalysis technique and re-training the classifier with the feature vector Y_I defined as

$$Y_I = [X_{I1}|X_{I2}|X_{I3}...]. \tag{1}$$

But in practice a number of problems arise with such an approach. These are:

- With the increasing number of features, the classifier becomes more susceptible to *curse of dimensionality* problem.
- Correlated and redundant features need to be excluded for better performance.
- Classifier needs to be re-designed every time a new component is added to the feature vector Y_I.
- Different feature compositions may require different designing approaches.

To elaborate on the last point, in our experiments we have observed that some feature vectors show much improvement with more computationally expensive non-linear classifiers, whereas others show very little improvement. Thus one need to take into consideration such factors when constructing the classifier.

2.2 Post-classification

In this case, the classifier is trained using a set of stego and cover feature vectors, thereby calculating the location of the decision hyper-plane in the high-dimensional feature space. Therefore, the trained classifier could be thought of as a function, f_{class}, that computes the perpendicular distance of I, in terms of the extracted features X_I to the decision hyperplane in the feature space. This distance, also called decision value, DV, is used to categorize the image I as either cover or stego. Hence, we have

$$DV_{X_I} = f_{class}(X_I) \tag{2}$$

Below we will discuss two different post-classification levels, with which the obtained decision values are processed. It is after this processing that the decision values obtained from a set of classifiers become comparable, and therefore could be fused.

Measurement Level. The obtained decision values need to be normalized in order to make them comparable among a set of classifiers. This could be done by converting the decision values to a conditional distribution, $P(stego|X_I)$, i.e., the posterior probability of image I represented by feature vector X_I carrying a secret message denoted as

$$P(stego|X_I) = f_{norm}(DV_{XI}) = f_{norm}(f_{class}(X_I)). \tag{3}$$

Since there are only two classes available (i.e. cover or stego), we have

$$P(cover|X_I) = 1 - P(stego|X_I). \tag{4}$$

This is the most widely used stage for fusion. Here, the measurement information obtained from a set of steganalyzers could be either input into a second stage classifier for a final decision, or could be combined using schemes such as the *Mean*, *Max*, *Min*, *Median*, and *Product* rules. In fact, Kittler et al. in [12] conduct a theoretical study of these rules, and show that the *Mean* rule is least susceptible to estimation errors in the conditional probability distributions. Given the results in [12] and based on our preliminary experimental study, we decided to only employ the *Mean* and *Max* rules in our experiments. These two rules are explained below:

- *Mean Rule:*
  $C = argmax_j \sum_{i=1}^{N} P(c_j|X_{Ii})$
  With this rule, the class c_j (for $j = \{stego, cover\}$), assigned to input image I, is the class with which the sum of the conditional probabilities for that class is maximized.
- *Max Rule:*
  $C = argmax_j max_i P(c_j|X_{Ii})$
  Here the class c_j is assigned to input image I with which the maximum conditional probability is obtained.

Abstract Level. Fusion could also be applied at the last stage of classification, in which conditional class distributions are thresholded (or alternatively the decision values are thresholded directly), and a decision is made as to the class of the image I:

$$P(stego|X_I) > .5 \Rightarrow I \in stego \tag{5}$$

$$P(stego|X_I) < .5 \Rightarrow I \in cover \tag{6}$$

In this case, voting rule could be used to obtain a collective decision from a set of steganalyzers. But since this stage is obtained by thresholding the conditional probability distribution values, yielding a binary value, it will provide minimal usable information for fusion.

3 Fusion Based Steganalysis

In a practical setting, the steganalyst will be unsure of the embedding technique being used, if any. Therefore the conventional approach is to employ a universal steganalyzer which could detect, although not perfectly, stego images. But the steganalyst could also have a set of specific steganalyzers at her disposal, that in some cases perform more accurately than the universal techniques, or even alternate universal steganalyzers. In such a scenario, the steganalyst could create a new composite steganalyzer and improve the detection performance by fusing the decision obtained from the available set of universal and/or specific steganalysis techniques as described in Section2.

