Image Steganalysis and Steganography in the Spatial Domain

DISSERTATION

Submitted to the Doctoral Programme in Network and Information Technologies of the Universitat Oberta de Catalunya in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Author: Daniel Lerch-Hostalot

Thesis Director: David Megías

February 2017

Acknowledgements

Vull aprofitar aquesta oportunitat per agrair a totes aquelles persones que, d'alguna manera, m'han ajudat a dur a terme aquesta tesi doctoral.

En primer lloc, vull agrair, a la meva dona Elisabet, la seva paciència durant aquests anys, en els que tantes vegades he estat massa concentrat en la recerca i no he tingut temps per a la família. Gràcies, Elisabet, pel teu suport i paciència. Estar al teu costat és el que me dona forces per a tirar endavant cada dia.

En segon lloc, però no menys important, vull agrair al meu fill Joel les hores d'alegria i felicitat que dóna un fill constantment als seus pares. Jugar amb ell sempre em fa oblidar els problemes i estressos de la vida i, per un moment, sentir que tot va bé. Aquests moments de desconnexió han estat de vital importància en tantes situacions en les que he estat encallat en algun punt de la recerca.

També vull agrair als meus pares i a la meva germana una infància feliç que m'ha portat a ser qui sóc. A ells els hi dec ser una persona amb ganes d'aprendre, que gaudeix dels reptes i preparada par a enfrontar una tasca gens fàcil, com es la realització d'una tesi doctoral.

En aquestes línies no pot faltar el Dr. David Megías, el meu director de tesi. A ell li he d'agrair el seu suport durant aquests anys de tesi i la seva excel·lent orientació en la recerca i els seus consells, tant a l'hora de triar les línies de treball com de millorar o complementar les meves investigacions. Però al David li he d'agrair alguna cosa mes. Ell va ser qui, abans inclús d'iniciar els meus estudis de Màster, em va orientar acadèmicament i qui em va introduir en el món de l'esteganografia i l'estegoanàlisi. A ell li he d'agrair, en bona part, haver recorregut el camí amb èxit.

This work was partly funded by the Spanish Government and the FEDER funds under the grants TSI2007-65406-C03-03 "E-AEGIS", CONSOLIDER INGENIO 2010 CSD2007-00004 "ARES", TIN2011-27076-C03-02 "CO-PRIVACY" and TIN2014-57364-C2-2-R "SMARTGLACIS".

Abstract

Modern steganography emerged at the same time as personal computers did, and what was a science with a very slow development through history began to evolve very quickly. Nowadays, steganographic techniques can be found in applications as diverse as privacy, malware, spying, terrorist communications or other types of criminal communication, and there is a growing number of steganographic tools available trough Internet. Nevertheless, steganography is not as developed as other related sciences, such as cryptography. In the recent years, there has been a considerable increase in the number of publications in this area, which constitutes an active research field.

One of the most used carriers to hide information using steganography are images, because of their widespread use in the Internet and due to the difficulty to model their statistics, making the detection of hidden information in this type of media a challenging task. The usual approach in the state of the art to hide information is to minimize the distortion introduced by the embedding process, obtaining a steganographic image (stego image) whose statistical properties are close to the original one (cover image).

To detect hidden information, the usual approach in the state of the art is to use machine learning based steganalysis. This type of steganalysis consists in preparing a set of images and hide information in some of them, thus obtaining a *training set*: a set of cover and stego images used for training a classifier. The setganalyzer uses this training set to extract some features and train a classifier that is able to detect which images are cover and which are stego, with some classification accuracy. The set of images that we want to classify is known as the *testing set*. This approach works very well in laboratory conditions, but there is a large room for improvement when this technique is applied to the real world data. The standard approach in the state of the art is to use Rich Models to extract features and Ensemble Classifiers for the training and classification processes.

In this dissertation, we propose different novel techniques both to detect hidden information (steganalysis) and to hide information (steganography). These techniques are presented in the form of a collection of five contributions, but sharing a common research problem. The first contribution presents three different methods to detect histogram shifting data hiding techniques, some of which are targeted attacks to specific schemes,

whereas others are more general. As a second contribution, in the area of machine learning steganalysis, we present a novel feature extractor to detect information hidden in the spatial domain, which can be used as an additional submodel in the rich models framework, and which outperforms the accuracy of the state-of-the-art steganalysis by subtractive pixel adjacency matrix (SPAM) with fewer features. In the same context, the third contribution is a steganographic algorithm that exploits the weakness of some submodels to deal with high dimensional data (which typically use a threshold to overcome the dimensionality problem). As a fourth contribution, we present a new framework for unsupervised steganalysis with accuracy higher than the supervised methods in the state of the art, while bypassing the cover source mismatch (CSM) problem. Finally, as a fifth contribution, we present a novel approach to address the CSM problem based on the set of machine learning techniques known as manifold alignment.

Resumen

La esteganografía moderna surgió al mismo tiempo que los ordenadores personales y, lo que fue una ciencia de desarrollo muy lento a lo largo de la historia, empezó a evolucionar rápidamente. Hoy en día, las técnicas esteganográficas pueden encontrarse en aplicaciones tan diversas como la privacidad, el *malware*, el espionaje y las comunicaciones terroristas u otros tipos de comunicación criminal, y existe un número creciente de herramientas de esteganografía disponibles a través de Internet. Sin embargo, la esteganografía no está tan desarrollada como otras ciencias relacionadas, como la criptografía. Por otra parte, en los últimos años, ha habido un incremento considerable en el número de publicaciones en esta área, la cual se ha convertido en un campo en activo desarrollo.

Uno de los medios más usados para ocultar información son las imágenes, dado su amplio uso en Internet y la dificultad de modelización estadística, que hace que la detección de información oculta sea un desafío. El enfoque habitual en el estado del arte, para ocultar información, consiste en minimizar la distorsión introducida durante la inserción del mensaje, obteniendo una imagen (imagen estego) cuyas propiedades estadísticas son muy parecidas a las de la imagen original (imagen cubierta).

Para detectar información oculta, el enfoque habitual en el estado del arte consiste en usar estegoanálisis basado en aprendizaje automático. Este tipo de estegoanálisis consiste en preparar un conjunto de imágenes y ocultar información en algunas de ellas, obteniendo un conjunto de *datos de entrenamiento*: imágenes cubierta y imágenes estego usadas para entrenar un clasificador. El estegoanalista usa estos datos de entrenamiento para extraer algunas características y entrenar un clasificador capaz de detectar qué imágenes son cubierta y qué imágenes son estego, con alta precisión. El conjunto de imágenes que queremos clasificar se conoce como *datos de prueba*. Este enfoque funciona muy bien en condiciones de laboratorio, pero todavía es necesaria mucha investigación para poder aplicarlo satisfactoriamente en entornos reales. En el estado del arte, habitualmente se usa el *framework* conocido como *rich models* para extraer las características de la imagen y el clasificador *ensemble classifiers* para realizar el entrenamiento y la clasificación.

En esta tesis, proponemos diferentes técnicas novedosas para detectar información oculta (estegoanálisis) y para ocultar información (esteganografía). Estas técnicas se pre-

sentan como una colección de cinco contribuciones, que comparten un problema común. La primera contribución presenta tres métodos diferentes para detectar información oculta usando técnicas de desplazamiento de histograma, algunas de las cuales son ataques dirigidos a esquemas concretos, mientras que las otras son más genéricas. En la segunda contribución, en el área del aprendizaje automático aplicado al estegoanálisis, presentamos un nuevo extractor de características para detectar información oculta en el dominio espacial, que se puede usar como submodelo adicional en el *framework rich models*, y que supera la precisión obtenida por el método del estado del arte *subtractive pixel adjacency matrix* (SPAM) usando un número inferior de características. En el mismo contexto, la tercera contribución es un algoritmo esteganográfico que explota la debilidad de algunos submodelos para tratar con datos de muchas dimensiones (los cuales suelen usar un umbral para superar el problema de la dimensionalidad). Como cuarta contribución, presentamos un nuevo *framework* de estegoanálisis dirigido no supervisado con una precisión superior a la de los métodos supervisados del estado del arte, y que permite eludir el problema del *cover source mismatch* (CSM). Finalmente, como quinta contribución, presentamos un nuevo enfoque al problema del CSM basado en un conjunto de técnicas de aprendizaje automático conocidas como *manifold alignment*.

Resum

L'esteganografia moderna va sorgir al mateix temps que els ordinadors personals i, el que va ser una ciència de desenvolupament molt lent al llarg de la història, va començar a evolucionar ràpidament. Avui en dia, les tècniques esteganogràfiques poden trobar-se en aplicacions tan diverses com la privacitat, el *malware*, l'espionatge i les comunicacions terroristes o altres tipus de comunicació criminal, i hi ha un nombre creixent d'eines de esteganografia disponibles a través d'Internet. No obstant això, l'esteganografia no està tan desenvolupada com altres ciències relacionades, com ara la criptografia. D'altra banda, en els últims anys hi ha hagut un increment considerable en el nombre de publicacions en aquesta àrea, la qual s'ha convertit en un camp en actiu desenvolupament.

Un dels mitjans més utilitzats per ocultar informació són les imatges, atès el seu ampli ús a Internet i la dificultat de modelització estadística, cosa que fa que la detecció d'informació oculta sigui un desafiament. L'apropament habitual en l'estat de l'art, per ocultar informació, consisteix a minimitzar la distorsió introduïda durant la inserció del missatge, obtenint una imatge (imatge estego) en què les propietats estadístiques són molt semblants a les de la imatge original (imatge coberta).

Per detectar informació oculta, l'enfocament habitual en l'estat de l'art consisteix a fer servir estegoanàlisi basada en aprenentatge automàtic. Aquest tipus d'estegoanàlisi consisteix a preparar un conjunt d'imatges i ocultar informació en algunes d'elles, obtenint un conjunt de *dades d'entrenament*: imatges coberta i imatges estego usades per entrenar un classificador. L'estegoanalista fa servir aquestes dades d'entrenament per extreure algunes característiques i entrenar un classificador capaç de detectar quines imatges són coberta i quines imatges són estego amb alta precisió. El conjunt d'imatges que volem classificar es coneix com *dades de prova*. Aquest apropament funciona molt bé en condicions de laboratori, però encara és necessària molta recerca per poder aplicar-lo satisfactòriament en entorns reals. En l'estat de l'art, habitualment es fa servir el *framework* conegut com a *rich models* per extreure les característiques de la imatge i el classificador *ensemble classifiers* per a realitzar l'entrenament i la classificació.

En aquesta tesi, proposem diferents tècniques novedoses per detectar informació oculta (estegoanàlisi) i per ocultar informació (esteganografia). Aquestes tècniques es presen-

ten en una col·lecció de cinc contribucions, que comparteixen un problema comú. La primera contribució presenta tres mètodes diferents per detectar informació oculta fent servir tècniques de desplaçament d'histograma, algunes de les quals són atacs dirigits a esquemes concrets, mentre que les altres són més genèriques. En la segona contribució, en l'àrea de l'aprenentatge automàtic aplicat a l'estegoanàlisi, presentem un nou extractor de característiques per detectar informació oculta en el domini espacial, que es pot fer servir com a submodel addicional en el *framework* de *rich models*, i que supera la precisió obtinguda pel mètode de l'estat de l'art *subtractive pixel adjacency matrix* (SPAM) fent servir un nombre inferior de característiques. En el mateix context, la tercera contribució és un algoritme esteganogràfic que explota la debilitat d'alguns submodels per tractar amb dades de moltes dimensions (els quals solen utilitzar un llindar per superar el problema de la dimensionalitat). Com a quarta contribució, presentem un nou *framework* d'estegoanàlisi dirigida no supervisada amb una precisió superior a la dels mètodes supervisats de l'estat de l'art, i que permet eludir el problema del *cover source mismatch* (CSM). Finalment, com a cinquena contribució, presentem un nou enfocament al problema del CSM basat en un conjunt de tècniques d'aprenentatge automàtic conegudes com a *manifold alignment*.

List of contributions

Contributions of the Ph.D. research

The validation of the research has been carried out with the publication of five original papers. All analysis and experimental or simulation results presented in the publications of this thesis have been produced by the author. The list of **published papers** is the following:

1. D. Lerch-Hostalot and D. Megías. Steganalytic Methods for the Detection of Histogram Shifting Data Hiding Schemes. In *XII Reunión Española sobre Criptología y Seguridad de la Información*, pages 381-386, 2012. ISBN: 978-84-615-9933-2.

   Citations: 2 (Google Scholar)

2. D. Lerch-Hostalot and D. Megías. LSB Matching Steganalysis Based on Patterns of Pixel Differences and Random Embedding. *Computers & Security*, 32:192–206, 2013.

   SCI JCR-indexed journal

   Impact factor (2013): 1.172; Quartile Q2 – Computer Science, Information Systems

   Citations: 7 (ISI Web of Knowledge), 10 (Scopus), 16 (Google Scholar)

3. D. Lerch-Hostalot and D. Megías. Esteganografía en Zona Ruidosas de la Imagen. In *XIII Reunión Española sobre Criptología y Seguridad de la Información*, pages 173-178, 2014. ISBN: 978-84-9717-323-0. (In Spanish).

   Citations: 1 (Google Scholar)

4. D. Lerch-Hostalot and Megías. Unsupervised steganalysis based on artificial training sets. *Engineering Applications of Artificial Intelligence* 50:45–59, 2016a.

   SCI JCR-indexed journal

Impact factor (2016): 2.368; Quartile Q1 – Computer Science, Artificial Intelligence

Citations: 1 (ISI Web of Knowledge), 1 (Scopus), 3 (Google Scholar)

5. D. Lerch-Hostalot and D. Megías. Manifold alignment approach to cover source mismatch in steganalysis. In *XIV Reunión Española sobre Criptología y Seguridad de la Información*, pages 123-128, 2016b. ISBN: 978-84-608-9470-4.

Other contributions

The author has other publications in the fields of cryptography and steganography. These publications are the following:

1. J. Serra Ruiz, D. Lerch-Hostalot and A. Muñoz. *Esteganografía y Estegoanálisis*. ZeroxWord Computing, Madrid, Spain, 2014. ISBN: 978-84-617-0021-9.

2. D. Lerch-Hostalot. Factorization Attack to RSA. *Hakin9 Magazine* 3:30–37, 2007.

3. D. Lerch-Hostalot. Criptografía de curva Elíptica, Ataque Rho de Pollard. *Hakin9 Magazine* 2–13, 2007.

Contents

Chapter 1

Introduction

1.1 Introduction to steganography and steganalysis

Steganography and steganalysis deal with different ways of hiding messages into other media and their detection, respectively. Although steganography is an ancient art, the science known today as steganography did not begin to develop before the nineties. Until then, this unusual way of secret communication was used only by spies. At that time, steganography could not be considered an area of research, since it was based solely on a set of tricks, without any theoretical basis, allowing only hiding secrets to non experts. However, with the emergence of the Internet and digital media, steganography experienced a considerable outburst. Applications (Fridrich, 2010) that allowed users to hide messages in media such as images, audio files or even text started to appear. At the same time, steganography caught the attention of academia, becoming a rigorous discipline with a well-grounded theoretical basis. Currently, steganography and steganalysis are active areas of research.

The word steganography combines the Greek words steganos (*steganos*), meaning "covered, concealed, or protected", and graphein (*graphia*) meaning "writing". Therefore, steganography deals with hiding messages. Unlike another more widely known science, cryptography, which ensures message confidentiality, steganography adds an additional layer of secrecy, keeping confidential even the mere fact that communication is taking place. On the other hand, the goal of steganalysis is not to extract the hidden message, but to detect a covert communication.

Steganography is formulated using the popular *prisoner problem* (Simmons, 1983): Alice and Bob are prisoners in separate cells and want to devise an escape plan. They are allowed to communicate, but all communications are supervised by Eve, the guardian of the prison. If Eve discovers that the prisoners are exchanging secret messages she will cut communications, putting Alice and Bob in isolation. The prisoners resort to

steganography to prepare their plan.

In the prisoner's problem, it is generally assumed that Eve has full access to the steganographic algorithm that Alice and Bob are using. All she does not know is the secret key used in the embedding process. This requirement has been adopted from cryptography and it is known as the Kerckhoffs's principle (Kerckhoffs, 1883), according to which the security of communications should not be based on keeping secret the used method, but only the (secret) key of the method. This principle stems from the experience that, through eavesdropping, the used method may fall into the enemy's hands. In this way, the security of the method shall not be based on its concealment.

In the digital age, the most frequently used media to hide messages through steganography are images, due to their widespread use in the Internet. There are many ways to embed information in images. In the recent years, most of the techniques are based on the modification of the bitmap image (Mielikäinen, 2006) and on the modification of the coefficients of the *Discrete Cosine Transform* (DCT) (Westfeld, 2001) –the transform used by JPEG images–. Likewise, steganalytic methods have always followed steganography closely, trying to expose every new technique that were proposed. Initially, there were steganalytic methods exploiting existing vulnerabilities in steganographic techniques but, later on, more generic methods we suggested, making use of classification techniques such as *machine learning* (Bishop, 2006; Mitchell, 1997).

The techniques used for hiding and detecting information have been extensively studied since the late nineties, reaching a considerable level of maturity. However, the science of steganography and steganalysis continues evolving and new methods to embed secret information and new ways of detecting it are being proposed continually.

One of the key factors for steganography is the understanding of the cover source; otherwise, one cannot be sure that the information is hidden without trace. In the case of images, the understanding of the statistical distribution of the image features is not yet complete. Many methods that appear to be secure –in terms of statistical undetectability– are exposed later on by steganalysts. The key for successful steganography is usually a careful analysis of the statistics of the image, something that machine learning can perform perfectly. Nowadays, most steganalytic techniques in the state of the art use machine learning to create a statistical model allowing to discriminate between an image with a hidden message ("stego image") and an original image with no embedded data ("cover image").

1.2 Objectives of the thesis

This thesis is mainly focused in steganalysis, but also with the aim of applying the lessons learned from steganalysis in the development of novel steganographic methods. Therefore, the main objective of this research is to find new techniques to improve steganalysis in images. More precisely, we can classify the objectives of the thesis into the following goals:

1. Propose new steganalytic techniques to attack histogram shifting data hiding methods.

2. Propose new steganalytic techniques to attack LSB matching steganography and its derivatives.

3. Develop new feature extractors to use in machine learning steganalysis.

4. Propose new techniques to deal with the Cover Source Mismatch problem in steganalysis.

Histogram shifting data hiding techniques are interesting because, unlike traditional steganography, they can usually recover the original image after message extraction. Although histogram shifting techniques are typically used for watermarking, they achieve significantly high embedding bit rates (around 0.5 bits per pixel and higher), especially when pixel predictions are used. Hence, although introducing minimum visual distortion is not an objective of steganography, but of watermarking, the high embedding capacity that can be achieved with histogram shifting techniques makes them attractive to steganographers. In this case, since the embedding method is standard, the steganalyst may not only detect if information has been hidden in the image through histogram shifting, but he/she may also try to determine the position of the hidden bits and finally extract the hidden message. Histogram shifting methods first perform a shift in the histogram of the image to produce one or more empty bins that can be used for embedding information afterwards. The knowledge of the pixel value (or prediction error, or interpolation error) used as the starting point of the shifting operation can be used to perform the reverse operation and extract the secret message. Therefore, a steganlyst would be interested in knowing, firstly, if the image contains information hidden using such kind of techniques and, secondly, which pixel (or value) was used as a base for the shifting. The second case is not always possible and depends on the specific technique used to hide data.

As remarked above, most steganalytic techniques in the state of the art use machine learning. Those methods generally use an image database (training set) containing samples of both cover images and stego images, prepared using the steganographic method that we want to detect. Firstly, we need to extract features from the images and use these

features and the corresponding labels as "cover" or "stego", to train a classifier. Once the classifier is prepared, we can use it to test images that we do not know if they are cover or stego (testing set).

From a research point of view there are two relevant issues in this approach. The first one is the feature extraction step. The second is the classification itself. The feature extraction step is highly relevant, since the success of the classification step depends on it. A good feature extractor should extract features that are very different when the image is cover from those corresponding to a stego image. One interesting property for a feature extractor is its performance, especially if we need to carry out steganalaysis in real time. The success of the classification step highly depends on the quality of the features extracted, but there is much room for improvement in the classification techniques. Although this issue is part of the machine learning field, in steganalysis, we need to deal with the design of dedicated classifiers to improve the classification exploiting the unique properties of the problem.

Finally, one of the main open challenges in steganalysis is the cover source mismatch (CSM) problem. This problem is related to the use of machine learning in steganalysis and occurs as a consequence of the training process. When we train a classifier, the result will be as good as our training data. If the training data are not good enough, for example, because we do not have access to the camera of the steganographer, the classifier will perform poorly. The CSM problem is an active area of research and there are different approaches to deal with it. For example, a common option is trying to avoid the CSM by training the classifier with a huge quantity of images with the hope that the training database is complete and CSM can be bypassed. However, the problem is still open and novel approaches are needed to deal with incomplete training sets.

1.3 Organization of the thesis

The rest of this thesis is organized as follows:

- Chapter 2 briefly presents the general background of data hiding and steganalysis, and summarizes the state of the art that is relevant to this research. It must be taken into account that each of the contributions presented in this thesis already contains an overview of the corresponding state of the art. Consequently, the state of the art presented in this chapter is intentionally short.

- Chapter 3 contains the five published contributions of this thesis, as detailed below:

  - In Section 3.1, the first published contribution of the thesis (Lerch-Hostalot and Megías, 2012) is presented. In this paper, several steganalytic techniques

designed to detect the existence of hidden messages using histogram shifting schemes are presented. Firstly, three techniques to identify specific histogram shifting data hiding schemes, based on detectable alterations on the histogram or abnormal statistical distributions, are suggested. Then, a general technique capable of detecting all the analyzed histogram shifting data hiding methods is suggested. This latter technique is based on the effect of histogram shifting methods on the "volatility" of the histogram of the difference image. The different behavior of volatility whenever new data are hidden makes it possible to identify stego and cover images.

– Section 3.2 presents the second published contribution of the thesis (Lerch-Hostalot and Megías, 2013). This paper proposes a novel method for detection of Least Significant Bit (LSB) matching steganography in grayscale images. This method is based on the analysis of the differences between neighboring pixels before and after random data embedding. In natural images, there is a strong correlation between adjacent pixels. This correlation is disturbed by LSB matching generating new types of correlations. The proposed method generates patterns from these correlations and analyzes their variation when random data are hidden. The experiments, performed for two different image databases, show that the proposed method yields better classification accuracy compared to prior art for both LSB matching and Highly Undetectable steGO (HUGO) steganography. In addition, although the method is designed for the spatial domain, some experiments show its applicability also for detecting JPEG steganography.

– In Section 3.3, the third published contribution of the thesis (Lerch-Hostalot and Megías, 2014) is presented. Most of the steganalytic techniques in the state of the art are based on the use of machine learning techniques, that is, in training classifiers to be able to differentiate a cover image from a stego counterpart. The research in this field shows that the areas in which it is more difficult to detect an embedded message are noisy ones, which correspond to edges and textures. In this paper, we present a novel steganographic method that embeds information in these areas. The effectiveness of the method has been tested using two image databases and two different steganalytic techniques. Our experiments show that the proposed algorithm improves the statistical undetectability of the method with respect to LSB matching steganography for the same embedding bit rate.

– Section 3.4 presents the fourth published contribution of the thesis (Lerch-Hostalot and Megías, 2016a). In this paper, an unsupervised steganalytic

method that combines artificial training sets and supervised classification is proposed. We provide a formal framework for unsupervised classification of stego and cover images in the typical situation of targeted steganalysis (i.e., for a known algorithm and approximate embedding bit rate). We also present a complete set of experiments using (1) eight different image databases, (2) image features based on Rich Models, and (3) three different embedding algorithms: LSB matching, HUGO and Wavelet Obtained Weights (WOW). The experimental results show that the proposed method outperforms prior art based on Rich Models in the majority of the tested cases. At the same time, the proposed approach bypasses the problem of Cover Source Mismatch –when the embedding algorithm and bit rate are known or if they can be approximated–, since it removes the need of a training database when we have a large enough testing set. Furthermore, we provide a generic proof of the proposed framework in the machine learning context. Hence, the results of this paper could be extended to other classification problems similar to steganalysis.

  - In Section 3.5, the fifth published contribution of the thesis (Lerch-Hostalot and Megías, 2016b) is presented. Cover source mismatch (CSM) is an important open problem in steganalysis. This problem, known as domain adaptation in the field of machine learning, deals with the decreasing of the classification accuracy when a classifier is moved from the laboratory into the real world. In this paper, we present an approach to CSM based on domain adaptation using manifold alignment algorithms. In this novel approach, we use manifold alignment to find a latent space where the two datasets (the one used for training and the one used for testing) have a common representation. We show that manifold alignment can significantly increase the accuracy of the classifier in crossdomain classification.

- Finally, Chapter 4 presents the concluding remarks of the thesis, together with some possible directions for future research.

Chapter 2

Background and state of the art

2.1 Data hiding techniques

Among the different techniques to hide information, watermarking and steganography deserve especial attention. *Watermarking* is the practice of altering a media in an imperceptible way to embed information about that media. On the other hand, *steganography* is the practice of altering a media in an undetectable way to embed a secret message (Cox et al., 2008).

There are some similarities between the techniques used in watermarking and steganography, but their objectives are quite different. Watermarking has its main application in copyright content protection, whereas steganography his mainly used for secret communications (privacy). Nevertheless, a more mature tool for privacy is cryptography. Cryptography allows to encode a message in such a way that an eavesdropper can not read it, but steganography provides another layer of protection, because it aims to conceal the fact that the communication is occurring.

The study of techniques to detect hidden information is called steganalysis. This work is focused in steganography and steganalysis in images, an important media because of their widespread use in the Internet.

There are different ways to hide information in images (Fridrich, 2010). The most direct way of hiding information is to modify the bitmap of the image, that is, to hide information in the spatial domain. In addition, we can also use a transformed domain, such as the Discrete Cosine Transform (DCT) (Ahmed et al., 1974) applied in the JPEG standard or the Discrete Wavelet Transform (DWT) (Croisier et al., 1976) exploited by the JPEG 2000 standard.

There exist different steganographic techniques for each case. The typical approach, both in the spatial and the transformed domains, is to perform an slight modification of the pixels or the transformed coefficients. In both cases, if the modification is small

enough, these changes cannot be perceived by the human eye. However, visual imperceptibility is not a typical requirement of steganography. For example, a red piece of clothing may be changed to blue in the stego image if this change does not raise any suspicion by a steganalyst. Steganography is not aimed at imperceptibility, but at statistical undetectability, which is a different property.

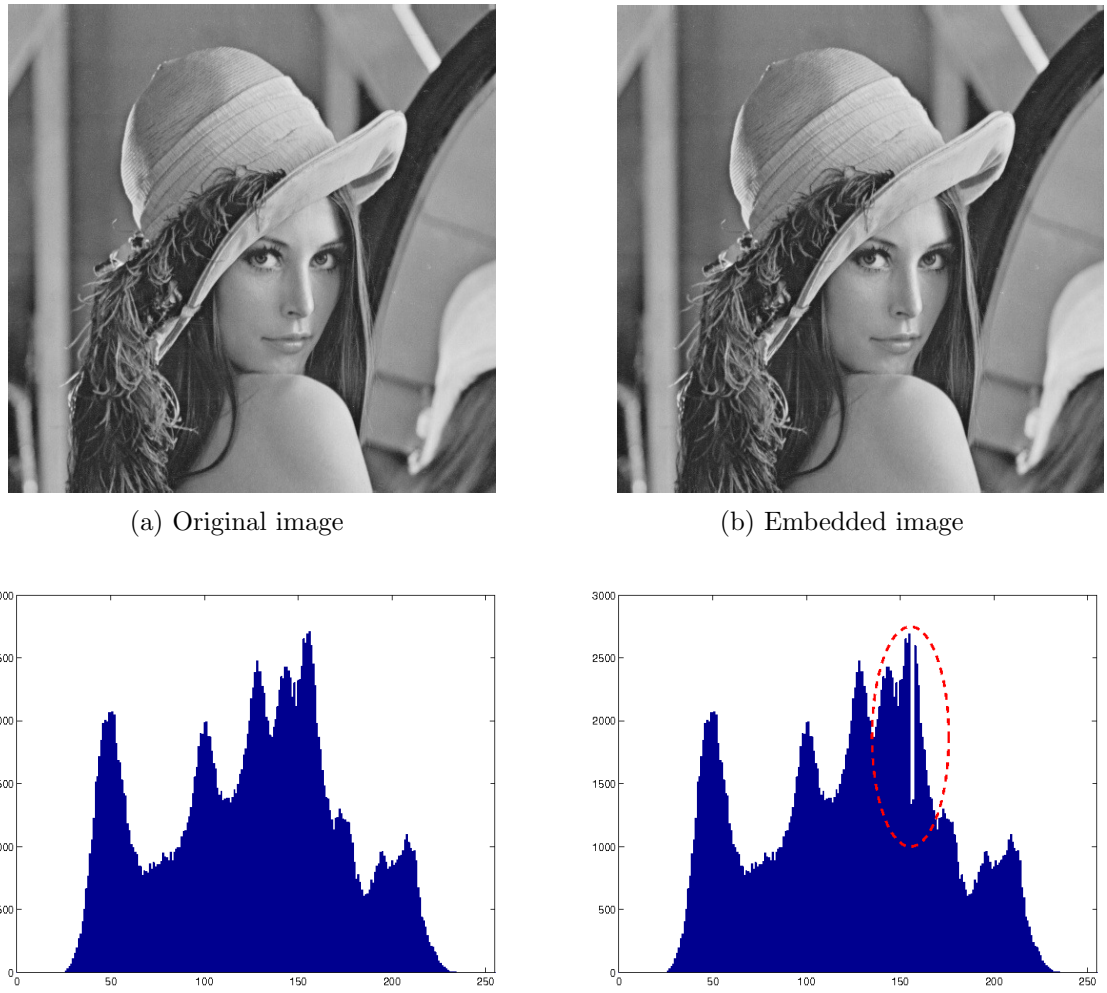
This dissertation is mainly focused in steganography and steganalysis in the spatial domain. Therefore, in the sequel, different techniques that allow hiding information in the bitmap of the image are described.

2.2 Reversible methods

Usually, in data hiding, the methods used for hiding information are non reversible, since we can not recover the original source after extracting the message. In fact, in steganography, the cover object is considered of no value, and recovering the original image is never a requirement. However, there is a family of reversible methods that are very useful when we need to recover the original image, especially in some applications related to watermarking valuable data, such as medical or satellite images.

There exist different reversible methods (Chang et al., 2006; Hong et al., 2008; Honsinger et al., 2001; Hwang et al., 2006; Ni et al., 2006). One of the best known reversible method is that of (Ni et al., 2006). This method uses a technique called *histogram shifting* to embed the secret message. To hide a message using histogram shifting, the first step is to obtain the histogram of pixel intensities from the image and identify a high enough value. Usually, the largest value (peak) in the histogram is chosen. The value of the pixel that produces the peak, say P, is used to hide information. Using the highest peak (or one of the highest peaks) is the best option because the capacity of the system is related to the number of pixels with value equal to P. Once the peak P is located, the second step is to shift the histogram from the value $P + 1$. For doing so, we have to add one to all the pixels of the image that have a value larger than P. The resulting histogram is similar to the first one, but contains a gap in the bin corresponding to the value $P + 1$. This process can be regarded as a "right shift" in the histogram, hence the name of the technique. Of course, we could, alternatively, perform a left shift by subtracting one to the values lower than P instead of adding one to the pixels with values larger than P. Finally, once the histogram is shifted, the message can be hidden in the pixels with value P by adding 1 to embed a binary '1' or leaving the same value to embed a binary '0'. With this procedure, the empty bin $P + 1$ will be partially filled and the method can be easily reverted. The recipient of the message, who knows what the peak is (Hwang et al., 2006) (as part of the secret key) only needs to read the message and restore all the pixels

with value $P+1$ to P. After that, then the histogram has to be shifted back to the left subtracting one to all the pixels of the image with value greater than P, and the original cover image can be completely restored.



(a) Original image

(b) Embedded image

(c) Original histogram

(d) Embedded histogram

Figure 2.1: Lena image and its histogram of pixel intensities before and after embedding using the algorithm proposed in (Ni et al., 2006)

Although Ni et al.'s (2006) embedding procedure embeds the information using the gap left by histogram shifting, this process does not completely restore the shape of the histogram. As a consequence, an artifact appears in the histogram, as it can be easily seen in Fig. 2.1. This artifact is not always easy to detect, since it depends on the hidden information and the original shape of the histogram. However, it is usually easy for an attacker to detect if the histogram has been altered in this way.

The histogram of pixel intensities is not the only one that can be obtained from an image. Actually, there are multiple options. One of these options is to use of the histogram

of prediction errors (Hong et al., 2008). This alternative usually allows embedding more information and does not usually produce evident visible artifacts in the histogram of pixel intensities.

To embed information using the histogram of prediction errors, we need a prediction function. This function provides an estimation of the value of a pixel based on the values of its neighbors. The difference between the real value and our prediction is the prediction error, and, with the values of the prediction errors of all the pixels of an image, we can build a histogram. The predicted value can be computed using a simple method, such as the value of one of the neighbouring pixels as a prediction, but the pit can also be far more complicated. Consider the pixel positions in Fig. 2.2a, where x corresponds to the position of the current pixel and a, b and c to the positions of its neighbors.

a	b
c	x

(a) 2×2 adjacent pixels

a	b	c
d	x	

(b) 2×3 adjacent pixels

Figure 2.2: Pixel locations for prediction equations

Now, we can generate a histogram of prediction errors using a vertical prediction calculating the prediction error as:

$$PE = P_x - P_c,$$

where P_z stands for the value of the pixel in the position z. Similarly, we can use an horizontal prediction error with:

$$PE = P_x - P_b,$$

as well as a diagonal prediction error:

$$PE = P_x - P_a.$$

On the other hand, we can use more sophisticated predictors, such as the median edge detector (MED) used in (Hong et al., 2008):

$$PE = \begin{cases} P_x - \min(P_b, P_c), & \text{if } P_a \geq \max(P_b, P_c), \\ P_x - \max(P_b, P_c), & \text{if } P_a \leq \min(P_b, P_c), \\ P_x - P_b + P_c - P_a, & \text{otherwise.} \end{cases}$$

Another possibility is the causal template prediction (using the positions specified in Fig. 2.2b):

$$PE = P_x - \frac{P_a + P_b + P_c + P_d}{4}.$$

Usually, the value of a pixel is similar to the value of its neighbours. Therefore, it is easy to build a new predictor using a combination of the values of the neighboring pixels. We can use any combination of near pixels and any mathematical operation aiming at obtaining a value that is as similar as possible to the actual value.

Since the prediction error is the result of a subtraction between the real and the predicted values, we expect a positive number if the current pixel value is greater than the predicted value, a negative number if the current pixel is lower than the predicted value, and a zero if the prediction matches the actual pixel value. If the predictor is accurate, we expect to obtain many zeroes as prediction errors, resulting in a symmetric histogram around the zero value with a triangular shape. The height of such a histogram is determined by the quality of the prediction function.

If prediction errors are used, the procedure to embed information is the same as in (Ni et al., 2006), but, in this case, there are some advantages: (1) we usually have more capacity because the histogram of prediction errors is symmetric around zero, which is typically the only peak of the histogram (with a large value); and (2), the prediction function can be part of the key and, if the steganalyzer does not know this function, he/she will not be able to build the histogram and the artifacts will be more difficult to detect. However, it must be taken into account that an attacker may eventually find out the prediction function, as discussed by Kerckhoffs (1883). In any case, these methods will be more difficult to detect compared to using the histogram of the pixel intensities.

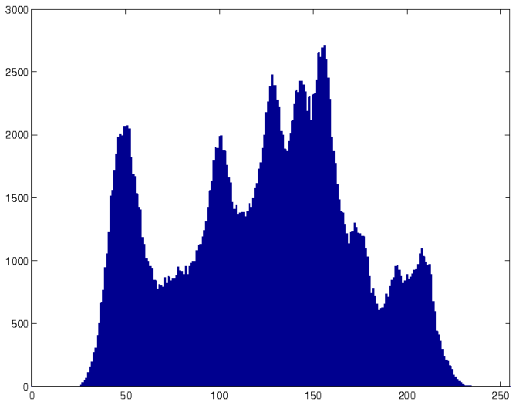
There is some literature about steganalysis for histogram shifting. The usual approach is to detect artifacts in the shape of the histogram. An example of this type of attacks can be found in (Thom et al., 2010), where the authors propose an attack to the histogram shifting method based on the difference image presented by Lee et al. (2004). Another example is the attack to the method of (Hwang et al., 2006) presented by Kuo and Lin (2008).

2.3 Non reversible methods

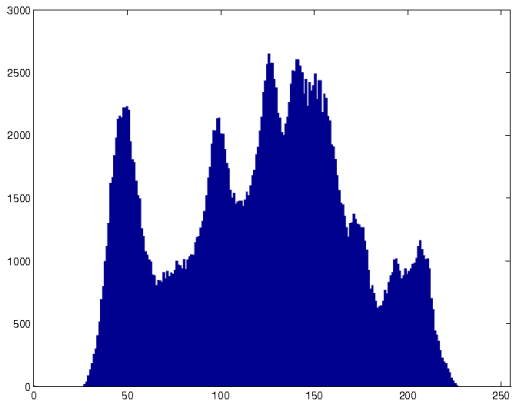
The histogram shifting technique has also been used as a non reversible method (Mohsenzadeh et al., 2009) with the aim of removing the artifacts produced by the method in (Ni et al., 2006). The proposal of Mohsenzadeh et al. (2009) also begins by finding the peak P of the histogram of pixel intensities. The shifting step, in this case, occurs in the two directions. First, the histogram is shifted to the right by adding one to the pixels greater than P and then, the histogram is shifted to the left by subtracting one to the pixels with

(a) Original image

(b) Embedded image



(c) Original histogram



(d) Embedded histogram

Figure 2.3: Lena image and its histogram of pixel intensities before and after embedding using the algorithm proposed by Mohsenzadeh et al. (2009)

value smaller than P. With these operations, two empty bins (gaps) are produced in the histogram. To fill these gaps and to embed the message, the proposed method starts by finding all the pixels with values $P+2$ and $P-2$. To embed '1', the method replaces the value of the pixel on the right of the selected pixel by $P+1$ (or $P-1$). Similarly, to embed '0', the same operation is carried out overwriting the pixel on the left of the selected pixel. With this procedure, the algorithm is generating new $P+1$ and $P-1$ values that fill the gaps left by the histogram shifting operation. Following this procedure, the shape of the histogram is corrected as shown in Fig. 2.3. With a careful visual analysis, we can notice a difference between the two histograms around the peak where the information is hidden. However, since the steganalyst does not have the original image, he/she can not compare both histograms. A visual examination of the resulting histogram does not

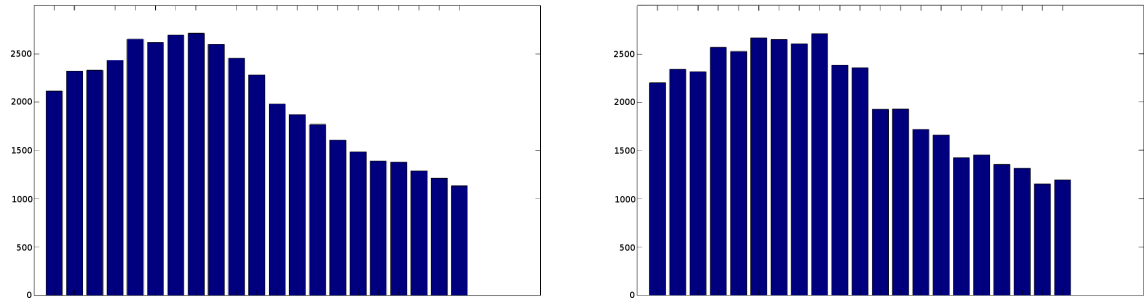
reveal any obvious artifact.

Nevertheless, histogram shifting is not the usual way of embedding information in a non reversible manner. One of the first non reversible techniques used in the spatial domain was least significant bit (LSB) replacement. This technique consists in replacing only the LSB of a pixel with the aim of modifying the original value as little as possible. This type of modification is not perceptible by the human visual system and, thus, we can not appreciate any visual difference in the image. In Fig. 2.4, the Lena image is shown both in original form and after embedding a bit in all its pixels.

(a) Original image (b) Embedded image

Figure 2.4: Lena image before and after embedding using LSB replacement with 1 bit per pixel

However, this slight modification of the pixel values introduces anomalies in the statistics of the image, allowing the detection of the method. The problem with this steganographic technique is that the replacement of the LSB of a given pixel is not a symmetric operation. If the value of the pixel is even, the result of the embedding operation can leave the pixel unchanged or increase its value (+1), whereas if the value of the pixel is odd, the result of the embedding operation preserves the value of the pixel or decreases it (−1). Therefore, an odd value never increases and an even value never decreases. The consequence of this asymmetric operation can be seen in a histogram representing the frequency of each pixel value of the image. If we hide enough information using LSB replacement, since the pixels with even values tend to increase and the pixels with odd values tend to decrease, the bins of the histogram representing consecutive values tend to have the same height. This can be appreciated in Fig. 2.5, especially in the right-hand side bins of the histogram. This is known as the histogram attack as presented by Westfeld and Pfitzmann (2000).