In what follows, we illustrate through the two possible scenarios, how a new steganalyzer could be built by fusing the results from a select set of steganalyzers. In the first scenario, Section3.1, we investigate the fusion of a number of universal steganalysis techniques, whereas in the second scenario, Section3.2, we investigate the fusion of universal and specific steganalysis techniques.

3.1 Fusing Universal Techniques

We will first study the fusion of three universal steganalysis techniques. Here we employed, binary similarity measures based steganalysis [13] denoted as BSM, wavelet transform coefficient features' based steganalysis [7,8] denoted as WBS, and DCT coefficient features' based steganalysis [9] denoted as FBS. An initial database consisting of 1800 natural images were used [14]. The images were converted to gray-scale and the borders around them were cropped, resulting in images of size 640x480 pixels, after which they were re-compressed with a quality factor of 75.

A stego dataset was created using the LSB and LSB +/- embedding techniques. In the LSB technique, the LSB of the pixels is replaced by the message bits to be sent. Usually the message bits are scattered around the image, by selecting the pixels to be modified in a random walk. Alternatively, LSB +/-, operates by incrementing or decrementing the last bit instead of replacing it. Message size was set as the ratio of bits per pixel in the image, more specifically we used the message sizes of 0.1 (3840 Bytes) and 0.2(7680 Bytes) in creating the stego set. A classifier was built for each message length using the feature vectors obtained by each steganalysis technique.

Fusion of the three steganalysis techniques was done at measurement level, using the *Max* and *Mean* rules discussed earlier in Section2. These rules operate on class conditional probabilities obtained from each steganalysis technique. For example with *Max* rule, the class of an input image is designated by the steganalyzer that has highest confidence in its decision, e.g., yielded the maximum conditional probability. With the *Mean* rule, the class to an input image is assigned so that the sum (or mean) of the conditional probabilities, associated with each steganalyzer, for that class is maximized.

The accuracy of the original steganalysis techniques, as well as the accuracy of the techniques when fused could be seen in terms of ROC curves in figure 2

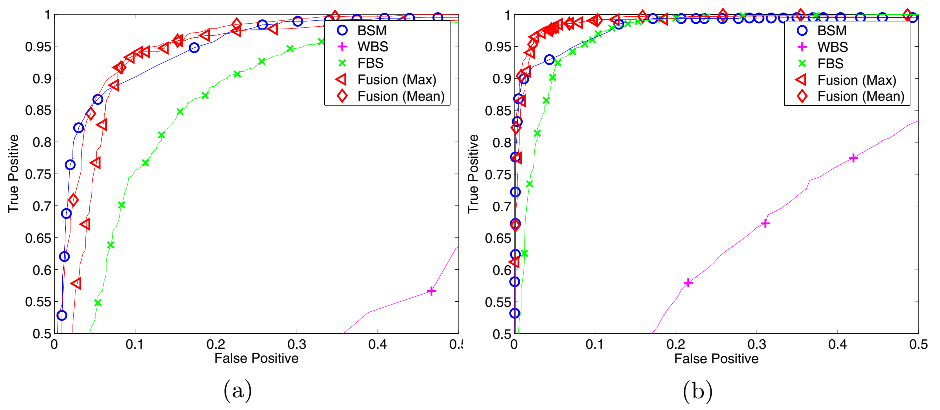

Fig. 2. ROC curves obtained for LSB embedded vs. unmarked images for three different steganalysis as well as two fusion techniques. **(a)** 0.1 messages size. **(b)** 0.2 message size.

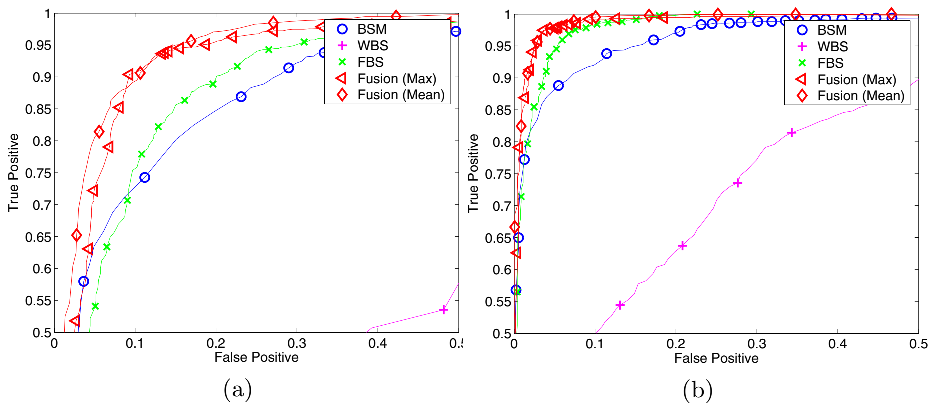

Fig. 3. ROC curves obtained for LSB +/- embedded vs. unmarked images for three different steganalysis as well as two fusion techniques. **(a)** 0.1 messages size. **(b)** 0.2 message size.

for the LSB, and in figure 3 for the LSB +/- embedding techniques. As seen in these figures fusion does provide considerable improvement over the three steganalysis techniques. To quantify this improvement the areas under the ROC curves are calculated and given in Table 1. We also observe that from the two fusion rules employed, the *Mean* rule outperforms the *Max* rule, therefore we will only employ the *Mean* rule for the rest of this work.

Here we should note that although the FBS steganalysis technique is proposed for DCT based embedding techniques, it performed satisfactorily on spatial domain embedding technique because of the dataset employed in these experiments. More specifically the original cover images used, were obtained from JPEG images compressed with a quality factor of 75. But, as observed in [15], if we

Table 1. Area under the ROC curves

	BSM	WBS	FBS	Fusion (Max)	Fusion (Mean)
LSB (0.1)	96.62	58.23	91.72	95.58	97.36
LSB (0.2)	98.79	73.97	98.12	99.24	99.54
LSB +/- (0.1)	90.92	59.11	92.09	94.69	96.34
LSB +/- (0.2)	97.51	80.61	98.78	99.14	99.43

use original BMP images as covers, where there exists no JPEG artifacts, the FBS technique is unable to distinguish between cover and LSB embedded stego images.

3.2 Fusing Specific and Universal Techniques

The second scenario we have looked at is one in which the embedder uses two types of embedding techniques, LSB and LSB +/- embedding techniques. We assume the steganalyst has available at her disposal the BSM universal steganalyzer [6], which performs well over both the LSB and LSB +/- embeddings, and the Pair steganalysis [16], which is a specific attack on the LSB embedding technique. From the large data set of gray scale images obtained for our benchmarking study in [10], we obtained about 13000 images with a quality factor of 85 and above and with a minimum width of 1000 pixels. These images were then down-sized to a size of 640x480 and saved as BMP. This was done to minimize the JPEG compression artifacts. We should note that the data set only consists of images that had their aspect ratio preserved after the down sampling operations.

Two BSM steganalyzers were trained independently, using the non-linear SVM [17], with the LSB and LSB +/- stego images. The message size used in creating the stego dataset was set as to the ratio of bits per pixel in the image, where in this case a value of 0.6 was used. Furthermore, in order to obtain the fusion result at different false positive rates (i.e. ROC curve), we opted to choose the output of the pair analysis attack as a feature value to be used by the classifier. This would also allow us to obtain classification confidence values which we will use when fusing the steganalysis techniques. Thus we have four steganalyzers available, including the fused steganalyzer, which was obtained using the *Mean* rule.