(a) Original histogram (b) Embedded histogram

Figure 2.5: Piece of histogram of the Lena image before and after embedding using LSB replacement

The histogram attack can only be applied if we embed information into all the pixels of the image or if we know which pixels contain hidden information. However, if the information is only hidden in a fraction of the pixels, we need more sophisticated attacks, such as those described by Fridrich et al. (2001) –RS[1] steganalysis– or Dumitrescu et al. (2002) –Sample Pairs Analysis (SPA)–, and their variations.

After the RS attack (Fridrich et al., 2001) and its later improvements, LSB replacement was no longer considered as a convenient method for steganography. The successor of LSB replacement was LSB matching (Sharp, 2001). This method is similar to LSB replacement, but introduces a slight difference: instead of replacing the LSB of the pixel, we increase $(+1)$ or decrease (-1) the value of the pixel by one randomly when the LSB differs from the bit to be embedded. Because of this, LSB matching is also known as ± 1 steganography. With this small modification, the LSB matching algorithm is similar to LSB replacement, but the statistical anomalies of the latter are prevented.

The information embedded using the LSB matching method causes no obvious statistical anomalies, and it is not enough to analyze first order statistics to detect it, requiring more sophisticated steganalityic methods. The tool chosen for dealing with the problem of detecting LSB matching is *machine learning*, which is introduced in Section 2.5.

2.4 Adaptive methods

There are regions of the images that are more difficult to model than others, mainly textures and edges. Taking this into account, new steganographic methods have been developed with the capability to adapt the hiding process to the content of the image, hiding information in the regions where the changes are more difficult to detect. This

[1] "RS" stems from "Regular and Singular groups", but only the initials are used to refer to this method in the literature.

family of methods is referred to as "adaptive".

Adaptive methods are based on the idea of minimizing the impact of embedding. One of the first successful methods using this approach was Highly Undetectable steGO (HUGO) (Pevný et al., 2010b), whose design principle is to minimize a suitably-defined distortion function consisting in a weighted difference of state-of-the-art feature vectors used in steganalysis. Actually, HUGO uses Subtractive Pixel Adjacency Matrix (SPAM) features (Pevný et al., 2010a). Therefore, HUGO can know which regions are prone to hide information, yielding a desirable content adaptivity. This allows HUGO to hide seven times more information than LSB matching with the same level of security (statistical undetectability).

Another method, known as Wavelet Obtained Weights (WOW) (Holub and Fridrich, 2012), address the same idea with a different strategy. The WOW method does not use a model to compute the embedding cost, but applies a bank of directional high-pass filters to obtain what is called *directional residuals*. These residuals are related to the predictability of the pixel in a specific direction. The method measures the impact of embedding on every directional residual and uses these impacts as a cost function forcing a high cost when the content is predictable in at least one direction (smooth areas of the image) and a low cost when the content is not predictable (noisy areas of the image). As a result, the WOW method is more difficult to detect than HUGO using the standard frameworks for steganalysis (Fridrich and Kodovský, 2012; Kodovský et al., 2012). A similar method is the Universal Wavelet Relative Distortion (UNIWARD) (Holub et al., 2014), which uses the same idea as WOW, but it is simpler and suitable for embedding in an arbitrary domain.

2.5 Machine learning in steganalysis

Nowadays, machine learning (Bishop, 2006; Mitchell, 1997) is the preferred tool for steganalysis. Typically, to use machine learning in image steganalysis, we need to represent images as a collection of points in an n-dimensional space by extracting some features from the images and using these features as the coordinates in this space. Once the data points are available, the first step in machine learning is to train a classifier. To do this, we need a training set: a set of data points (representing images) and a set of labels indicating which images are cover and which ones are stego. With this information, the classifier learns how to differentiate between cover and stego images. Later on, the classifier can be used to classify images for which we do not know if they are cover or stego. The performance of the classifier is evaluated using a testing set, i.e. a set of images for which we can verify if the classifier works accurately.

In Fig. 2.6, we can see an example of a two-dimensional data set being classified. The different color tones indicate the probability of classification of the samples in different zones of the diagram.

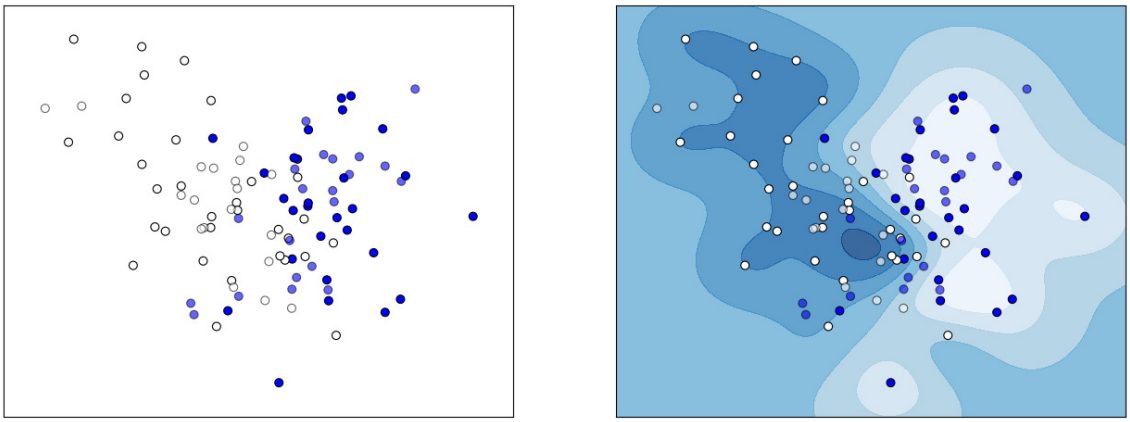


Figure 2.6: Example of a data set and its classification

There are several alternatives to extract features from images. Many feature extractors in the state of the art try to exploit the properties of the cover source and the effects of the embedding algorithm. The Subtractive Pixel Adjacency Matrix (SPAM) (Pevný et al., 2010a) feature vector, based on the ideas presented in (Sullivan et al., 2005; Zou et al., 2006), allowed detecting LSB matching steganography with very high accuracy. The SPAM feature vector is constructed as follows. Consider a grayscale image represented by a $m \times n$ matrix:

$$\{I_{i,j} | I_{i,j} \in \mathbb{N}, i \in \{1, \ldots, m\}, j \in \{1, \ldots, n\}\}.$$

To calculate a feature for the SPAM model, we start by computing the difference matrix D. This matrix can be computed in different directions and this is specified using a superscript $\{\leftarrow, \rightarrow, \uparrow, \downarrow \nwarrow, \searrow, \swarrow, \nearrow\}$. For example, the difference array in a horizontal direction left-to-right is:

$$D^{\rightarrow}_{i,j} = I_{i,j} - I_{i,j+1},$$

for $i \in \{1, \ldots, m\}, j \in \{1, \ldots, n-1\}$. These differences tend to be concentrated around zero and only small values are frequent. Hence, the SPAM model uses a threshold T to accept only the values that fall in the range $[-T, T]$.

The SPAM features are modeled using a Markov process. For the horizontal left-to-right direction this leads to:

$$M^{\rightarrow}_{u,v,w} = P(D^{\rightarrow}_{i,j+2} = u | D^{\rightarrow}_{i,j+1} = v | D^{\rightarrow}_{i,j} = w),$$

where $u, v, w \in [-T, T]$. Finally, the SPAM model assumes that, in natural images, there is a symmetry with respect to mirroring and flipping and it averages the horizontal and vertical matrices as well as diagonal matrices:

$$F_{1,\ldots,k} = \frac{1}{4}[M.^{\rightarrow} + M.^{\leftarrow} + M.^{\downarrow} + M.^{\uparrow}],$$

$$F_{k+1,\ldots,2k} = \frac{1}{4}[M.^{\nwarrow} + M.^{\searrow} + M.^{\swarrow} + M.^{\nearrow}],$$

where $k = (2T+1)^2$ for first-order features and $k = (2T+1)^3$ for second-order features. Usually, SPAM uses $T = 3$, leading to $2k = 686$ second-order features.

Although SPAM performs extremely well for detecting LSB matching, it does not provide a good solution for the detection of modern steganographic algorithms like HUGO (Pevný et al., 2010b). Based on previous methods, (Fridrich et al., 2011a,b; Gul and Kurugollu, 2011), to attack HUGO, a new feature extractor based on the use of multiple models was developed. This feature extractor is called Rich Models (Fridrich and Kodovský, 2012). The goal of Rich Models is to capture a large number of dependencies between neighboring pixels to detect a large number of embedding algorithms. Using a simple model an enlarging it with a higher threshold (T for the SPAM features) is not a good option, because it would have many underpopulated bins (Fridrich et al., 2011a). Therefore, Rich Models attack the problem assembling different submodels formed by joint distributions of neighboring samples from noise residuals obtained using high-pass filters.

Rich Models uses residuals of a type called "'spam" by their similarity to the SPAM (Pevný et al., 2010a) feacture vector, and a type called "minmax" that uses two or more linear filters obtaining the minimum or the maximum to introduce non-linearity. Rich Models also use different residual classes called "1st", "2nd", "3rd", "SQUARE", "EDGE3x3" and "EDGE5x5", depending on the central pixel predictor used. The whole method is too long to be included here and we refer the reader to the original paper (Fridrich et al., 2011a).

There exist many different classifiers used in steganalysis, but, in the last few years, support vector machines (SVM) (Chang and Lin, 2011; Vapnik, 1982, 1995) were one of the most popular choices (Fridrich, 2004; Lyu and Farid, 2002; Pevný and Fridrich, 2007; Pevný et al., 2010a). The fact that there is a significant number of high quality implementations of SVM and their good performance, even when the number of features is larger than the size of the training set, are two of the reasons for this success. However, when the number of features increases and we want to train a classifier using several thousands of images, we need to use more efficient machine learning algorithms because the training time of a SVM could be too long. The classifier proposed to be used with Rich Models is Ensemble Classifiers (Kodovský et al., 2012), which consists of different

simple base classifiers that use a randomly selected subset of the feature space. Ensemble Classifiers make a decision by aggregating the decisions of every base classifier.

Recently, the use of deep learning in steganalysis (Pibre et al., 2016; Qian et al., 2015) is gaining popularity. This is a branch of machine learning that deals with algorithms that use multiple processing layers composed of non-linear transformations. There exist different deep learning architectures such deep neural networks, convolutional deep neural networks, deep belief networks or recurrent neural networks.

2.6 The cover source mismatch problem

When machine learning is used, the classifier obtained with a set of training data will be as good as the training data. If the model is incomplete, the classifier may not work properly. Each data set depends on many factors, such as the steganographic algorithm used for hiding data, the algorithm used for feature extraction or the properties of the cover source in different aspects like size, noise or the hardware (camera) used for acquisition. Therefore, if the training and the testing datasets are similar, the classification usually works accurately, but if very different sets are used, this can produce a significative degradation in the classification results. In steganalysis, this problem was initially reported by Cancelli et al. (2008). Machine learning literature (Bishop, 2006; Mitchell, 1997) refers to this problem as domain adaptation, but in steganalysis it is called cover source mismatch (CSM), being an important challenge in the field (Ker et al., 2013).

In the last few years, we can find different approaches to deal with CSM. In the BOSS competition (Filler et al., 2010), the BOSSrank database –a image database that suffers from CSM– was presented. Some competitors tried to include part of the images of the testing set into the training set (Fridrich et al., 2011a; Gul and Kurugollu, 2011) using an idea called "training on a contaminated database". The images were previously denoised to estimate their cover version and new information was hidden to produce a stego sample.

Later on, in 2012, an approach based on training using a huge number of images with an online classifier was proposed by Lubenko and Ker (2012). In 2013, a universal method was presented in (Pevný and Ker, 2013), in which the authors apply linear projections informed by embedding methods and an anomaly detector, trying to make these projections sensitive to stego content and insensitive to cover variation.

Since 2014, the literature related with the CSM problem has increased significantly. Ker and Pevný (2014) show different methods to deal with the CSM problem. They propose centering features to solve a possible shift, by subtracting an estimated centroid, or to use weighted ensemble methods to deal with a possible displacement in different directions after embedding. In (Kodovský et al., 2014), we find different methods to

deal with CSM: training with a mixture of different sources, the use of different classifiers trained on different sources, and the use of the classifier corresponding to the closest source to the testing set or the closest source for a given testing image. Finally, in (Pasquet et al., 2014), an approach based on Ensemble Classifiers with feature selection (Chaumont and Kouider, 2012) is presented. They use the "Islet approach", a pre-processing step in which images are organized in clusters and a different steganalyzer is assigned to each of them.

Chapter 3

Contributions of the thesis

This chapter provides the different contributions of this dissertation, all of them published in conferences or journals. The versions given in this thesis are not exactly the published ones, because the author does not have permission to redistribute the published papers. However, the text provided is identical to that of the published versions of the papers.

3.1 Steganalytic methods for the detection of histogram shifting data hiding schemes

D. Lerch-Hostalot and D. Megías. Steganalytic methods for the detection of histogram shifting data hiding schemes. In *Actas de la XII Reunión Española sobre Criptología y Seguridad de la Información*, pages 381–386. 2012. ISBN: 978-84-615-9933-2. `http://recsi2012.mondragon.edu/es/programa/recsi2012_submission_06.pdf`.

Abstract

In this paper, several steganalytic techniques designed to detect the existence of hidden messages using histogram shifting schemes are presented. Firstly, three techniques to identify specific histogram shifting data hiding schemes, based on detectable visible alterations on the histogram or abnormal statistical distributions, are suggested. Afterwards, a general technique capable of detecting all the analyzed histogram shifting data hiding methods is suggested. This technique is based on the effect of histogram shifting methods on the "volatility" of the histogram of the difference image. The different behavior of volatility whenever new data are hidden makes it possible to identify stego and cover images.

Steganalytic methods for the detection of histogram shifting data hiding schemes

Daniel Lerch-Hostalot
Universitat Oberta de Catalunya,
Internet Interdisciplinary Institute (IN3),
Estudis d'Informàtica, Multimèdia i Telecomunicació,
Rambla del Poblenou, 156,
08018 Barcelona, Catalonia, Spain,
Email: dlrech@uoc.edu

David Megías
Universitat Oberta de Catalunya,
Internet Interdisciplinary Institute (IN3),
Estudis d'Informàtica, Multimèdia i Telecomunicació,
Rambla del Poblenou, 156,
08018 Barcelona, Catalonia, Spain,
Email: dmegias@uoc.edu

***Abstract*—In this paper, several steganalytic techniques designed to detect the existence of hidden messages using histogram shifting schemes are presented. Firstly, three techniques to identify specific histogram shifting data hiding schemes, based on detectable visible alterations on the histogram or abnormal statistical distributions, are suggested. Afterwards, a general technique capable of detecting all the analyzed histogram shifting data hiding methods is suggested. This technique is based on the effect of histogram shifting methods on the "volatility" of the histogram of the difference image. The different behavior of volatility whenever new data are hidden makes it possible to identify stego and cover images.**

***Index Terms*—Communication security, steganalysis, steganography.**

I. Introduction

Data hiding [16] is a collection of techniques to embed secret data into digital media such that its existence becomes undetectable by some attacking party. Data hiding can be applied to secret communications, copyright protection, authentication of digital contents and other applications. The most common carriers used for data hiding are images because of their widespread use in the Internet.

To hide data into a cover image, pixel values are changed and image distortion occurs. Usually, the distortion due to data hiding is not reversible and the original image can not be recovered. However, there are techniques that have the ability to restore the original image. These techniques are known as reversible data hiding [14], [5], [8], [11], [3], [7], [9], [6].

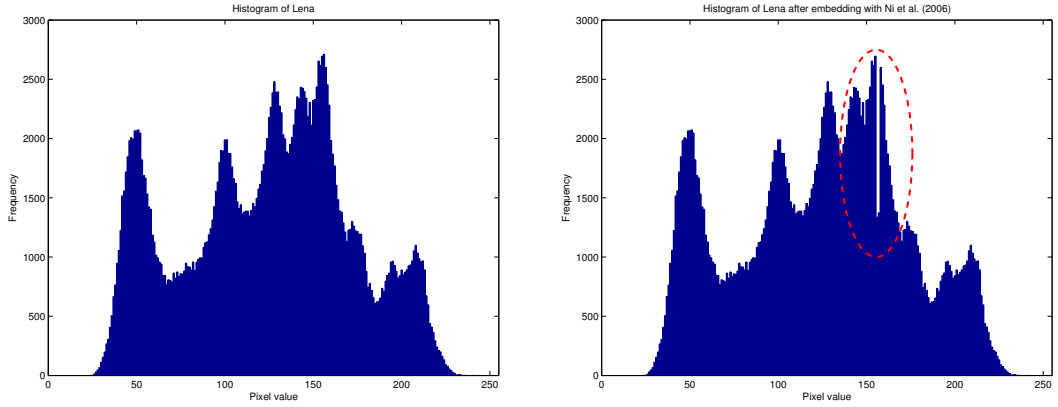
The simplest non-reversible data hiding method consists of modifying the least significant bit (LSB) of some (or all) pixel values, which is often referred to as LSB steganography. In [15], several attacks on LSB steganography are described. Later, in [4], the RS attack is introduced, which can reliably detect messages even for embedding capacities as low as 0.03 bits per pixel (bpp). In general, much work has been devoted to develop steganalytic tools for LSB steganography, LSB matching [12] or JPEG steganography [13], but little attention has been paid to other data hiding strategies, such as the histogram shifting methods analyzed in this paper.

A reversible data hiding method based on histogram shifting was proposed in [11]. This scheme uses the information about peaks and zeros of the histogram of the cover image to perform a partial shift, leaving a gap to hide data. In the Ni *et al.*'s method [11], the embedded secret data cannot be recovered when the knowledge of peak and zero point of histogram are not transmitted to the receiver. In order to overcome the above drawback, Hwang *et al.* [7] proposed a robust reversible data hiding scheme based on the histogram shifting method. This new approach proposes the use of a location map to store the information needed to reverse the process when the minimum point of histogram is non-zero.

Later, in [5], Hong *et al.* presented a scheme which performs a shift of the histogram of prediction errors. This method is based on [11], but has greater capacity. In their paper, Hong *et al.* use the median edge detector (MED) to predict pixel values (as detailed in Section II-C). Since the histogram of prediction errors is sharply centered at zero, we can use the concept of histogram shifting to hide information without determining the peak and zero points, unlike Ni *et al.*'s method. Although the histogram shifting technique is commonly used in reversible data hiding, several methods have recently emerged and are used as non-reversible ones [10].

There is some work on steganalysis applied to histogram shifting methods. Particularly, a few of them perform the detection based on changes in the shape of the histogram. In [14], a technique to attack the method based on shifting the histogram of the difference image of [9] is presented. This technique is based on detecting an unusual shape in the histogram, similar to the attack we present in Section II-A. However, this technique is not applicable to [11]. In [8], a technique to attack the method of [7] is presented. As in the previous case, this technique is based on finding an unusual shape in the histogram but, again, it is not applicable to the histogram shifting method of [11]. In both cases, this irregularity affects seven of the histogram bins. Therefore, it hardly ever occurs in cover (unmarked) images. In [11], the irregularity affects only four bins, making it harder to detect.

This paper presents different steganalytic tools which can be used to detect histogram shifting steganography for Ni *et al.*'s method [11], Mohsenzadeh *et al.*'s method [10] and different histogram shifting of prediction errors techniques, such as that



(a) Histogram of the original Lena image

(b) Histogram of the Lena image marked with [11]

Fig. 1: Histogram comparison

of [5]. In addition, the concept of the "volatility" of a histogram of prediction errors is introduced, and the measurement of this volatility before and after random embedding is shown to be effective to detect all the above histogram shifting data hiding schemes with a significantly high accuracy.

The rest of the paper is organized as follows. Section II proposes four steganalytic techniques. Three of these techniques are specific for particular data hiding schemes, whereas the fourth one is generic and can be applied to any of them. Section III shows the experimental results obtained with the different specific techniques and the generic one. Finally, Section IV draws the conclusions and outlines some guidelines for future research.

II. Proposed Steganalytic Methods

In this section, we present four steganalytic techniques. The first method detects Ni *et al.*'s scheme [11] by finding anomalies in the histogram of pixel intensities. The second steganalytic technique detects Mohsenzadeh *et al.*'s [10] method by searching for an unusual statistical distribution introduced by the embedding algorithm. The third steganalytic technique detects Histogram Shifting of Prediction Errors (HSPE) methods [5] studying the "volatility" of the histogram. Finally, the fourth technique extends the third one to create a generic method which can be applied to detect all of the above.

A. *Ni* et al.*'s Method*

In 2006, Ni *et al.* [11] presented a reversible data hiding method which consists in shifting the histogram of the image in order to create a gap to hide secret data. Their method uses a simple but effective algorithm:

Procedure 1 (Ni et al.*'s method [11]):*

1) Find the maximum (or peak) of the histogram, which corresponds to a pixel value P, and then find a zero to the right of P, which corresponds to the pixel value Z.
2) Shift the histogram to the right, from the peak to the zero point. To do this, all the pixels of the image with values between $P+1$ and $Z-1$ (included) are increased by 1.
3) To embed the message, it is necessary to scan the entire image looking for all the pixels with value P. These pixels are replaced by $P+1$ to embed '1', or keep the same value (P) to embed '0'.

In Fig.1, we can see the histogram of pixel intensities for the grayscale Lena image with 512×512 pixels [2], before and after data have been hidden with the method described in Procedure 1 [11]. If we compare the original histogram with the histogram of the marked image, there is a visible notch caused by histogram shifting and data hiding, as highlighted in Fig.1(b) with a dashed ellipse.

The abnormal shape in the histogram of the marked image can be detected with some reliability by applying the following observations. Let h_i, h_{i+1}, h_{i+2} and h_{i+3} be four consecutive bins of the histogram, then a peak replacement in h_{i+1} can be detected as follows:

1) $h_{i+1} + h_{i+2}$ is greater that any bin of the histogram,
2) h_{i+1} and h_{i+2} are approximately equal and
3) h_i or h_{i+3} are not much smaller than $h_{i+1} + h_{i+2}$.

The last two conditions require some thresholds as detailed in Section III.

B. *Mohsenzadeh* et al.*'s Method*

In 2009, Mohsenzadeh *et al.* [10] presented a steganographic method which is able to thwart histogram based steganalysis. Their method uses histogram shifting techniques to hide non-reversible data with the following algorithm.

Procedure 2 (Mohsenzadeh et al.*'s method [10], Alg. 1):*

1) Find the maximum bin (or peak) of the histogram, which corresponds to a pixel value P, and then find the first

zero to the left (Z_l) and the first zero to the right (Z_r) of the peak.
2) Shift the histogram to the right, from $P+1$ to Z_r-1, and do the same to the left from $P-1$ to Z_l+1. To do this, 1 is added to each pixel of the image with value between $P+1$ and Z_r-1 and 1 is subtracted from each pixel between $P-1$ and Z_l+1 .
3) To embed the message, it is necessary to scan the entire image in zigzag order looking for all pixels $I_{\text{zig}}(i)$ with values $P+2$ or $P-2$. To embed '0', set $I_{\text{zig}}(i-1) := P+1$ (or $P-1$) and, to embed '1', set $I_{\text{zig}}(i+1) := P+1$ (or $P-1$).

Mohsenzadeh *et al.* [10] present a second algorithm that uses a secret key to randomize the position of the modified pixels. For each selected pixel (with value $P+2$ or $P-2$), the message bit is embedded in one of its eight neighboring pixels, rather than using the neighbors in zigzag order as done in Algorithm 1. This variant is referred to as Algorithm 2.

Procedure 2 (or Algorithm 1 as defined in [10]) produces a significant statistical anomaly, since there is always a $P+1$ (or $P-1$) value next to $P+2$ (or $P-2$). For this reason, we can detect hidden data with this algorithm, counting those occurrences next to all the pixels (in a zigzag traversal of the image). If we consider each pixel of the image as a potential $P+2$ or $P-2$, we can check if this pixel has a $P+1$ or $P-1$ neighbor to its left or right (in zigzag order). If so, we can assume that P is a peak candidate and count this occurrence. Statistically, the maximum number of pixels that satisfy this constraint corresponds to the peak used to embed data. Therefore, if we draw a histogram with the frequency of pairs $(P+2, P+1)$ and $(P-2, P-1)$ counted for each peak candidate P, the highest bin corresponds to the true peak P. This is illustrated in Fig.2 for the Lena image, where the peak P occurs for the value 156. Note that this is not a standard histogram of the pixel intensities. Each pixel value is considered to be $P+2$ (or $P-2$) and, if $P+1$ (or $P-1$) occurs next to the pixel, then we increase the bin of P.

Apart for the highest frequency found for P, there is a detectable anomaly in this histogram since the bins of the values $P+1$, $P-1$, $P+2$ and $P-2$ decrease more than what is statistically expectable. We can also observe this fact in Fig.2. This provides a powerful mechanism for identifying stego images marked with this method, since we only have to verify that, given seven consecutive bins of this histogram: h_j for $j = i, i+1, \ldots, i+6$, the following conditions hold:

1) h_i and h_{i+6} are greater than h_{i+1}, h_{i+2}, h_{i+4} and h_{i+5} and
2) h_{i+3} is the greatest value of the histogram.

This technique, which does not require any threshold, is not applicable to Algorithm 2, due the randomized positions for the modified values $P+1$ and $P-1$.

C. Histogram Shifting of Prediction Errors Methods

Histogram Shifting of Prediction Errors (HSPE) methods were presented in [5]. There are many different HSPE data hiding methods and the most popular ones have been selected for the experiments presented in this paper. HSPE schemes obtain a histogram of the differences between the pixels values and a prediction p computed using some prediction equation. This histogram can be used to embed a message using techniques analogous to that of [11].

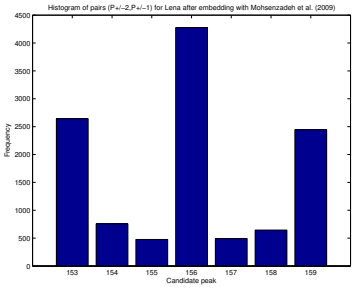


Fig. 2: Frequency of the candidate peaks P for the Lena image marked with [10]

The alterations of histograms of prediction errors are more difficult to detect than those of pixel intensities, since histograms of prediction errors can be generated from different prediction formulas. However, neighboring pixels are often used for this prediction. For example, Hong et al. [5] use the median edge detector (MED) prediction to calculate the predicted value (p) of a pixel x:

$$p = \begin{cases} \min(b,c), & \text{if } a \geq \max(b,c), \\ \max(b,c), & \text{if } a \leq \min(b,c), \\ b+c-a, & \text{otherwise.} \end{cases}$$

where a, b and c are three neighbors of the pixel x, as shown in Fig.3(a).

a	b
c	x

(a) 2×2 adjacent pixels

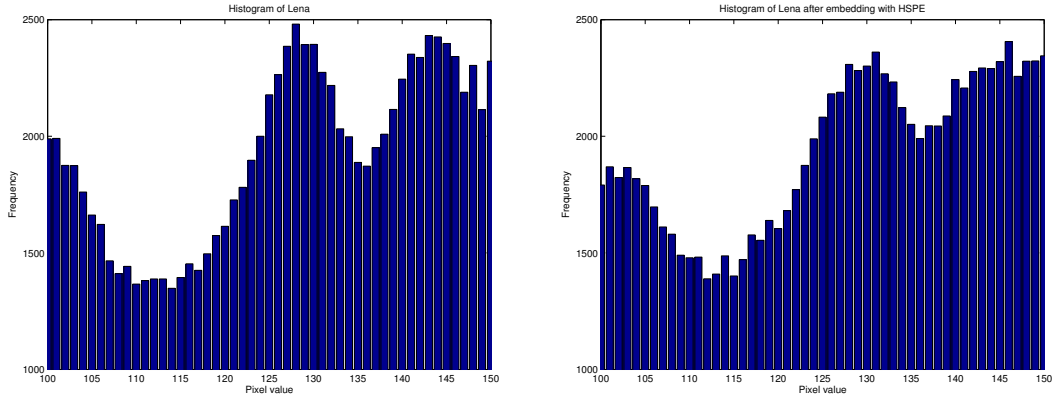
a	b	c
d	x	

(b) 2×3 adjacent pixels

Fig. 3: Pixel locations for prediction equations

There are simpler methods such as horizontal prediction: $p = c$, vertical prediction: $p = b$, diagonal prediction: $p = a$ and others even more sophisticated, such as a causal template prediction, like $p = \lfloor (a+b+c+d)/4 \rfloor$, where a, b, c and d are shown in Fig.3(b) with respect to the pixel x to be predicted and $\lfloor y \rfloor$ stands for the largest integer which is lower than or equal to y.

In this case, it is not possible to analyze the histogram of prediction errors directly, since the specific prediction formula will not be known (in general). Thus, It is necessary to take another approach. One of the common traits of HSPE methods is that they modify areas where the pixels are similar. When similar pixels are modified by adding one, these pixels will advance to the next bin of the histogram. As this occurs



(a) Histogram of the original Lena image

(b) Histogram of the Lena image marked with HSPE

Fig. 4: Histogram comparison before and after HSPE embedding

throughout the histogram, after shifting in the difference space, most bins give some of their pixels to their neighbors, producing a less volatile histogram. This situation is illustrated in Fig.4, where it can be seen that the frequencies of the histogram obtained after HSPE embedding are more similar to those of their neighbors. Therefore, the difference of a bin with respect to the preceding and the succeeding ones decreases in Fig.4(b) as compared to the original histogram shown in Fig.4(a). Although the difference from some bins to their neighbors may increase locally, the total difference from all bins to their neighbors decreases globally.

We can measure the volatility V of the histogram comparing the value of each bin h_i with its neighbors h_{i-1} and h_{i+1} as follows:

$$V = \sum_{i=1}^{254} \left| \frac{h_{i-1} + h_i + h_{i+1}}{3} - h_i \right|,$$

which yields:

$$V = \sum_{i=1}^{254} \left| \frac{h_{i-1} - 2h_i + h_{i+1}}{3} \right|. \quad (1)$$

Expression 1 is presented for clarity, but the following normalized expression has been used in the implementation of the methods:

$$V = \sum_{i=1}^{254} \frac{\max(\widetilde{h_i}, h_i) - \min(\widetilde{h_i}, h_i)}{\max(\widetilde{h_i}, h_i)}, \quad (2)$$

where $\widetilde{h_i} = (h_{i-1} + h_i + h_{i+1})/3$. If $h_{i-1} = h_i = h_{i+1}$, the i-th bin is not included in the computation of V (since it contributes with 0 to the overall volatility).

The experiments show that the volatility of the image histogram of pixel intensities is significantly reduced when a message is embedded into a cover image. However, when it is embedded into a stego image, the volatility is reduced to a smaller extent. This behavior provides a detection mechanism: the volatility of the test image can be compared with the volatility after embedding a new message into it. If this process significantly reduces the volatility, the image is cover, otherwise it is stego. The new message is embedded by choosing a random binary mask with the same dimensions of the test image and adding it to the pixels values. This means that, on average, 50% of the pixel values are increased by 1, whereas the other 50% remains unchanged.

D. Generic Steganalytic Method

The detection technique presented in the previous section is useful because it exploits some common characteristics of different data hiding systems. However, it does not detect the methods introduced by Ni *et al.* [11] or Mohsenzadeh *et al.* [10]. The reason for this is that these schemes affect only a reduced group of values of the frequencies of the histogram, whereas some other frequencies are only shifted. Thus, it becomes necessary to use a different histogram for which most bins are affected by the different data hiding methods introduced above. We have found that a histogram of differences is suitable for this purpose if the following prediction equation is used:

$$p = \left\lfloor \frac{a + b + c}{3} \right\rfloor, \quad (3)$$

where a, b and c are as shown in Fig.3(a). Once all such predictions are computed, a histogram based on the value of the differences $|x - p|$ is obtained.

When analyzing this histogram of differences, volatility goes the opposite way as compared to the histogram of pixel intensities. When embedding data, the volatility of the histogram of differences increases instead of decreasing. However, after embedding a new message into a stego image, the volatility remains almost unchanged.

Now, the method proceeds as described in the previous section. Firstly, the volatility of the histogram of differences,

with the prediction of Expression 3, is calculated as per Expression 2. Then, a new message is embedded (using a random binary mask as described in the previous section) into the image and, finally, the volatility is computed again. If the volatility increases significantly after embedding, then the image is declared cover, otherwise it is detected as stego.

III. Experimental results

In this section, we present the experimental results obtained with all the proposed algorithms. In these experiments, the National Resource Conservation System (NRCS) [1] database of 1371 images has been used. These images have been embedded with the different methods described above. More precisely, a testing set consisting of 2742 images, half of them stego and half of them cover have been used for each of the experiments detailed below (except for the mixed experiments of Section III-D which have more specific settings).

For the generic algorithm, we have used a threshold of 15%. This means that an image is considered stego if its volatility increases less than 15% when embedding a new message, otherwise it is considered cover. The results of the experiments show that this threshold is appropriate.

A. *Ni* et al.*'s Method*

As detailed in Section II-A, this specific detection technique requires two thresholds. The first threshold is to verify that h_{i+1} and h_{i+2} have a similar value. We have used a maximum difference of 10%. The second threshold is to verify that h_i or h_{i+3} are not much smaller than $h_{i+1} + h_{i+2}$. A maximum difference of 30% has been used for this condition. The experiments show that these thresholds are appropriate.

TABLE I: Experimental results for Ni *et al.*'s method [11]

Results	Specific	Generic
Successful	85.19%	85.22%
Positive	40.29%	44.93%
Negative	44.89%	40.29%
False positive	5.10%	9.70%
False negative	9.70%	5.06%

The results are shown in Table I. The row "Successful" refers to the percentage of correctly identified images (either as cover or stego), the row "Positive" reports the percentage of the correctly identified stego images (the maximum is 50%), "Negative" reports the number of correctly identified cover (unmarked) images (again, the maximum is 50%), "False positive" reports the percentage of cover images incorrectly identified as stego and, finally, "False negative" is the percentage of stego images which are not correctly detected by the technique. Note that the number of positives plus false negatives equals 50%. Analogously, the number of negatives plus false positives also equals 50% of the total number of images.

It can be observed that both the specific and the generic methods correctly identify more than 85% of the images. The specific scheme has a higher percentage of false negatives, whereas the generic one has a higher ratio of false positives.

B. *Mohsenzadeh* et al.*'s Method*

For Mohsenzadeh *et al.*'s method with Algorithm 1, the specific algorithm described in Section II-B does not need any threshold, just analyzes the shape of the histogram as shown in Fig.2.

TABLE II: Experimental results for Mohsenzadeh *et al.*'s method [10] – Algorithms 1 and 2

Results	Algorithm 1		Algorithm 2
	Specific	Generic	Generic
Successful	90.99%	81.65%	90.18%
Positive	42.19%	41.35%	49.89%
Negative	48.79%	40.29%	40.29%
False positive	1.23%	9.70%	9.70%
False negative	7.76%	8.64%	0.10%

The results shown in Table II for Algorithm 1 indicate higher scores for the specific algorithm, but the generic method also yields remarkable results. For Mohsenzadeh *et al.*'s method with Algorithm 2, only the generic algorithm has been applied, since the exact position of the neighboring pixels used to embed data is protected by means of a secret key and cannot be used by a specific attack. The results, which are shown in the same table, indicate a large ratio of success using the generic algorithm. In addition, in this case, the reliability of the detection of positives is remarkable, with a 49.89% of success for a maximum of 50%.

C. *HSPE Methods*

For detecting HSPE methods, the specific algorithm presented in Section II-C has been used with a threshold of 15%. This means that an image is considered stego if its volatility decreases less than 15% when embedding a new message, otherwise it is considered cover. The experiments show that this threshold is appropriate.

Table III shows the results for HSPE steganography with five different prediction equations: horizontal, vertical, diagonal, causal and MED. As horizontal prediction is concerned, both the specific and the generic methods obtain similar success ratios, with somewhat better results for the generic one. The same goes to the results for diagonal prediction errors, with an almost identical performance for both methods and a small difference in favor of the generic one. For vertical prediction errors, it can bee seen that the success ratios are, again, similar with both methods, but this time the results are slightly better for the specific one.

As causal prediction is concerned, the results of Table III show that the specific method is particularly unsuitable for this embedding technique using the same thresholds as for the other HSPE methods. By modifying the threshold of the specific method slightly, success ratios of above 80% can be obtained, but this also increases the number of false positives. Finally, when MED prediction errors are used, the results are analogous to those obtained with causal prediction errors. Again, the successful identification ratio with the specific

TABLE III: Experimental results for HSPE data hiding

Results	Horizontal		Vertical		Diagonal		Causal		MED	
	Specific	Generic	Specific	Generic	Specific	Generic	Specific	Generic	Specific	Generic
Successful	86.94%	87.16%	88.84%	87.19%	87.52%	88.65%	61.19%	86.10%	63.78%	85.88%
Positive	43.47%	46.86%	45.36%	46.90%	44.05%	48.35%	17.72%	45.80%	20.31%	45.58%
Negative	43.47%	40.29%	43.47%	40.29%	43.47%	40.29%	43.47%	40.29%	43.47%	40.29%
False positive	6.52%	9.70%	6.52%	9.70%	6.52%	9.70%	6.52%	9.70%	6.52%	9.70%
False negative	6.52%	3.13%	4.63%	3.09%	5.94%	1.64%	32.27%	4.19%	29.68%	4.41%

technique could be improved over 80% by modifying the thresholds at the price of increasing the false positive ratio.

D. Mixed Experiments

In this section, an experiment was performed with 1000 cover and 1000 stego images. The set of stego images contains a mixture of all the presented methods in equal parts. *I.e.*, the 1000 stego images are marked using Ni *et al.*'s method, Mohsenzadeh *et al.*'s Algorithms 1 and 2, and HSPE with horizontal, vertical, diagonal, causal and MED predictions. Hence, eight different embedding methods are used and 125 images are embedded with each method.

TABLE IV: Experimental results for different mixed histogram shifting data hiding methods

Results	Generic
Successful	86.05%
Positive	46.15%
Negative	39.90%
False positive	10.10%
False negative	3.85%

As shown in Table IV, the generic algorithm for a mixture of histogram shifting data hiding schemes yields a successful classification ratio of above 86%.

IV. Conclusions

In this paper, we have shown that histogram shifting-based data hiding schemes cause alterations in the image histogram and that these alterations can be detected. We have introduced a technique, based on the analysis of the histogram's volatility, which can be applied to several data hiding methods. The experimental results show that the analysis of the histogram's volatility can be used to detect relevant changes in the histogram, and that the analysis of the histogram of differences provides remarkable results, being able to identify between 80% and 90% of the test images correctly as cover or stego.

As future work, it would be advisable to study the use of other histograms to estimate volatility, as well as exploring the applicability of this steganalytic technique to other data hiding schemes.

Acknowledgments

This work was partly funded by the Spanish Government through projects TSI2007-65406-C03-03 E-AEGIS, TIN2011-27076-C03-02 CO-PRIVACY and CONSOLIDER INGENIO 2010 CSD2007-0004 ARES.

References

[1] "National Resource Conservation System (NRCS) Photo Gallery," accessed on May 30, 2012. [Online]. Available: http://photogallery.nrcs.usda.gov

[2] "Standard test images," accessed on May 30, 2012. [Online]. Available: http://www.ece.rice.edu/~wakin/images/

[3] C.-C. Chang, W.-L. Tai, and C.-C. Lin, "A reversible data hiding scheme based on side match vector quantization," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 16, no. 10, pp. 1301–1308, oct. 2006.

[4] J. Fridrich, M. Goljan, and R. Du, "Detecting LSB steganography in color, and gray-scale images," *Multimedia, IEEE*, vol. 8, no. 4, pp. 22–28, oct-dec 2001.

[5] W. Hong, T.-S. Chen, and C.-W. Shiu, "Reversible data hiding based on histogram shifting of prediction errors," in *Intelligent Information Technology Application Workshops, 2008. IITAW '08. International Symposium on*, dec. 2008, pp. 292–295.

[6] C. W. Honsinger, P. Jones, M. Rabbani, and J. C. Stoffel, "Lossless recovery of an original image containing embedded data," United States Patent N.: 6,278,791 B1, 2001, accessed on May 30, 2012. [Online]. Available: http://www.freepatentsonline.com/6278791.html

[7] J. Hwang, J. Kim, and J. Choi, "A reversible watermarking based on histogram shifting," in *Digital Watermarking*, ser. Lecture Notes in Computer Science, Y. Shi and B. Jeon, Eds. Springer Berlin / Heidelberg, 2006, vol. 4283, pp. 348–361.

[8] W.-C. Kuo and Y.-H. Lin, "On the security of reversible data hiding based-on histogram shift," in *Proceedings of the 2008 3rd International Conference on Innovative Computing Information and Control*, ser. ICICIC '08. Washington, DC, USA: IEEE Computer Society, 2008, p. 174.