The four steganalyzers are tested against the cover dataset and 3 different stego datasets, namely, an LSB dataset, an LSB +/- dataset, and a dataset consisting of equal number of unique LSB and LSB +/- stego images. The obtained ROC curves for each dataset could be seen in figure 4, and the AUR [2] values could be seen in Table 2. From the results we observe that the specific attack works perfectly and has an accuracy of 100% when distinguishing between cover and LSB stego images, and when tested against the LSB +/- dataset the technique fails as expected. But when the pair analysis is tested against the mixed

[2] AUR: The area under the ROC curve is generally used to obtain a single comparative performance value for each classifier.

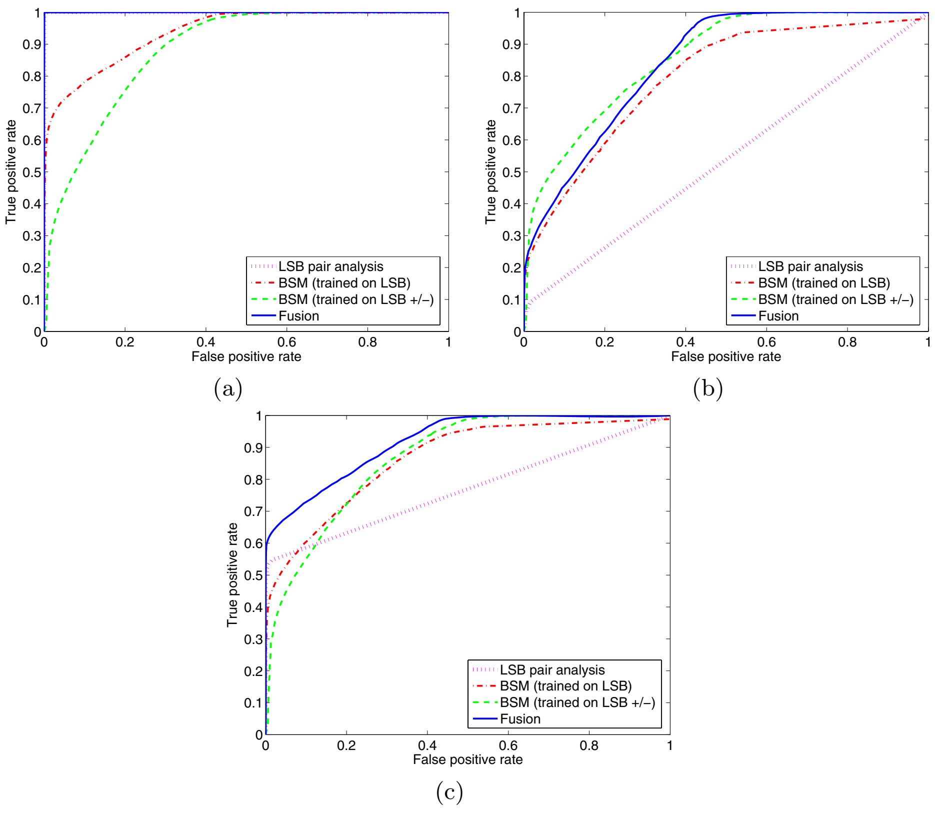

Fig. 4. Obtained ROC curves for the 4 steganalyzers when employed against the same cover but different stego datasets. (a) The stego test images consist of only LSB images. (b) The stego test images consist of only LSB +/- images. (c) The stego test images consist of both LSB and LSB +/- images. Here the number of LSB and LSB +/- stego images is the same, and we have avoided using the same image from both sets.

Table 2. AUR obtained from the ROC curves, when fusing universal and specific steganalysis techniques

	Specific Attack	BSM (LSB)	BSM(LSB +/-)	Fusion
LSB (0.6)	99.96	93.72	88.04	99.06
LSB +/- (0.6)	53.84	79.49	85.72	84.15
Mixed Set	76.94	86.52	86.90	92.07

dataset, its performance reduces since it is only effective in detecting the LSB images and not the LSB +/- images.