[9] S.-K. Lee, Y.-H. Suh, and Y.-S. Ho, "Lossless data hiding based on histogram modification of difference images," in *Proceedings of the 5th Pacific Rim conference on Advances in Multimedia Information Processing - Volume Part III*, ser. PCM'04. Berlin, Heidelberg: Springer-Verlag, 2004, pp. 340–347.

[10] Y. Mohsenzadeh, J. Mohajeri, and S. Ghaemmaghami, "Histogram shift steganography: A technique to thwart histogram based steganalysis," in *Proceedings of the 2009 Second International Workshop on Computer Science and Engineering - Volume 02*, ser. IWCSE '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 166–170.

[11] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 16, no. 3, pp. 354–362, mar. 2006.

[12] T. Pevný, P. Bas, and J. Fridrich, "Steganalysis by subtractive pixel adjacency matrix," *Information Forensics and Security, IEEE Transactions on*, vol. 5, no. 2, pp. 215–224, june 2010.

[13] T. Pevný and J. Fridrich, "Merging Markov and DCT features for multi-class JPEG steganalysis," in *Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents IX, San Jose, CA, January 29–February 1*, E. Delp and P. Wong, Eds., vol. 6505, January 2007, pp. 03–14.

[14] C. T. H. Thom, H. V. Canh, and T. N. Tien, "Steganalysis for reversible data hiding," *International Journal of Database Theory and Application*, vol. 3, no. 2, pp. 21–30, jun. 2010.

[15] A. Westfeld and A. Pfitzmann, "Attacks on steganographic systems," in *Information Hiding*, ser. Lecture Notes in Computer Science, A. Pfitzmann, Ed. Springer Berlin / Heidelberg, 2000, vol. 1768, pp. 61–76.

[16] W. Zeng, "Digital watermarking and data hiding: technologies and applications (invited talk)," in *Proc. International Conference on Information Systems, Analysis and Synthesis*, vol. 3, 1998, pp. 223–229.

3.2 LSB matching steganalysis based on patterns of pixel differences and random embedding

D. Lerch-Hostalot and D. Megías. LSB matching steganalysis based on patterns of pixel differences and random embedding. *Computers & Security*, 32: 192–206, Feb. 2013. http://dx.doi.org/10.1016/j.cose.2012.11.005.

Abstract

This paper presents a novel method for detection of LSB matching steganography in grayscale images. This method is based on the analysis of the differences between neighboring pixels before and after random data embedding. In natural images, there is a strong correlation between adjacent pixels. This correlation is disturbed by LSB matching generating new types of correlations. The presented method generates patterns from these correlations and analyzes their variation when random data are hidden. The experiments performed for two different image databases show that the method yields better classification accuracy compared to prior art for both LSB matching and HUGO steganography. In addition, although the method is designed for the spatial domain, some experiments show its applicability also for detecting JPEG steganography.

LSB Matching Steganalysis Based on Patterns of Pixel Differences and Random Embedding

Daniel Lerch-Hostalot and David Megías (corresponding author)
Universitat Oberta de Catalunya,
Internet Interdisciplinary Institute (IN3),
Estudis d'Informàtica, Multimèdia i Telecomunicació,
Rambla del Poblenou, 156,
08018 Barcelona, Catalonia, Spain,
E-mail {dlerch,dmegias}@uoc.edu.

Abstract

This paper presents a novel method for detection of LSB matching steganography in grayscale images. This method is based on the analysis of the differences between neighboring pixels before and after random data embedding. In natural images, there is a strong correlation between adjacent pixels. This correlation is disturbed by LSB matching generating new types of correlations. The presented method generates patterns from these correlations and analyzes their variation when random data are hidden. The experiments performed for two different image databases show that the method yields better classification accuracy compared to prior art for both LSB matching and HUGO steganography. In addition, although the method is designed for the spatial domain, some experiments show its applicability also for detecting JPEG steganography.

Keywords: Communication security, steganalysis, steganography, LSB matching, Support Vector Machines.

1. Introduction

Data hiding is a collection of techniques to embed secret data into digital media such that its existence becomes undetectable by an attacking party. These techniques can be used in many different application scenarios, such as secret communications, copyright protection or authentication of digital contents, among others. The most common carriers used for data hiding are images because of their widespread use in the Internet.

Steganography is a major branch of data hiding in which the main goal is secret communication. On the other hand, Steganalysis works in the detection of messages hidden using steganography. A popular steganography technique in spacial-domain images is Least Significant Bit (LSB) replacement. This technique consists of replacing the LSB of each pixel by the bit of the message to be inserted. The LSB replacement technique introduces some anomalies in the histogram of the image that makes it detectable (Westfeld and Pfitzmann, 2000). This anomaly has been successfully exploited by the RS attack, capable to detect LSB steganography even for bit rates as low as 0.03 (Fridrich et al., 2001). LSB matching, also known as ± 1 embedding, is a variation of LSB replacement where the pixel value is increased or decreased randomly to match the bit of the message to be hidden (Sharp, 2001). This technique is more difficult to detect and has become the basis of most state of the art steganographic systems in the spatial domain.

More recently, Pevný et al. (2010) proposed the Highly Undetectable steGO (HUGO) steganographic tool based on ± 1 steganography. HUGO uses a high dimensional model in order to choose between $+1$ and -1 when both possibilities are valid. The choice is made such that the distortion from the cover to the stego model is minimized according to the model. It also uses a Syndrome-Trellis Code (STC) in order to reduce the number of modified pixels. This allows HUGO to be much more undetectable than the standard LSB matching. HUGO can embed up to seven times more data than LSB matching for the same detection accuracy. Although the computational cost of HUGO is very high, it can be assumed that the parties willing to have a secret communication will devote much effort in making the message as undetectable as possible.

Different strategies have been proposed to detect LSB matching methods. Some of these methods are based on analysing the features of the image histogram (Cai et al., 2010; Zhang et al., 2009; Mankun et al., 2008). Cai et al. (2010) show that the peak value of the histogram of the difference image (the differences of adjacent pixels) decreases after LSB matching embedding. However, the renormalized histogram (the ratio of the histogram to the peak value) increases. The peak value and the renormalized histogram are used as features for classification. Zhang et al. (2009) propose a method that uses the fact that the values of local maxima of an image histogram decrease and those of the local minima increase after embedding. The resulting area, that can be seen as an envelope, is smaller than that of a cover image. The authors combine this property with the fact that LSB matching embedding corresponds to low-pass filtering of the histogram producing differences in the higher order statistical moments of high frequencies of the histogram. Mankun et al. (2008) model the LSB matching problem as a kind of image degradation with additive pulse noise proportional to the embedding rate. This method obtains an estimation of the cover image by wavelet denoising and extracts features from the

test images and the estimated ones.

In other methods, the LSB matching embedding is modeled in the context of additive noise (Li et al., 2008; Harmsen and Pearlman, 2003; Ker, 2005). In (Harmsen and Pearlman, 2003), it is assumed that the histogram of the hidden message is a convolution of the noise probability mass function (PMF) and the original function. In the frequency domain, this convolution can be viewed as a multiplication of the histogram characteristic function (HCF) and the noise characteristic function. The method exploits the fact that embedding produces a decrease in the HCF center of mass (HCF COM). Ker (2005) proposes two variations to the Harmsen and Pearlman's method: the calibration of the output using downsampled images and the use of the adjacency histogram. These variations improve the classification reliability substantially. Li et al. (2008) study the application of the method to the difference image (the difference of the adjacent pixels) rather than using the original image.

Other systems analyze the relationship between adjacent pixels (Zhang et al., 2010; Zhao et al., 2010; Pevný et al., 2010). Zhang et al. (2010) propose a method to estimate the number of zero different values using the non-zero difference values according to the LSB matching steganography. The difference between the actual and the estimated value can be used as a feature to detect stego images. On the other hand, Zhao et al. (2010) use a median filter to remove noise and, after that, propose a detector based on a 12-dimensional feature vector constructed counting the different number of overlapping 3×3 blocks of the images formed by the last 2, 3 and 4 bitplanes and the "horizontal sum" image. Currently, one of the most successful methods for LSB matching steganalysis in the state-of-the-art is the Steganalysis by Substractive Pixel Adjacency Matrix (SPAM) method proposed by Pevný et al. (2010). The SPAM method uses first-order (SPAM-1) and second-order (SPAM-2) Markov chains to create a model of the difference of adjacent pixels in natural images. By analyzing the deviations between tested images and these models, the system identifies stego images with the LSB matching technique with very high accuracy: 0.167 error rate for an embedding bit rate of 0.25 bits per pixel (bpp) in the Natural Resources Conservation Service (NRCS) image database (NRCS, n.d) with SPAM-2.

Other successful approaches to detect LSB matching steganography are based on Markov chains and/or discrete cosine transform (DCT) features. Shi et al. (2007) use a Markov process to model the differences between JPEG 2-D arrays along horizontal, vertical and diagonal directions and generate features for training a Support Vector Machine (SVM). Fridrich (2005) introduces a new steganalytic method based on DCT features and uses it for comparing JPEG steganography algorithms and evaluating its embedding mechanisms. Pevný and Fridrich (2007) combine these two methods (DCT and Markov chains features) to create a 274-

dimensional feature vector that is used to train an SVM. The resulting method provides significantly more reliable results compared to previous works. Kodovský and Fridrich (2009) propose a modified calibration procedure that improves the practical results of the steganalyzer in (Pevný and Fridrich, 2007).

Most of the methods reported above are of the "single-model" type. They basically compute a low-dimensional set of features (typically less than 1000) which can be used to classify images as stego or cover using some classification technique, such as SVM. More recently, new steganalytic methods based on combining different models (rich models) have been proposed (Fridrich and Kodovský, 2012) and the efficient ensemble classifiers have been suggested to replace SVM (Kodovský et al., 2012). Methods based on rich models built a high-dimensional set of features (up to 34,761 features in (Fridrich and Kodovský, 2012)) from which sub-models with less features (around 3000) can be selected to be used for classification much more efficiently. The objective of this paper is not to overcome the results of these high-dimensional techniques, but to present a new low dimensional set of features (a single model) that overcomes the results of other single-model steganalizers. The suggested features may be incorporated into these new rich models to improve their performance. It must be remarked that the computational burden associated to rich models' methods are much higher than that of a single-model technique and, thus, their use in real-time applications will require powerful hardware. The main aim of this paper is to improve the current state-of-the-art of less computationally demanding methods.

This paper proposes a new single-model steganalytic method based on the detection and counting of patterns of pixel differences (PPD) before and after random data embedding. Henceforth, the name PPD is used to refer to the proposed system. The basic idea is to gather significant information about neighboring pixels and use this information to create patterns. After that, the number of patterns of the test image are counted and random data is embedded into the test image. Then, the number of patterns after random embedding is counted again. This procedure makes it possible to analyze the behavior of the patterns with different noise conditions. The relation between the number of patterns in each case is significantly different if the test image is cover or stego. Hence, the ratio in the number of patterns before and after embedding can be used as a vector of features in an SVM (Steinwart and Christmann, 2008) successfully. By training this SVM, it is possible to distinguish cover images from stego ones reliably. The performance of PPD is shown for both standard LSB matching and for its more undetectable version provided by HUGO. In addition, the system is shown to also detect steganography in the transform domain, such as the block DCT used in JPEG.

The rest of the paper is organized as follows. Section 2 defines the concept of patterns of pixel differences and shows the distribution of these patterns in typical

images. Section 3 analyzes the effect of random data embedding on the variation of these patterns and shows that different behavior occurs for stego and cover images. This result is the basis of the proposed method. Section 4 describes the proposed algorithm for the extraction of the PPD features. Section 5 presents the experimental results obtained with the proposed PPD method and compares them with the state-of-the-art SPAM method (Pevný et al., 2010) for LSB matching, HUGO and JPEG steganography. In the latter case, the comparison is presented not only for SPAM but also for the Merged features steganalyzer (Pevný and Fridrich, 2007). Finally, the most relevant concluding remarks and directions for future research are summarized in Section 6.

2. Patterns of Pixel Differences

The proposed method consists of analysing the different number of patterns of pixel differences before and after random data embedding. Please note that this random data embedding is part of the testing method to obtain the features used in the SVM. This means that the test image, either stego or cover, is analysed and an embedding step is applied. After this embedding, the PPD patterns are obtained again and compared with those before embedding. Since the variation in the number of PPD patterns is greater for cover than stego images, the proposed method provides with a reliable system to detect LSB matching steganography.

2.1. Computation of Patterns of Pixel Differences

Given a block of 3×3 pixels, we can arrange it into two horizontal pairs, two vertical pairs and four diagonal pairs. Taking into account the pixel distribution illustrated in Fig.1, the horizontal pairs are formed by (x_{22}, x_{21}) and (x_{22}, x_{23}), the vertical pairs are formed by (x_{22}, x_{12}) and (x_{22}, x_{32}), and the diagonal pairs by (x_{22}, x_{11}), (x_{22}, x_{13}), (x_{22}, x_{31}) and (x_{22}, x_{33}).

Creating patterns of pixels gathering all this information would be unpractical due to its high dimensionality. If patterns of 9 pixels for grayscale images with a color depth of 8 bpp are created, 256^9 different patterns would be required, which is an exceedingly high number. However, not all the combinations of pixels are equally probable. For example, in an image, it is not usual to find a pixel with intensity 255 next to (or even close to) a pixel with intensity 0, i.e. the neighboring pixels tend to have similar values. If the difference between neighbors is used to create patterns, values will be typically small. We can, thus, define a threshold or limit for the difference between neighboring pixels similar to the one proposed in (Pevný et al., 2010). Let $d(x, y)$ be the limited difference between two neighboring

pixels x and y defined as follows:

$$d(x,y) = \begin{cases} S-1, & \text{if } |x-y| > S-1, \\ |x-y|, & \text{if } |x-y| \leq S-1, \end{cases} \tag{1}$$

where S represents the number of possible values of the pixel differences. This way, all pixel differences are limited to the values $0, 1, \ldots, S-1$ (*i.e.* S different values).

Even in this case, the number of patterns obtained with most of the values for S is still too large. For this reason, we need to reduce the number of pixels involved in the patterns. At this point, we can exploit the fact that part of the pairs contain redundant information because of their intrinsic symmetry. For example, the pair (x_{22}, x_{23}) has similar information to (x_{22}, x_{21}), the pair (x_{22}, x_{12}) has similar information to (x_{22}, x_{32}) and so on. It is thus possible to neglect part of this redundant information and consider only one horizontal, one vertical and two diagonals, as shown in Fig.2. This corresponds to the pairs (x_{22}, x_{12}), (x_{22}, x_{13}), (x_{22}, x_{23}) and (x_{22}, x_{33}), yielding a reduced block $B = \{x_{12}, x_{22}, x_{13}, x_{23}, x_{33}\}$.

If one of the pixels of the block is chosen as a reference, after this reduction and the limited difference of Expression 1, the number of patterns is reduced from 256^9 to S^4, which is a significant decrease. This approach can provide more information if more than one reference pixel is used for each block. For example, if the block of pixels $B = \{110, 111, 110, 113, 112\}$ is considered, we can use the minimum value $b = x_{12} = 110$ and subtract it from the rest. In this case, the pattern generated using x_{12} as a reference and $S = 3$ is the tuple $(0, 1, 0, 2, 2)$. Similarly, we can use the maximum value $b = x_{23} = 113$ as a reference to obtain the pattern $(2, 2, 2, 0, 1)$ by subtracting the pixel values of the block from 113 and limiting the difference to $S - 1 = 2$. Note that these tuples have always at least one zero (the reference pixel corresponding to the maximum or minimum, denoted as b) which does not need to be represented nor stored, whereas the remaining four values of pixel differences are in the range $[0, S-1]$, yielding S^4 possible patterns.

Since the reference pixel b can be chosen from any of the positions x_{12}, x_{13}, x_{22}, x_{23} and x_{33} in the block, a specific order must be defined for the other four pixels of the block to construct the pattern. The chosen order is depicted in Fig.3 for all possible locations of the reference pixel b. The arrow pointing to the "center" of the table is used to define the "left" (l), "upper left" (ul), "upper right" (ur), and "right" (r) neighbors of the reference pixel. The pixel located to the left of the arrow is chosen to be l, the one to the right is chosen to be r, and the remaining pixels (ul and ur) are defined by considering the clockwise path from l to r. The chosen order provides a unique form to obtain the four values of the tuple (pattern) for each reference pixel.

The pattern P is finally obtained from the tuple (l, ul, ur, r) as depicted in Fig.4. Let b be the value used as a reference, $P[1..4]$ an array (or row vector, if more formal language is used) that forms the pattern and N an array of neighbors sorted such that the value used as a reference lies in the center (Fig.3), *i.e.* $N[1] = l$, $N[2] = ul$, $N[3] = ur$ and $N[4] = r$. Then the pattern array P can be obtained as follows:

$$P[i] = d(b, N[i]), \text{ for } i = 1, \ldots, 4.$$

where d is the limited difference of neighboring pixels as defined by Expression 1 for some value of S.

With the same example values provided above, $B = \{x_{12} = 110, x_{13} = 111, x_{22} = 110, x_{23} = 113, x_{33} = 112\}$, using the maximum and minimum pixel values as references, two different patterns can be obtained, namely, $P_{\min}$ for the minimum $x_{12} = 110$ and using the pixel positions shown in Fig.3(a): $(l, ul, ur, r) = (x_{13}, x_{23}, x_{33}, x_{22})$, and $P_{\max}$ for the maximum value $x_{23} = 113$ and using the pixel positions shown in Fig.3(d): $(l, ul, ur, r) = (x_{33}, x_{22}, x_{12}, x_{13})$. This results in the patterns $P_{\min} = [0, 2, 2, 1]$ and $P_{\max} = [1, 2, 2, 2]$. Note that there are other three (or even four if the maximum and minimum pixel values are identical) other possible selections of the reference pixel are possible. In the example, the patterns obtained with x_{13}, x_{22} and x_{33} are not considered. These other patterns are discarded since they do not provide any improvement in the results obtained with the proposed steganalytic method.

Choosing the pixel orders instead depicted in Fig.3 instead of using a fixed sorting like

$$x_{12}, x_{13}, x_{22}, x_{23}, x_{33}$$

and removing the zero in the reference point b is a convenient way to take advantage of the symmetries and spatial properties of the patterns. For example, with the values of pixels of Fig.5, the pattern obtained taking the minimum value (23) as a reference is the same in both cases (a) and (b). For Fig.5(a), the pattern is constructed using Fig.3(e), whereas for Fig.5(b), the pattern is built using Fig.3(b). In both cases, the obtained pattern is identical $P_{\min} = [1, 2, 2, 1]$. This is consistent with the fact that the 24 values are closer to the minimum (23) in both blocks, whereas the 25 pixel values are further from the minimum (23). The block shown in Fig.3(b) can be considered as a rotated version of the block of Fig.3(a), and the steganalytic method suggested in the paper obtains better results if both blocks produce the same pattern. Note that if the fixed order $x_{12}, x_{13}, x_{22}, x_{23}, x_{33}$ was used, two different patterns would be obtained: $P'_{\min} = [2, 2, 1, 1]$ for Fig.3(a) and $P'_{\min} = [1, 2, 1, 2]$ for Fig.3(b). Hence, two spatially analogous situations would produce two different patterns, which would not be convenient for classification purposes.

Finally, the obtained patterns are arrays $[P[1], P[2], P[3], P[4]]$, where each component $P[i]$ (for $i = 1, 2, 3, 4$) is in the range $[0, S-1]$. Now, each pattern can be uniquely mapped to an integer if the components of P are taken to be digits of a number expressed in the basis S, with $P[1]$ being the most significant digit and $P[4]$ the least significant one, as follows:

$$M(P, S) = P[1]S^3 + P[2]S^2 + P[3]S + P[4] + 1, \tag{2}$$

where 1 is added in this definition to provide values strictly greater than 0 (between 1 and S^4) in such a way that they can be directly used to select a component of the array of features (T) in Algorithm 2 (Section 4). We assume that the first component of an array is indexed by 1 (as in MATLAB) instead of 0 (as in C).

With this mapping function and $S = 3$, the patterns $P_{\min} = [0, 2, 2, 1]$ and $P_{\max} = [1, 2, 2, 2]$ found for the previous example can be mapped to:

$$M([0, 2, 2, 1], 3) = 0 \cdot 3^3 + 2 \cdot 3^2 + 2 \cdot 3 + 1 + 1 = 26,$$
$$M([1, 2, 2, 2], 3) = 1 \cdot 3^3 + 2 \cdot 3^2 + 2 \cdot 3 + 2 + 1 = 54.$$

2.2. Pattern Distribution

Prior to discussing the idea of using these pattern counters as features to discriminate stego and cover images, this section analyzes how these patterns are distributed in images. After that, Section 3 analyzes how these patterns behave when random data are embedded in cover and stego images so that they can be used to construct features to classify images in a steganalyzer.

Firstly, we analyze the distribution of patterns in a random situation where all pixel values are uniformly distributed, which is not a realistic case, but may be useful to understand the frequency of each pattern in real images. The pattern frequencies obtained with $S = 4$ (256 different patterns) for a hypothetical image containing all possible pixel blocks of Fig.2, where $B = \{x_{12}, x_{22}, x_{13}, x_{23}, x_{33}\}$ are in the range $[0, 7]$, is shown in Fig.6(a). Ideally, the pixel value range would have to be $[0, 255]$ for 8 bpp grayscale images. However, this would mean generating $256^5 = 2^{40} > 10^{12}$ different blocks, which is a extremely large number that would require a very long CPU time. In addition, it is not likely that neighboring pixels in the same block have all possible values. The differences between pixel values within a block will mostly be limited to a much lower number, and the range $[0, 7]$ (which limits such differences to up to 7) is a more realistic model. By limiting the pixel value range, the number of blocks is reduced to $8^5 = 32{,}768$, which makes it possible to obtain the ideal pattern frequencies with a much shorter CPU time.

It can be seen that, for example, patterns with $M(P,4) \in \{64, 128, 192, 256\}$ have higher values than others. These mapped values correspond to the patterns $[0,3,3,3]$, $[1,3,3,3]$, $[2,3,3,3]$ and $[3,3,3,3]$. Since pixel differences higher than 3 are limited to 3, it is obvious that, in the case that all pixel blocks appear with the same frequency in the image, patterns having a greater number of components equal to the limit (3) are more frequent than other patterns. In particular, the pattern $P_{256} = [3,3,3,3]$ (with $M(P_{256},4) = 256$) is the one showing the highest frequency.

The frequency of patterns for the Lena 512×512 grayscale image with a color depth of 8 bpp, provided by the Image Communications Lab at UCLA (Wakin, 2003), is shown in Fig.6(b). The vertical axis of the plot has been clipped (the value for the 256th pattern is larger than $4 \cdot 10^4$) to make it possible to see more details in the figure. The peaks in patterns with $M(P,4) \in \{64, 128, 192, 256\}$ are still noticed, but the behavior is not as regular as that of Fig.6(a) due to the use of a real image with a non-uniform distribution of the pixel block values.

The main idea behind the use of the pattern counters introduced in this section is that the variation of these counters after embedding random data is significantly different in stego and cover images, as shown in Section 3.

3. Random Data Embedding

This section analyses the effect of random data embedding in the pattern counters of both cover and stego images. The following algorithm has been used for random embedding. Consider the test image $I = [p_{i,j}]$ for $i = 1, 2, \ldots, H$ and $j = 1, 2, \ldots, W$, where $p_{i,j}$ are the pixel values, H is the image height and W is the image width. LSB matching steganography with a bit rate of 1 bpp can be implemented as follows:

Algorithm 1 (Random data embedding).

For $i := 1$ *to* H*,*
  For $j := 1$ *to* W*,*
  (a) *Compute a pseudo-random number* r_1 *with uniform distribution in the interval* $[0,1]$.
  (b) *If* $r_1 < 0.5$ *then let* $b_{i,j} := 0$ *and, otherwise, let* $b_{i,j} := 1$.
  (c) *If* $\mathrm{mod}(p_{i,j}, 2) = b_{i,j}$ *then let* $p'_{i,j} := p_{i,j}$*; else let* r_2 *be another pseudo-random number with uniform distribution in the interval* $[0,1]$. *If* $r_2 < 0.5$ *then let* $p'_{i,j} := p_{i,j} + 1$ *else let* $p'_{i,j} := p_{i,j} - 1$.

Where $\mathrm{mod}(\cdot,\cdot)$ is the remainder of the integer division. This way, a stego image $I' = [p'_{i,j}]$ is obtained, where each pixel of the stego image contains an embedded bit using the ± 1 technique.

The behavior of pattern counters is illustrated in Fig.7. The histogram of pattern counters, with $S = 4$, for the grayscale cover image NRCSAK97001 of the NRCS database (NRCS, n.d) is shown in Fig.7(a) and, after embedding random data with a 1 bpp bit rate and ± 1 steganography (Algorithm 1), the patterns change as shown in Fig.7(b). It can be seen that some of the patterns are less frequent after embedding, especially those which have components with smaller values (some components lower than the limit $S - 1 = 3$). The ratio between both histograms is shown in Fig.8(a), where we notice that some of these ratios are quite greater than 1 (even around 3), specially for the first half of the patterns.

The experiment is repeated with different embedding information using, again, Algorithm 1. Hence, Fig.7(c) is almost identical to Fig.7(b), since they correspond to a similar situation. After a new data embedding (with the same settings as in the other cases) the resulting histogram of patterns is shown in Fig.7(d), and the ratio between them is shown in Fig.8(b). The values of these ratios are much more regular around one than those obtained in Fig.8(a) for a cover image. In Fig.8(b), the peak points are not as high and the valleys are not as deep as those in Fig.8(a). Thus, this example illustrates how the PPD method can extract features which will be used to discriminate cover and stego images.

3.1. Pattern Shift After Random Data Embedding

As introduced above, the key element of the proposed PPD scheme is the embedding of a random message in the test image (using ± 1 steganography and 1 bpp embedding rate) to detect an atypical behavior in the variation of the pattern counters which makes it possible to discriminate stego from cover images. This section deepens this study by grouping the patterns in such a way that a more detailed analysis is obtained.

Firstly, let us define the "maximum distance" concept for a pixel block or a pattern. Given the parameter S, which limits the value of the pixel differences as described above, the maximum distance D of a pixel block is defined as the maximum component (or infinity norm) of either $P_{\max}$ or $P_{\min}$. It can be easily seen that, for a block of pixels $B = \{x_{12}, x_{22}, x_{13}, x_{23}, x_{33}\}$, the maximum distance D satisfies

$$D = \|P_{\min}\|_\infty = \|P_{\max}\|_\infty = \min(S - 1, \max_{x \neq y}\{|x - y|, x, y \in B\}),$$

where $\|\cdot\|_\infty$ is the maximum absolute value of the components of a vector. Given S, this makes it possible to classify blocks of pixels (or patterns) in S classes referred to as d-patterns: patterns with maximum distance $D = d$. For $S = 4$, we have 0-patterns, 1-patterns, 2-patterns and 3-patterns.

Fig.9 shows an example of pixel blocks producing 1-patterns (a), 2-patterns (b) and 3-patterns (c). For $S = 4$, the block pixel of Fig.9(a) produces $P_{\min} = [1, 1, 0, 0]$ and $P_{\max} = [0, 1, 1, 1]$ (1-patterns), Fig.9(b) produces $P_{\min} = [2, 1, 0, 2]$ and $P_{\max} = [1, 2, 0, 2]$ (2-patterns) and Fig.9(b) produces $P_{\min} = [2, 2, 3, 1]$ and $P_{\max} = [2, 3, 1, 1]$ (3-patterns).

Now we can analyze how these patterns change in case of random embedding using Algorithm 1. We have performed 1000 independent random bit insertions in the blocks of pixels of Fig.9 and counted the resulting patterns. Note that the 1000 experiments are always carried out with the patterns of Fig.9, *i.e.* the 1000 random embeddings are not subsequent.

Since each block of pixels produces two patterns, after 1000 embedding experiments we obtain 2000 patterns which represent the different possibilities to shift the patterns of Fig.9 to other patterns. The results are shown in Fig.10, where each bar represents the number of 0-patterns, 1-patterns, 2-patterns and 3-patterns obtained after 1000 random embeddings. Although these results are particular for the patterns of Fig.9 , the same behavior has been observed for all patterns with the same maximum difference. It can be seen that, after random embedding, the highest probability for 1-patterns is to become 2-patterns or 3-patterns, although some probability exists for them to remain as 1-patterns and a limited shift to a 0-pattern also occurs. For 2-patterns, the highest probability is to shift to a 3-pattern, although a significant chance of remaining as 2-pattern exists (and a smaller one to become a 1-pattern). As 3-patterns are concerned, they most possibly remain as 3-patterns, although some shifting occurs to 2-patterns or (less likely) to 1-patterns. Hence, random embedding will tend to produce patterns with greater maximum differences compared to cover images. When an image is already stego, some of its patterns have already been shifted to higher differences and, thus, the pattern shifting from lower maximum differences will be not so large as that of cover images. This is the basis of the detection method suggested in this paper.

The analysis is completed by applying random embedding to a real image, the grayscale cover image NRCSAK97001 of the NRCS database (NRCS, n.d). The results (again for $S = 4$) are shown in Fig.11. The black bar is used for the cover image, and the gray bars represent random embedding to the previous image. Hence, the white bar represents the results after three subsequent random embeddings. It can be seen that the number of 1-patterns decreases abruptly after the first data insertion. After subsequent data insertions the decrease continues, but not so sharply. As the number of 2-patterns are concerned, after the first data insertion, there is an increase (the second bar for 2-patterns is higher than the first one) mostly produced by some 3-patterns that are shifted to 2-patterns. This effect is relevant since the number of 3-patterns is much higher than those of 0, 1 and 2-patterns. In subsequent data embeddings, the number of 2-patterns decreases

instead of increasing. This means the number of 2-patterns that shift to 1 or 3-patterns is greater than the number of 1 or 3-patterns that shift to 2-patterns. This behavior is steady in successive data insertions. Finally, with regard to 3-patterns, their number increases after each data insertion, though the increase is smaller as more insertions occur.

In short, the behavior of patterns of pixel differences after random embedding provides with much information which can be exploited in the form of features by a classifier to detect stego images. For a cover image, the comparison between the black and the dark gray bars indicates that 1-patterns will decrease sharply, 2-patterns may increase slightly and 3-patterns will show a large increase after random data insertion. For a stego image, the comparison between the dark gray and the light gray bars points out that 1-patterns will decrease, but not so sharply as for a cover image, 2-patterns will most possibly decrease (although this depends on the embedding rate of the stego image) and 3-patterns will increase, but to a lesser extent compared to cover images.

4. Proposed Method

The algorithm to construct a vector of features to identify stego images with the suggested PPD method using an SVM is as follows. Consider the test image $I = [p_{i,j}]$ as defined in the previous section.

Algorithm 2 (Extraction of the PPD features).

1. *Let T be an array to store the number of times that each pattern occurs in the image I. First, initialize this array with zeroes as follows: for $k := 1$ to S^4 set $T[k] := 0$.*
2. *For $i := 2$ to $H - 1$,*
   (a) *For $j := 1$ to $W-1$, build the block of neighbors $\{x_{12}, x_{13}, x_{22}, x_{23}, x_{33}\}$ for $x_{22} = p_{i,j}$ as shown in Fig.2.*
   (b) *Obtain the patterns of pixel differences $P_{\max}$ and $P_{\min}$ for this block using its maximum and minimum pixel values as a reference b respectively (limited to $S - 1$ as per Expression 1).*
   (c) *Increase $T[M(P_{\max}, S)]$ and $T[M(P_{\min}, S)]$ by one using the pattern array to integer mapping $M(\cdot, \cdot)$ defined in Expression 2.*
3. *Embed random data at the test image producing a modified image $I' = [p'_{i,j}]$ for $i = 1, 2, \ldots, H$ and $j = 1, 2, \ldots, W$ using Algorithm 1.*
4. *Let T' be an array to store the number of times that each pattern occurs in the image I'. Initialize this array with zeroes as follows: for $k := 1$ to S^4, set $T'[k] := 0$.*

5. *For* $i := 2$ *to* $H - 1$,
   (a) *For* $j := 1$ *to* $W-1$, *build the block of neighbors* $\{x_{12}, x_{13}, x_{22}, x_{23}, x_{33}\}$ *for* $x_{22} = p'_{i,j}$ *as shown in Fig.2.*
   (b) *Obtain the patterns of pixel differences* $P'_{\max}$ *and* $P'_{\min}$ *for this block using its maximum and minimum pixel values as a reference* b *respectively (limited to* $S - 1$ *as per Expression 1).*
   (c) *Increase* $T'[M(P'_{\max}), S]$ *and* $T'[M(P'_{\min}, S)]$ *by one using the pattern array to integer mapping* $M(\cdot, \cdot)$ *defined in Expression 2.*
6. *For* $k := 1$ *to* S^4 *the feature array* $F[k]$ *is defined as the ratio between the counters of patterns of pixel differences before and after embedding random data,* i.e.

$$F[k] := T[k]/T'[k].$$

   These features $F[k]$ *can be used in an SVM to discriminate between cover and stego images.*
7. *Finally the feature array is normalized to produce values between 0 and 1. The maximum and minimum values of the feature array are obtained as follows:*

$$\alpha := \min_{k=1,\ldots,S^4} \{F[k]\},$$
$$\beta := \max_{k=1,\ldots,S^4} \{F[k]\}.$$

   Then the normalized features can be computed, for $k := 1$ *to* S^4*, as*

$$\widetilde{F}[k] := \frac{F[k] - \alpha}{\beta - \alpha}.$$

Note that the third step of the algorithm requires embedding random data into the test image. We have observed that embedding data in stego images produces significantly different patterns of pixel differences from those of cover images. This is exploited by the proposed PPD steganalyzer to discriminate between stego and cover images. In this step, ± 1 steganography is used to embed random bits in the test image with a 1 bpp embedding rate. This embedding process is, thus, a step of the analysis of the test image in the suggested PPD method. It must be noted that this step is **always** carried out with Algorithm 1 irrespective of the steganographic method to be detected. Hence, the experiments shown to detect HUGO and JPEG steganography in Section 5 also embed random data using LSB matching steganography with 1 bpp of embedding rate.

In addition, it is worth pointing out that two normalization steps are carried out in the feature array. The sixth step does not only compute the relationship

between the counters of patterns for the test I and the embedded I' images, but it also normalizes the values of the features since the counters themselves are not uniformly distributed in the space $\{0, 1, \ldots, S^4\}$ (as shown in Fig.6). This ratio makes it possible to compensate the different intrinsic frequencies obtained for different patterns. The second normalization occurs in the last step, which produces features in the interval $[0, 1]$ for all images.

5. Experimental Results

5.1. Detection of LSB Matching Steganography

In this section, an experimental evaluation of the suggested PPD method for two different databases is presented. The first database consists of 1000 grayscale images with a fixed size of 2100×1500 pixels from the NRCS database (NRCS, n.d). The second database consists of 1000 grayscale images with variable sizes about 4000×2500 pixels from the Break Our Steganographic System! (BOSS) (Filler et al., 2010) contest. Half of these images have been chosen randomly, hiding data using LSB matching with different bit rates. The results are also compared to those of the subtractive pixel adjacency matrix (SPAM) method (Pevný et al., 2010), possibly the best available single-model steganalyzer for LSB matching steganography.

We have used a classifier based on SVM with a Gaussian Kernel. This classifier must be adjusted to provide optimal results. In particular, the values of the parameters C and γ must be adjusted. These values should be chosen to give the classifier the ability to generalize. In the comparison with other steganalyzers, these parameters have been fixed in a neutral manner using the LIBSVM tools (Chang and Lin, 2012) to choose the optimal values. The process has been performed as described in (Hsu et al., 2010). For all the experiments of the paper, we have used cross-validation on the training set using the following multiplicative grid for C and γ:

$$C \in \left\{2^{-5}, 2^{-3}, 2^{-1}, 2^{1}, 2^{3}, \ldots, 2^{15}\right\},$$
$$\gamma \in \left\{2^{-15}, 2^{-13}, 2^{-11}, \ldots, 2^{-1}, 2^{1}, 2^{3}\right\}.$$

This grid is even more exhaustive than the one used in (Pevný et al., 2010) for SPAM (except some very small values for C).

Prior to the experiments, the two sets are divided into a training and a testing set of equal sizes, with the same number of cover and stego images. Thus, it is ensured that the images in the testing set were not used in any form during the training process or conversely.

The results of the proposed PPD algorithm and those of the SPAM method are shown in Tables 1 and 2 respectively. The SPAM method has been used with

Table 1: Results for the proposed PPD method with $S = 4$

PPD method (Proposed)							
Database	**Bit rate**	**Accuracy**	**TP**	**FP**	**TN**	**FN**	**Training**
NRCS	100%	99.00%	496	6	494	4	100.0%
NRCS	50%	90.90%	460	51	449	40	100.0%
NRCS	25%	79.10%	393	102	398	107	93.60%
BOSS	100%	100.0%	500	0	500	0	100.0%
BOSS	50%	99.90%	500	1	499	0	100.0%
BOSS	25%	98.90%	495	6	494	5	100.0%

Table 2: Results for the SPAM method (Pevný et al., 2010)

SPAM method							
Database	**Bit rate**	**Accuracy**	**TP**	**FP**	**TN**	**FN**	**Training**
NRCS	100%	95.90%	496	37	463	4	99.20%
NRCS	50%	83.20%	445	113	387	55	88.40%
NRCS	25%	68.00%	388	208	292	112	69.00%
BOSS	100%	99.60%	499	3	497	1	99.90%
BOSS	50%	98.80%	498	10	490	2	98.70%
BOSS	25%	96.90%	500	31	469	0	96.60%

Table 3: CPU time for the PPD (proposed) and SPAM (Pevný et al., 2010) methods

Method	User time (s)	System time (s)	CPU time (s)
PPD (proposed)	18.373	0.872	19.245
SPAM	29.002	2.440	31.442

second order Markov chains ("2nd SPAM" as denoted in (Pevný et al., 2010)) and $T = 3$ which is the optimum value, yielding 686 different features. On the other hand, the PPD method has been implemented with $S = 4$ which produces 256 features.

The information provided in the different columns of these tables is the following: name of the database, bit rate of data embedding, accuracy of the classification (percentage of correctly classified images), true positives (TP), false positives (FP), true negatives (TN), false negatives (FN) and classification accuracy with the training set. Logically, the accuracy obtained with the training set is always greater than that obtained with the testing set. It can be seen, from the results of both tables, that the proposed PPD algorithm performs better than SPAM in all the cases. Please note that there are exactly 500 cover and 500 stego images in the experiments. Hence, TP + FN $= 500$ and TN + FP $= 500$ always.

In Fig.12, we can see the Receiver Operating Characteristic (ROC) curves that compare the proposed method with the SPAM method using the NRCS set with 1 bpp (a), 0.5 bpp (b) and 0.25 bpp (c). The curves obtained with the BOSS database are too close to the ideal characteristic to compare both methods. It can be seen that the ROC curves of the proposed method are better than those of the SPAM method for all tested embedded bit rates. For the case of 1 bpp embedding bit rate, the abscissa axis has been limited to $[0, 0.25]$ to magnify the details of both curves.

Another key issue for comparison, apart from the quality of the classification, is CPU time. The proposed PPD method needs 256 features (for $S = 4$), less than half the 686 required by the SPAM method (2nd. order and $T = 3$). Table 3 shows the "user", "system" and total "CPU" times obtained for computing the features of the same 20 images with both methods. It can be noticed that the CPU time required by the SPAM method is more than 50% larger than that required by the proposed PPD one. These results have been obtained with an Intel Core 2 Duo CPU P8600 processor with 2.40 GHz and 4 Gb of RAM memory.

In short, these experiments show that the proposed PPD method outperforms the SPAM method for LSB matching steganography using less CPU time, which is quite remarkable since SPAM is possibly the best single-model steganalysis tool for LSB matching steganography in the literature.

5.2. Tuning Analysis and Low Embedding Rates

The objective of this section is twofold. On the one hand an analysis of the effect of the parameter S in the detection accuracy of the proposed method is presented. On the other hand the accuracy of the method for low embedding rates (between 0.05 and 0.20 bpp) is discussed.

The results for 0.05, 0.1, 0.15, 0.20, 0.5 and 1 bpp are shown in Fig.13 for $S \in [2, 9]$. All the experiments have been performed with the NRCS database. For each embedding ratio, a training set of 1000 images, 500 of which are stego and 500 of which are cover, is created. The tests are carried out for a different set of 1000 images, again with 500 stego and 500 cover ones. It can be seen that the best results are obtained for $S = 3$ (solid, 'o'), $S = 4$ (solid, '*'), $S = 5$ (solid, '+'). Among these three values of S, $S = 4$ performs better than the other two options for low embedding ratios (lower than 0.2 bpp), and is always very close to the best option for the other embedding ratios. Of all the tested values for the parameter S, $S = 4$ is the one producing the best overall results.