The two BSM steganalzyers, one trained with cover and LSB stego images the other with cover and LSB +/- stego images, perform around the same level with all three datasets. But when the outputs of these three steganalyzers (tested against the mixed dataset) are fused together, we observe a 15.3%, 5.55%, and 5.17% performance improvement from the results obtained if we had used only the specific attack, BSM trained with LSB stego images, and BSM trained with LSB +/- stego images, respectively.

It should be noted that we also tried a decision tree approach in which at the root we had placed the pair steganalysis technique. But the results of such fusion technique were poor, due to the inaccuracy of the pair steganalysis technique in identifying LSB +/- stego images.

4 Fusion Based Adaptive Steganalysis

Although in theory universal steganalysis techniques are meant to detect any stego embedding technique, even ones unseen to it at the training stage, in our experiments (as will be discussed later in this section), we have observed otherwise. That is, a trained steganalyzer using embedding technique A, which also performs well when tested on stego images of type A, performs quite inaccurately if it is asked to classify stego image obtained from embedding technique B. This is best illustrated in figure 5, where we show two stego sets denoted as *stego1* and *stego2*.

If the training dataset only consists of *cover* and *stego1* images then the classifiers might have a decision plane following the line A, with which most of *stego2* images will be classified correctly. But if the training dataset consists of *cover* and *stego2* images then the classifiers decision plane will follow line B, with which half of the the *stego1* images will be misclassified as cover images. In order to avoid such a problem, the training set needs to include both *stego1* and *stego2* images so that the classifier's decision plane will follow line C, and it will be able to correctly classify both *stego1* and *stego2* images.

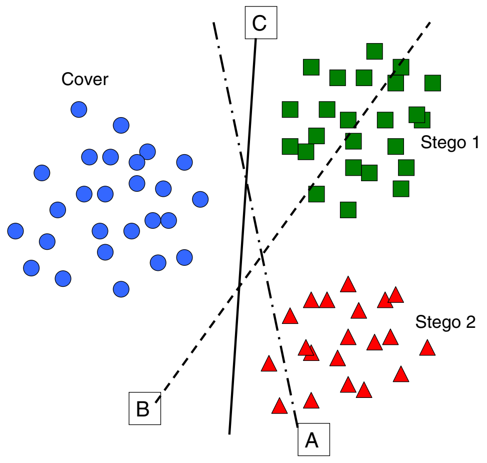

Fig. 5. Effects of training set on the performance of universal steganalysis techniques

Fridrich et al. [18], exploits the above deficiency of the universal steganalysis techniques to address another interesting problem. In their work, a multi-class classifier is designed using cover and stego images created with a number of embedding techniques. Since, presumably, stego images from each embedding technique occupy a unique space in the feature space, the steganalyst, not only differentiates among stego and cover images, but also is able to distinguish between different types of stego images based on the embedding technique used.

We should note that the above problem could potentially be avoided using one-class SVMs, but that approach has its own downsides. One-class SVMs are designed with only one class of images by creating a hyper-sphere in the feature space so that all images that fall inside the hyper-sphere are defined to be cover images and images that fall outside of the hyper-sphere are deemed to be stego images. Hence, the accuracy of such classifiers greatly depends on how well the cover images, represented by a set of extracted features, could be enclosed by a hyper-sphere. Because of the difficulty of this requirement two-class SVM classifiers, which have access to both cover and stego images at the design stage, outperform one-class SVMs.

In universal steganalysis, as the number of stego techniques represented in the training dataset increases the size of the training dataset needs to grow as well. This is so that a minimal number of stego images from each technique could be represented in the dataset. However, this increase in the dataset size also increases the classifier's training cost, thus making this approach unscalable and prohibitive. To show the relationship between the training set size and computational time, we have conducted a simple experiment in which we trained a set of classifiers each using training sets with varying sizes that consist of cover and stego images obtained by Model Based steganoggraphic embedding technique. The images dataset from Section3.2 was used, with the message length set to .08 bits per image pixel. The training set size vs. computational curve is given in figure 6. From the figure, we observe that the computational time increases rapidly as the training set size increases for the linear SVM classifier. This increase is more drastic when we used the superior non-linear SVM classifier. For example if our training set consists of 110000 images, then it would take more than 11 hours to design the non-linear SVM classifier.