It should be noted that increasing the value of S beyond 5 does not improve the accuracy of the analysis. For $S = 6$ (dashed, $\triangledown$), $S = 7$ (dashed, $\square$), $S = 8$ (dashed, $\triangle$) and $S = 9$ (dashed, $\diamond$) the accuracy results are worse than those for $S \in [3, 5]$, especially for low embedding ratios. The only exception is $S = 6$ for 1 bpp, which produces accuracy results nearly as good as the best one, which is obtained for $S = 5$. It can be noticed that $S = 9$ is even the worse choice for bit ratios of 0.15 bpp or less. This indicates that no advantage is obtained of using a large number of patterns ($9^4 = 6561$) when the embedding ratio is low. Finally, $S = 2$ (which leads only to $2^4 = 16$ patterns) does not provide enough detection accuracy, being the worst choice for relatively high embedding ratios (over 0.2 bpp).

This figure also shows that the detection accuracy (with $S = 4$ or $S = 3$) is quite reliable even for embedding bit rates as low as 0.15 bpp. For 0.15 bpp the detection accuracy with $S = 4$ is close to 70%, whereas an embedding ratio of 0.2 bpp can be detected with almost 75% accuracy. For bit rates of 0.1 bpp or lower, the accuracy drops quite abruptly and more sophisticated techniques should be used.

Obviously, as S increased, more features must be computed, but the number of patterns is always the same (exactly two patterns for each pixel). As CPU times for obtaining the feature vectors are concerned, the variation for different values of S is really small. For obtaining the features of 2000 images (1000 for training and 1000 for testing), the CPU times vary just about a 10% from the best to the worst case: the value of S producing the fastest results requires about 90% the CPU time of that of the slowest results. Since, for each pixel, two patterns are obtained before and

Table 4: Results for the proposed PPD method and SPAM for HUGO

Bit rate (bpp)	**Analyzer**	**Accuracy**	**TP**	**FP**	**TN**	**FN**	**Training**
1.00	SPAM	84.90%	443	94	406	57	87.00%
	PPD	97.70%	489	12	488	11	99.80%
0.80	SPAM	64.10%	436	295	205	64	64.10%
	PPD	82.40%	408	86	414	92	92.80%
0.65	SPAM	50.90%	138	129	371	362	51.00%
	PPD	66.50%	345	180	320	155	78.40%
0.50	SPAM	50.50%	373	368	132	127	50.20%
	PPD	50.10%	161	160	340	339	51.00%

after random embedding, the threshold S does not introduce significant variations in the computations of the features. However, as more features are obtained, the SVM classifier takes longer to complete a test. The test time for a kernel SVM (as the ones used in this paper) is linear in the number of features. Hence, testing an image with $S = 6$ (1296 features) takes roughly 5 times the testing time with $S = 4$ (256 features).

5.3. Detection of HUGO Steganography

As mentioned in Section 1, the HUGO (Pevný et al., 2010) steganographic system uses a complex model to reduce the success ratio of steganalyzers designed for LSB matching. The accuracy of PPD (with $S = 4$) and SPAM-2 (with $T = 3$) for images embedded with HUGO is analyzed in this section.

The results obtained for 1000 images in the NRCS database are shown in Table 4. A training set of 1000 images and a testing set of another 1000 images have been created for each embedding bit rate. It can be seen that the suggested PPD method outperforms SPAM for all the experiments and that the detection accuracy with the suggested PPD analyzer is quite remarkable for large enough bit rates. For low bit rates (0.5 bpp) or less, the HUGO steganographic tool is undetectable with PPD.

A graphical comparison for the PPD (with $S = 4$) and SPAM-2 (with $T = 3$) analyzers for both LSB matching (± 1) and HUGO steganalysis with different embedding ratios is given in Fig.14. It can be seen that the proposed PPD method ('*') outperforms SPAM ('o') for all cases, and the difference in favor or PPD is even more significant for HUGO. With SPAM, HUGO is already undetectable for bit rates of 0.65 bpp, whereas PPD is able to detect it with more than 50% accuracy for all tested bit rates larger than 0.5 bpp.

Table 5: Accuracy results for JPEG steganography for the PPD (proposed), SPAM (Pevný et al., 2010) and Merged features (Pevný and Fridrich, 2007) methods

Embedding algorithm	Bit rate	PPD (proposed)		SPAM		Merged features	
		Accuracy	Training	Accuracy	Training	Accuracy	Training
StegHide	20%	85.20%	94.80%	51.30%	56.80%	99.60%	100.0%
StegHide	10%	69.10%	94.00%	49.40%	51.40%	99.60%	99.90%
F5	20%	98.30%	100.0%	95.50%	95.00%	100.0%	100.0%
F5	10%	97.80%	99.90%	94.10%	94.70%	100.0%	100.0%
JP Hide&Seek	20%	94.00%	99.60%	81.40%	84.00%	99.50%	100.0%
JP Hide&Seek	10%	91.40%	93.80%	76.00%	80.40%	99.50%	99.80%
Perturbed Quant.	20%	99.80%	100.0%	100.0%	100.0%	99.60%	99.90%
Perturbed Quant.	10%	99.70%	100.0%	100.0%	100.0%	99.60%	99.80%

Table 6: True positive, false positive, true negative and false negative results for JPEG steganography for the PPD (proposed), SPAM (Pevný et al., 2010) and Merged features (Pevný and Fridrich, 2007) methods

Embedding algorithm	Bit rate	PPD (proposed)				SPAM				Merged features			
		TP	FP	TN	FN	TP	FP	TN	FN	TP	FP	TN	FN
StegHide	20%	435	83	417	65	254	241	259	246	497	1	499	3
StegHide	10%	336	145	355	164	319	325	175	181	496	0	500	4
F5	20%	493	10	490	7	496	41	459	4	500	0	500	0
F5	10%	492	14	486	8	479	38	462	21	500	0	500	0
JP Hide&Seek	20%	480	40	460	20	496	182	318	4	498	3	497	2
JP Hide&Seek	10%	467	53	447	33	484	224	276	16	498	3	497	2
Perturbed Quant.	20%	499	1	499	1	500	0	500	0	499	3	497	1
Perturbed Quant.	10%	499	2	498	1	500	0	500	0	498	2	498	2

5.4. Detection of JPEG Steganography

Like SPAM (Pevný et al., 2010), the proposed PPD method has been conceived to detect spatial domain steganography and, more precisely, LSB matching. However, SPAM is proved to be quite successful also for the detection of steganography for JPEG images, i.e., steganography based on the transform domains, such as the discrete cosine transform (DCT) which is the basis of the JPEG compression algorithm.

This section is devoted to show the performance of PPD against different JPEG steganographic tools, namely StegHide (Hetzl and Mutzel, 2005), F5 (Westfeld, 2001), JP Hide & Seek (Latham, 1999) and Perturbed Quantization (Fridrich et al., 2004) for 10% and 20% embedding bit rates.

In order to test the performance of the proposed PPD algorithm for JPEG

steganography, classification experiments have been carried out using the BOSS (Filler et al., 2010) database, with 1000 training images (500 stego and 500 cover) and 1000 test images (500 stego and 500 cover). Tables 5 and 6 show the accuracy, true positives, false positives, true negatives and false negatives results obtained with three different steganalyzers, namely the proposed PPD method with $S = 4$, SPAM-2 with $T = 3$ (Pevný et al., 2010) and Merged features (Pevný and Fridrich, 2007), which is the best performing state-of-the-art analyzer for JPEG images. Unsurprisingly, the best accuracy results are obtained for the latter method, but it can be seen that the proposed PPD algorithm outperforms SPAM for all embedding strategies but Perturbed Quantization (Fridrich et al., 2004), for which SPAM performs even better than the Merged features analyzer with a very small difference between all the tested tools. It can be seen that the suggested PPD approach produces over 90% accuracy in all cases except the StegHide method (Hetzl and Mutzel, 2005).

The reason why the suggested PPD method also detects JPEG steganography is the fact that embedding data in the DCT domain (as JPEG steganography does) also implies changes in the distribution of patterns in the spatial domain. This is illustrated in Fig.15 for the NRCSAK97001 image. This figure shows the variation in 0, 1, 2 and 3-patterns after embedding random data with LSB matching steganography for the cover image (JPEG-compressed with quality 85) and the stego image (JPEG compression with quality 85 and 20% data insertion rate using StegHide steganography). It can be seen that the sign of the variation are the same in both cases: the number of 0-patterns decreases (negative variation), whereas the number of 1, 2 and 3-patterns increases (positive variation). However, there are subtle differences in the amount of these variations, which are more noticeable for 1-patterns and 3-patterns. The increase of 3-patterns is somewhat lower for the stego image compared to the cover one, whereas the increase in 1-patterns is greater for the stego image. Although the differences are subtle in this aggregated form, they provide the SVM classifier with enough information to detect JPEG steganography. Note that the SVM uses 256 features, one per each pattern, and not only the four groups shown in Fig.15.

6. Conclusions

This paper presents the PPD method to detect LSB matching steganography using patterns of differences of neighboring pixels together with (additional) random data embedding. The obtained results show that the proposed PPD method is able to outperform SPAM which is possibly the most accurate state-of-the-art single-model steganalytic tool for LSB matching detection in the literature. The

PPD method is also shown to yield better results than SPAM also for the HUGO steganographic tool.

Furthermore, the proposed method stands out for its simplicity since it just uses pixel differences and a specific order of pixels. In addition, the number of features is quite small compared to other techniques. The key idea of the suggested method is that the pattern distribution before and after random data embedding significantly differs from cover and stego images, which is exploited for SVM classification. This reduced number of features is also translated into reduced computation time compared to SPAM (SPAM requires 50% more CPU time than the proposed PPD approach). The PPD method, which was initially designed to detect LSB matching steganography, is also illustrated to produce remarkable results for JPEG steganography, outperforming SPAM for the tested embedding schemes, but still inferior to the results of the state-of-the-art Merged features steganalyzer.

For future research, the inclusion of the PPD features into the recent rich-model techniques is a possible continuation line. In addition, the idea of using the difference between the test image before and after random data embedding may be extended to use it in the transform domain in order to improve the results of JPEG steganography.

Acknowledgments

This work was partly funded by the Spanish Government through projects TSI2007-65406-C03-03 “E-AEGIS”, TIN2011-27076-C03-02 “CO-PRIVACY” and CONSOLIDER INGENIO 2010 CSD2007-0004 “ARES”.

We would like to thank the authors of SPAM, HUGO, JP Hide&Seek, StegHide, F5, Perturbed quantization and the Merged features techniques for having the code available for download.

References

Cai, K., Li, X., Zeng, T., Yang, B., Lu, X., 2010. Reliable histogram features for detecting LSB matching. In: Image Processing (ICIP), 2010 17th IEEE International Conference on. pp. 1761–1764.

Chang, C.-C., Lin, C.-J., 2012. LIBSVM - A Library for Support Vector Machines. `http://www.csie.ntu.edu.tw/~cjlin/libsvm`, accessed on September 2012.

Filler, T., Pevný, T., Bas., P., 2010. Break our steganographic system (boss). `http://exile.felk.cvut.cz/boss/`.

Fridrich, J., 2005. Feature-based steganalysis for JPEG images and its implications for future design of steganographic schemes. In: Fridrich, J. (Ed.), Information Hiding. Vol. 3200 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 67–81.

Fridrich, J., Goljan, M., Du, R., 2001. Detecting LSB steganography in color, and gray-scale images. Multimedia, IEEE 8 (4), 22–28.

Fridrich, J., Goljan, M., Soukal, D., 2004. Perturbed quantization steganography with wet paper codes. In: Proceedings of the 2004 workshop on Multimedia and security. MM&Sec'04. ACM, New York, NY, USA, pp. 4–15.

Fridrich, J., Kodovský, J., 2012. Rich models for steganalysis of digital images. Information Forensics and Security, IEEE Transactions on(To appear).

Harmsen, J., Pearlman, W. A., 2003. Steganalysis of additive noise modelable information hiding. In: Delp, E., Wong, P. (Eds.), SPIE, Electronic Imaging, Security and Watermarking of Multimedia Contents V. Vol. 5020. SPIE, Santa Clara, CA, pp. 131–142.

Hetzl, S., Mutzel, P., 2005. A graph-theoretic approach to steganography. In: Proceedings of the 9th IFIP TC-6 TC-11 international conference on Communications and Multimedia Security. CMS'05. Springer-Verlag, Berlin, Heidelberg, pp. 119–128.

Hsu, C.-W., Chang, C.-C., Lin, C.-J., 2010. A practical guide to support vector classification. `http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf`, accessed on September 2012.

Ker, A., 2005. Steganalysis of LSB matching in grayscale images. Signal Processing Letters, IEEE 12 (6), 441–444.

Kodovský, J., Fridrich, J., 2009. Calibration revisited. In: Proceedings of the 11th ACM workshop on Multimedia and security. MM&Sec '09. ACM, New York, NY, USA, pp. 63–74.

Kodovský, J., Fridrich, J., Holub, V., 2012. Ensemble classifiers for steganalysis of digital media. Information Forensics and Security, IEEE Transactions on 7 (2), 432 –444.

Latham, A., 1999. JP Hide&Seek. `http://linux01.gwdg.de/~alatham/stego.html`, accessed on September 2012.

Li, X., Zeng, T., Yang, B., 2008. Detecting LSB matching by applying calibration technique for difference image. In: Proceedings of the 10th ACM workshop on Multimedia and security. MM&Sec '08. New York, NY, USA, pp. 133–138.

Mankun, X., Tianyun, L., Xijian, P., 2008. Steganalysis of LSB matching based on histogram features in grayscale image. In: Communication Technology, 2008. ICCT 2008. 11th IEEE International Conference on. pp. 669–672.

NRCS, n.d. National Resource Conservation System (NRCS) Photo Gallery. http://photogallery.nrcs.usda.gov, accessed on September 2012.

Pevný, T., Bas, P., Fridrich, J., 2010. Steganalysis by subtractive pixel adjacency matrix. Information Forensics and Security, IEEE Transactions on 5 (2), 215–224.

Pevný, T., Fridrich, J., 2007. Merging Markov and DCT features for multi-class JPEG steganalysis. In: Delp, E., Wong, P. (Eds.), Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents IX, San Jose, CA, January 29-February 1. Vol. 6505. pp. 03–14.

Pevný, T., Filler, T., Bas, P., 2010. Using high-dimensional image models to perform highly undetectable steganography. In: Böhme, R., Fong, P., Safavi-Naini, R. (Eds.), Information Hiding. Vol. 6387 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 161–177.

Sharp, T., 2001. An implementation of key-based digital signal steganography. In: Moskowitz, I. (Ed.), Information Hiding. Vol. 2137 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 13–26.

Shi, Y., Chen, C., Chen, W., 2007. A Markov process based approach to effective attacking JPEG steganography. In: Camenisch, J., Collberg, C., Johnson, N., Sallee, P. (Eds.), Information Hiding. Vol. 4437 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 249–264.

Steinwart, I., Christmann, A., 2008. Support Vector Machines, 1st Edition. Springer.

Wakin, M., 2003. Standard test images. http://www.ece.rice.edu/~wakin/images/, accessed on September 2012.

Westfeld, A., 2001. F5–a steganographic algorithm (high capacity despite better steganalysis). In: Moskowitz, I. (Ed.), Information Hiding. Vol. 2137 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 289–302.

Westfeld, A., Pfitzmann, A., 2000. Attacks on steganographic systems - breaking the steganographic utilities ezstego. In: Jsteg, Steganos, and S-Tools - and Some Lessons Learned, Lecture Notes in Computer Science. Springer-Verlag, pp. 61–75.

Zhang, J., Hu, Y., Yuan, Z., 2009. Detection of LSB matching steganography using the envelope of histogram. Journal of Computers 4 (7).

Zhang, T., Li, W., Zhang, Y., Ping, X., 2010. Detection of LSB matching steganography based on the laplacian model of pixel difference distributions. In: Image Processing (ICIP), 2010 17th IEEE International Conference on. pp. 221–224.

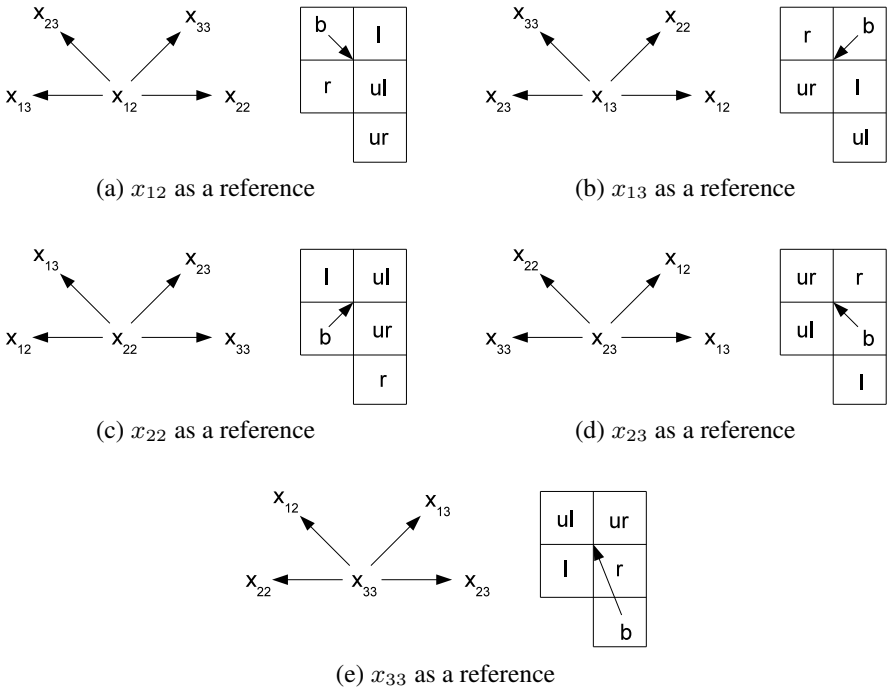
Zhao, Y., Ping, X., Wang, R., Zheng, E., 2010. Steganalysis for LSB matching by counting image blocks with the same gray level. In: Signal Processing (ICSP), 2010 IEEE 10th International Conference on. pp. 1845–1848.

x_{11}	x_{12}	x_{13}
x_{21}	x_{22}	x_{23}
x_{31}	x_{32}	x_{33}

Figure 1: Block of 3×3 pixels

x_{12}	x_{13}
x_{22}	x_{23}
	x_{33}

Figure 2: Block reduction



(a) x_{12} as a reference

(b) x_{13} as a reference

(c) x_{22} as a reference

(d) x_{23} as a reference

(e) x_{33} as a reference

Figure 3: Pattern positions generated from different reference points

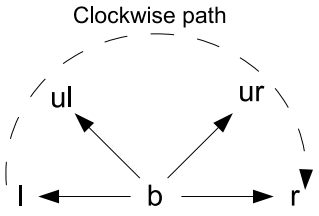
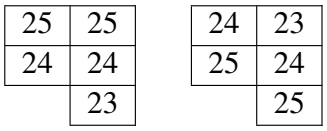
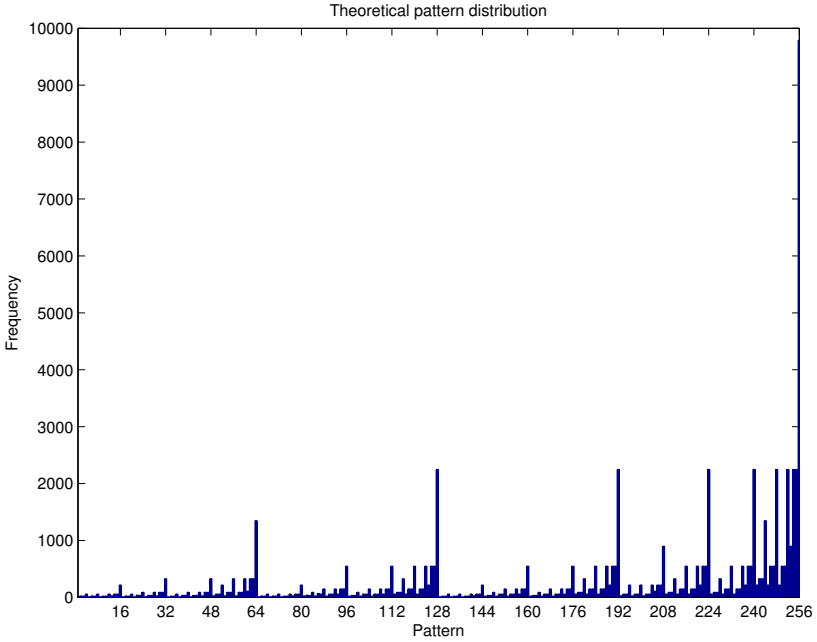


Figure 4: Pattern structure

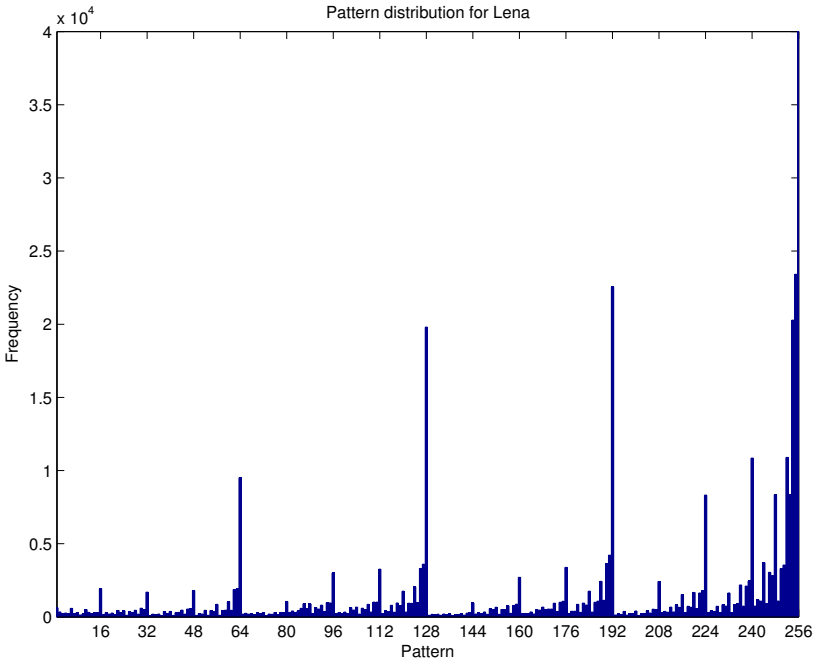


(a) (b)

Figure 5: Example of symmetric patterns

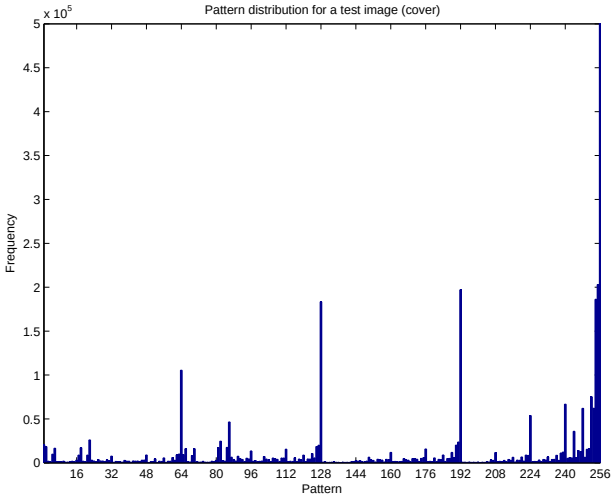


(a) Theoretical uniform distribution

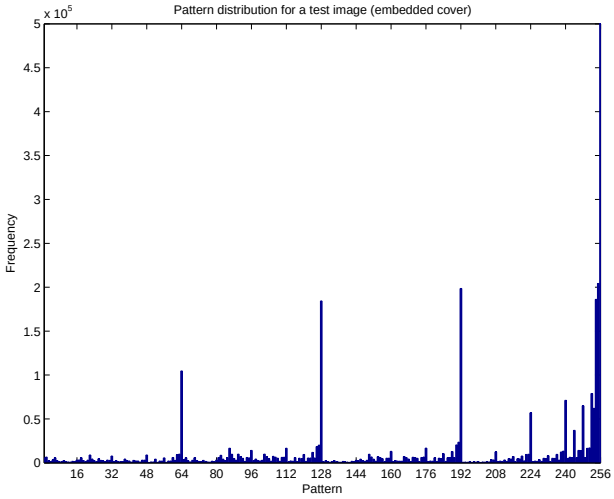


(b) Distribution for Lena

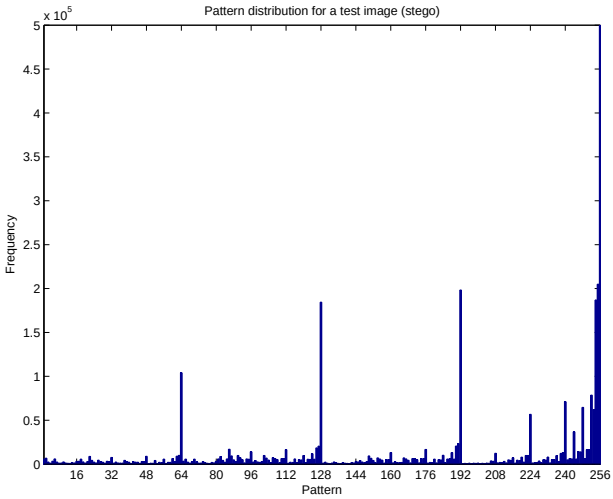
Figure 6: Pattern distribution for a “uniform” theoretical image and for Lena



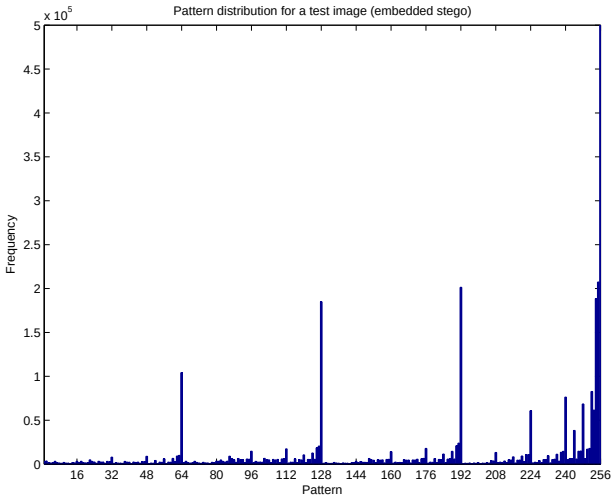
(a) Cover image



(b) Cover image after random embedding

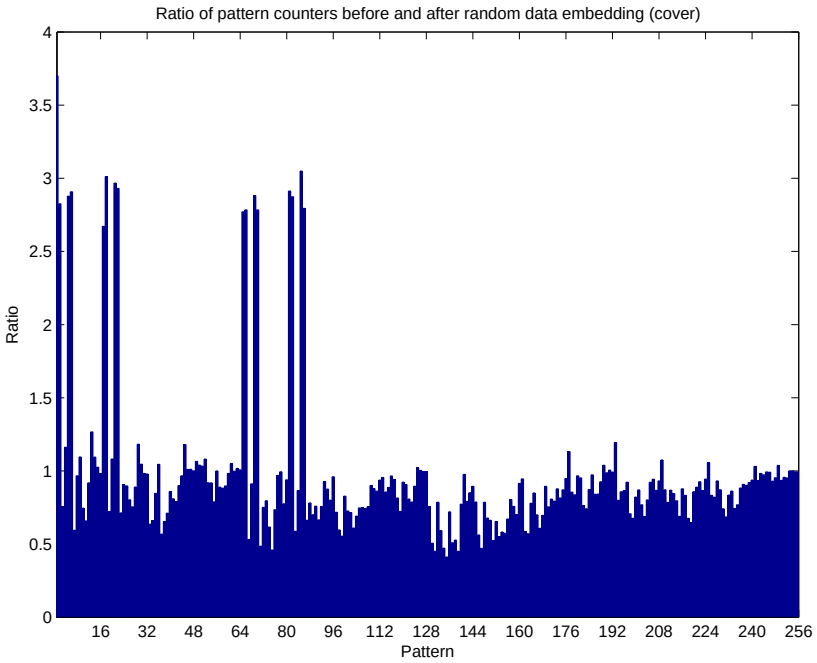


(c) Stego image

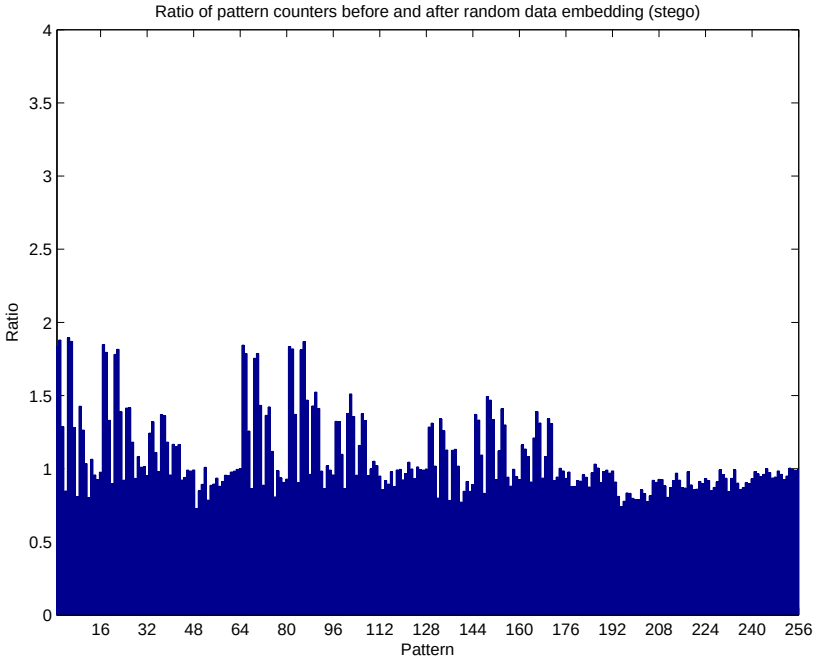


(d) Stego image after random embedding

Figure 7: Pattern distributions for a cover and a stego image (before and after random embedding)



(a) Ratio of pattern counters (cover image)



(b) Ratio of pattern counters (stego image)

Figure 8: Ratio of pattern counters for a cover and a stego image (before and after random embedding)

100	101
100	101
	100

(a) 1-pattern

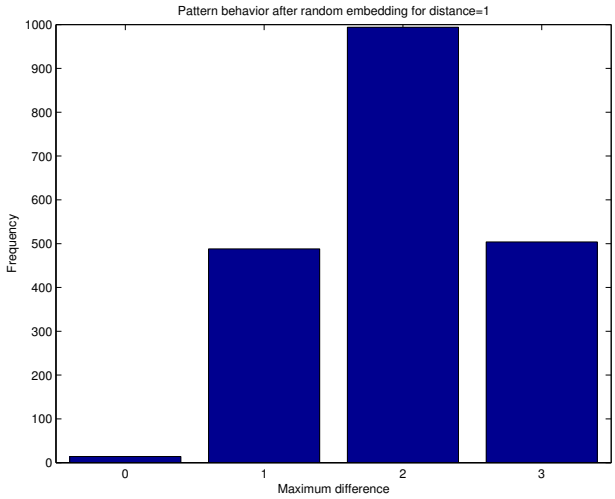
100	102
102	101
	100

(b) 2-pattern

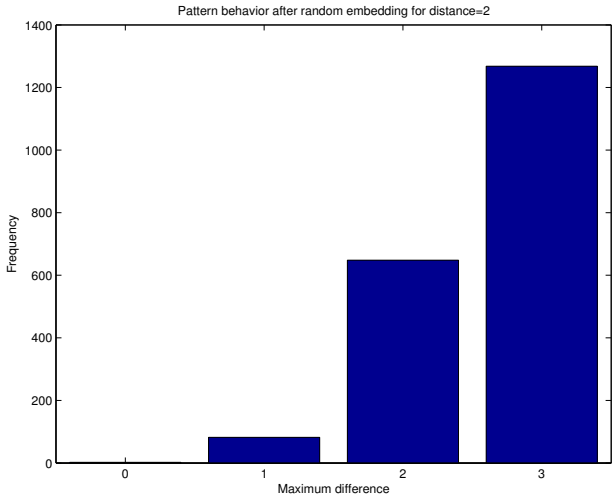
102	103
102	101
	100

(c) 3-pattern

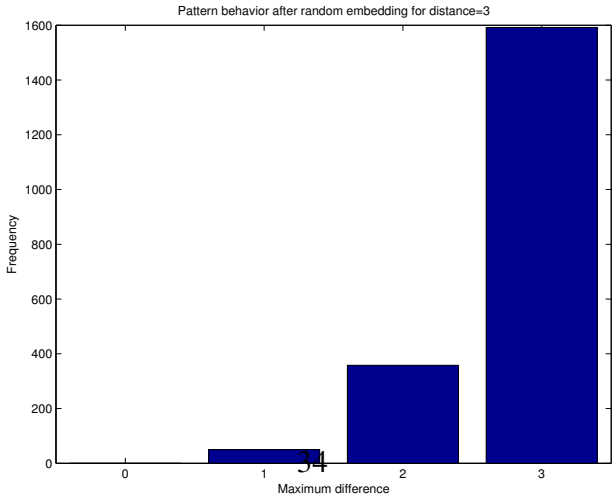
Figure 9: Example of block pixels producing 1, 2 and 3-patterns



(a) 1-patterns



(b) 2-patterns



(c) 3-patterns

Figure 10: Pattern shifting likelihood after random embedding

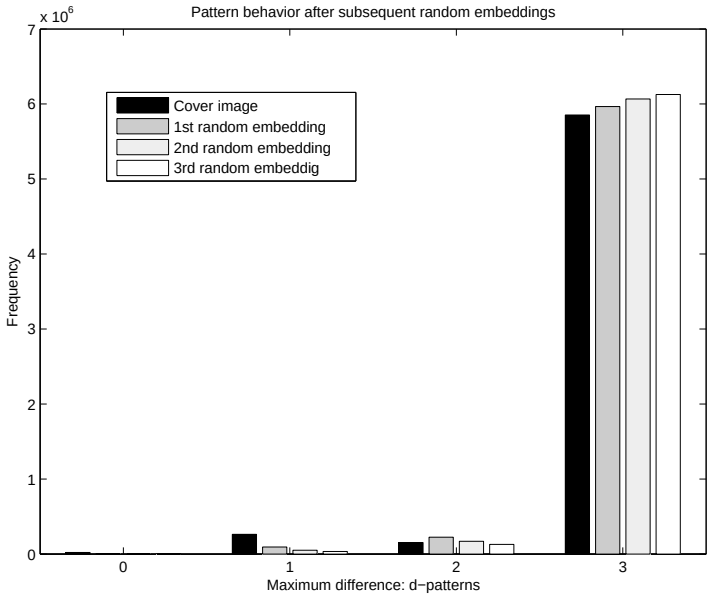
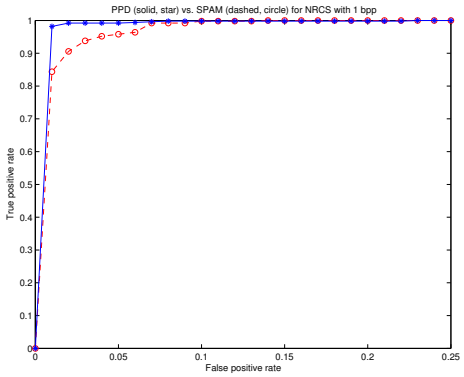
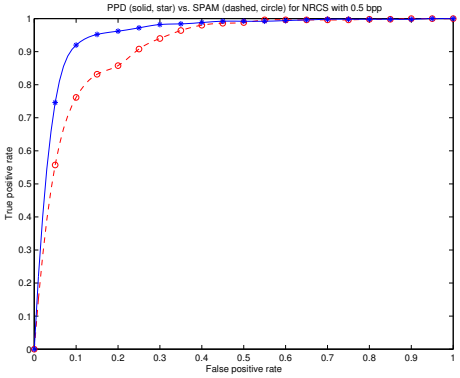


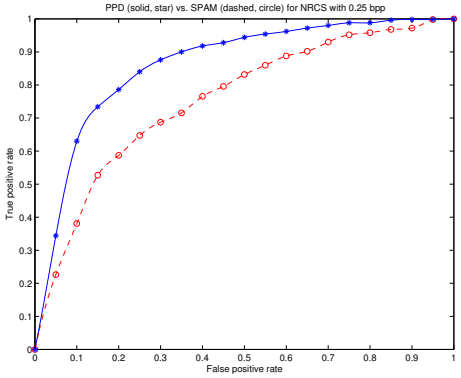
Figure 11: Pattern shifting after three subsequent random embeddings for the NRCSAK97001 image



(a) Embedding bit rate 1 bpp



(b) Embedding bit rate 0.5 bpp



(c) Embedding bit rate 0.25 bpp

Figure 12: ROC curves for the proposed PPD (solid, star) and SPAM methods (dashed, circle)

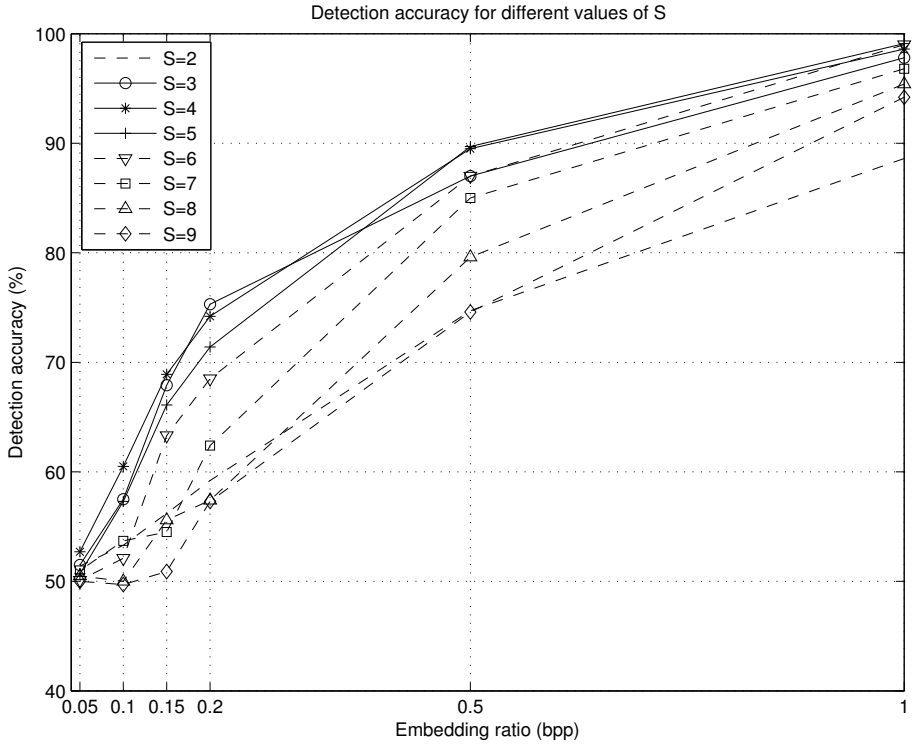


Figure 13: Comparison of accuracy results for $S \in [2, 9]$

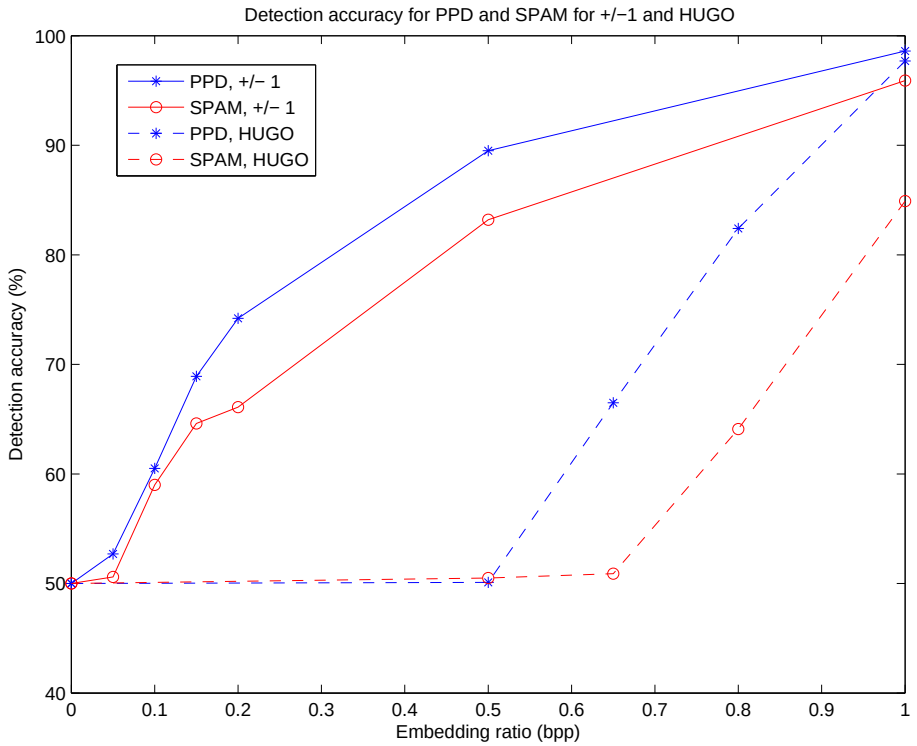


Figure 14: Comparison of accuracy results for PPD and SPAM against LSB matching and HUGO

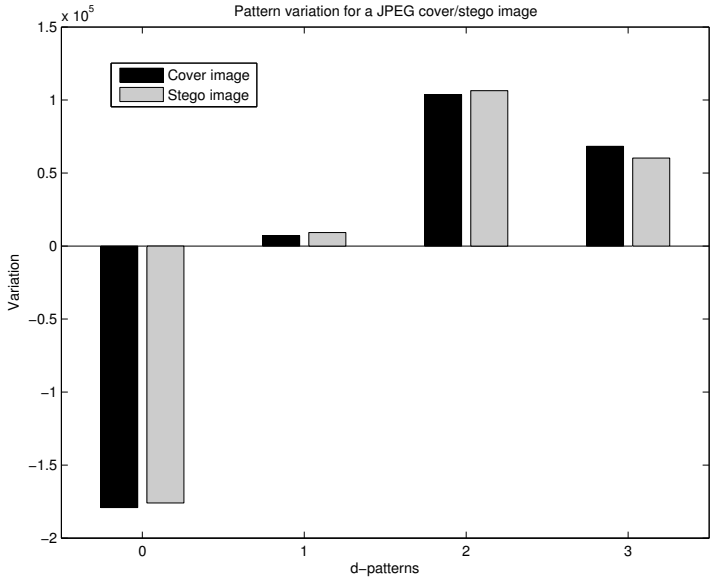


Figure 15: Pattern variation for the JPEG NRCSAK97001 (cover and stego) image

3.3 Steganography in noisy areas of the image

D. Lerch-Hostalot and D. Megías. Esteganografía en zonas ruidosas de la imagen. In *Actas de la XIII Reunión Española sobre Criptología y Seguridad de la Información*, pages 173–178. 2014. ISBN: 978-84-9717-323-0. http://hdl.handle.net/10045/40424.