The above described problem is further exacerbated due to the fact that the training operation has to be repeated every time with images from a new steganograhic embedding technique are added to the training dataset. The use of fusion strategies, aside from addressing decision aggregation problem, also offers a solution to this problem. This can be realized by designing a separate classifier for each available steganographic technique and then fusing the decisions obtained by testing an image against all available classifiers. Therefore, when a new steganographic technique is introduced or dataset is changed, re-training at a global scale is not needed. But the question to be answered is whether, with fusion, we will be able to obtain accuracy results as well as those obtained from a steganalyzer trained with a dataset containing stego images created with available steganographic techniques.

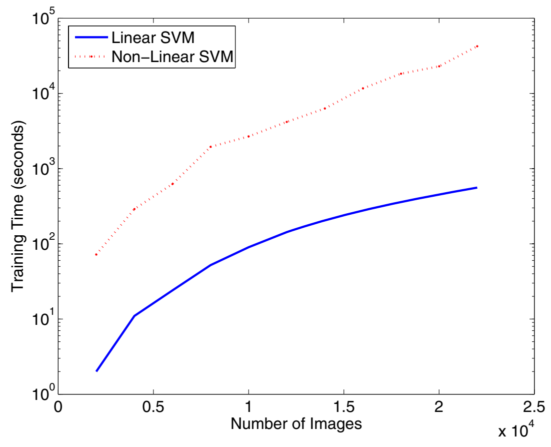

Fig. 6. Computational time for different training set sizes, using the linear and non-linear SVM classifier. Training was done on a machine with a Xeon 2.8GHz processor, and 1GB of memory, running linux. In the case of the non-linear svm, the parameter grid search was done from 2^0 to 2^{15} with steps of 2^3 for parameter c, and from 2^{-10} to 2^5 with steps of 2^3 for parameter g.

To investigate the above question, we obtained a set of cover images, as in Section3.2, and used Outguess (+) [3] [19], F5 [2], and Model Based [4] techniques to create three stego datasets. The message lengths were set to 0.06 bits per image pixel. As for the steganalyzer, we employed the FBS technique. A steganalyzer was trained independently for each of the three cover and stego image pairs. We further designed a steganalyzer using a training set which consists of cover images and a stego dataset compromised of equal number of stego images from all three embedding techniques.

To show the importance of the training set on the performance of the universal steganalyzers, we tested each trained steganalyzer against the same cover but three different stego datasets. The obtained ROC curves are seen in figure 7, and the calculated AURs are presented in Table 3. For example, we observe from these results that the steganalyzer trained solely on the Outguess (+) stego images, when asked to distinguish between cover and Outguess (+) images, obtains an accuracy of 98.49%. But, its accuracy for distinguishing cover images from F5 and Model Based images is 54.37% and 66.45%, respectively.

Afterwards, the output of the three steganalyzers, each trained for one of the three embedding techniques, are fused using the *Mean* rule. Alternatively a steganalyzer is trained using all three available stego images. The obtained results are in figure 7, and the calculated AURs are presented in Table 3. Based on these

[3] Outguess (+): The plus sign indicates the usage of the *statistical steganalysis foiling feature* with the Outguess program. With this feature, a set of reserved DCT coefficients are adjusted after the message has been embedded with the aim of preserving the original histogram of DCT coefficients.