Abstract

Most of the steganalysis in the state of the art is based on the use of machine learning*, that is, in training classifiers to distinguish between* cover *and* stego *images. Research in this field shows that the areas of the images that are more difficult to model and, consequently, in which it is more difficult to detect a hidden message than others, are noisy areas. These areas correspond to edges and textures. In this paper, we present a new steganographic method that allows hiding information in these zones, making the detection more difficult. The effectiveness of this method has been tested using two image databases and two recent feature extractors. The experiments show that the presented algorithm improves the statistical undetectability significantly with respect to the LSB matching method for the same embedding capacity.*

Esteganografía en zonas ruidosas de la imagen

Daniel Lerch-Hostalot
Universitat Oberta de Catalunya,
Internet Interdisciplinary Institute (IN3),
Estudis d'Informàtica, Multimèdia i Telecomunicació,
Rambla del Poblenou, 156,
08018 Barcelona,
Email: dlerch@uoc.edu

David Megías
Universitat Oberta de Catalunya,
Internet Interdisciplinary Institute (IN3),
Estudis d'Informàtica, Multimèdia i Telecomunicació,
Rambla del Poblenou, 156,
08018 Barcelona,
Email: dmegias@uoc.edu

***Resumen*—La mayor parte del estegoanálisis en el estado del arte se basa en el uso de técnicas de *machine learning*, es decir, en entrenar clasificadores para que sean capaces de diferenciar una imagen portadora de una imagen con mensaje oculto. Las investigaciones realizadas en este campo muestran que las zonas de la imagen más difíciles de modelar y, en consecuencia, aquellas en las cuales es más difícil detectar un mensaje incrustado, son las zonas ruidosas. Estas corresponden a líneas y texturas. En este artículo presentamos un nuevo método de esteganografía que permite ocultar información en dichas zonas, dificultando así su detección. La efectividad del método se ha comprobado usando dos bases de datos de imágenes diferentes y dos estegoanalizadores recientes. Los experimentos demuestran que el algoritmo propuesto mejora significativamente la indetectabilidad estadística respecto al sistema LSB *matching* para la misma capacidad de incrustación.**

Esteganografía, Estegoanálisis.

I. Introducción

La esteganografía estudia diferentes técnicas para la ocultación de datos en otros objetos, conocidos como objetos portadores. Actualmente, estos objetos portadores suelen ser medios digitales, como por ejemplo imágenes, vídeos o archivos de sonido. No obstante, sin lugar a dudas, el medio más utilizado en la actualizad son las imágenes, por su amplia difusión en Internet.

Uno de los métodos más usados para ocultar información en imágenes de mapas de bits es la sustitución del bit menos significativo (*Least Significant Bit*, LSB). Este método divide el mensaje original en bits y oculta cada uno de ellos en un píxel de la imagen. La variación en el valor del píxel es tan poco significativa que no puede ser detectada visualmente, pero resulta suficiente para ocultar información. Sin embargo, como puede verse en [16], esta técnica presenta algunos inconvenientes. La sustitución del LSB es una operación asimétrica, pues los píxeles con un valor par tenderán a incrementar su valor (cuando se incruste un '1'), mientras que los píxeles con un valor impar tenderán a disminuirla (cuando se incruste un '0'). Esto crea anomalías estadísticas en la imagen, como por ejemplo parejas de barras (frecuencias) que tienden a igualarse en el histograma de luminosidad de la imagen [16]. Finalmente, esta debilidad de la sustitución del LSB ha culminado en ataques como el RS [5] o el SPA [4], los cuales han conseguido detectar la presencia de información oculta incluso cuando el número de bits incrustados apenas alcanza el 3 % del número de píxeles de la imagen. Debido a estos ataques, el método de sustitución del LSB ha dejado de ser considerado como seguro.

El sistema que ha tomado el relevo de la sustitución del LSB es el conocido como LSB *matching* [15]. Este método es muy similar al anterior, pues solo tiene una pequeña diferencia: en lugar de sustituir el valor del LSB directamente por el bit a incrustar, lo que hace es modificarlo sumando o restando uno al valor total del píxel cuando el bit a incrustar no coincide con el LSB del píxel correspondiente. El efecto sobre el LSB es el mismo, así como la dificultad para detectarlo visualmente. Sin embargo, al proceder de esta manera, ya no se trata de una operación asimétrica y no se introducen anomalías estadísticas tan evidentes. De hecho, con este método resulta muy difícil diferenciar un mensaje oculto del ruido existente en todas las imágenes que aparece como consecuencia del proceso de captura.

Para detectar este método de ocultación de información en imágenes los estegoanalistas han recurrido al uso de técnicas de *machine learning* [2]. Para ello es necesario preparar una base de datos de imágenes que se usarán para entrenar un clasificador y verificar que este funciona correctamente. Este clasificador será el encargado de diferenciar las imágenes con mensaje oculto (imágenes esteganográficas) de las imágenes no alteradas (imágenes portadoras). En este tipo de estegoanálisis el trabajo del estegoanalista se basa principalmente en detectar aquellas características de la imagen que son más susceptibles de ser alteradas cuando se oculta información. Estas características son las usadas para entrenar al clasificador. Si bien se han propuesto diferentes métodos de estegoanálisis basados en clasificadores que ofrecen buenos resultados [13], [8], [11], todavía queda mucha investigación para avanzar en este campo.

Una de las lecciones aprendidas en los últimos años de investigación en estegoanálisis usando clasificadores es que existen zonas que son mucho más difíciles de modelar que otras: los bordes y las texturas. Estas zonas contienen mucho ruido y, en ellas, es muy difícil extraer características ade-

Este trabajo está financiado parcialmente por el Ministerio de Economía y Competitividad a través de los proyectos TIN2011-27076-C03-01/02 "CO-PRIVACY" y CONSOLIDER INGENIO 2010 CSD2007-0004 "ARES".

cuadas para entrenar al clasificador. Esta es la base que usan algunas técnicas modernas de esteganografía, como por jemplo las presentadas en [14], [9]. En este artículo se presenta una nueva técnica de ocultación de información que usa estas zonas ruidosas de la imagen para incrustar los bits del mensaje.

El resto del artículo se organiza de la manera siguiente. En la Sección II se presenta el método de esteganografía propuesto en el artículo. En la Sección III se analiza experimentalmente el método y se comprueba su indetectabilidad usando software de estegoanálisis. Finalmente, la Sección IV presenta las conclusiones extraídas de este trabajo.

II. Método propuesto

II-A. Motivación

Los métodos modernos de estegoanálisis usan clasificadores para modelar las propiedades estadísticas de las imágenes, pero existen zonas que son especialmente difíciles de modelar, como los bordes o las texturas. Por ello, la investigación en nuevos sistemas de esteganografía se centra, en gran parte, en la construcción de métodos que permitan ocultar información en esas zonas.

Sistemas de estegoanálisis como [13], [8], [11] modelan las características extraídas de la imagen como diferencias entre píxeles vecinos. Tomemos como ejemplo el método basado en patrones de diferencias de píxeles (*patterns of pixel diferences*, PPD) presentado en [11]. En ese artículo se usan cinco píxeles vecinos para modelar la imagen, como se muestra en la Fig.1, tomando uno de ellos como referencia a restar de los demás.

a	b
c	d
	e

Figura 1: Extracción de características basadas en bloques PPD

De esta forma pueden obtenerse vectores formados por cuatro posibles diferencias. Por ejemplo, si tomamos b como base, podemos obtener una representación en cuatro dimensiones: $v = [a-b, c-b, d-b, e-b]^{\mathrm{t}}$, donde $[\cdot]^{\mathrm{t}}$ denota el operador de transposición de un vector (o una matriz).

Suponiendo el caso más sencillo, es decir, usando imágenes en escala de grises con una profundidad de color de 8 bits, cada píxel puede tomar un valor de 0 a 255. Por lo tanto, en el peor de los casos, la diferencia entre dos píxeles vecinos será de 255. Así pues, con el modelo presentado, podrían generarse hasta 255^4 características, lo que resulta numéricamente impracticable para los clasificadores actuales. Con los sistemas propuestos en [13] y [8] la situación es similar. No obstante, no es habitual que un píxel con valor 0 sea el vecino de un pixel con valor 255, dado que en las imágenes el valor de los píxeles suele cambiar en forma de degradado. Por lo tanto, ignorar diferencias muy grandes entre valores vecinos no suele perjudicar a los sistemas de estegoanálisis. Es por ello que se utiliza un parámetro para reducir el número de características. Por ejemplo, en [11] se sugiere el uso de un umbral $S = 4$. De esta manera, en lugar de obtener 255^4 características, se obtienen 4^4, que es una cifra mucho más manejable. Con esta aproximación lo métodos de estegoanálisis pueden atacar el problema sin que la explosión en el número de características les impida modelar la imagen.

Sin embargo, este no es el único motivo para el uso de un umbral para reducir el número de dimensiones. Otro problema asociado a la dimensionalidad, y que perjudica seriamente al estegoanálisis, es el de la obtención de muestras insuficientes. Este tipo de problema, detectado previamente en estegoanálisis [7], se produce al usar modelos de grandes dimensiones. Cuantas más dimensiones tenga el modelo, más difícil es encontrar muestras en la imagen para todas las posibilidades que ofrece. Durante la extracción de características, se reparten todas las muestras extraídas de la imagen entre cada uno de los patrones de los que dispone el modelo. En el caso de PPD, por ejemplo, existen T^4 patrones, la frecuencia de los cuales dependerá de la imagen y de su contenido. Dado que el número total de muestras es fijo y a medida que crece el valor de T aumenta el número de patrones, la frecuencia de cada patrón será cada vez más pequeña. Los patrones menos frecuentes serán los primeros en llegar a frecuencias tan bajas que su valor no será representativo y perjudicarán al entrenamiento del clasificador. Por este motivo existe un límite en el valor del umbral a partir del cual los métodos de estegoanálisis dejan de ser efectivos. El método que se propone en este artículo, pretende explotar esta debilidad.

II-B. Zonas de inserción

Como se ha descrito en el apartado anterior, el objetivo del método presentado en este artículo es ocultar la información en las zonas más difíciles de modelar de la imagen. El problema que se presenta cuando se desea ocultar información únicamente en unas zonas concretas de la imagen es cómo comunicar al receptor del mensaje (de la imagen) en qué zonas debe leer y en qué zonas no. Si se desarrolla un procedimiento para identificar las zonas ruidosas, estas pueden cambiar (dejar de ser ruidosas) al ocultar información, por lo que el receptor puede acabar leyendo en zonas donde no hay mensaje e ignorando zonas donde sí lo había.

Un enfoque válido, tomado en los métodos [14], [9], es el uso de *Wet Paper Codes* (WPC) [6]. Estos métodos, que son muy adecuados para el problema presentado, son relativamente complejos, lo que implica un procesamiento muy lento en la inserción del mensaje. En este trabajo se expone un procedimiento alternativo, que resulta muy rápido y sencillo en comparación con el uso de WPC.

Para detectar las zonas de inserción se establecerá un umbral T que nos indicará las zonas difíciles de modelar, de la misma forma que lo hacen los sistemas de estegoanálisis. Se agruparan los píxeles en parejas de píxeles vecinos (a, b),de manera que solo los tomaremos en consideración para ocultar información si su diferencia es mayor o igual al umbral T, es decir si $|a - b| \geq T$.

(a) Imagen Lenna en escala de grises (b) Zonas de ocultación para $T = 4$ (c) Zonas de ocultación para $T = 10$

Figura 2: Imagen Lenna y zonas donde se oculta la información para diferentes valores de T

En la Fig.2(b) se muestran, con píxeles negros, las zonas donde se ocultaría la información en caso de usar $T = 4$ para la imagen Lenna de la Fig.2(a). Análogamente, la Fig.2(c) muestra las zonas de ocultación de información para el caso $T = 10$. En ellas podemos ver cómo se evitan las zonas más uniformes de la imagen, mientras que los bordes, principalmente, y algunas texturas son las zonas que se usan para ocultar información.

Para ocultar información el procedimiento consiste en recorrer la imagen tomando cada vez una pareja diferente de píxeles vecinos. Las parejas se forman sin solapamiento, de manera que, dados cuatro píxeles vecinos, (a, b, c, d), se formarán las parejas (a, b) y (c, d), mientras que la pareja (b, c) no se tendrá en consideración. Las parejas así formadas no se usarán para incrustar información si su diferencia esta por debajo de umbral T. La imagen se puede recorrer tomando las parejas de forma horizontal o vertical, aunque esto no es determinante para el funcionamiento del método propuesto.

II-C. Procedimiento de incrustación

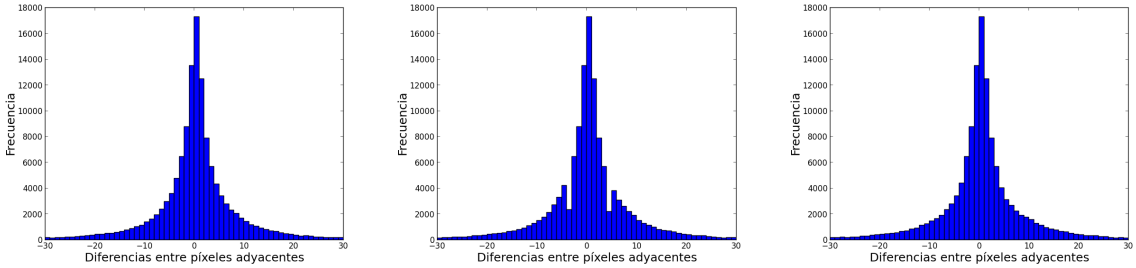
El método propuesto usa cada pareja que supera el umbral para ocultar un bit. Concretamente se oculta alterando el píxel de la izquierda, es decir, el etiquetado como a de la pareja (a, b). Para ello, se usa el LSB de a como bit de información, dejándolo tal y como está si su valor es igual al del bit del mensaje que se quiere ocultar y modificándolo si su valor no coincide. Esta modificación se realizará siempre incrementando la diferencia entre los valores de los píxeles de la pareja. Así pues, a', el nuevo valor de a, vendrá determinado por la ecuación siguiente:

$$a' = \begin{cases} a, & \text{si } a \bmod 2 = m, \\ a + 1, & \text{si } a > b \\ a - 1, & \text{si } a < b. \end{cases}$$

La idea es incrementar siempre la diferencia entre los píxeles de la pareja, nunca disminuirla. El motivo es que incrementar o disminuir aleatoriamente, de forma similar a como se hace en LSB *matching*, llevaría en el caso $|a - b| = T$ a dejar de cumplir el umbral establecido para algunas parejas, que pasarían a tener una diferencia $T - 1$. Al no cumplir el umbral de lo que consideramos un píxel adecuado para ocultar información, el receptor no sabría que debe leer información de él y se perdería la información oculta en ese píxel.

Lógicamente, esta operación para ocultar información introduce una anomalía estadística, pues no se aplica de forma equitativa sobre todas las parejas de píxeles. Al incrementar la diferencia entre parejas con un valor superior a $T+1$, parte de esas parejas (en las que se quiere ocultar un bit diferente al LSB de a) se convierten en parejas con diferencia $T + 2$. Análogamente, algunas parejas con diferencia $T + 2$ pasan a tener diferencia $T+3$ y así sucesivamente. En general, aunque varias parejas con diferencia $T + n$ pasan a tener diferencia $T + n + 1$, este hecho se ve compensado por el número de parejas nuevas con diferencia $T + n$ que aparecen al incrustar información en parejas con diferencia $T + n - 1$. Pero hay un caso especial, el de las parejas con diferencia T, pues mientras que parte de estas pasan a tener diferencia $T + 1$, el número de parejas con diferencia T no se ve retroalimentado y solo decrece. Esta anomalía puede observarse en el histograma de los valores de las diferencias entre píxeles adyacentes representado en la Fig.3(b), en comparación con el histograma de la imagen original que aparece en la Fig.3(a).

Para eliminar esta anomalía en el histograma, se puede repartir la responsabilidad de ser la primera pareja (la que tiene diferencia igual a T) entre diferentes parejas. De esta forma, se consigue que las barras del histograma correspondientes no decrezcan lo suficiente como para generar una anomalía. Para ello, se usa un valor de T dinámico, que dependerá del pixel que se esté modificando. La idea es inicializar un generador de números pseudoaleatorios (*Pseudo-Random Number Generator*, PRNG) con una semilla (por ejemplo una contraseña) que deberán conocer tanto el emisor del mensaje como el receptor. Este PRNG se utiliza para generar una secuencia de valores dinámicos para el umbral T. Se usa



(a) Imagen original (b) Imagen esteganográfica con $T = 4$ (c) Imagen esteganográfica con $T_{\text{mín}} = 4$ y $T_{\text{máx}} = 9$

Figura 3: Histograma de diferencias entre parejas de píxeles adyacentes para la imagen original, la imagen esteganográfica con $T = 4$ y la imagen esteganográfica con umbral dinámico entre $T_{\text{mín}} = 4$ y $T_{\text{máx}} = 9$

un rango de valores de T entre un mínimo, $T_{\text{mín}}$, y un máximo, $T_{\text{máx}}$. De esta manera los valores dinámicos de T se generarán con la función $T = \text{PRNG}(T_{\text{mín}}, T_{\text{máx}})$, que devuelve un número pseudoaleatorio con distribución uniforme en el intervalo $[T_{\text{mín}}, T_{\text{máx}}]$. De esta manera, como se puede apreciar en la Fig.3(c), la anomalía en el histograma desaparece.

El uso del PRNG para seleccionar un valor dinámico del umbral no solo sirve para eliminar la anomalía estadística, sino que además ofrece una capa de seguridad adicional que dificulta el estegoanálisis, dado que no se puede saber con exactitud en qué píxeles se ha ocultado información sin disponer de la semilla.

II-D. Algoritmos de incrustación y de extracción

Tal y como se explica en los apartados anteriores, ya tenemos todas las piezas necesarias para el algoritmo de inserción de datos de la imagen (Algoritmo 1).

La extracción de datos es similar. Basta con inicializar el PRNG, recorrer la imagen extrayendo parejas de píxeles e ir leyendo los LSB del primer píxel de cada pareja que cumple con el umbral T (Algoritmo 2).

III. Resultados experimentales

El método se ha verificado con dos sistemas de estegoanálisis: PPD [11] y SPAM [13] (en la versión más efectiva de este, que usa características de segundo orden). Estos sistemas permiten extraer características de las imágenes. Como clasificador, se ha usado una implementación de una *Support Vector Machine* (SVM) [3], por ser uno de los clasificadores que ofrece mejores resultados en estegoanálisis.

La SVM debe ser ajustada para que proporcione unos resultados óptimos. Concretamente, es necesario seleccionar valores para los parámetros C y γ. Estos valores serán escogidos para dar al clasificador la capacidad de generalizar. Para escoger dichos parámetros, se ha seguido el proceso especificado en [10], es decir, realizando una validación cruzada en el conjunto de entrenamiento de todos los posibles valores de los parámetros C y γ que se especifican a continuación:

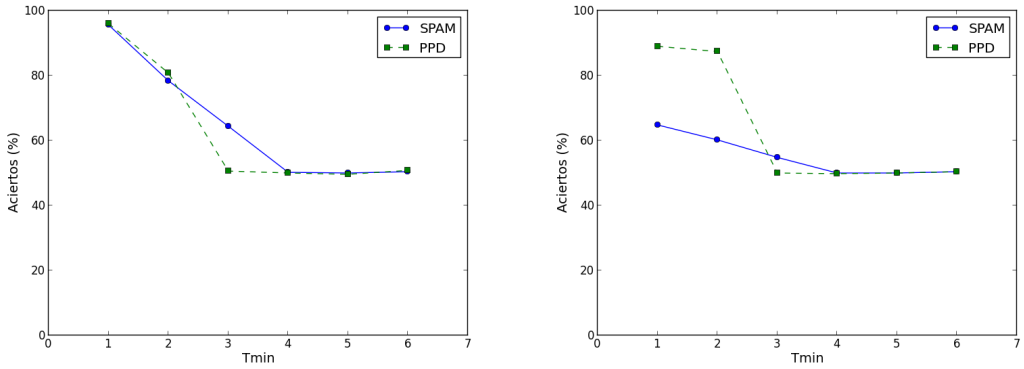
Algorithm 1 Ocultar mensaje

Input: $M, I, \text{Seed}, T_{\text{mín}}, T_{\text{máx}}$
M: Mensaje a ocultar
I: Matriz $[1..H, 1..W]$ que contiene la imagen original
Seed: Semilla del generador PRNG
$T_{\text{mín}}$: Valor mínimo para el cálculo de T
$T_{\text{máx}}$: Valor máximo para el cálculo de T
Output: I'
I': Matriz que contiene la imagen con el mensaje oculto

1: InitializePRNG(Seed)
2: **for all** $i \in [1, H]$ **do**
3: **for all** odd $j \in [1, W-1]$ **do**
4: $T \leftarrow \text{PRNG}(T_{\text{mín}}, T_{\text{máx}})$
5: $a \leftarrow I[i, j]$
6: $b \leftarrow I[i, j+1]$
7: **if** $|a - b| \geq T$ **then**
8: $m \leftarrow \text{nextBit}(M)$
9: $I'[i,j] \leftarrow \begin{cases} a, & \text{if } a \bmod 2 = m, \\ a+1, & \text{if } a > b, \\ a-1, & \text{if } a < b. \end{cases}$
10: **end if**
11: **end for**
12: **end for**

$$C \in \left\{2^{-5}, 2^{-3}, 2^{-1}, 2^{1}, 2^{3}, \ldots, 2^{15}\right\},$$
$$\gamma \in \left\{2^{-15}, 2^{-13}, 2^{-11}, \ldots, 2^{-1}, 2^{1}, 2^{3}\right\}.$$

Los experimentos se han realizado con la base de datos BOSS, presentada en [1], por ser una de las más usadas en esteganografía, y también en la base de datos pública NRCS [12], por disponer de imágenes de alta resolución muy ruidosas, significativamente diferentes de las de BOSS. Para cada base de datos, se han creado dos grupos de imágenes, uno que se usa como conjunto de entrenamiento y otro que se usa como conjunto de verificación. Cada uno de ellos esta formado por 500 imágenes, 250 de ellas sin incrustar (portadoras) y



(a) Base de datos BOSS

(b) Base de datos NRCS

Figura 4: Porcentajes de detección correcta en función de $T_{\text{mín}}$, usando las bases de datos de imágenes BOSS y NRCS

Algorithm 2 Extraer mensaje

Input: I, Seed, $T_{\text{mín}}, T_{\text{máx}}$
I: Matriz $[1..H, 1..W]$ que contiene la imagen con el mensaje oculto
Seed: Semilla del generador PRNG
Output: M
M: Mensaje extraído

```
1: InitializePRNG(Seed)
2: for all i ∈ [1, H] do
3:   for all odd j ∈ [1, W − 1] do
4:     T ← PRNG(T_mín, T_máx)
5:     a ← I[i, j]
6:     b ← I[i, j + 1]
7:     if |a − b| ≥ T then
8:       M ← addBit(M, a)
9:     end if
10:  end for
11: end for
```

las otras 250 con información incrustada (imágenes esteganográficas). El umbral usado por defecto en PPD es $T = 4$ (parámetro S especificado en [11], mientras que en SPAM es de $T = 3$. Los experimentos se han diseñado para verificar que, marcando con umbrales superiores a los establecidos por las herramientas de estegoanálisis, el método presentado no se detecta. Se ha incrustado información en las imágenes usando diferentes valores para $T_{\text{mín}}$ y $T_{\text{máx}}$, tal y como se muestra en el Cuadro I.

Como se puede ver en la Fig.4, el porcentaje de detección cae al 50 % (equivalente a decisión aleatoria, o sea, a no detección) aproximadamente al llegar a $T_{\text{mín}} = 4$. En los gráficos se aprecia como los métodos de estegoanálisis fallan cuando la información esta oculta en zonas que no pueden modelar. Los experimentos se han realizado sobre dos bases

Cuadro I: Valores mínimos y máximos del umbral usados en los experimentos

$T_{\text{mín}}$	$T_{\text{máx}}$
1	6
2	7
3	8
4	9
5	10
6	11

de datos de imágenes muy diferentes, y en ambos casos, el algoritmo propuesto no es detectado cuando $T_{\text{mín}}$ supera el umbral usado por los métodos de estegoanálisis.

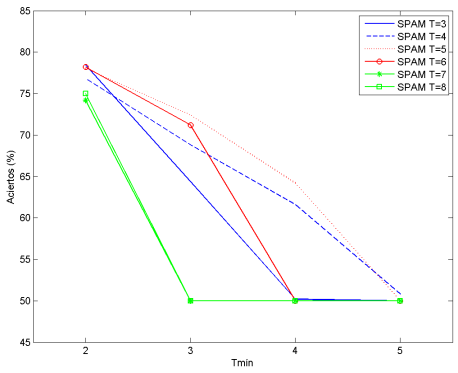


Figura 5: Porcentaje de detección correcta en función de $T_{\text{mín}}$, usando la base de datos de imágenes BOSS y diferentes valores de T, para el método de estegoanálisis SPAM

Sin embargo, podría parecer que el motivo por el que los métodos de estegoanálisis no detectan al método de esteganografía propuesto es por la elección de un umbral superior

al que ellos usan y que bastaría con subir también el umbral usado en estos métodos. Pero esto no es así dado que, si se incrementa el umbral, aumenta el número de dimensiones y aparecen combinaciones para las que no existen muestras o para los que existen muy pocas. Esto, como se ha comentado en la Sección II, empeora considerablemente los resultados del estegoanálisis. En la Fig.5 se puede observar que no existe ningún umbral T que permita detectar el método propuesto con SPAM si $T_{\text{mín}} \geq 5$. Lo mismo sucede para PPD si $T_{\text{mín}} \geq 4$.

La cantidad de información que puede incrustarse (es decir, la capacidad del método) en una imagen depende de las zonas ruidosas de esta, por lo que no es sencillo de determinar *a priori*. A nivel orientativo, usando $T_{\text{mín}} = 4$ y $T_{\text{máx}} = 9$ en las imágenes de BOSS, se ha realizado la inserción con una ratio media del 9 %, es decir incrustando un bit en el 9 % de los píxeles (0,09 bits por píxel). En las imágenes de NRCS la ratio de incrustación es del 13 %. Para mostrar la efectividad del sistema propuesto, se han comparado los resultados de indetectabilidad de este con los obtenidos usando la esteganografía LSB *matching* tradicional [15]. Para ello, se han usado los estegoanalizadores PPD y SPAM con el objetivo de calcular los porcentajes de detección cuando se incrusta en LSB *matching* usando una ratio del 9 % en BOSS y del 13 % en NRCS. De esta manera se puede realizar una comparación en igualdad de condiciones en cuanto a capacidad se refiere. Los resultados de detección se muestran en II. Como se puede observar, para las mismas ratios de inserción que no se detectan (porcentaje de aciertos del 50 %) con el algoritmo presentado, la esteganografía LSB *matching* se detecta con los estegoanalizadores SPAM y PPD (porcentaje de aciertos superior al 50 %).

Cuadro II: Detección de la esteganografía LSB *matching*, para la misma capacidad que el método propuesto, usando PPD y SPAM

Base de datos	Método de detección	Porcentaje de aciertos
BOSS	SPAM	85.00 %
BOSS	PPD	81.60 %
NRCS	SPAM	58.00 %
NRCS	PPD	64.00 %

IV. Conclusión

En este artículo se presenta un nuevo método para ocultar información en zonas difíciles de modelar de la imagen. Para detectar estas zonas, el método propuesto intenta explotar dos debilidades de los sistemas de estegoanálisis existentes: el crecimiento exponencial del número de características y la imposibilidad de extraer información útil de patrones con pocas muestras. Ambas debilidades tienen en común un umbral T, usado como base en el método presentado.

Por otra parte, se trata de un método que no requiere de ningún cálculo complejo, a diferencia de otros que persiguen objetivos similares, como los basados en WPC, por lo que es adecuado para entornos en los que la velocidad de ejecución o el rendimiento sean un factor clave. Los resultados muestran la indetectabilidad del método ante dos sistemas de estegoanálisis: PPD [11] y SPAM [13], viendo como la selección de un umbral T adecuado es suficiente para eludir la detección. También se comprueba que el ajuste del parámetro T en los métodos de estegoanálisis no permite la detección del método propuesto.

En futuros trabajos sería interesante estudiar si existen otros modelos similares que tengan en cuenta grupos de píxeles mayores que una pareja y si esto puede mejorar el algoritmo. Además, sería recomenable realizar un estudio teórico para el cálculo del valor óptimo de T.

Referencias

[1] T. Filler, T. Pevný, and P. Bas, *Break our steganographic system (BOSS)*, 2010. [Online]. Disponible: http://exile.felk.cvut.cz/boss/ [Accedido el 24 de julio de 2014].

[2] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics Series)*. New York, Secaucus, NJ, USA: Springer, 2006.

[3] C.-C. Chang and C.-J. Lin, "LIBSVM - A Library for Support Vector Machine," [Online]. Disponible: http://www.csie.ntu.edu.tw/~cjlin/libsvm [Accedido el 24 de julio de 2014].

[4] S. Domitrescu, X. Wu and N. D. Memon, "On Steganalysis of Random LSB Embedding in Continuous-tone Images," In *Proc. International Conference on Image Processing, ICIP 2002*, Rochester, NY, USA: IEEE, pp. 324-339.

[5] J. Fridrich, M. Goljan and R. Du, "Detecting LSB steganography in color and grayscale Images," In *Proc. ACM Workshop on Multimedia and Security*, Ottawa, Canada: ACM, pp. 22-28, 2001.

[6] J. Fridrich et al., "Writing on Wet Paper," *IEEE Trans. on Signal Processing*, vol. 53, no. 10, Oct. 2005, pp. 3923-3935.

[7] J. Fridrich, J. Kodovský, V. Holub, M. Goljan. "Breaking HUGO - the process discovery ," *Information Hiding, 13th International Workshop, Lecture Notes in Computer Science*.

[8] J. Fridrich and J. Kodovský, "Rich Models for Steganalysis of Digital Images," *IEEE Trans. on Information Forensics and Security*, vol. 7, no. 3, June 2012, pp. 868-882.

[9] V. Holub and J. Fridrich, "Designing Steganographic Distortion Using Directional Filters," In: *Proc. IEEE Workshop on Information Forensic and Security (WIFS)*, Tenerife, Spain: IEEE, pp. 234-239, 2012.

[10] C. W. Hsu, C. C. Chang, and C.J. Lin, "A practical guide to support vector classification," Department of Computer Science, National Taiwan University, 2003. [Online]. Disponible: http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf [Accedido el 24 de julio de 2014].

[11] D. Lerch-Hostalot and D. Megías, "LSB matching steganalysis based on patterns of pixel differences and random embedding," *Computers & Security*, vol. 32, Feb. 2013, pp. 192-206.

[12] National Resource Conservation Service, *NRCS Photo Gallery*, [Online]. Disponible: http://photogallery.nrcs.usda.gov [Accedido el 24 de julio de 2014].

[13] T. Pevný, P. Bas and J. Fridrich, "Steganalysis by Subtractive Pixel Adjacency Matrix," In *Proc. ACM Multimedia and Security Workshop*, Princeton, NJ, USA: ACM, pp. 75-84, 2009.

[14] T. Pevný, T. Filler, P. Bas, *Using High-Dimensional Image Models to Perform Highly Undetectable Steganography*. Information Hiding. Lecture Notes in Computer Science Volume 6387, 2010, pp 161-177.

[15] T. Sharp, "An Implementation of Key-Based Digital Signal Steganography," In *Information Hiding, Lecture Notes in Computer Science*, vol. 2137, Berlin-Heidelberg, Germany: Springer, 2001, pp. 13-26.

[16] A. Westfeld and A. Pfitzmann, "Attacks on Steganographic Systems," In *Information Hiding, Lecture Notes in Computer Science Volume*, vol. 1768, Berlin-Heidelberg, Germany: Springer, 2000, pp. 61-76

3.4 Unsupervised steganalysis based on artificial training sets

D. Lerch-Hostalot and David Megías. Unsupervised steganalysis based on artificial training sets. *Engineering Applications of Artificial Intelligence*, 50:45–59, Apr. 2016. `http://dx.doi.org/10.1016/j.engappai.2015.12.013`.

Abstract

In this paper, an unsupervised steganalysis method that combines artificial training sets and supervised classification is proposed. We provide a formal framework for unsupervised classification of stego and cover images in the typical situation of targeted steganalysis (i.e., for a known algorithm and approximate embedding bit rate). We also present a complete set of experiments using 1) eight different image databases, 2) image features based on Rich Models, and 3) three different embedding algorithms: Least Significant Bit (LSB) matching, Highly undetectable steganography (HUGO) and Wavelet Obtained Weights (WOW). We show that the experimental results outperform previous methods based on Rich Models in the majority of the tested cases. At the same time, the proposed approach bypasses the problem of Cover Source Mismatch –when the embedding algorithm and bit rate are known–, since it removes the need of a training database when we have a large enough testing set. Furthermore, we provide a generic proof of the proposed framework in the machine learning context. Hence, the results of this paper could be extended to other classification problems similar to steganalysis.

Unsupervised Steganalysis Based on Artificial Training Sets

Daniel Lerch-Hostalot[a], David Megías[a]

[a]*Estudis d'Informàtica Multimèdia i Telecomunicació, Internet Interdisciplinary Institute (IN3), Universitat Oberta de Catalunya, Av. Carl Friedrich Gauss, 5, 08660 Castelldefels, Catalonia, Spain.*

Abstract

In this paper, an unsupervised steganalysis method that combines artificial training sets and supervised classification is proposed. We provide a formal framework for unsupervised classification of stego and cover images in the typical situation of targeted steganalysis (i.e., for a known algorithm and approximate embedding bit rate). We also present a complete set of experiments using 1) eight different image databases, 2) image features based on Rich Models, and 3) three different embedding algorithms: Least Significant Bit (LSB) matching, Highly undetectable steganography (HUGO) and Wavelet Obtained Weights (WOW). We show that the experimental results outperform previous methods based on Rich Models in the majority of the tested cases. At the same time, the proposed approach bypasses the problem of Cover Source Mismatch –when the embedding algorithm and bit rate are known–, since it removes the need of a training database when we have a large enough testing set. Furthermore, we provide a generic proof of the proposed framework in the machine learning context. Hence, the results of this paper could be extended to other classification problems similar to steganalysis.

Keywords: Unsupervised steganalysis, Cover source mismatch, Machine learning

1. Introduction

Data hiding is a collection of techniques to embed secret data into digital media. These techniques can be used in many different application scenarios, such as secret communications, copyright protection or authentication of digital contents, among others. Images are the most common carriers for data hiding because of their widespread use in the Internet.

Within data hiding, steganography is a major branch whose goal is to secretly communicate data, making it undetectable for an attacker. On the other hand, steganalysis is another branch whose goal is to detect messages previously hidden using steganography.

Many of the image steganalysis methods in the state of the art (Pevný et al., 2010a; Fridrich and Kodovský, 2012) use feature-based steganalysis and machine learning classification. In order to apply this methodology, the steganalyst needs to extract a set of features

Email addresses: dlerch@uoc.edu (Daniel Lerch-Hostalot), dmegias@uoc.edu (David Megías)

from a training data set and train a classifier. Then, the classifier is tested using a testing data set and, if the results are satisfactory, the classifier is considered successful.

This approach is widely adopted in classification tasks. The classifier is trained with a specific data set and, consequently, its classification capabilities usually decrease as the testing data set differs from the training data. As a result, this methodology is not fully effective when used in real-world scenarios.

The data set obtained after feature extraction depends on many factors, such as the steganographic algorithm used for hiding data into the cover source, the algorithm used for feature extraction or the properties of the cover source in different aspects (e.g. size, noise and hardware used for acquisition). If similar cover source is used, the feature extraction process provides data sets with similar representation and, therefore, the machine learning tools work properly and the classification results are satisfactory. However, if different cover source is used, the data sets obtained by feature extraction are also different, producing a degradation of the classification results. Machine learning (Mitchell, 1997; Bishop, 2006) literature refers to this problem as *domain adaptation*, whereas the term used to refer to this situation in steganalysis is *cover source mismatch (CSM)*. This constitutes an important open problem in the field (Ker et al., 2013), which was initially reported in (Cancelli et al., 2008).

Several approaches to deal with the CSM problem have been proposed in the recent years. In the BOSS competition (Filler et al., 2010), the BOSSrank database (which suffers from CSM) had to be used as a testing set. Some participants of the competition tried to include the testing set images in the training set (Gul and Kurugollu, 2011; Fridrich et al., 2011). This idea was called "training on a contaminated database" (Fridrich et al., 2011). This approach consists in applying denoising algorithms to estimate the cover sources of the testing set and using these estimated covers to generate new stego samples, by embedding new information into them. After that, these new estimated cover and stego samples are included in the training set.

In 2012, a solution based on training a classifier with a huge variety of images was proposed (Lubenko and Ker, 2012). This approach consists in applying machine learning to millions of images. Due to the high time and memory requests, this step is performed using on-line classifiers. Later on, in 2013, the use of rich features in universal steganalysis was analyzed (Pevný and Ker, 2013). Since rich features are not sensitive enough for their application in universal steganalysis, the authors apply linear projections informed by embedding methods and an anomaly detector. This approach tries to make these projections sensitive to stego content and, at the same time, insensitive to cover variation.

In 2014, different methods to deal with CSM were presented. Ker and Pevný (2014) show the possibility of centering features when there is a shift in the cover sources, by subtracting an estimated centroid of the cover features. Ker and Pevný also use weighted ensemble methods to deal with situations in which the features are moving in different directions after embedding. Kodovský et al. (2014) present three different strategies to deal with CSM. The first one consists in training with a mixture of different cover sources. The second approach uses different classifiers trained on different sources and, in the testing step, the testing set is classified using the closest source. The third strategy is similar to the second one, but

testing each image separately using the closest source. In 2014, another approach, based on Ensemble Classifiers with Feature Selection (EC-FS) (Chaumont and Kouider, 2012), was proposed by Pasquet et al. (2014). In this new method, Pasquet et al. use the EC-FS classifier with the *Islet approach*, a pre-processing step that consists in organizing images in clusters and assigning a steganalyzer to each cluster. Using this technique, a classifier can manage the diversity of the images more easily, after learning with a set of close feature vectors (in each cluster, the distance between the feature vectors is relatively small). This allows reducing the number of required images from millions to a few thousands.

In this paper we present a new approach based on bypassing the CSM problem rather than addressing it. The proposed technique consists in creating an "artificial" training set from the testing set. This artificial training set is formed by applying the targeted steganographic algorithm to the testing data (the data set A) twice. If the testing set A is formed by stego and cover images, a first application of the steganographic algorithm results in a "transformed" set B with "double stego" and stego images. The second application of the steganographic algorithm produces a "double transformed" set C that includes "triple stego" and "double stego" images. We show how the sets A and C can be used as artificial training data to finally classify the set B into stego and "double stego" images. Since there is a bijection between the elements of A and B, this is equivalent to the classification of the images in A as cover or stego.

The idea behind the proposal is that part of the images that we want to classify – the cover images– can be transformed into images that belong to the other class that we want to classify: the class of stego images. This fact is exploited to create an artificial training set that is used to find a boundary between classes with remarkable accuracy. This classification technique can thus have a relevant impact in the way in which steganalysis is usually approached, since it allows classifying the images without a real training set, which constitutes the direct cause of the CSM problem.