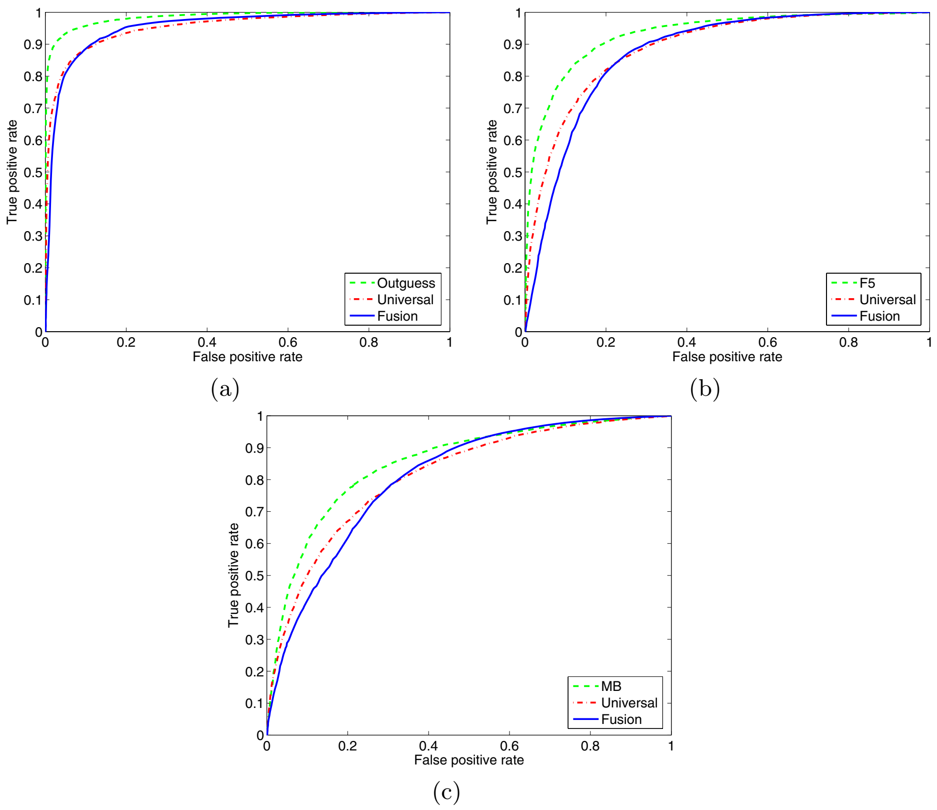

Fig. 7. Obtained ROC curves for the designed steganalyzers when employed against the same cover but different stego datasets. (a) Tested against the Outguess(+) stego set. (b) Tested against the F5 stego set. (c) Tested against the MB stego set.

Table 3. AUR obtained from the ROC curves, when fusing steganalyzers to obtain scalability

	FBS (Out+)	FBS (F5)	FBS (MB)	FBS (Universal)	FBS (Fusion)
Out+	98.49	44.06	86.29	95.36	95.34
F5	54.37	93.12	65.47	88.84	87.16
MB	66.45	63.73	85.22	81.34	80.62

results, we make two observations, first of all, as argued earlier the composition of training set plays an important role on the performance of the steganalyzer. Secondly, we see a performance degradation ranging roughly from 3% to 4% when we test our datasets against the steganalyzer trained with all three stego techniques as opposed to steganlyzers trained with only one stego technique. As evident from the results the fused steganalyzer has performance values very close to the steganalyzer trained with all the three embedding techniques included in its training dataset. Thus, fusion allows us to train steganalyzers only using one embedding technique and then fuse the outputs together, therefore avoiding the re-training of the classifier with new embedding techniques included in the training dataset. We believe that these results will generalize over alternate universal steganalysis techniques, and are not specific to the technique studied here, although the magnitude of the effects may vary.

5 Discussion

With the availability of large number of steganalysis techniques proposed in the literature, one might feel that the steganalyst has a good chance of distinguishing between cover and stego images. But in practice, the steganalyst will have to select one or more techniques which she will employ on a set of suspected stego images. However, the question of what to do when the results produced by various steganalysis techniques are in contradiction was not answered, previously. In this work, we investigated how fusion techniques could be applied in steganalysis to resolve such questions.

As the first application, we illustrated how a new steganalyzer could be created by fusing a number of steganalysis techniques while at the same time improving the detection accuracy. As the second application of fusion, we discussed the importance of the training set for universal steganalysis techniques and argued that incorporation of a new steganographic embedding technique into the an already designed steganalyzer is a costly and unscalable procedure. As an alternative, we proposed fusing decisions from a set of steganalysis technique trained independently using only one embedding technique. We illustrated through experimentation that the obtained accuracy results matches that of a steganalyzer trained with stego images from all embedding techniques studied, while at the same time providing scalability.

We believe that the applications of fusion techniques are not limited to the examples we have studied in this work. For example, as noted earlier, Fridrich et al. [18] illustrate how they could identify between a set of stego images, based on the embedding technique used to create them. In their work, only one steganalysis technique is employed, but with the help of fusion, one could improve and expand the results, by including more steganalyzers. Further more, such approach could be extended to alternate post-steganalysis operations such as estimation of the embedded message length. This form of information would be quite valuable in any forensic analysis of the stego images that intends to recover the hidden message.

References

1. Kharrazi, M., Sencar, H.T., Memon, N.: Image steganography: Concepts and practice. to appear in Lecture Note Series, Institute for Mathematical Sciences, National University of Singapore (2004)
2. Westfeld, A.: F5 steganographic algorithm: High capacity despite better steganalysis. 4th International Workshop on Information Hiding. (2001)

3. Fridrich, J., Goljan, M., Hogea, D., Soukal, D.: Quantitive steganalysis of digital images: Estimating the secret message lenght. ACM Multimedia Systems Journal, Special issue on Multimedia Security (2003)
4. Sallee, P.: Model-based steganography. International Workshop on Digital Watermarking, Seoul, Korea. (2003.)
5. Bhme, R., Westfeld, A.: Breaking cauchy model-based jpeg steganography with first order statistics. 9th European Symposium on Research in Computer Security, Sophia Antipolis, France, September (2004)
6. Avcibas, I., Kharrazi, M., Memon, N., sankur, B.: Image steganalysis with binary similarity measures. To appear in EURASIP Journal on Applied Signal Processing (2005.)
7. Lyu, S., Farid, H.: Detecting hidden messages using higher-order statistics and support vector machines. 5th International Workshop on Information Hiding. (2002.)
8. Lyu, S., Farid, H.: Steganalysis using color wavelet statistics and one-class support vector machines. SPIE Symposium on Electronic Imaging, San Jose, CA, (2004.)
9. Fridrich, J.: Feature-based steganalysis for jpeg images and its implications for future design of steganographic schemes. Proc. 6th Information Hiding Workshop, Toronto, Canada, May 23-25 (2004)
10. Kharrazi, M., Sencar, T.H., Memon, N.: Benchmarking steganographic and steganalysis techniques. EI SPIE San Jose, CA, January 16-20 (2005)
11. Jain, A.K., Nandakumar, K., Ros, A.: Score normalization in multimodal biometric systems. to appear in Pattern Recognition (2005)
12. Kittler, J., Hatef, M., Duin, R., Matas, J.: On combining classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence **20**(3) (1998) 226–239
13. Avcibas, I., Memon, N., sankur, B.: Image steganalysis with binary similarity measures. IEEE International Conference on Image Processing, Rochester, New York. (September 2002.)
14. Greenspun, P.: Images obtained from. philip.greenspun.com (-)
15. Kharrazi, M., Sencar, T.H., Memon, N.: Benchmarking steganographic and steganalysis techniques. Submitted to the Journal of Electronic Imaging (2006)
16. Dumitrescu, S., Wu, X., Memon, N.: On steganalysis of random lsb embedding in continuous-tone images. IEEE International Conference on Image Processing, Rochester, New York. (September 2002.)
17. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. (2001) Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.
18. Fridrich, J., Pevny, T.: Multiclass blind steganalysis for jpeg images. SPIE Electronic Imaging, Photonics West, San Jose,CA (2006)
19. Provos, N.: Defending against statistical steganalysis. 10th USENIX Security Symposium (2001)