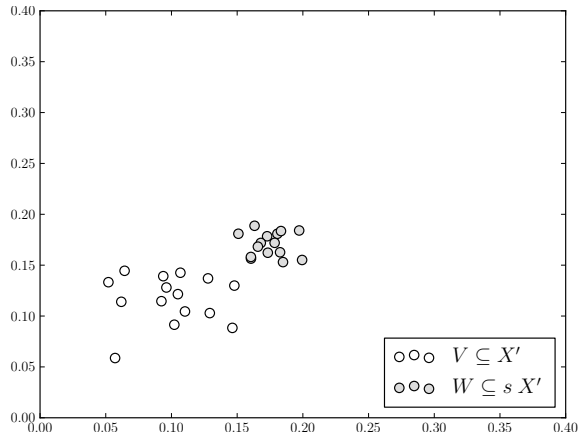
In this paper, we provide results for three different steganographic methods, namely, Least Significant Bit (LSB) matching (LSB matching) (Mielikäinen, 2006), Highly undetectable steganography (HUGO) (Pevný et al., 2010b) and Wavelet Obtained Weights (WOW) (Holub and Fridrich, 2012). Nevertheless, the proposed method is general and can be applied to any steganographic system, such as the more recent methods suggested by Karakiş et al. (2015).

The rest of this paper is organized as follows. Section 2 presents the proposed method, which is formalised in Section 3. Section 4 presents the experimental results obtained using the proposed method for eight different image databases in a cross-domain environment. Finally, Section 5 summarizes the conclusions of this work and suggests some directions for further research.

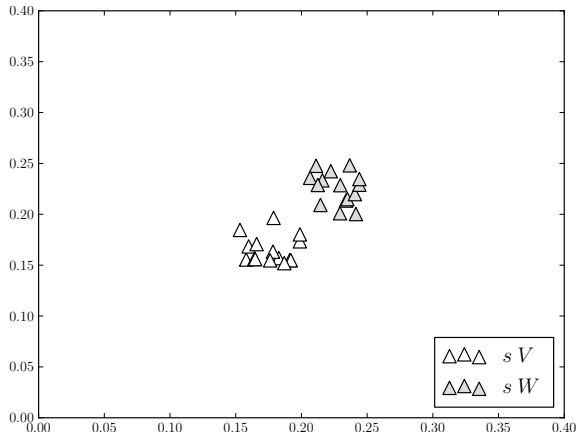
2. Proposed Method

This section presents a description of the method proposed in the paper.

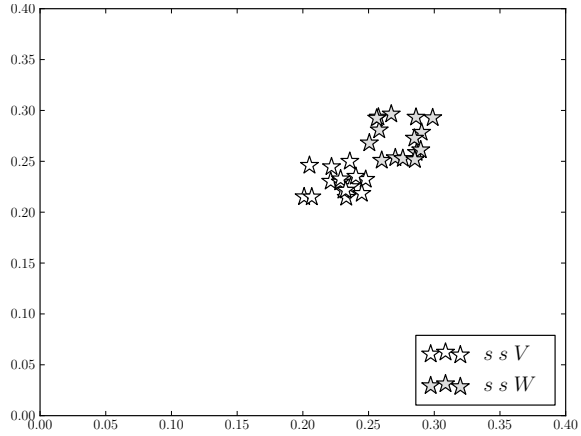
First of all, we assume that the embedding algorithm and the approximate bit rate used by the steganographer are known. Using the same algorithm and bit rate, we can perform



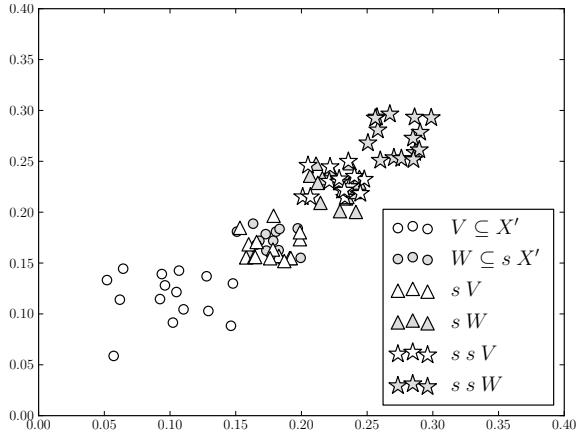
(a) Original samples: data set A



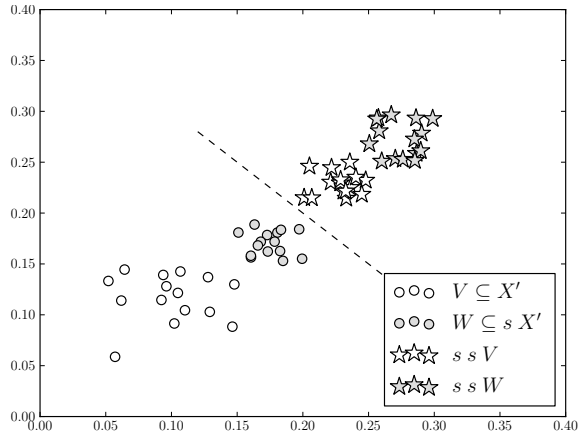
(b) "Transformed" samples: data set B



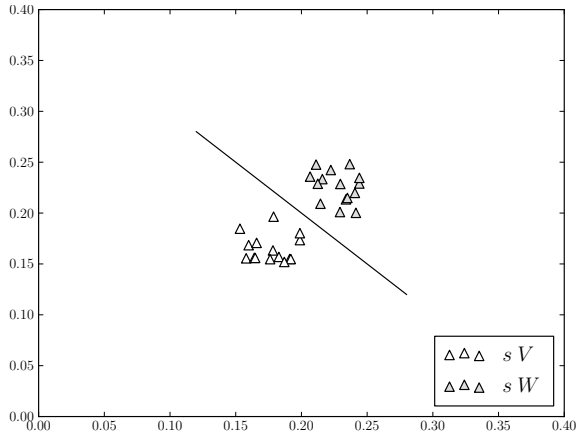
(c) "Double transformed" samples: data set C



(d) All data sets



(e) Learning boundary A/C



(f) Classification of the data set B

Figure 1: Simple graphical representation of the proposed method

new embedding operations to all the images in the testing database. Let A be the testing data set; B the transformed set, obtained after embedding data in all the images of A; and C the double transformed set, obtained after embedding data in all the images of B. As a result, A contains cover and stego images, B contains stego and "double stego" images, and C contains "double stego" and "triple stego" images. The interesting fact of this situation is that, if we create an artificial training set formed by A and C, we can train a classifier to learn the boundary between A and C. In principle, the same boundary can be used to classify the transformed set B into stego and "double stego" images. Furthermore, the existing bijection between the elements of B and A makes it possible to relate each stego image of B with a cover image in A, and each "double stego" image in B with a stego image in A. Hence, classifying B as stego or "double stego" images is equivalent to classifying A as cover or stego images. The bijection between the elements of A and B can be recorded in order to complete the classification of the original testing set A.

A simple graphical representation of this approach is shown in Fig.1 to illustrate the rationale behind the proposed algorithm. In Fig.1a, we can see the set A, with the cover and stego samples depicted as white and gray circles, respectively. Although the cover and stego images are shown with circles of different color, note that this set is not labeled for classification (since it is the testing data set). In Fig.1b, the set B, resulting from the application of the steganographic algorithm to all the images of the A, are shown. In this case, the cover images of A become stego images of B (white triangles), whereas the stego images of A become "double stego" images of B (gray triangles). Similarly, in Fig.1c, we can see the "double transformed" set C, which contains "double stego" (white stars) and "triple stego" images (gray stars). In Fig.1d, all the data sets A, B and C are shown together.

The boundary between A and C can be found using machine learning with $A \cup C$ as a training set, as shown in Fig.1e. In this step, two different labels must be used, one for the images of A and the other one for those of C. Then, this trained classifier can be applied to the set B as depicted in Fig.1f, where the learnt boundary is used to classify the images of B as stego or "double stego". Finally, this result can be applied to separate A into cover and stego images, since stego images in B match cover images in A, whereas "double stego" images in B match stego images in A.

A flowchart of the proposed algorithm is shown in Fig.2. For the sake of notational simplicity, (A, λ'_1) and (B, λ'_2) stand for the Cartesian products $A \times \{\lambda'_1\}$ and $B \times \{\lambda'_2\}$, respectively. Similarly, a function applied to a set, e.g. $E_{br}(A)$, means that the function is applied to all the elements (images) of that set, and the resulting image set is returned. The call to the function *Train* returns a classifier *Clf*. This classifier is then used to separate B into stego (λ'_1) and "double stego" (λ'_2) images, by calling the function *Classify*. The final loop "translates" the classification of the images of B, as stego (λ'_1) or "double stego" (λ'_2), into the classification of the images of A, as cover (λ_1) or stego (λ_2). In this loop, $n = |A| = |B| = |C|$, where $|\cdot|$ stands for the cardinality of a set.

It must be taken into account that the notation in the flowchart is deliberately abused, since the training and classification procedures are not carried out directly with the images, but with their feature vectors which must be extracted before.

A theoretical analysis of the approach is provided in the next section. The theoretical

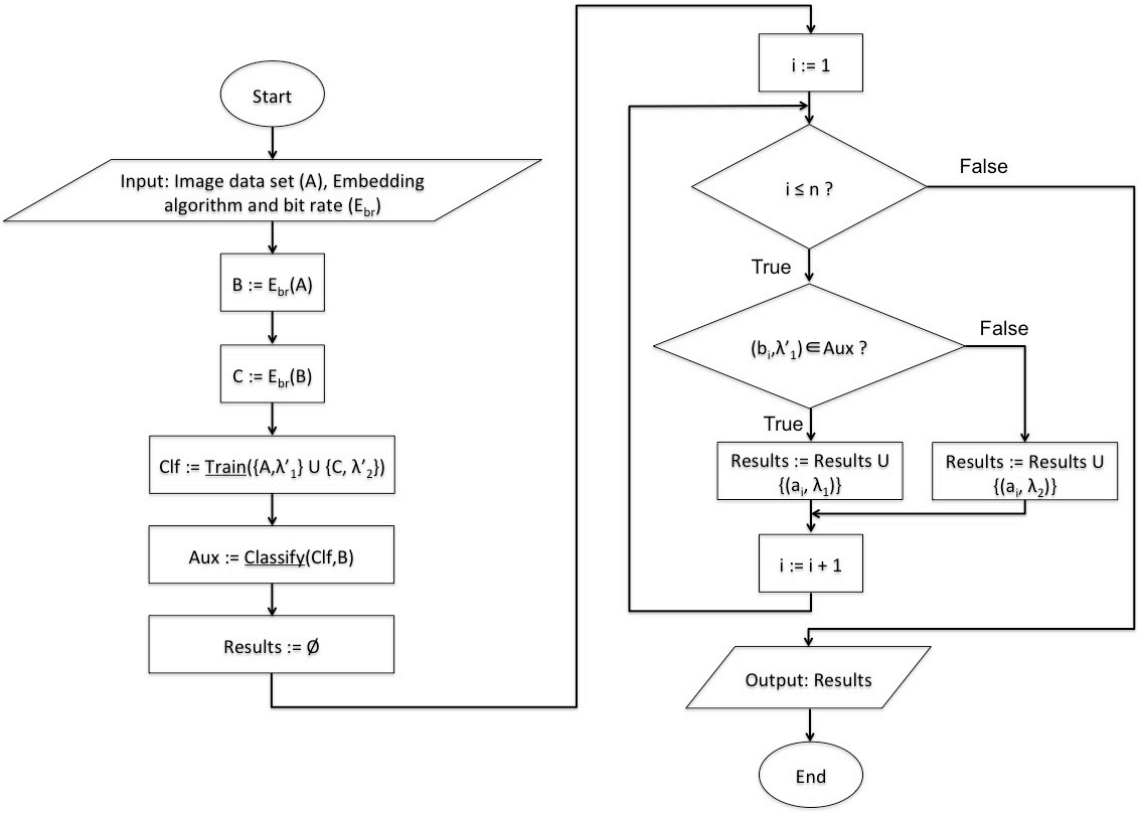


Figure 2: Flowchart of the proposed method

analysis includes a theorem and a proof based on the assumptions taken (which are standard assumptions in the field of targeted steganalysis).

3. Theoretical Analysis

This section is aimed at providing a theoretical framework for the suggested method, by means of definitions, lemmas, a theorem and proofs.

First of all, we provide a formalisation of learning algorithms with some basic definitions. Consider an input space X (of samples) and an output space $L = \{\lambda_1, \lambda_2, \ldots, \lambda_M\}$ (of labels), and assume that the pairs $(x_i, l_i) \in X \times L$ are random variables i.i.d. according to an unknown distribution D, such that each x_i is associated to a unique label l_i.

The goal of a classification (learning) algorithm is to find a function $h : X \to L$ that predicts l_i from x_i. The function h is an approximation of an unknown labeling function $t : X \to L$ that relates each value of X with its corresponding unique label in L.

There are two types of classification methods:

- *Supervised classification*: in this case, we use a *training set* of m **known** samples (pairs) $\{(x_1, l_1), (x_2, l_2), \ldots, (x_m, l_m)\} \subset X \times L$ in a training fase to obtain the function h.

After that, the function h is used to classify a *testing set* of n vectors $\{x'_1, x'_2, \ldots, x'_n\} \subset X$, without any label assigned to them.

- *Unsupervised classification*: in this case, the learning algorithm obtains the function h to classify a testing set without requiring any training set.

Now, we provide some definitions required for the proposed method and a formal proof by means of a theorem.

Definition 1. We call *splitting function* to a function $s : X' \rightarrow s[X']$, where $X', s[X'], s[s[X']], s[s[s[X']]] \subset X$, such that:

1. When $s(x_i) = x_j$ and $t(x_i) = \lambda_k$, then $t(x_j) = \lambda_{k+1}$ (with $\lambda_k \in L$ for $k < M$),
2. $X' \cap s[X'] = \emptyset$,
3. $X' \cap s[s[X']] = \emptyset$, and
4. $X' \cap s[s[s[X']]] = \emptyset$.

Definition 2. Given a set V of n_1 vectors $\{v_1, v_2, \ldots, v_{n_1}\} \subseteq X'$, such that $t(v_i) = \lambda_1$ for all $i = 1, 2, \ldots, n_1$, and $X' \subset X$, and a set W of n_2 vectors $\{w_1, w_2, \ldots, w_{n_2}\} \subseteq s[X']$, whereby $s[X'] \subset X$, for some *splitting function* s (and hence $X' \cap s[X'] = \emptyset$, $X' \cap s[s[X']] = \emptyset$ and $X' \cap s[s[s[X']]] = \emptyset$), we call $A = V \cup W$ an *s-partable set.*

Remark 1. Note that the label corresponding to the vectors of W, according to Def.1, is λ_2.

Remark 2. The sets V and W are disjoint as per Condition 2 of Def.1 and, thus, we assume that some feature set exists to classify the elements of A into V and W by using machine learning. In an s-partable set, we should be able to approximately classify the set $A = V \cup W$ into the subsets V (with label λ_1) and W (with label λ_2), hence the name "s-partable". This would occur if the splitting function produces some "measurable difference" when applied to the original data vectors.

Lemma 1. *Given an s-partable set $A \subseteq X' \cup s[X']$, for some splitting function s, then $s[s[X']] \cap A = \emptyset$.*

Proof.

$$\begin{aligned} s[s[X']] \cap A &\subseteq s[s[X']] \cap (X' \cup s[X']) \\ &= (s[s[X']] \cap X') \cup (s[s[X']] \cap s[X']) \\ &= \emptyset \cup \emptyset \\ &= \emptyset. \end{aligned}$$

Note that $(s[s[X']] \cap X') = \emptyset$ due to Condition 3 in Def.1 and $(s[s[X']] \cap s[X']) = \emptyset$ due to Condition 2 in Def.1 (with $Y' = s[X']$). □

Lemma 2. *Given an s-partable set $A \subseteq X' \cup s[X']$, for some splitting function s, then $s[X'] \cap s[s[A]] = \emptyset$.*

Proof.

$$\begin{aligned} s[X'] \cap s[s[A]] &\subseteq s[X'] \cap (s[s[X']] \cup s[s[s[X']]]) \\ &= (s[X'] \cap s[s[X']]) \cup (s[X'] \cap s[s[s[X']]]) \\ &= \emptyset \cup \emptyset \\ &= \emptyset. \end{aligned}$$

Note that $(s[X'] \cap s[s[X']]) = \emptyset$ due to Condition 2 in Def.1 (with $Y' = s[X']$) and $(s[X'] \cap s[s[s[X']]]) = \emptyset$ due to Condition 3 in Def.1 (again, with $Y' = s[X']$). □

Lemma 3. *Given an s-partable set $A \subseteq X' \cup s[X']$, for some splitting function s, then $A \cap s[s[A]] = \emptyset$.*

Proof. The proof directly follows from considering

$$A \cap s[s[A]] \subseteq (X' \cup s[X']) \cap (s[s[X']] \cup s[s[s[X']]]),$$

and then applying the distributive property of set algebra and Conditions 2, 3 and 4 of Def.1. □

Theorem 1. Unsupervised classification *of an s-*partable *set $A = \{a_1, a_2, \ldots, a_n\}$, formed as per Def.2 for some splitting function s, can be achieved with* supervised classification *of $B = s[A]$ as a* testing set *if the function s is known (or can be approximated) by constructing an artificial training set.*

Rationale behind the proof. The proof provided below is based on the following principles:

1. The set A of samples to be classified and the set of "double transformed" samples $C = s[s[A]]$ are disjoint as per Lemma 3. Hence, we assume that a machine learning algorithm (for some set of features) can be trained to separate A and C. Thus, we merge these two sets for training (using different labels for the elements of A and C).
2. The set of "transformed samples" $B = s[A]$ has elements either in $s[X']$ or in $s[s[X']]$. These two sets are disjoint as per Condition 2 of Def.1.
3. The intersection between the subset of the elements of B belonging to $s[X']$ and C is empty, but the intersection of the same subset with A is not. Hence, if we use the trained algorithm to classify B, there is a high probability that the elements of B belonging to $s[X']$ will be classified with the label used for the elements of A.
4. Similarly, the intersection between the subset of the elements of B belonging to $s[s[X']]$ and A is empty, but the intersection of the same subset with C is not. Hence, the elements of B belonging to $s[s[X']]$ will be classified mainly with the same label as those of C.

5. Separating the elements of B into $s[X']$ and $[s[s[X']]$ is equivalent to separating the elements of A into X' and $s[X']$, due to the existing bijection between the elements of A and B.

Proof. Since A is an *s-partable* set by definition (for the splitting function s), some elements a_i belong to the set X' and some other belong to $s[X']$. A classification function must assign the labels λ_1 and λ_2 for the elements a_i belonging to X' and $s[X']$, respectively.

Consider two additional sets $B = \{b_1, b_2, \ldots, b_n\} = s[A]$, with $b_i = s(a_i)$, and $C = \{c_1, c_2, \ldots, c_n\} = s[B] = s[s[A]]$, with $c_i = s(b_i) = s(s(a_i))$. Since $A \subseteq X' \cup s[X']$, then $B \subseteq s[X'] \cup s[s[X']]$ and $C \subseteq s[s[X']] \cup s[s[s[X']]]$. Now, construct an auxiliary *training set* T as follows:

$$T = \{(a_1, \lambda'_1), (a_2, \lambda'_1), \ldots, (a_n, \lambda'_1), (c_1, \lambda'_2), (c_2, \lambda'_2), \ldots, (c_n, \lambda'_2)\},$$

such that $T \subset (A \cup C) \times L'$ for $L' = \{\lambda'_1, \lambda'_2\}$. Note that the label λ'_1 is used for the elements of A and the label λ'_2 is used for the elements of $C = s[s[A]]$.

If we use *supervised classification* with the training set T, we can obtain a classifying function $h : T \to L'$. This classification is possible since, as proven in Lemma 3, the intersection between A and $C = s[s[A]]$ is the empty set. Now, if the function h can classify T into A and C, the same function would classify B into $s[X']$ and $s[s[X']]$, since $s[s[X']] \cap A = \emptyset$ (as proven in Lemma 1) and $s[X'] \cap C = \emptyset$ (as proven in Lemma 2). On the other hand, $s[X'] \cap A$ and $s[s[X']] \cap C$ are not empty. Note also that, by Def.1, $s[X'] \cap s[s[X']] = \emptyset$.

Thus, using the same function h, we can classify each b_i with the label λ'_1 (if it belongs to $s[X']$) or λ'_2 (if it belongs to $s[s[X']]$). Therefore, this function also classifies A into X' and $s[X']$, since $b_i = s(a_i)$. If b_i is classified with the label λ'_1, this means that $b_i = s(a_i)$ belongs to $s[X']$ and a_i belongs to X'. Similarly, if b_i is classified with the label λ'_2, then $b_i = s(a_i)$ belongs to $s[s[X']]$ and a_i belongs to $s[X']$. Hence, the classification of $b_i = s(a_i)$ as λ'_1 or λ'_2 is equivalent to the classification of a_i as λ_1 or λ_2, respectively. Consequently, we can finally classify A without labeled samples and, by definition, this is *unsupervised classification.* □

Remark 3. Unsupervised classification of A is thus achieved due to the application of the splitting function s to A (twice), which allows creating an artificial *training set* and to use *supervised classification.* Hence, if we have a *training set* and we use it to find a function $h : X \to L$ for classifying a *testing set*, by definition, this is *supervised classification.*

Remark 4. Note, however, that some degree of misclassification of the elements in B is possible, because the elements in B will not be either in A nor C. Since $A = V \cup W$ will usually be a strict subset of $X' \cup s[X']$, the elements $b_i \in s[X']$ will be outside W (and outside A), and some of them may be classified incorrectly with the label λ'_2. Similarly, the elements $b_j \in s[s[X']]$ will be outside $s[s[V]]$ (and outside C), and some of them may be classified incorrectly with the label λ'_1. This will lead to some degree of error in the unsupervised classification of B. The larger the difference introduced by the *splitting function* s is, the least likely the misclassification will become. In any case, it must be taken into account that machine learning classification processes are not error-free.

3.1. Application to Steganalysis

The application of the framework described above to steganalysis can be carried out in the following way:

- The set X is formed by samples of feature vectors of images (some of them stego and some of them cover).
- The splitting function s is the steganographic method we want to detect (using approximately the same embedding bit rate). We assume no knowledge about the secret keys of the steganographic scheme.
- The set X' represents the features of cover images, whereas the set $s[X']$ represents the features of stego images. Note that $X' \cap s[X'] = \emptyset$, since an image cannot be cover and stego at the same time.
- Note that the above condition also implies that successive applications of the embedding method produce disjoint sets. Since the condition applies to any subset $X' \subset X$, we also require, for example, that $s[X'] \cap s[s[X']] = \emptyset$. Some steganographic algorithms may not satisfy this condition. For example, LSB replacement[1] with an embedding bit rate close to 1 bit per pixel (bpp) may not produce significant differences between $s[X']$ and $s[s[X']]$. Since the proposed method is designed for targeted steganalysis, this condition can be considered known by the steganalyst.
- Another problem related to this condition is the fact that the "splitting" properties of the embedding process must be fulfilled for some specific feature set, which is used for training the artificial training set and then classifying the "transformed samples". For example, the Adaptive Steganography by Oracle (ASO) embedding algorithm (Kouider et al., 2013) is particularly designed to enhance its undetectability with respect to the features used by Ensemble Classifiers as proposed in (Kodovský et al., 2012). Hence, if we use this set of features for classification, it is likely that the suggested approach is not effective for ASO steganography.
- The set $s[s[X']]$ is formed by the features of "double stego" images and, again, we have $X' \cap s[s[X']] = \emptyset$, since an image cannot be cover and "double stego" at the same time. The same applies for "triple stego" images: $X' \cap s[s[s[X']]] = \emptyset$.
- The s-partable set $A = V \cup W$ is the testing set for the machine learning classification problem. This set is formed by some cover images (belonging to $V \subset X'$) and some stego images (belonging to $W \subset s[X']$).
- The label λ_1 corresponds to cover images, whereas λ_2 refers to stego images.

[1] Not to be mistaken for LSB matching.

- The application of the splitting function s to V obviously produces stego images with no possible intersection with the set of cover images (X'). The same thing occurs after different applications of the splitting function, producing "double stego" images, "triple stego" images, and so on.
- $B = s[A]$ is formed by "stego" ($s[V] \subset s[X']$) and "double stego" ($s[W] \subset s[s[X']]$) images.
- $C = s[B] = s[s[A]]$ is formed by "double stego" ($s[s[V]] \subset s[s[X']]$) and "triple stego" ($s[s[W]] \subset s[s[s[X']]]$) images.

With these definitions, the method described in this section can be applied for the classification of stego and cover images, yielding an unsupervised steganalytic system.

4. Experimental Results

This section presents the results obtained with the proposed method and a comparison with existing techniques in the literature to illustrate the performance of the suggested approach.

In order to test the proposed approach, we have selected eight distinct image databases:

- The BOSS database is the set of training images for the competition Break Our Steganographic System! (Filler et al., 2010). This database is formed by 10,000 cover images with a fixed size of 512×512 pixels, obtained with seven different cameras (Bas et al., 2011). The cover images were provided together with a stego set, obtained after embedding the cover images using HUGO steganography with 0.40 bpp. Hence, the whole training set was formed by 20,000 images. However, after the competition, a weakness was found (Kodovský et al., 2011) in the creation of the stego set, which can be removed embedding with a different threshold ($T = 255$ instead of the default value $T = 90$ that was used in the BOSS challenge). Therefore, we have repaired the database of 20,000 images by replacing the stego set with the 10,000 cover images embedded with HUGO steganography and 0.40 bpp, using the correct value of the threshold.
- The RANK database (often referred to as BOSSrank) is the set of testing images from the competition Break Our Steganographic System! (Filler et al., 2010). This database is formed by 1,000 cover images with a fixed size of 512×512 pixels. These images were first provided in the competition as a testing set and, hence, they were unlabeled and different from those of BOSS. 847 of the RANK images were taken using one of the cameras used to generate the BOSS database, but the remaining 153 images were taken with a camera not included in the BOSS database (Bas et al., 2011). Thus, these 153 images exhibit the CSM problem with respect to the BOSS database. After the competition, the whole cover set was released. Again, we have repaired the stego set using the correct threshold for HUGO steganography with 0.40 bpp (i.e., $T = 255$). The repaired database is formed by 942 images, 471 of which are cover and 471 stego.

- The NRCS database consists of images from the National Resource Conservation System (NRCS, n.d) with a fixed size of 2100×1500 pixels.
- The ESO database consists of images from the European Southern Observatory (ESO, n.d) with variable sizes about 1200×1200 pixels.
- The Interactions database consists of images from Interactions.org (INTE, n.d) with variable sizes about 600×400 pixels.
- The NOAA database consists of images from the National Oceanic and Atmospheric Administration (NOAA, n.d) with variable sizes about 2000×1500 pixels.
- The Albion database consists of images from the Plant Image Database of the Albion College (ALBI, n.d) with a fixed size of 1024×685 pixels.
- The Calphotos database consists of images from the Regents of the University of California (CALP, n.d) with variable sizes about 700×500 pixels.

The steganographic algorithms used in the experiments are LSB matching (Mielikäinen, 2006), with embedding bit rates of 0.25 and 0.10 bpp, HUGO (Pevný et al., 2010b), with embedding bit rates of 0.40 and 0.20 bpp, and WOW (Holub and Fridrich, 2012), with embedding bit rates of 0.40 and 0.20 bpp.

Since we are addressing targeted steganalysis, the splitting function used for the proposed unsupervised approach is the same steganographic algorithm and the same embedding bit rate, unless otherwise explicitly specified. Note, however, that although we are using the same embedding bit rate, the secret key of the steganographic algorithm (which determines the exact embedded pixels) is not used in our splitting function (since the secret key is assumed unknown by the steganalyzer).

The steganalysis approach taken in this paper stems from the well-known Rich Models framework (Fridrich and Kodovský, 2012), which proposes a feature extraction step using multiple submodels, and a classification step based on Ensemble Classifiers (Breiman, 2001; Kodovský et al., 2011, 2012). Ensemble Classifiers scale very well with the number of training samples and dimensionality, and provide similar accuracy compared to the well-known Support Vector Machine (SVM) approach if enough training samples are used (Kodovský et al., 2011, 2012). On the other hand, SVMs –particularly the Gaussian Kernel based SVM (G-SVM)– are more accurate with a relatively small number of samples and dimensions.

For a fair comparison between the traditional supervised approach and the proposed unsupervised method, we used two different classifications techniques. The supervised approach used the full Spatial domain Rich Model (SRM) feature set and was trained with the BOSS database. In most of the experiments, we used 19,000 images for training (9,500 cover and 9,500 stego), chosen randomly from the full database of 10,000 cover and 10,000 stego images. For a particular experiment (Section 4.4), we used a different training set, as explicitly detailed below.

On the other hand, it must be taken into account that the proposed method is designed for small sets of images, since we do not have a training set and, therefore, Ensemble

Classifiers are not the best choice. For this reason, we used a G-SVM together with a standard feature selection phase (34,671 dimensions are too many for training a G-SVM). More specifically, the feature selection step chooses the best 500 features based on their ANOVA F-value. Hence, we provide the best possible settings for both the supervised and the proposed methods, which allows a fair comparison between them.

Whenever SVMs were used, we chose an SVM with a Gaussian kernel. This classifier must be adjusted to provide optimal results. In particular, the values for the parameters C and γ must be selected to provide the classifier with the ability to generalize. This process was carried out as described in (Hsu et al., 2003). For all the experiments with SVMs in this paper, we used cross-validation on the training set applying the following multiplicative grid for C and γ:

$$C \in \left\{2^{-5}, 2^{-3}, 2^{-1}, 2^{1}, 2^{3}, \ldots, 2^{15}\right\},$$
$$\gamma \in \left\{2^{-15}, 2^{-13}, 2^{-11}, \ldots, 2^{-1}, 2^{1}, 2^{3}\right\}.$$

4.1. Same Number of Cover and Stego Images and CSM

To compare the results obtained with the supervised method and the proposed approach, we carried out different experiments. As mentioned above, these experiments were performed using Ensemble Classifiers in the case of supervised classification and G-SVM with feature selection for the proposed method.

In this section, all the experiments with supervised classification were performed training with a database of 19,000 images. The remaining 1,000 images were used as the testing set when the training and testing databases matched. For the proposed method, all the experiments were carried out using only the testing set.

In the first group of experiments, the testing sets consisted of 250 images, 125 of which were cover and the other 125 were stego. The aim of this experiment was to compare how the CSM problem affects the results obtained with supervised classification in contrast with those provided by the proposed method. For this reason, we used the BOSS database as training set for the supervised method and all the databases (including BOSS) for testing. When the testing database is different from BOSS, the CSM problem came along.

In Table 1, we can see the results using LSB Matching with embedding bit rates of 0.25 and 0.10 bpp, HUGO (Pevný et al., 2010b) with embedding bit rates of 0.40 and 0.20 bpp and WOW (Holub and Fridrich, 2012) with embedding bit rates of 0.40 and 0.20 bpp. In all the cases, we provide the results using both supervised classification (column "SUP") and the proposed method (Artificial Training Sets, column "ATS").

It can be observed that the suggested unsupervised approach provided the best results for almost all cases (the best results are boldfaced). Note, also, that the CSM problem is completely avoided with the proposed method, since it does not need a training database, which represents one of the major drawbacks of supervised steganalysis. The worst case occurs with the NRCS database and HUGO with an embedding bit rate of 0.20 bpp, for which none of the two methods succeeded in classifying the images correctly (yielding classification accuracies about 0.5, i.e., equivalent to random guessing).

One may think that the results obtained with the suggested method and those of the supervised approach should be almost identical when no CSM problem occurs. However, even when there was no CSM (first row of Table 1), we obtained a classification accuracy difference in favor of the suggested approach. We think that the reason for this difference, even when there is no CSM, is the fact that the artificial training step required by our method is performed exactly with the same images of the testing set. Even within the same database, there will be significant differences between different images. This will lead to relevant differences between the training and the testing sets and degraded classification results when traditional supervised methods are used. The suggested "artificial training" step is applied to exactly the same images used for testing, which prevents this "degradation".

Another experiment was conducted using the RANK database. This is the testing database of the Break Our Steganographic System! competition, formed by 1,000 images, which contain both samples from the BOSS database domain and others from a different domain. Hence, this experiment also exhibits the CSM problem. The results are shown in the first two rows of Table 2 for both the supervised approach and the proposed method. In this case the testing set was formed by 471 cover and 471 stego images. We can notice that the accuracy of the proposed method is, again, much higher than that of the supervised counterpart.

4.2. Reduced and Unbalanced Number of Stego Samples

In this section, we illustrate a real-world situation of steganalysis, namely, the lack (or a reduced number) of stego samples in the testing set. Generally, communicating parties that use steganography do not embed information in many images. Therefore, we can probably find enough cover sources from these parties, but relatively fewer stego images.

To test this scenario, we carried out different experiments similar to those presented in the previous section, but using 125 cover images and only 50 stego images in the testing set. After that, other experiments were carried out using 125 cover and only 10 stego images. The tested steganographic systems were LSB matching with 0.25 and 0.10 bpp, and HUGO with 0.40 and 0.20 bpp.

The results for 50 stego images are shown in Table 3 for LSB matching and in Table 4 for HUGO. The results using 10 stego images are shown in Table 5 for LSB matching and in Table 6 for HUGO. These tables do not only show the classification accuracy ("Acc"), but also the details about true positives ("TP"), true negatives ("TN"), false positives ("FP") and false negatives ("FN"). In both cases, we can see that the accuracy decreased compared to that obtained with a greater number of stego samples, but a remarkable level of detection was still achieved. For example, using the BOSS database and LSB matching with a 0.10 bpp bit rate, we obtained a classification accuracy of 94% when the number of stego and cover images was the same (as shown in Table 1). When we used only 50 stego images, the accuracy decreased to 78%. Finally, when we used only 10 stego images, the accuracy was further reduced to 70%. Although the accuracy decreased, a remarkable level of detection was achieved even for only 10 stego images. In particular, we notice that the number of true positives was very high with LSB matching for both cases (10 and 50 stego images), and still quite acceptable for HUGO steganography with 50 stego images. This means that

a stego image was very likely to be detected (and further analysis might be carried out to discard false positives).

The worst situation for the proposed system came out when the number of stego images was reduced to 10. In this case, the results obtained with the supervised approach were better than those of the proposed method when no CSM occurs (the first row of Tables 5 and 6). In case of CSM (the other rows), the proposed system provided better results than the supervised method for some testing databases and worse results for others. This illustrates that the performance of the proposed system decreases when the ratio between the number of stego images and the total number of testing images is very small. This situation is further analysed by means of other experiments in this section.

We also performed an experiment using the RANK database with two different ratios between cover and stego images: 70%/30% (471/202) and 90%/10% (471/52). The results, shown in Table 2, illustrate that the proposed method provided excellent detection accuracy (greater than or equal to 80%) for these cases. This shows that the accuracy is still high for unbalanced testing sets. The next experiment was performed in order to determine the threshold to obtain convenient detection results (classification accuracy) with the proposed approach as the ratio between stego and cover images is considered.

Since the classification accuracy seems to decrease when we have an unbalanced testing set, considering the relative number of cover and stego images, we carried out the next family of experiments to investigate how the proposed method behaved with different ratios of stego images. In Table 7, the detection accuracy obtained with an increasing ratio of stego images is shown for a testing set taken from the BOSS database embedded using HUGO with 0.40 bpp. We began with a testing database of 125 cover and 0 stego images (125/0) and, in each step, we removed five cover and added five stego images, until a testing set of 0 cover and 125 stego images (0/125) was formed for the last experiment. We can see that the proposed ATS method provided convenient results (accuracy over 70%) when the ratio of cover and stego images was roughly between 10% and 95%. The accuracy decreased when either the number of stego or cover images was very small (less than 10% of stego images or less than 5% of cover images). A particularly difficult situation occurred when the testing set was formed exclusively by cover images. Even in that case, the proposed approach tried to separate the testing set into cover and stego images, leading to poor classification results (only 26% of the testing images were correctly classified as cover images). This situation shall be prevented, since testing sets formed only by cover images can be very usual in real-world scenarios. This problem shall be addressed in the future research to avoid such a high level of false positives.

Another interesting real-world scenario comes along when the total number of testing images is very small. This situation may occur, for example, when only a few images are found in a USB stick and they have to be evaluated by a steganalyst. In order to test this scenario, we carried out a set of experiments using very small sets (with only 20, 15, 10 and 8 images) with variable cover/stego ratios. In Table 8, the results obtained with 1) ten cover and ten stego images, and 2) five cover and five stego images, are shown. Similarly, in Table 9, we can see the results obtained with 1) five cover and ten stego images, and 2) ten cover and five stego images. Finally, Table 10, shows the results obtained with 1) five

cover and three stego images, and 2) three cover and five stego images. The algorithms and embedding bit rates used in the experiments were LSBM with 0.25 bpp and HUGO with 0.40 bpp. The results shown in the tables are the average values obtained after repeating each experiment ten times with a different selection of images. Because of this, the true and false positive and negative values are not integers. It is worth pointing out that the results are remarkable even when the number of stego/cover images were 3/5 and 5/3, with detection accuracies close to or higher than 70%, except for the NRCS database. Taking into account the reduced size of these testing sets, we can conclude that the reliability of the proposed system is quite noteworthy.

4.3. Testing Set from Mixed Databases

This section presents an even more challenging situation for the proposed steganalysis approach. In the next experiments, the images (both cover and stego) came from different databases with an uneven proportion. The mixed data set was formed by 280 images as follows: 70 images from BOSS, 60 images from INTE, 50 images from CALP, 40 images from ESO, 30 images form ALBN, 20 images from NOAA and ten images from NRCS. Then, we selected 140 of these images as cover and the other 140 images as stego, randomly. After that, we embedded a message in the 140 images selected as stego using LSB matching with an embedding bit rate of 0.10 bpp, and removed the corresponding original (cover) images from the testing set. We can see the results after applying the proposed steganalysis method in the first row of Table 11, which shows that the classification accuracy is still quite large (79%), despite the hostile scenario.

After that, we applied the same procedure as above but using 50 stego images only (and a total of 190 images between cover and stego). In this case, the results obtained after applying the proposed steganalysis method are shown in the second row of Table 11. Since the images in the different databases have different sizes, we repeated the same experiment but with all the images clipped to 512×512 pixels, as shown in the last two rows of the table.

We can see that the only problem with the proposed approach was a relatively high number of false positives. However, even in this challenging situation, the accuracy of the suggested method was still above 70%. Again, the most remarkable property of these results is the ability of the method to provide a very large number of true positives. Note that using images with the same size the accuracy increases about 10 percentage points for the proposed method and 20 percentage points for the supervised approach. In any case, the best classification results are always those provided by the proposed method.

On the other hand, the supervised approach, trained with the BOSS database consisting of 19,000 images, does not yield good classification results, probably due to the CSM problem. Since the other databases do not have that many images, including them in the training set would not be a convenient strategy, since the training set would be unbalanced (it would contain many more images from BOSS than from the other databases). Nevertheless, there are some techniques in the state of the art that allow to deal with the CSM problem with supervised classification, such as the methods of (Pasquet et al., 2014), in which a clustering

approach is used, and (Lubenko and Ker, 2012), in which millions of images are used for training.

4.4. Testing Set with Mixed Embedding Bit Rates

As detailed in Sections 2 and 3, the proposed approach requires using the targeted steganographic system as a splitting function with approximately the same embedding bit rate. The experiment presented in this section uses three of the selected databases and the stego images were generated with the same steganographic method (LSB matching) but with five different embedding bit rates: 0.25, 0.20, 0.15, 0.10 and 0.05 bpp. We took 250 images for the testing set: 125 cover images (without any change) and 125 stego images obtained using the embedding method and one of the five possible embedding bit rates. Once embedded, the original sources were removed from the testing set. The stego image set was thus formed by 25 images for each embedding bit rate.

The main difficulty with addressing this steganalysis problem is the fact that we do not know the embedding bit rate. Even worse, the embedding bit rate is different for different images in the stego set. However, the suggested approach requires a splitting function that is approximately the same that the targeted steganographic method. Thus, we need to choose an appropriate splitting function.

It can be easily understood (from the analysis presented in Section 3) that using LSB matching with a low embedding bit rate will not conveniently separate the testing set. A stego image with a 0.10 bpp embedding bit rate would have 10% of pixels selected for embedding and, on average, half of them (a 5% of the total) would be modified compared to the corresponding cover image (the other half will already have the correct value in the LSB). If this stego image is embedded again, with an embedding bit rate of 0.10 bpp, the number of modified pixels with respect to the corresponding cover image would be, approximately, 5% (already modified) + 95% $\cdot$ 5% (modified by the second embedding) = 9.75%. This number is not exact, since some already modified pixels could be reverted to the original value in the second embedding, but it suffices to illustrate the problem. Similarly, the number of pixels modified for a cover image embedded with 0.20 bpp is 10% on average. The "double stego" image (embedded with 0.10 bpp twice) will have differences of up to ± 2 for a few pixels (about a 0.25% of them), whereas a 9.5% of them will have differences of ± 1 compared to the cover image. Hence, it would be very difficult to separate "double stego" images with 0.10 bpp from stego images with 0.20 bpp. This means that the condition $X' \cap s[X'] = \emptyset$ would not be satisfied.

On the other hand, a too large embedding rate, e.g. 0.50 bpp, would make it difficult to separate the 0.05 bpp stego images from the cover images: a 0.05 bpp stego image embedded again with a 0.50 bpp bit rate would modify approximately $2.5\% + 97.5\% \cdot 25\% = 26.875\%$ of pixels on average, compared to the corresponding cover image, most of them with a difference of ± 1, whereas a cover image embedded with a 0.50 bpp bit rate modifies 25% of the pixels, on average, with a difference of ± 1.

Hence, the selection of the appropriate embedding bit rate for the splitting function in this situation is critical. To overcome this difficulty, for this particular experiment, we used

LSB matching with 0.25 bpp as a splitting function to check if the separation property could still be achieved.

The results, shown in Table 12, illustrate that this approach leads to excellent classification results, greater than 85% in accuracy, for three different databases. For the supervised method, we used a training set formed with 9,500 cover images, and $5 \cdot 9{,}500 = 47{,}500$ stego images, since we embedded the chosen images once for each tested embedding bit rate (i.e. 0.25, 0.20, 0.15, 0.10 and 0.05 bpp). The training was carried out with the BOSS database and, hence, the results of the last two rows of the table also exhibit the CSM problem. It must be pointed out that, although the training set contained stego images embedded with all the tested bit rates, the supervised approach could not classify the testing set even when there was no CSM problem (first row).

This experiment shows that the proposed method can still be applied even if we do not know the exact embedding bit rate of the targeted steganographic system. However, a method for selecting the optimal embedding bit rate for the splitting function is required and must be addressed in the future research.

4.5. Testing Set with Unknown Message Length

The usual scenario in steganography assumes, by the Kerckhoffs' principle, that the steganalyst knows all details about the steganographic channel except the secret keys. However, in a real-world scenario, some details, such as the embedding bit rate, are rarely known. This problem is often referred to as detecting messages of unknown length (Pevný, 2011).

In the state of the art, there are quantitative steganalyzers that try to discover the embedding bit rate. In (Pevný, 2011) and (Kodovský and Fridrich, 2013), the authors use a regression algorithm for obtaining a mapping $F : \mathcal{F} \mapsto \mathcal{P} \subseteq \mathbb{R}$, where $\mathcal{F} \equiv \mathbb{R}^n$ is a feature space representation of images and $\mathcal{P}$ is a compact subset representing the space of embedding bit rates. Unfortunately, there is not a direct way of applying this framework in the proposed method. The suggested system is unsupervised and there is no training set with which we can train the regression algorithm. Despite that, in this section, we present some experiments about how to deal with a testing set of images for which the embedding bit rate is not fully known, which is a challenging situation for the proposed system. Below, we propose a methodology based on the measure of distance between the centroids of the different classes.

As a result of applying the proposed method (see Section 2 and Fig.1), we obtain a classification of B into stego and "double stego" images and, by the existing bijection between the elements of B and A, a classification of A into cover and stego images. Therefore, if the classification is successful, we expect that the elements of the stego part of A be similar to those of the stego part of B (see Fig.1d). Likewise, we expect that the elements of the cover part of A be dissimilar to those of the stego part of A (see Fig.1a). To exploit this idea, we can calculate the centroid of the cover part of A, namely $C_{A_{\text{cover}}}$, the centroid of the stego part of A, namely $C_{A_{\text{stego}}}$, and the centroid of the stego part of B, namely $C_{B_{\text{stego}}}$, and compute the distances between them: $d(C_{A_{\text{stego}}}, C_{B_{\text{stego}}})$ and $d(C_{A_{\text{cover}}}, C_{A_{\text{stego}}})$. This computation

can be carried out using a standard distance measure d, such as the Euclidean distance:

$$d(x,y) = \sqrt{\sum_{i=0}^{n}(x_i - y_i)^2}.$$

We expect short distances for $d(C_{A_{\text{stego}}}, C_{B_{\text{stego}}})$, because they are similar, and large distances for $d(C_{A_{\text{cover}}}, C_{A_{\text{stego}}})$, since they are dissimilar. Then, we can compute a score value S as follows:

$$S = \frac{d(C_{A_{\text{stego}}}, C_{B_{\text{stego}}})}{d(C_{A_{\text{cover}}}, C_{A_{\text{stego}}})}.$$

This value will be small when the classification is correct, that is, when we have managed to guess the correct algorithm and embedding bit rate. Therefore, the steganalyst can select a list of tentative bit rates and try them sequentially. The bit rate that provides the lowest score will most possibly be the right one (or close to it).

Although there are several open questions with respect to this methodology, which shall be addressed in the future research, some results are shown in Table 13. These experiments were performed using 200 images of the BOSS database, 100 of which were cover and 100 stego. Some tests were also carried out with only cover images. In that case, we only used 100 images in the testing set. The distances were calculated using 50 features, selected from the best F-values of an ANOVA test. The details of the different experiments are discussed below.

We carried out experiments with an unknown message length for three different steganographic algorithms: LSB matching, HUGO and WOW, with an embedding bit rate of 0.40 bpp in all cases. The steganalyst needed to find out what the real embedding bit rate was and, for this purpose, he/she prepared a list of tentative bit rates, namely 0.10, 0.20, 0.30, 0.40, 0.50 and 0.60 bpp. The procedure is straightforward: the steganalyst only had to apply the proposed method one time for each tentative bit rate and keep the results with lowest score. As shown Table 13, in all three cases, the lowest score indicated the correct bitrate.

4.6. Real-time Construction of the Testing Set

Another relevant scenario to consider for the proposed scheme is how to proceed when we do not have access to the whole set of images (formed with stego and cover samples) at the same time. This can be the typical situation that arises when we are eavesdropping the communications between two (or more) parties. If these parties exchange a few images (or even one image) from time to time, it is not possible to run the proposed classification method with the complete set of images, as shown in the previous sections.

In this case, the testing set A will be built dynamically. To deal with this realistic scenario, we propose the following procedure:

1. Collect the images of the testing set A one by one until $|A| \geq n_{\min}$, where $n_{\min}$ is a minimum number of images required to apply the method. Hence, the set $A = \{I_1, I_2, \ldots, I_{n_{\min}}\}$ is built, where I_j stands for the image obtained at the j-th iteration.

2. Classify the set A into cover (V) and stego (W) images applying the proposed method. Output the label of each image ("cover" or "stego").
3. When a new image I_k is obtained, add the new image into the set $A := A \cup \{I_k\}$, and repeat Step 2.

As shown in Section 4.2, $n_{\min}$ can be very small and still provide remarkable detection accuracy. In the experiments presented below, the value $n_{\min} = 10$ has been used.

Although the strategy of repeating the classification with the whole set may appear simplistic, in fact, it is the best option from the accuracy point of view. As shown below by means of several experiments, the classification accuracy increases with the number of classified images. Hence, when a new image is obtained, it is better to classify the whole set of images again and output the result of the last classification. Needless to say, this strategy also requires more computational effort. Accuracy is thus obtained in exchange for computational cost. We consider this strategy realistic, since the classification of even hundreds of images can be carried out in just a few minutes with standard hardware. If reduced computation time is a strong requirement in a real-time implementation of the scheme, accuracy can be sacrificed by selecting a subset of the collected images for each classification.

In addition, this re-classification of the whole testing set each time a new image is obtained produces a side effect: for each image, we do not only have the last classification result (as "cover" or "stego") but also the results obtained in all the previous classifications (iterations). In fact, assuming that we obtain the images one by one, if $n = |A|$ is the current number of images (i.e., I_n is the last image that has been included into the set A), we have the following number of classification results (labels), n_l, per each image:

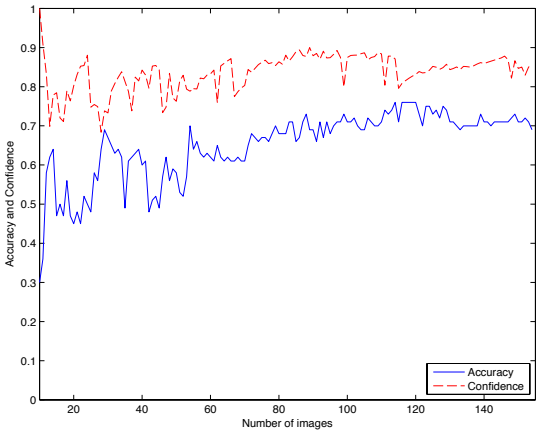
$$n_l(I_k) = \begin{cases} \max(0, n - n_{\min} + 1), & \text{if } k \leq n_{\min}, \\ \max(0, n - k + 1), & \text{otherwise.} \end{cases}$$

Now, we can compute $m_l(I_k)$ as the number of times that each image has been classified **with the same label** as in the last classification experiment. In the best case for classification "confidence", we will have $m_l(I_k) = n_l(I_k)$, that is, the image I_k has been classified with the same label in all the iterations. In the worst case, we will have $m_l(I_k) = 1$, meaning that the previous $n_l(I_k) - 1$ classification experiments yielded the opposite label compared to the current experiment. With these considerations, we can define a simple "confidence level" for the classification result of each image:

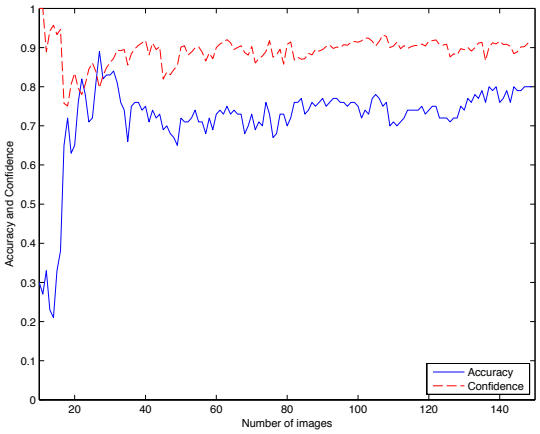
$$c(I_k) = \frac{m_l(I_k)}{n_l(I_k)}, \text{ if } n_l(I_k) > 0. \tag{1}$$

The value of the confidence level provided in Equation 1 satisfies $c(I_k) \in (0, 1]$. The closer $c(I_k)$ to 1, the more confident we can be about the classification of I_k and, conversely, the closer $c(I_k)$ to 0, the less confident we can be about the label assigned to that image. One of the advantages of such a measurement is that it can be computed without knowledge of the true category of each image. The confidence level of the classification of an image I_k always begins with $c(I_k) = 1$ the first time it is classified, and it will typically have a

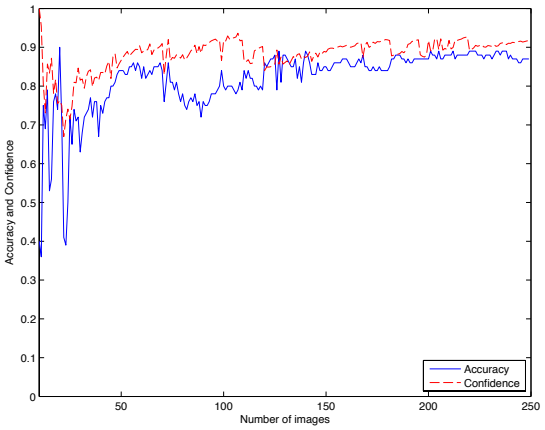
lower value as it is re-classified each time a new image is included into the set A. The values provided by this indicator are also analysed in the experiments below.



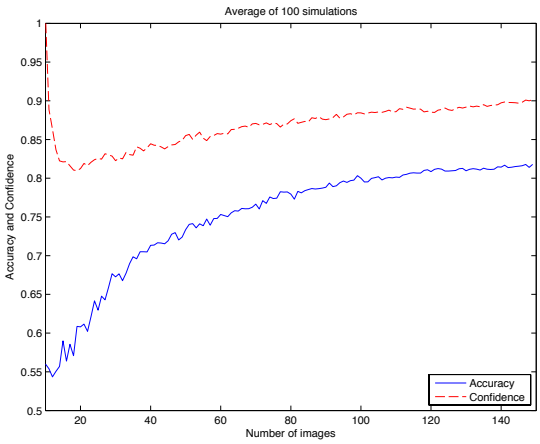
(a) 124 cover and 31 stego images



(b) 105 cover and 45 stego images



(c) 125 cover and 125 stego images



(d) 105 cover and 45 stego images (average of 100 simulations)

Figure 3: Accuracy (solid line) and confidence level (dashed line) for testing sets constructed in real time for the BOSS database using HUGO with 0.40 bpp

The first experiment to test this scenario was simulated taking 155 images, 124 of which were cover and the remaining 31 have been embedded using HUGO steganography with 0.40 bpp (i.e. 20% of the images were stego). At each iteration, a random image was selected from the set and added into A. As remarked above, we set $n_{\min} = 10$. Hence, the first classification experiment was carried out when the 10-th image was selected. The results, in terms of average classification accuracy and average confidence level are shown in Fig.3a, using a solid line for accuracy and a dashed one for confidence. At each iteration, the average accuracy and the average confidence were computed for the set of available images. As accuracy is concerned, the classification results with a few images (approximately less

than 50) were low, but when the number of images reaches this threshold, the accuracy increased, reaching a 70% (0.7) in the last iteration. On the other hand, the confidence, started from a high value, showed a significant decrease in the first few iterations and at some iteration (again about 50 images) started a quite regular increase. We can also observe a strong correlation between both curves, specially when the number of collected images was significant enough. This situation is really remarkable, since in a real experiment we would not know the real accuracy, but we would be able to compute the confidence measurement, since it only depends on the past classification results. When the confidence reaches a "steady state" behavior, the same occurs with the accuracy. Hence, this provides with a strong indicator of the quality of the classification results.

The same procedure was applied for other testing sets of images. Fig.3b shows the results obtained with 150 images, 105 of which were cover and the remaining 45 stego (i.e. 30% of stego images). The behavior of both variables was similar to that of the previous experiment, but the threshold in the number of images required for a high accuracy and confidence was lower (about 30 images). In this case, the final accuracy was also higher, about 80% (0.8).

Fig. 3c provides the results obtained for 250 images, 125 of which were stego (i.e. 50% of stego images). The situation was similar to that of the previous two experiments. From a certain iteration (again about 30 images) the accuracy and confidence curves showed a more regular behavior, mostly increasing. In this case, the final accuracy was higher, close to 90% (0.9), due to the larger number of images in the final testing set (250 images, compared to 155 for the first experiment and 150 for the second one).

The correlation between accuracy and confidence could also be observed for the iterations in which there was a sudden variation in both variables. When accuracy shows a sudden increase or decrease, a similar situation occurs with confidence, but the increase or decrease was sometimes reversed compared to accuracy. There is a simple explanation for this situation. A large variation in confidence means that many images changed their label at that iteration. This change of label could be right, leading to an increased accuracy, or wrong, leading to a decrease in accuracy. Only occasionally, these variations were compensated as right and wrong classification results (leading to small variations in accuracy). It can also be observed that the higher the number of images, the more likely the variations in accuracy and confidence had the same sign. This explains the strong positive correlation between both curves when the number of images reaches a minimum threshold (about 30-50).

The results shown in Figs.3a-3c provide only one simulation in each case. Such a simulated experiment tries to reproduce the situation that we would find in a real-life experiment, i.e. the images would be obtained one by one, or in a small number at each iteration. However, the results obtained in this way can be biased due to the generation of the pseudo-random numbers used to select the image order. In order to avoid this bias, we carried out 100 simulations with the same set of 105 cover and 45 stego images, but using a different seed for the pseudo-random selection (ordering) of the images at each simulation. Fig.3d shows the averaged results of these 100 simulations for both accuracy and confidence (please note the scale change in the vertical axis with respect to the other three cases). In this figure, we can notice that accuracy became a much "softer" curve, starting from low values (about 55%) and reaching high accuracy (over 80%) for the complete set. The increase in accuracy

is almost monotonic, and the oscillations were almost suppressed by averaging the results of 100 experiments. In addition, we can also observe the same "softer" shape of the confidence measurement. The correlation between accuracy and confidence when the number of images was larger than 30 was almost perfect. Both curves increased approximately linearly, with roughly the same slope.

These experiments suggest that using all the available images increases the accuracy and justifies the method proposed for a real-time implementation of our scheme. No accuracy gain can be expected, a priori, from discarding some images from the testing set.

It must be taken into account that if the ratio of stego images varies too much in real-time, the value of accuracy may not increase as regularly as shown in Fig.3. As discussed in Section 4.2, if the percentage of stego images is below 10% or above 95%, the detection accuracy results may decrease significantly. This situation may occur if, from some given moment, (nearly) only cover or (nearly) only stego images were obtained, leading to a very low or a very high percentage of stego images. This kind of scenario must be prevented to maintain the accuracy results in acceptable values.

5. Conclusion

In this paper, a novel unsupervised steganalysis method is presented. We show how unsupervised steganalysis can be addressed by using an artificial training set and supervised classification if we know the algorithm and the embedding bit rate used for steganography. Hence, the suggested method is applicable in targeted steganalysis. Using the proposed approach, we can also bypass the CSM problem and outperform the state-of-the-art methods. Removing the necessity of a training data set in the machine learning problem is the major contribution of this paper.

The proposed approach has been tested using three steganographic methods: LSB matching, HUGO and WOW. It is shown that we can achieve better classification accuracy than that obtained using traditional supervised steganalysis (Rich Models, Ensemble Classifiers and SVM), while avoiding the CSM problem that makes the performance of supervised steganalysis decrease significantly.

Furthermore, through the different experiments presented in the paper, we show that the proposed method can address complex real-world situations, in which we do not have a clear training database (e.g., when the images come from different databases), the number of stego images is reduced or the images are obtained one by one, in real time. We show that the suggested method provides remarkable performance even if the images are selected unevenly from different databases or if the embedding bit rate is unknown and variable for different stego samples.

As future work, it would be worth researching how the suggested method can be applied in situations for which we do not know the image database, the steganographic algorithm or the embedding bit rate used, as it would occur in real-world steganalysis. The experiments with images taken unevenly from different databases show some decrease in the accuracy results of the method. Besides, if the message length is unknown, the approaches presented in the paper shall be analyzed more deeply, though the preliminary results are promising.

Similarly, the case when the testing set is formed only by cover images needs be specifically addressed to avoid a large number of false positives. In addition, it would also be interesting to investigate how to proceed when we cannot estimate the splitting function, e.g. when we deal with a novel and unknown steganographic scheme. This would mean porting the proposed approach to the problem of universal steganalysis.

Finally, we propose to investigate the application of this approach in other fields, beyond steganalysis, with similar properties. The idea behind the proposed method could be exploited whenever a splitting function can be defined in a machine learning classification problem.

Acknowledgment

This work was partly funded by the Spanish Government through grants TIN2011-27076-C03-02 "CO-PRIVACY" and TIN2014-57364-C2-2-R "SMARTGLACIS".

References

ALBI, n.d. Plant Image DataBase from Albion College. Available online at: `http://www4.albion.edu/plants/`, accessed on May 24, 2016. [Online].

Bas, P., Filler, T., Pevný, T., 2011. "break our steganographic system": The ins and outs of organizing boss. In: Proceedings of the 13th International Conference on Information Hiding. IH'11. Springer-Verlag, pp. 59–70.

Bishop, C. M., 2006. Pattern Recognition and Machine Learning. Information Science and Statistics Series. Springer.

Breiman, L., 2001. Random Forests. Machine Learning 45 (1), 5–32.

CALP, n.d. Regents of the University of California, Berkeley. Available: `http://calphotos.berkeley.edu/`, accessed on May 24, 2016. [Online].

Cancelli, G., Doërr, G., Barni, M., Cox, I. J., 2008. A Comparative Study of ±1 Steganalyzers. In: Multimedia Signal Processing, 2008 IEEE 10th Workshop on. pp. 791–796.

Chaumont, M., Kouider, S., 2012. Steganalysis by Ensemble Classifiers with Boosting by Regression, and Post-Selection of Features. In: Image Processing (ICIP), 2012 19th IEEE International Conference on. pp. 1133–1136.

ESO, n.d. European Southern Observatory. Available: `http://www.eso.org/public/images/`, accessed on May 24, 2016. [Online].

Filler, T., Pevný, T., Bas, P., 2010. Break our Steganographic System (BOSS). `http://exile.felk.cvut.cz/boss/`, accessed on September 21, 2012.

Fridrich, J., Kodovský, J., 2012. Rich Models for Steganalysis of Digital Images. IEEE Trans. Information Forensics and Security 7 (3), 868–882.

Fridrich, J., Kodovský, J., Holub, V., Goljan, M., 2011. Breaking HUGO: The Process Discovery. In: Proceedings of the 13th International Conference on Information Hiding. IH'11. Springer-Verlag, pp. 85–101.

Gul, G., Kurugollu, F., 2011. A New Methodology in Steganalysis : Breaking Highly Undetectable Steganography (HUGO). In: Proceedings of the 13th International Conference on Information Hiding. IH'11. Springer-Verlag, pp. 71–84.

Holub, V., Fridrich, J. J., 2012. Designing Steganographic Distortion Using Directional Filters. In: International Workshop on Information Forensics and Security (WIFS). IEEE, pp. 234–239.

Hsu, C.-W., Chang, C.-C., Lin, C.-J., 2003. A Practical Guide to Support Vector Classification. Tech. rep., Department of Computer Science, National Taiwan University, accessed on May 24, 2016. [Online].

INTE, n.d. Interactions.org Particle Physics News and Resources. Available: http://www.interactions.org/cms/?pid=1900, accessed on May 24, 2016. [Online].

Karakiş, R., Güller, I., Çapraz, I., Bilir, E., 2015. A novel fuzzy logic-based image steganography method to ensure medical data security. Computers in Biology and Medicine 67, 172–183.

Ker, A. D., Bas, P., Böhme, R., Cogranne, R., Craver, S., Filler, T., Fridrich, J., Pevný, T., 2013. Moving Steganography and Steganalysis from the Laboratory into the Real World. In: Proceedings of the First ACM Workshop on Information Hiding and Multimedia Security. ACM, pp. 45–58.

Ker, A. D., Pevný, T., 2014. A Mishmash of Methods for Mitigating the Model Mismatch Mess. Proceedings of SPIE - The International Society for Optical Engineering" 9028, 90280I–90280I–15.

Kodovský, J., Fridrich, J., 2013. Quantitative Steganalysis Using Rich Models. Proceedings of SPIE - The International Society for Optical Engineering" 8665, 86650O–86650O–11.

Kodovský, J., Fridrich, J., Holub, V., 2011. On Dangers of Overtraining Steganography to Incomplete Cover Model. In: Proceedings of the Thirteenth ACM Multimedia Workshop on Multimedia and Security. MM&Sec '11. ACM, pp. 69–76.

Kodovský, J., Fridrich, J. J., Holub, V., 2012. Ensemble Classifiers for Steganalysis of Digital Media. IEEE Transactions on Information Forensics and Security 7 (2), 432–444.

Kodovský, J., Sedighi, V., Fridrich, J., 2014. Study of Cover Source Mismatch in Steganalysis and Ways to Mitigate its Impact. Proceedings of SPIE - The International Society for Optical Engineering" 9028.

Kouider, S., Chaumont, M., Puech, W., 2013. Adaptive steganography by oracle (ASO). In: International Conference on Multimedia and Expo (ICME). IEEE, pp. 1–6.

Lubenko, I., Ker, A. D., 2012. Going from Small to Large Data in Steganalysis. In: Media Watermarking, Security, and Forensics 2012. Vol. 8303 of Proceedings of SPIE - The International Society for Optical Engineering. pp. 0M01–0M10.

Mielikäinen, J., 2006. LSB Matching Revisited. Signal Processing Letters 13, 285–287.

Mitchell, T. M., 1997. Machine Learning. McGraw-Hill, Inc.

NOAA, n.d. National Oceanic and Atmospheric Administration (NOAA). Available: http://www.photolib.noaa.gov/, accessed on May 24, 2016. [Online].

NRCS, n.d. National Resource Conservation System (NRCS) Photo Gallery. Available: http://photogallery.nrcs.usda.gov/res/sites/photogallery/, accessed on May 24, 2016. [Online].

Pasquet, J., Bringay, S., Chaumont, M., 2014. Steganalysis with Cover-Source Mismatch and a Small Learning Database. In: Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European. pp. 2425–2429.

Pevný, T., 2011. Detecting Messages of Unknown Length. Proceedings of SPIE - The International Society for Optical Engineering" 7880, 78800T–78800T–12.

Pevný, T., Bas, P., Fridrich, J. J., 2010a. Steganalysis by Subtractive Pixel Adjacency Matrix. IEEE Transactions on Information Forensics and Security 5, 215–224.

Pevný, T., Filler, T., Bas, P., 2010b. Using High-Dimensional Image Models to Perform Highly Undetectable Steganography. In: Information Hiding - 12th International Conference. pp. 161–177.

Pevný, T., Ker, A. D., 2013. The Challenges of Rich Features in Universal Steganalysis. Proceedings of SPIE - The International Society for Optical Engineering 8665, 86650M–86650M–15.

Comparative analysis between supervised classification and ATS												
	LSB Matching				HUGO				WOW			
	0.25 bpp		0.10 bpp		0.40 bpp		0.20 bpp		0.40 bpp		0.20 bpp	
Test DB	SUP	ATS	SUP	ATS	SUP	ATS	SUP	ATS	SUP	ATS	SUP	ATS
BOSS	0.96	**0.98**	0.90	**0.94**	0.78	**0.87**	0.67	**0.79**	0.82	**0.94**	0.62	**0.82**
ESO	0.37	**0.94**	0.50	**0.98**	0.50	**0.83**	0.45	**0.86**	0.53	**0.98**	0.50	**0.94**
CALP	0.51	**0.96**	0.57	**0.94**	0.48	**0.90**	0.48	**0.84**	0.52	**0.95**	0.51	**0.85**
INTE	0.47	**0.98**	0.50	**0.98**	0.51	**0.95**	0.51	**0.95**	0.50	**1.00**	0.50	**0.96**
NRCS	0.49	**0.68**	0.55	**0.61**	0.49	**0.63**	**0.50**	0.46	0.50	**0.86**	0.50	**0.68**
ALBN	0.57	**0.99**	0.65	**0.98**	0.50	**0.97**	0.50	**0.91**	0.54	**0.95**	0.49	**0.92**
NOAA	0.35	**1.00**	0.50	**0.98**	0.50	**1.00**	0.44	**0.96**	0.50	**0.98**	0.50	**1.00**

Table 1: Accuracy of supervised classification ("SUP") compared with the proposed method ("ATS") for different image databases ("DB") embedded with different algorithms and bit rates

BOSS RANK Experiments						
HUGO 0.40 bpp						
Method	**Cover/Stego**	**Acc**	**TP**	**TN**	**FP**	**FN**
SUP	471/471	0.61	461	114	357	10
ATS	471/471	0.86	394	414	57	77
ATS 70/30	471/202	0.84	176	391	80	26
ATS 90/10	471/52	0.81	43	382	89	9

Table 2: Classification results of different experiments performed with supervised classification ("SUP") and the proposed method ("ATS") using the RANK database embedded with HUGO and 0.40 bpp: Accuracy ("Acc"), True positives ("TP"), True negatives ("TN"), False positives ("FP") and False negatives ("FN")

LSB Matching Steganography with 50 stego images																				
	SUP 0.25 bpp					**ATS 0.25 bpp**					**SUP 0.10 bpp**					**ATS 0.10 bpp**				
DB	**Acc**	**TP**	**TN**	**FP**	**FN**	**Acc**	**TP**	**TN**	**FP**	**FN**	**Acc**	**TP**	**TN**	**FP**	**FN**	**Acc**	**TP**	**TN**	**FP**	**FN**
BOSS	**0.95**	47	119	6	3	0.85	48	100	25	2	**0.89**	47	108	17	3	0.78	48	89	36	2
ESO	0.47	12	71	54	38	**0.82**	47	96	29	3	0.30	45	8	117	5	**0.89**	48	108	17	2
CALP	0.72	4	117	0	46	**0.94**	43	121	4	7	0.72	16	105	12	34	**0.85**	37	112	13	13
INTE	0.66	2	114	11	48	**0.97**	49	120	5	1	0.29	50	0	125	0	**0.92**	49	112	13	1
NRCS	**0.66**	5	103	11	45	0.59	41	63	62	9	0.39	40	24	90	10	**0.52**	37	54	71	13
ALBI	0.61	22	84	41	28	**0.98**	50	122	3	0	0.55	42	55	70	8	**0.97**	49	121	4	1
NOAA	0.46	17	64	61	33	**0.89**	50	105	20	0	0.29	50	0	125	0	**0.95**	49	117	8	1

Table 3: Classification results of the supervised approach ("SUP") and the proposed method ("ATS") for different databases ("DB") embedded with LSB matching steganography using only 50 stego images: Accuracy ("Acc"), True positives ("TP"), True negatives ("TN"), False positives ("FP") and False negatives ("FN")

HUGO Steganography with 50 stego images																				
	SUP 0.40 bpp					**ATS 0.40 bpp**					**SUP 0.20 bpp**					**ATS 0.20 bpp**				
DB	**Acc**	**TP**	**TN**	**FP**	**FN**	**Acc**	**TP**	**TN**	**FP**	**FN**	**Acc**	**TP**	**TN**	**FN**	**FP**	**Acc**	**TP**	**TN**	**FP**	**FN**
BOSS	0.77	43	91	34	7	**0.79**	46	92	33	4	**0.63**	36	75	50	14	0.62	40	69	56	10
ESO	0.43	34	41	84	16	**0.82**	49	95	30	1	0.33	34	23	102	16	**0.76**	47	86	39	3
CALP	0.44	21	51	66	24	**0.77**	36	89	28	9	0.49	13	66	51	32	**0.69**	37	75	42	8
INTE	0.61	22	85	40	28	**0.94**	50	114	11	0	0.43	27	48	77	23	**0.89**	49	106	19	1
NRCS	0.35	29	25	89	11	**0.68**	36	68	46	4	0.49	25	51	63	15	**0.72**	7	104	10	33
ALBI	0.30	48	5	120	2	**0.95**	49	117	8	1	0.32	46	10	115	4	**0.74**	42	87	38	8
NOAA	0.31	49	6	119	1	**0.98**	49	123	2	1	0.30	49	3	122	1	**0.90**	49	109	16	1

Table 4: Classification results of the supervised approach ("SUP") and the proposed method ("ATS") for different databases ("DB") embedded with HUGO steganography using only 50 stego images: Accuracy ("Acc"), True positives ("TP"), True negatives ("TN"), False positives ("FP") and False negatives ("FN")

LSB Matching Steganography with 10 stego images																				
	SUP 0.25 bpp					**ATS 0.25 bpp**					**SUP 0.10 bpp**					**ATS 0.10 bpp**				
DB	**Acc**	**TP**	**TN**	**FP**	**FN**	**Acc**	**TP**	**TN**	**FP**	**FN**	**Acc**	**TP**	**TN**	**FP**	**FN**	**Acc**	**TP**	**TN**	**FP**	**FN**
BOSS	**0.96**	10	119	6	0	0.86	10	106	19	0	**0.87**	9	108	17	1	0.70	9	86	39	1
ESO	0.54	2	71	54	8	**0.82**	10	101	24	0	0.13	9	8	117	1	**0.58**	10	68	57	0
CALP	**0.92**	0	117	0	10	0.29	6	33	92	4	**0.87**	5	105	12	5	0.26	4	31	94	6
NTE	**0.84**	0	114	11	10	0.64	8	78	47	2	0.07	10	0	125	0	**0.62**	10	74	51	0
NRCS	**0.83**	0	103	11	10	0.47	9	54	71	1	0.26	8	24	90	2	**0.41**	8	48	77	2
ALBI	0.69	9	84	41	1	**0.91**	10	113	12	0	0.47	8	55	70	2	**0.81**	10	100	25	0
NOAA	0.48	1	64	61	9	**0.72**	9	88	37	1	0.07	10	0	125	0	**0.73**	10	88	37	0

Table 5: Classification results of the suppervised approach ("SUP") and the proposed method ("ATS") for different databases ("DB") embedded with LSB matching steganography using only 10 stego images: Accuracy ("Acc"), True positives ("TP"), True negatives ("TN"), False positives ("FP") and False negatives ("FN")

HUGO Steganography with 10 stego images																				
	SUP 0.40 bpp					**ATS 0.40 bpp**					**SUP 0.20 bpp**					**ATS 0.20 bpp**				
DB	**Acc**	**TP**	**TN**	**FP**	**FN**	**Acc**	**TP**	**TN**	**FP**	**FN**	**Acc**	**TP**	**TN**	**FP**	**FN**	**Acc**	**TP**	**TN**	**FP**	**FN**
BOSS	**0.75**	10	91	34	0	0.58	10	68	57	0	**0.60**	6	75	50	4	0.51	10	59	66	0
ESO	**0.36**	7	41	84	3	0.21	10	19	106	0	0.22	7	23	102	3	**0.43**	9	49	76	1
CALP	**0.40**	0	51	66	9	0.19	4	20	97	5	**0.52**	0	66	51	9	0.16	3	17	100	6
INTE	**0.64**	2	85	40	8	0.24	10	23	102	0	**0.38**	3	48	77	7	0.27	10	27	98	0
NRCS	0.26	7	25	89	3	**0.51**	8	55	59	2	0.45	5	51	63	5	**0.49**	8	53	61	2
ALBI	0.11	10	5	120	0	**0.65**	10	78	47	0	0.15	10	10	115	0	**0.57**	9	68	57	1
NOAA	0.11	9	6	119	1	**0.60**	10	71	54	0	0.09	9	3	122	1	**0.76**	10	93	32	0

Table 6: Classification results of the supervised approach ("SUP") and the proposed method ("ATS") for different databases ("DB") embedded with HUGO steganography using only 10 stego images: Accuracy ("Acc"), True positives ("TP"), True negatives ("TN"), False positives ("FP") and False negatives ("FN")

Variable ratio of stego images						
HUGO 0.40 bpp						
Cover/Stego	**Percent of stego**	**Acc**	**TP**	**TN**	**FP**	**FN**
125 / 0	0%	0.26	0	33	92	0
120 / 5	4%	0.54	5	63	57	0
115 / 10	8%	0.53	10	56	59	0
110 / 15	12%	0.73	15	76	34	0
105 / 20	16%	0.73	19	72	33	1
100 / 25	20%	0.76	23	72	28	2
95 / 30	24%	0.81	28	73	22	2
90 / 35	28%	0.82	31	72	18	4
85 / 40	32%	0.80	37	63	22	3
80 / 45	36%	0.82	42	61	19	3
75 / 50	40%	0.81	44	57	18	6
70 / 55	44%	0.87	52	57	13	3
65 / 60	48%	0.81	51	50	15	9
60 / 65	52%	0.85	58	48	12	7
55 / 70	56%	0.84	62	43	12	8
50 / 75	60%	0.85	61	45	5	14
45 / 80	64%	0.85	67	39	6	13
40 / 85	68%	0.84	71	34	6	14
35 / 90	72%	0.86	74	33	2	16
30 / 95	76%	0.85	80	26	4	15
25 / 100	80%	0.82	82	20	5	18
20 / 105	84%	0.80	80	20	0	25
15 / 110	88%	0.79	85	14	1	25
10 / 115	92%	0.75	86	8	2	29
5 / 120	96%	0.75	89	5	0	31
0 / 125	100%	0.65	81	0	0	44

Table 7: Classification results of the proposed method with the BOSS database ("DB") embedded with HUGO and 0.40 bpp: Accuracy ("Acc"), True positives ("TP"), True negatives ("TN"), False positives ("FP") and False negatives ("FN")

HUGO/LSBM with few testing images						
Cover/Stego	**DB/Algorithm**	**Acc**	**TP**	**TN**	**FP**	**FN**
10/10	BOSS / LSBM 0.25	0.89	9.1	8.8	1.2	0.9
10/10	BOSS / HUGO 0.40	0.73	7.8	6.8	3.2	2.2
10/10	ESO / LSBM 0.25	0.82	8.6	7.9	2.1	1.4
10/10	ESO / HUGO 0.40	0.68	7.8	5.8	4.2	2.2
10/10	CALP / LSBM 0.25	0.84	8.5	8.3	1.7	1.5
10/10	CALP / HUGO 0.40	0.80	7.7	7.4	2.2	1.5
10/10	INTE / LSBM 0.25	0.94	10	8.8	1.2	0
10/10	INTE / HUGO 0.40	0.77	9.3	6.1	3.9	0.7
10/10	NRCS / LSBM 0.25	0.53	6.3	4.3	5.7	3.7
10/10	NRCS / HUGO 0.40	0.53	4.7	5.2	4.3	4.2
10/10	ALBN / LSBM 0.25	0.93	9.4	9.2	0.8	0.6
10/10	ALBN / HUGO 0.40	0.85	9.2	7.9	2.1	0.8
10/10	NOAA / LSBM 0.25	0.93	9.6	9.1	0.9	0.4
10/10	NOAA / HUGO 0.40	0.83	9.2	7.5	2.5	0.8
5/5	BOSS / LSBM 0.25	0.83	4.0	4.3	0.7	1.0
5/5	BOSS / HUGO 0.40	0.61	3.2	2.9	2.1	1.8
5/5	ESO / LSBM 0.25	0.71	4.0	3.1	1.9	1.0
5/5	ESO / HUGO 0.40	0.68	3.9	2.9	2.1	1.1
5/5	CALP / LSBM 0.25	0.79	3.7	4.2	0.8	1.3
5/5	CALP / HUGO 0.40	0.63	3.6	2.3	2.4	1.0
5/5	INTE / LSBM 0.25	0.86	4.9	3.7	1.3	0.1
5/5	INTE / HUGO 0.40	0.73	4.4	2.9	2.1	0.6
5/5	NRCS / LSBM 0.25	0.51	3.5	1.6	3.4	1.5
5/5	NRCS / HUGO 0.40	0.59	2.7	2.6	2.0	1.7
5/5	ALBN / LSBM 0.25	0.85	4.3	4.2	0.8	0.7
5/5	ALBN / HUGO 0.40	0.77	4.1	3.6	1.4	0.9
5/5	NOAA / LSBM 0.25	0.82	4.5	3.7	1.3	0.5
5/5	NOAA / HUGO 0.40	0.78	4.6	3.2	1.8	0.4

Table 8: Average classification results of the proposed method for different databases ("DB") with HUGO and LSB matching steganography for very small testing sets: Accuracy ("Acc"), True positives ("TP"), True negatives ("TN"), False positives ("FP") and False negatives ("FN")

HUGO/LSBM with few testing images						
Cover/Stego	**DB/Algorithm**	**Acc**	**TP**	**TN**	**FP**	**FN**
5/10	BOSS / LSBM 0.25	0.77	7.8	3.8	1.2	2.2
5/10	BOSS / HUGO 0.40	0.69	7.4	3.0	2.0	2.6
5/10	ESO / LSBM 0.25	0.80	8.7	3.4	1.6	1.3
5/10	ESO / HUGO 0.40	0.77	7.8	3.8	1.2	2.2
5/10	CALP / LSBM 0.25	0.77	7.8	3.9	1.1	2.2
5/10	CALP / HUGO 0.40	0.83	8.1	3.8	0.7	1.7
5/10	INTE / LSBM 0.25	0.93	9.6	4.4	0.6	0.4
5/10	INTE / HUGO 0.40	0.75	7.6	3.7	1.3	2.4
5/10	NRCS / LSBM 0.25	0.61	7.0	2.3	2.7	3.0
5/10	NRCS / HUGO 0.40	0.49	4.1	2.4	2.3	4.5
5/10	ALBN / LSBM 0.25	0.92	9.5	4.4	0.6	0.5
5/10	ALBN / HUGO 0.40	0.86	8.6	4.3	0.7	1.4
5/10	NOAA / LSBM 0.25	0.94	9.6	4.5	0.5	0.4
5/10	NOAA / HUGO 0.40	0.89	8.9	4.5	0.5	1.1
10/5	BOSS / LSBM 0.25	0.81	4.5	7.7	2.3	0.5
10/5	BOSS / HUGO 0.40	0.59	3.4	5.5	4.5	1.6
10/5	ESO / LSBM 0.25	0.73	4.7	6.3	3.7	0.3
10/5	ESO / HUGO 0.40	0.66	4.2	5.8	4.2	0.8
10/5	CALP / LSBM 0.25	0.82	3.9	8.4	1.6	1.1
10/5	CALP / HUGO 0.40	0.57	3.5	4.7	4.6	1.5
10/5	INTE / LSBM 0.25	0.87	4.9	8.2	1.8	0.1
10/5	INTE / HUGO 0.40	0.69	4.9	5.6	4.4	0.1
10/5	NRCS / LSBM 0.25	0.48	3.6	3.6	6.4	1.4
10/5	NRCS / HUGO 0.40	0.52	2.8	4.3	5.0	1.6
10/5	ALBN / LSBM 0.25	0.88	4.7	8.6	1.4	0.3
10/5	ALBN / HUGO 0.40	0.76	4.7	6.7	3.3	0.3
10/5	NOAA / LSBM 0.25	0.88	4.8	8.5	1.5	0.2
10/5	NOAA / HUGO 0.40	0.71	4.7	6.1	3.9	0.3

Table 9: Average classification results of the proposed method for different databases ("DB") with HUGO and LSB matching steganography for very small testing sets: Accuracy ("Acc"), True positives ("TP"), True negatives ("TN"), False positives ("FP") and False negatives ("FN")

HUGO/LSBM with few testing images						
Cover/Stego	**DB/Algorithm**	**Acc**	**TP**	**TN**	**FP**	**FN**
5/3	BOSS / LSBM 0.25	0.75	2.5	3.5	1.5	0.5
5/3	BOSS / HUGO 0.40	0.55	2.1	2.3	2.7	0.9
5/3	ESO / LSBM 0.25	0.64	2.7	2.4	2.6	0.3
5/3	ESO / HUGO 0.40	0.59	2.4	2.3	2.7	0.6
5/3	CALP / LSBM 0.25	0.75	1.9	4.1	0.9	1.1
5/3	CALP / HUGO 0.40	0.61	2.4	2.3	2.4	0.6
5/3	INTE / LSBM 0.25	0.77	3.0	3.2	1.8	0
5/3	INTE / HUGO 0.40	0.71	2.9	2.8	2.2	0.1
5/3	NRCS / LSBM 0.25	0.45	2.1	1.5	3.5	0.9
5/3	NRCS / HUGO 0.40	0.49	1.3	2.3	2.3	1.2
5/3	ALBN / LSBM 0.25	0.65	2.3	2.9	2.1	0.7
5/3	ALBN / HUGO 0.40	0.64	2.5	2.6	2.4	0.5
5/3	NOAA / LSBM 0.25	0.79	2.9	3.4	1.6	0.1
5/3	NOAA / HUGO 0.40	0.61	3.0	1.9	3.1	0
3/5	BOSS / LSBM 0.25	0.81	4.3	2.2	0.8	0.7
3/5	BOSS / HUGO 0.40	0.58	3.1	1.6	1.4	1.9
3/5	ESO / LSBM 0.25	0.67	3.8	1.6	1.4	1.2
3/5	ESO / HUGO 0.40	0.74	4.5	1.4	1.6	0.5
3/5	CALP / LSBM 0.25	0.69	3.5	2.0	1.0	1.5
3/5	CALP / HUGO 0.40	0.61	2.9	1.4	1.1	1.6
3/5	INTE / LSBM 0.25	0.88	5.0	2.1	0.9	0
3/5	INTE / HUGO 0.40	0.84	4.6	2.1	0.9	0.4
3/5	NRCS / LSBM 0.25	0.54	3.5	0.8	2.2	1.5
3/5	NRCS / HUGO 0.40	0.54	2.7	1.4	1.6	1.9
3/5	ALBN / LSBM 0.25	0.64	3.5	1.6	1.4	1.5
3/5	ALBN / HUGO 0.40	0.79	3.9	2.4	0.6	1.1
3/5	NOAA / LSBM 0.25	0.86	4.8	2.1	0.9	0.2
3/5	NOAA / HUGO 0.40	0.65	3.4	1.8	1.2	1.6

Table 10: Average classification results of the proposed method for different databases ("DB") with HUGO and LSB matching steganography for very small testing sets: Accuracy ("Acc"), True positives ("TP"), True negatives ("TN"), False positives ("FP") and False negatives ("FN")

LSB Matching Embedding with mixed DBs (**0.10** bpp)										
	SUP					**ATS**				
Stego/Total samples	**Acc**	**TP**	**TN**	**FP**	**FN**	**Acc**	**TP**	**TN**	**FP**	**FN**
140/280	0.47	78	54	84	62	**0.79**	116	106	34	24
50/190	0.42	25	54	84	25	**0.74**	50	91	49	0
140/280 (same size)	0.69	122	72	67	18	**0.87**	122	122	18	18
50/190 (same size)	0.61	44	72	67	6	**0.78**	46	103	37	4

Table 11: Classification results of the supervised approach ("SUP") and the proposed method ("ATS") in an unevenly mixed database ("DB") embedded with LSB matching and 0.10 bpp: Accuracy ("Acc"), True positives ("TP"), True negatives ("TN"), False positives ("FP") and False negatives ("FN")

LSB Matching Steganography with Multiple Embedding Bit Rates **0.25, 0.20, 0.15, 0.10 and 0.05 bpp**										
	SUP					**ATS**				
DB	**Acc**	**TP**	**TN**	**FP**	**FN**	**Acc**	**TP**	**TN**	**FP**	**FN**
BOSS	0.53	124	9	116	1	**0.87**	99	119	6	26
CALP	0.53	81	47	70	44	**0.89**	103	120	5	22
INTE	0.47	117	0	125	8	**0.88**	100	119	6	25

Table 12: Classification results of the supervised approach ("SUP") and proposed method ("ATS") using multiple embedding bit rates with three different databases ("DB"): Accuracy ("Acc"), True positives ("TP"), True negatives ("TN"), False positives ("FP") and False negatives ("FN")

Unknown Message Length				
Algorithm	**Real bit rate (bpp)**	**Tentative bit rate (bpp)**	**Score**	**Accuracy**
LSBM	0.40	0.10	0.66	0.72
		0.20	0.52	0.84
		0.30	0.25	0.89
		0.40	**0.04**	0.89
		0.50	0.41	0.82
		0.60	1.19	0.85
HUGO	0.40	0.10	0.79	0.57
		0.20	1.05	0.57
		0.30	0.67	0.77
		0.40	**0.24**	0.81
		0.50	0.37	0.76
		0.60	0.48	0.63
WOW	0.40	0.10	0.86	0.53
		0.20	0.80	0.59
		0.30	0.69	0.73
		0.40	**0.40**	0.76
		0.50	0.45	0.71
		0.60	0.58	0.62

Table 13: Classification results of different experiments using the BOSS database embedded with LSBM, HUGO and WOW with an unknown message length

3.5 Manifold alignment approach to cover source mismatch in steganalysis

D. Lerch-Hostalot and D. Megías. Manifold alignment approach to cover source mismatch in steganalysis. In *Actas de la XIV Reunión Española sobre Criptología y Seguridad de la Información*, pages 123-128, 2016b. ISBN: 978-84-608-9470-4. `http://recsi16.uib.es/wp-content/uploads/2016/10/ACTAS-RECSI-2016.pdf`.

Abstract

Cover source mismatch (CSM) is an important open problem in steganalysis. This problem, known as domain adaptation in the field of machine learning, deals with the decrease in the classification accuracy when a classifier is moved from the laboratory into the real world. In this paper, we present an approach to CSM based on domain adaptation using manifold alignment algorithms. In this novel approach, we use manifold alignment to find a latent space where the two datasets (the one used for training and the one used for testing) have a common representation. We show that manifold alignment can significantly increase the accuracy of the classifier in crossdomain classification.

XIV Reunión Española sobre Criptología y Seguridad de la Información

Manifold alignment approach to cover source mismatch in steganalysis

Daniel Lerch-Hostalot
Universitat Oberta de Catalunya,
Internet Interdisciplinary Institute (IN3),
Estudis d'Informàtica, Multimèdia i Telecomunicació,
Av. Carl Friedrich Gauss, 5,
08860 Castelldefels (Barcelona)
Email: dlerch@uoc.edu

David Megías
Universitat Oberta de Catalunya,
Internet Interdisciplinary Institute (IN3),
Estudis d'Informàtica, Multimèdia i Telecomunicació,
Av. Carl Friedrich Gauss, 5,
08860 Castelldefels (Barcelona)
Email: dmegias@uoc.edu

***Abstract*—Cover source mismatch (CSM) is an important open problem in steganalysis. This problem, known as domain adaptation in the field of machine learning, deals with the decrease in the classification accuracy when a classifier is moved from the laboratory into the real world. In this paper, we present an approach to CSM based on domain adaptation using manifold alignment algorithms. In this novel approach, we use manifold alignment to find a latent space where the two datasets (the one used for training and the one used for testing) have a common representation. We show that manifold alignment can significantly increase the accuracy of the classifier in cross-domain classification.**

***Index Terms*—Steganalysis, Cover source mismatch, Domain adaptation, Manifold alignment, Machine learning**

I. Introduction

Steganography is a collection of techniques for hiding data into apparently innocent objects with the aim of establing secret communications. Nowadays, these objects are usually digital media. One of the most common carriers are images because of their widespread use in the Internet.

On the other hand, steganalysis refers to different techniques aiming at detecting whether a cover source has been modified to hide data. The most successful techniques for carrying out steganalysis are based on *machine learning* [20], [7], [14]. The usual methodology implies preparing a training set formed by true covers (referred to as "the *cover* set") and covers that are embedded with hidden information (referred to as "the *stego* set"). This training set is used for training a classifier that is applied later on to test sets of unknown media and determine if they are cover or stego.

Steganalysis based on machine learning is usually successful in laboratory conditions, i.e. if it is assumed that we have access to a set of media of the same type as the set of media used by the steganographer. However, in the real world, the set of media used by the steganographer is not of the same type used for training the classifier [13]. In the case of images, for example, the test set may be taken with a different camera and with different properties of the cover, such as size, lighting conditions, and so on. Therefore, the domain of the training set is usually different from the domain of the testing set, and the classifier will not usually perform appropriately in the latter. In machine learning literature, this problem is known as the problem *domain adaptation* [17], [2], whereas, in steganalysis, this situation is referred to as *cover source mismatch* (CSM), being an important open problem in the field [13].

This paper stems from a family of algorithms known as *manifold alignment* [26] used in a subfield of machine learning called *manifold learning* [24], [22], [16]. This concept was first introduced in [9] adding a manifold constraint to the problem of correlating sets of high-dimensional vectors [10]. This initial idea has evolved in subsequent publications [29], [26], [27], [28] and it is becoming a powerful tool in the field of machine learning. Manifold alignment can be used as a framework for discovering a unifying representation of multiple datasets. In this paper, we show how to apply this framework to minimize the impact of CSM in steganalysis.

The rest of this paper is organized as follows. Section II presents the manifold alignment strategy and the proposed steganalysis algorithm applying this technique. Section III presents the experimental results obtained using the proposed method for seven different image databases in the cross-domain case. Finally, Section IV summarizes the conclusions of this work and suggests some directions for further research.

II. Manifold Alignment

This section presents a general overview of manifold alignment and specific algorithms to apply this technique in steganalysis.

A. Background

Manifold alignment [26] can be used as a framework for discovering a unifying representation of multiple datasets. In this paper, we show how it can be used in steganalysis with the aim of obtaining a common representation of the training and the testing datasets, being a solution to mitigate the problem of cover source mismatch.

The basic idea of manifold alignment is to use the similarities within each dataset and correspondences between datasets for mapping initially disparate datasets to a joint latent space, usually a low dimensional one. The new representation is extracted by modeling the local geometry of each dataset

and by constraining the dimensionality reduction with the correspondence among datasets.

The problem we want to solve can be formalized as follows: given two data sets $X = \{x_1, x_2, ..., x_n\}$ and $Y = \{y_1, y_2, ..., y_m\}$ in $\mathbb{R}^D$, along with correspondence information $x_i \in X \leftrightarrow y_j \in Y$, we need to map X and Y into a new space preserving the local geometry of each dataset and matching the instances (or samples) of both sets in correspondence.

The local geometry within each dataset is modeled with an adjacency matrix [8], usually constructed using a nearest neighbor graph. This means that we construct a graph using every sample of the dataset as a vertex and adding an edge between two vertices of the graph if they are in the set of the k-nearest neighbors. Therefore, we build the adjacency matrix of X as follows:

$$A_x(i,j) = \begin{cases} 1, & \text{if } x_i \text{ and } x_j \text{ are neighbors,} \\ 0, & \text{otherwise.} \end{cases}$$

When the two vertices are in the same dataset, we can improve the adjacency matrix using the geodesic distance [16]:

$$A_x(i,j) = \begin{cases} e^{-||x_i - x_j||}, & \text{if } x_i \text{ and } x_j \text{ are neighbors,} \\ 0, & \text{otherwise.} \end{cases}$$

The adjacency matrix for Y can be built in the same way:

$$A_y(i,j) = \begin{cases} e^{-||y_i - y_j||}, & \text{if } y_i \text{ and } y_j \text{ are neighbors,} \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

The correspondence information between datasets is modeled through a matrix of correspondences. The matrix of correspondences is an adjacency matrix, but, since the two datasets lie in different manifolds, the geodesic distance between samples cannot be used. This matrix is constructed as follows:

$$C_{xy}(i,j) = \begin{cases} 1, & \text{if } x_i \text{ corresponds to } y_j, \\ 0, & \text{otherwise.} \end{cases}$$

The alignment algorithm needs both the correspondence and the local geometry as base information for finding the new representation. With the local geometry, we ensure that the local properties of the datasets are preserved and, with the correspondence information, we can obtain a common representation of both datasets in a different space.

Using the adjacency and the correspondence matrices, we can create the *joint adjacency matrix* [26] as follows:

$$W = \begin{pmatrix} A_x & C_{xy} \\ C_{xy}^{\mathrm{T}} & A_y \end{pmatrix}, \quad (2)$$

where $[\cdot]^{\mathrm{T}}$ denotes the transposition operator. With this joint adjacency matrix, we can compute the Laplacian matrix [8], defined as $L = D - W$, where D is the degree matrix, given by

$$D(i,i) = \sum_{j=1}^{n} W(i,j). \quad (3)$$

The projection into the new space of d dimensions is carried out optimizing a cost function. The new coordinates are given by the d smallest nonzero eigenvectors of $Lf = \lambda Df$ [26]. This optimization can be regarded as a way of obtaining a new graph in which the connected nodes with high weights come closer than the nodes with lower weights (see Wang *et al.* [26] for more details).

The above approach summarizes the basic idea of manifold alignment. However, in our application, we propose a variation in the construction of the adjacency matrix for X. Since X is part of the training set, we know if a sample of X is cover or stego. Therefore, we can apply a multiplier to the geodesic distance in each case, with the intention of increasing the weight when the pair is in the same class and to decrease it otherwise. The purpose of this operation is that, in the new representation, the samples belonging to the same class remain close to each other, whereas the samples of different classes draw away from each other. The new adjacency matrix is defined as follows:

$$A_x(i,j) = \begin{cases} m_1 e^{-||x_i - x_j||}, & \text{if } x_i \text{ and } x_j \text{ are neighbors and they are in the same class,} \\ m_2 e^{-||x_i - x_j||}, & \text{if } x_i \text{ and } x_j \text{ are neighbors and they are in different classes,} \\ 0, & \text{otherwise,} \end{cases} \quad (4)$$

where m_1 and m_2 are the chosen multipliers.

B. Correspondence Information

In the process of alignment, the algorithm needs the correspondence information (the matrix C_{xy}) to find the new representation. However, the process of domain adaptation in steganalysis is unsupervised and, thus, we do not have information of these correspondences (since we do not know which images are stego or cover in the testing dataset). Since this correspondence information is required, we need to estimate it from the original data. We propose an approach that, to the best of our knowledge, is novel in the manifold alignment literature. The suggested procedure is as follows:

1) Use the first domain as a training dataset.
2) Use the second domain as a testing dataset.
3) Assuming that both datasets have some similarities, pre-classify the testing dataset using some machine learning algorithm.
4) Use the pre-classification results obtained in the previous step to estimate the correspondence information.

More precisely, the correspondence information is constructed as follows. The correspondence matrix shall provide a relationship between one sample in the training domain and one sample in the testing domain. We establish this correspondence based on the probability of classification of each sample obtained from the machine learning classifier. To do this, we create a correspondence between the sample with

the highest probability of being stego in the training domain and the sample with the highest probability of being stego in the testing domain. Then, we create another correspondence between the second sample with the highest probabilities of each set, and so on. Finally, we have a correspondence between each sample of each domain, and this information is used in the alignment process for finding the joint space.

Consequently, the correspondence matrix can be computed as

$$C_{xy}(i,j) = \begin{cases} 1, & \text{if } x_i \text{ and } y_j \text{ are in the same} \\ & \text{position of the pre-classified} \\ & \text{sets (ordered by probability),} \\ 0, & \text{otherwise.} \end{cases} \quad (5)$$

In fact, the method described above for constructing the correspondence matrix applies only when the number of samples of each set and the number of samples belonging to each class (cover or stego) in each set are identical. Nevertheless, this is not a limitation of our method, since it is still possible to a construct the correspondence matrix linking only a subset of X to a subset of Y. For example, we may take only those samples for which we are more confident about their classification. However, we have not developed a final strategy for the selection of these subsets yet. Hence, in the sequel, we consider the homogeneous situation in which both X and Y have the same number of stego and cover images. The specific solution for the non-homogeneous case is left for the future research.

C. Alignment Algorithm

Using the building blocks described in the previous sections, we can define the algorithm for domain adaptation applying manifold alignment.

Given a dataset $X = \{x_1, x_2, ..., x_n\}$ of samples of the training set (vectors of image features), the associated set of labels $X_l = \{l_1, l_2, ..., l_n\}$, where each label indicates whether the corresponding image is cover or stego, and the target dataset $Y = \{y_1, y_2, ..., y_m\}$ of samples of the testing set (vectors of features), we proceed as follows:

1) **k-Nearest Neighbors Graphs.** Each sample of X becomes a vertex of a graph. A vertex (sample of the training dataset) x_i is connected with another vertex (sample) x_j by an edge if x_i is among the k-nearest neighbors of x_j. The same process is used to create the graph for the testing dataset.
2) **Adjacency Matrices.** Build the adjacency matrices A_x and A_y for the k-nearest neighbors graphs of X and Y using the geodesic distance as shown in Expressions (4) and (1).
3) **Pre-classification.** Classify the second domain (testing dataset) using the first domain (training dataset) as a training set. Use an algorithm that provides the probability of classification for each sample.
4) **Correspondence Information.** Establish the correspondence relationship, using the pre-classified sets ordered by probability, as per Expression (5).
5) **Joint Adjacency Matrix.** Use the adjacency matrices and the correspondence information for building the joint adjacency matrix as per Expression (2).
6) **Eigenmaps.** Compute the eigenvectors and eigenvalues for the generalized eigenvector problem: $Lf = \lambda Df$, where D is the diagonal weight matrix (degree matrix). The coefficients of this matrix are column (or row, since W is symmetric) sums of W, as shown in Expression (3), and L is the Laplacian matrix: $L = D - W$.
The projection into the new space is given by the d minimum nonzero eigenvectors $f_1, \ldots, f_d$ of the eigenvalue decomposition. Each of these vectors, f_j, has $n + m$ components, where n is the number of samples of the training dataset X and m is the number of samples of the testing dataset Y. The projection of X and Y, denoted as $\widetilde{X}$ and $\widetilde{Y}$, to the joint space is formed by taking the components of these vectors as follows:

$$\tilde{x}_j = \begin{bmatrix} (f_1)_j & (f_2)_j & \cdots & (f_d)_j \end{bmatrix}^{\mathrm{T}},$$

for $j = 1, \ldots, n$, and

$$\tilde{y}_j = \begin{bmatrix} (f_1)_{n+j} & (f_2)_{n+j} & \cdots & (f_d)_{n+j} \end{bmatrix}^{\mathrm{T}},$$

for $j = 1, \ldots, m$, where $(f_j)_k$ denotes the k-th component of the vector f_j.
Note that projected samples, $\tilde{x}_j$ and $\tilde{y}_j$, have dimension d, which must be lower than or equal to the dimension of the samples of the original datasets (D).

D. Steganalysis

Finally, in this section, we consider the problem of steganalyzing a target set of images from the real world [13]. In this case, we do not know which images are cover and which images are stego in the target set and we do not have information about the conditions in which these images were taken. In such scenario, we assume that the source and the destination domains are different and they need to be adapted. As already remarked, the only assumption we make is about an equal number of stego and cover images in each dataset. This is not required by our method, but we limit the experiments to this case only due to the fact that we do not have a final strategy for the non-homogeneous problem.

In these conditions, we propose the following procedure:

1) **Training Set Feature Extraction.** Choose a training database, embed secret information in some of the images (stego) and leave the other as cover. Perform feature extraction and obtain a set X of samples (vectors of features) and the corresponding set X_l of labels (stego or cover).
2) **Target Set Feature Extraction.** Extract the features from the target database (containing some cover and some stego images), obtaining a set Y of samples (vectors of features).
3) **Domain Adaptation.** Adapt the training and target domains using the algorithm detailed in Section II-C, yielding the adapted sets $\widetilde{X}$ and $\widetilde{Y}$.

4) **Training.** Train the classifier using the adapted set of "projected features" $\widetilde{X}$ and the set of labels X_l.
5) **Classification.** Using the classifier trained in the previous step, classify the adapted testing dataset, using their "projected features" $\widetilde{Y}$, into cover and stego images.

III. Experimental Results

This section presents the experiments conducted to show the effectiveness of the proposed approach for the cover source mismatch problem.

A. Image Databases

In order to test the proposed approach, we selected seven distinct image databases:

- The NRCS database consists of images from the National Resource Conservation System [19] with a fixed size of 2100×1500 pixels.
- The BOSS database consists of images from Break Our Steganographic System! [6] with a fixed size of 512×512 pixels.
- The ESO database consists of images from the European Southern Observatory [5] with variable sizes about 1200×1200 pixels.
- The Interactions (INTE) database consists of images from Interactions.org [12] with variable sizes about 600×400 pixels.
- The NOAA database consists of images from the National Oceanic and Atmospheric Administration (NOAA) [18] with variable sizes about 2000×1500 pixels.
- The Albion (ALBN) database consists of images from the Plant Image Database of the Albion College [1] with a fixed size of 1024×685 pixels.
- The Calphotos (CALP) database consists of images from the Regents of the University of California [25] with variable sizes about 700×500 pixels.

For each database, we generated four sets of 500 images each. Theses sets consist of 250 cover images and 250 stego ones. In two of these sets, the stego images were generated with an embedding rate of 0.5 bits per pixel (bpp), whereas an embedding rate of 0.25 bpp was used for the other two sets. The steganographic system used for embedding is LSB matching [23]. A set of the group with embedding rate of 0.5 bpp was used for training and the other one for testing. The same goes for the two sets with embedding rate of 0.25 bpp.

B. Classification

The experiments show how the classification accuracy is improved after domain adaptation. For comparison purposes, we need the results before and after domain adaptation.

To perform the experiments in a neutral manner, we chose a unique feature extractor and a unique classifier for all cases. As a feature extractor, we used Patterns of Pixel Differences (PPD) [14] for its low CPU time and lower number of features compared to other methods. For obtaining the PPD features, we set the threshold for the pixel differences to $S = 4$, which yields 256 features per image. As a classifier, we chose the implementation LibSVM [3] of Support Vector Machines (SVM) for its efficiency and accuracy.

We used an SVM with a Gaussian Kernel. This classifier must be adjusted to provide optimal results. In particular, the values of the parameters C and γ must be selected. These values should be chosen to give the classifier the ability to generalize. The process has been performed as described in [11]. For all the experiments of the paper, we have used cross-validation on the training set using the following multiplicative grid for C and γ:

$$\begin{aligned} C &\in \left\{2^{-5}, 2^{-3}, 2^{-1}, 2^{1}, 2^{3}, \ldots, 2^{15}\right\}, \\ \gamma &\in \left\{2^{-15}, 2^{-13}, 2^{-11}, \ldots, 2^{-1}, 2^{1}, 2^{3}\right\}. \end{aligned} \tag{6}$$

In the standard classification process (with no domain adaptation) we selected the training set and trained the SVM. After that, we tested the classifier with the testing set. This process is typically used in machine learning. On the other hand, in the classification with domain adaptation, we first took the training and the testing datasets (samples of features) and projected them into the joint space using the algorithm proposed in Section II-C. After that, we followed the rest of the steganalysis process as detailed in Section II-D.

C. Tuning Parameters

As discussed in the previous sections, we need to tune some parameters to apply domain adaptation. Since we are working in an unsupervised environment, we have no information about the testing domain other than some grade of similarity with the training domain. Therefore, it is not convenient to tune the parameters using a specific domain, because they might not work with other domains. Hence, we decided to make some reasonable assumptions to select their values.

The first parameter to tune is k, required in the k-nearest neighbors algorithm used for obtaining the graph in the Alignment Algorithm (Section II-C). A common option is to choose the square root of the total number of samples: $k = \sqrt{n+m}$ as suggested in [4].

In addition, we need to tune the weights m_1 and m_2. The weight m_1 affects the pairs which lay on the same class, and we want it as large as possible (in order to decrease its distance in the projected space). On the ohter hand, m_2 affects the pairs in different classes, and we want it as small as possible (so as to increase its distance in the projected space). We tested different values experimentally, and the results always improved when m_1 increased and m_2 decreased, until we reached a value at which no improvement occured and it remained unchanged. After this process, we finally selected $m_1 = 1000$ and $m_2 = 0.001$.

For the pre-classification step of the Alignment Algorithm, we also used an SVM with a multiplicative grid and cross-validation, as described in [11], and the specific values used for the experiments are given in Expression (6). In addition, in this step, we need the membership probabilities for each sample. For this reason, we used an SVM with this capability, more precisely, LibSVM [3] with Platt scaling [21], [15].

Manifold alignment approach to cover source mismatch in steganalysis

Another relevant parameter is the number of dimensions of the latent (joint) space (d). Unfortunately, there is no standard strategy for choosing this value. We made tests with different values of d between 1 and 250 (the original dimension was 256). The classification accuracy (the percentage of correctly classified images) was almost identical for all $d < 100$ and it decreased for larger values of d. According to this result, we selected $d = 2$ in all the other experiments, since $d = 2$ yielded somewhat better results than $d = 1$. It is worth pointing out that just two dimensions were enough to achieve remarkable classification results in the latent space. This illustrates that the data in the latent space are very rich to discriminate between cover and stego images.

D. Results

We can see the results for all the experiments in Table I. Each row in the table corresponds to a training database (first domain) and a testing database (second domain). Firstly, we show the classification accuracy without domain adaptation (column "Acc"), where we can see the effects of CSM. In the next column ("Acc DA"), the classification accuracy results **using** domain adaptation through manifold alignment are shown. Here, we can notice how the accuracy changes with manifold alignment. The results are provided for the two selected embedding rates: 0.5 bpp and 0.25 bpp. The table is divided into several blocks, each of which corresponding to the same training database. The first row of each block shows the results obtained using training and testing databases in the same domain (the same image database). This is not a case of domain adaptation, because the domain would not need to be adapted. However, it is an interesting case, since it allows to analyze how manifold alignment affects the accuracy results when adaptation is not required. In addition, since we are assuming an unsupervised environment, we do not know whether the testing domain needs to be adapted or not. The classification accuracy improves in almost all cases with two different domains (improvement is highlighted in boldface). However, the adaptation of two domains that not need to be adapted usually results in a decreased accuracy. In this case, the decrease in accuracy is typically small, with the only exception of the Albion and Calphotos databases.

As shown in the last row of the Table I, on average, the classification accuracy results with domain adaption improve from 75.00% to 82.00% in the case of 0.5 bpp embedding, and from 67.00% to 72.00% for 0.25 bpp embedding. The increase in classification accuracy is even more remarkable if we consider only those cases that do not need adaptation, that is, when the training and the testing domain differ. In that case, the classification accuracy increases from 72.00% to 81.00% for 0.5 bpp and from 63.00% to 70.00% for 0.25 bpp. Hence, a method to detect when the domains really need to be adapted will improve the effectiveness of the proposed approach.

TABLE I: Classification accuracy ("Acc") with and without domain adaptation ("DA") for LSB matching with embedding bit rates of 0.5 bpp and 0.25 bpp

Embedding bit rate		**0.5 bpp**		**0.25 bpp**	
Train	**Test**	**Acc (%)**	**Acc DA (%)**	**Acc (%)**	**Acc DA (%)**
CALP	CALP	89.6	**91.4**	**89.6**	86.0
CALP	ALBN	53.0	**66.2**	**50.2**	49.8
CALP	BOSS	74.4	**82.8**	53.6	**75.0**
CALP	ESO	95.4	**97.8**	52.6	**87.4**
CALP	INTE	91.4	**98.8**	51.2	**93.8**
CALP	NOAA	90.4	**99.8**	50.0	**99.4**
CALP	NRCS	51.0	**62.6**	50.0	**52.8**
BOSS	BOSS	**88.2**	85.6	**84.4**	76.6
BOSS	ALBN	57.2	**63.4**	**57.0**	53.2
BOSS	CALP	84.6	**90.6**	55.6	**75.2**
BOSS	ESO	64.4	**94.2**	81.4	**87.4**
BOSS	INTE	60.6	**98.0**	83.4	**87.4**
BOSS	NOAA	53.0	**97.6**	71.8	**97.8**
BOSS	NRCS	62.8	**68.6**	**56.0**	54.6
NRCS	NRCS	**89.2**	74.8	**74.4**	60.4
NRCS	ALBN	**70.8**	67.0	**61.4**	58.4
NRCS	BOSS	73.2	**79.2**	**72.2**	70.6
NRCS	CALP	67.4	**87.2**	61.2	**68.0**
NRCS	ESO	78.6	**94.2**	62.8	**92.4**
NRCS	INTE	82.0	**98.4**	62.2	**91.0**
NRCS	NOAA	66.8	**99.2**	55.0	**98.4**
ALBN	ALBN	**76.0**	57.8	**58.4**	53.6
ALBN	BOSS	76.4	**84.4**	**52.2**	42.0
ALBN	CALP	51.2	**82.0**	**50.0**	48.6
ALBN	ESO	76.0	**81.2**	**50.0**	47.4
ALBN	INTE	73.0	**88.4**	50.0	**51.6**
ALBN	NOAA	58.0	**98.6**	**50.0**	49.8
ALBN	NRCS	**72.0**	66.4	54.8	54.8
ESO	ESO	**98.0**	97.2	97.2	**97.4**
ESO	ALBN	50.0	**54.8**	50.0	**51.2**
ESO	BOSS	72.8	**74.4**	55.4	**64.8**
ESO	CALP	72.4	**74.4**	**64.4**	64.2
ESO	INTE	**100.0**	99.6	**99.8**	98.2
ESO	NOAA	**100.0**	99.8	100.0	100.0
ESO	NRCS	50.2	**66.0**	50.0	**54.2**
INTE	INTE	**99.8**	99.6	**99.2**	97.6
INTE	ALBN	**56.4**	52.8	50.6	**58.4**
INTE	BOSS	74.4	**77.0**	**69.4**	65.6
INTE	CALP	72.6	**77.6**	**62.2**	59.0
INTE	ESO	**97.0**	96.8	**95.8**	95.4
INTE	NOAA	100.0	100.0	100.0	100.0
INTE	NRCS	56.6	**59.4**	51.0	**55.4**
NOAA	NOAA	**100.0**	99.8	100.0	100.0
NOAA	NRCS	51.0	**54.0**	50.0	**51.0**
NOAA	ALBN	**51.0**	45.6	**50.0**	46.4
NOAA	BOSS	64.8	**66.0**	**65.8**	57.8
NOAA	CALP	**62.6**	61.2	54.2	**54.8**
NOAA	ESO	95.6	**96.0**	**95.2**	95.0
NOAA	INTE	99.4	**100.0**	**98.2**	97.4
AVERAGE		75.0	**82.0**	67.0	**72.0**

IV. Conclusion

In some applications of classification, like steganalysis, a problem appears when the classifier is moved into the real world. In steganalysis, this problem is known as cover source mismatch and is caused by the existence of significant differences between the training and the testing datasets. In this paper, we present a novel approach for dealing with CSM based on manifold alignment techniques. As shown in the paper, the manifold alignment algorithm seems to be a convenient approach for dealing with the CSM problem, since the classification accuracy results improve when using this technique if the training and testing datasets differ. The results have been obtained for seven different databases showing a relevant improvement compared to the non-adapted case.

For future research, some issues need to be addressed. To begin with, we need to develop a strategy to choose the optimal value of d (although the preliminary results indicate that a very low dimensional space is a convenient choice). Similarly, the effect of the parameter k used in the k-nearest neighbors algorithm should be analysed. We have used the value of k suggested in machine learning literature, but a deeper analysis is required. Finally, we have noticed that adaption is not advisable when the training and testing domains do not need to be adapted. This drawback may be overcome by exploring the degree of similarity between domains prior to deciding whether to use adaptation.

Acknowledgments

This work was partly funded by the Spanish Government through grants TIN2011-27076-C03-02 "CO-PRIVACY" and TIN2014-57364-C2-2-R "SMARTGLACIS".

References

[1] "Plant image database," Available: http://www4.albion.edu/plants/, Albion College, accessed on April 2, 2016 [Online].

[2] C. M. Bishop, *Pattern Recognition and Machine Learning*, ser. Information Science and Statistics Series. New York, Secaucus, NJ, USA: Springer, 2006.

[3] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, available: http://www.csie.ntu.edu.tw/~cjlin/libsvm, accessed on April 2, 2016.

[4] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Wiley-Interscience, 2000.

[5] "ESO images," Available: http://www.eso.org/public/images/, European Southern Observatory, accessed on April 2, 2016 [Online].

[6] T. Filler, T. Pevný, and P. Bas, "Break our steganographic system (BOSS)," http://agents.fel.cvut.cz/stegodata/, 2010, accessed on April 2, 2016 [Online].

[7] J. Fridrich and J. Kodovský, "Rich models for steganalysis of digital images," *IEEE Trans. Information Forensics and Security, June 2012*, vol. 7, no. 3, pp. 868–882, 2012.

[8] C. Godsil and G. Royle, *Algebraic Graph Theory*. Springer, Apr. 2001.

[9] J. H. Ham, D. D. Lee, and L. K. Saul, "Learning high dimensional correspondences from low dimensional manifolds," in *Workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, 2003, pp. 34–41.

[10] H. Hotelling, "Relations between two sets of variates," *Biometrika*, vol. 28, no. 3/4, pp. 321–377, Dec. 1936.

[11] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, "A practical guide to support vector classification," Department of Computer Science, National Taiwan University, Tech. Rep., 2003, accessed on April 2, 2016 [Online]. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/papers.html

[12] "ImageBank," Available: http://www.interactions.org/cms/?pid=1900, Interactions.org Particle Physics News and Resources, accessed on April 2, 2016 [Online].

[13] A. D. Ker, P. Bas, R. Boehme, R. Cogranne, S. Craver, T. Filler, J. Fridrich, and T. Pevný, "Moving steganography and steganalysis from the laboratory into the real world," in *Proc. 1st IH&MMSec. Workshop*. Montpellier, France: ACM, Jun. 2013, pp. 45–58.

[14] D. Lerch-Hostalot and D. Megías., "LSB matching steganalysis based on patterns of pixel differences and random embedding," *Computers & Security*, vol. 32, pp. 192–206, 2013.

[15] H.-T. Lin, C.-J. Lin, and R. C. Weng, "A note on platt's probabilistic outputs for support vector machines," *Mach. Learn.*, vol. 68, no. 3, pp. 267–276, Oct. 2007.

[16] Y. Ma and Y. Fu, *Manifold Learning Theory and Applications*, 1st ed. Boca Raton, FL, USA: CRC Press, Inc., 2011.

[17] T. M. Mitchell, *Machine Learning*, 1st ed. New York, NY, USA: McGraw-Hill, Inc., 1997.

[18] "NOAA photo library," Available: http://www.photolib.noaa.gov/, National Oceanic and Atmospheric Administration, accessed on April 2, 2016 [Online].

[19] "NRCS photo gallery," Available: http://photogallery.nrcs.usda.gov, National Resource Conservation System, accessed on April 2, 2016 [Online].

[20] T. Pevný, P. Bas, and J. J. Fridrich, "Steganalysis by subtractive pixel adjacency matrix." *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 2, pp. 215–224, 2010.

[21] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances In Large Margin Classifiers*. MIT Press, 1999, pp. 61–74.

[22] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, Dec. 2000.

[23] T. Sharp, "An implementation of key-based digital signal steganography," in *In Information Hiding*, ser. Lecture Notes in Computer Science. Berlin-Heidelberg, Germany: Springer, 2001, vol. 2137, pp. 13–26.

[24] J. B. Tenenbaum, V. Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[25] "CalPhotos," Available: http://calphotos.berkeley.edu/, University of California, Berkeley, accessed on April 2, 2016 [Online].

[26] C. Wang and S. Mahadevan, "A general framework for manifold alignment." in *AAAI Fall Symposium: Manifold Learning and Its Applications*, ser. AAAI Technical Report, vol. FS-09-04. AAAI, 2009.

[27] ——, "Manifold alignment without correspondence," in *Proceedings of the 21st International Jont Conference on Artifical Intelligence*, ser. IJCAI'09. Morgan Kaufmann Publishers Inc., 2009, pp. 1273–1278.

[28] ——, "Manifold alignment preserving global geometry," in *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, ser. IJCAI'13. AAAI Press, 2013, pp. 1743–1749.

[29] L. Xiong, F. Wang, and C. Zhang, "Semi-definite manifold alignment," in *ECML*, ser. Lecture Notes in Computer Science, J. N. Kok, J. Koronacki, R. L. de Mántaras, S. Matwin, D. Mladenic, and A. Skowron, Eds., vol. 4701. Springer, 2007, pp. 773–781.

Chapter 4

Conclusions and future research

4.1 Conclusions

In this thesis, different techniques to detect steganographic methods are proposed, and a novel steganographic algorithm is suggested. In the case of steganalysis, the proposed techniques include methods for the detection of histogram shifting techniques, methods for the detection of state of the art steganographic algorithms, and some results focused to mitigate or bypass the CSM problem. In the case of steganography, a method is described that takes advantage of the difficulty that some feature extractors have in dealing with high dimensions.

The conclusions for these different works are summarized in the following sections.

4.1.1 Histogram shifting steganalysis

Four steganalytic techniques to detect information hidden using histogram shifting methods are presented.

The first method detects the orginal scheme of Ni et al. (2006) by finding anomalies in the histogram of pixel intensities. The anomalies produced by the embedding can be detected applying the following observations: given four consecutive bins of the histogram (h_i, h_{i+1}, h_{i+2} and h_{i+3}) the peak used to hide information is usually located in a bin that satisfies that $h_{i+1} + h_{i+2}$ is greater than any bin of the histogram, that h_{i+1} and h_{i+2} have similar heights and that h_i or h_{i+3} are not much smaller than $h_{i+1} + h_{i+2}$. Some thresholds are proposed to deal with the peak detection. Once the peak is detected, the hidden message can be extracted.

The second steganalytic technique presented detects Mohsenzadeh et al.'s 2009 scheme by detecting an unusual statistical distribution introduced by the embedding algorithm. The embedding algorithm produces a significant statistical anomaly because there is al-

ways a $P+1$ (or $P-1$) value next to a pixel of value $P+2$ (or $P-2$). We have proposed an algorithm based on counting those occurrences next to all the pixels. The basic idea is to consider each pixel as a potential $P+2$ (or $P-2$) candidate, and check if this pixel has a $P+1$ (or $P-1$) neighbor. In this case, we consider the pixel a P candidate counting its occurrences. The P candidate with more occurrences is considered as the peak where the data was hidden.

The third steganalytic technique detects histogram shifting of prediction errors (HSPE) methods (Hong et al., 2008), by analyzing the *volatility* of the histogram. We have used the following expression to detect this volatility:

$$V = \sum_{i=1}^{254} \frac{\max(\widetilde{h}_i, h_i) - \min(\widetilde{h}_i, h_i)}{\max(\widetilde{h}_i, h_i)},$$

where $\widetilde{h}_i = (h_{i-1} + h_i + h_{i+1})/3$.

Finally, the fourth technique is a generalization of the concept of volatility to detect all the steganographic methods mentioned above.

4.1.2 Feature extraction

A new feature extractor –patterns of pixel differences (PPD)– for the detection of LSB matching steganography is presented. The PPD extractor is based on the analysis of the differences between neighboring pixels before and after random data embedding. The PPD method is compared with the state-of-the-art SPAM feature extractor, and it can be seen that PPD outperforms SPAM in both accuracy and computation time. The PPD feature extractor is also tested with the HUGO steganographic algorithm.

The PPD feature extractor uses blocks of 5 pixels to cover the horizontal, vertical and the two diagonal directions from the central pixel. Similarly as for SPAM, the proposed method uses a threshold to reduce the high dimensionality produced by an hypothetical vector with 256^5 values (features) to a vector with T^5 values (features), where T is a threshold chosen to be 4. This reduction would lead to $4^5 = 1024$ features, but we introduced the use of a pixel as a reference that allows decreasing the dimensionality to only $4^4 = 256$ features, while capturing the same information. The procedure to prepare the vector consists in subtracting the pixel with the lower value from the other pixels of the block. This leads to a values from zero to 256 in gray scale images. After applying the threshold, we obtain a vector with five elements between zero and three. Since we always have a zero value, we can remove it, obtaining a vector with four components and reducing the number of features to 256. Another novelty in the presented feature extractor is the analysis of the patterns before and after embedding random data. This allows the feature extractor to measure how the image responds to random data insertion,

a behavior that seems to be different for cover and stego images. Although it was not the target for our feature extractor, we show that PPD also works with remarkable results with JPEG steganography.

4.1.3 Steganography beyond the detection threshold

A steganographic method that tries to take advantage of some lessons learned in our research about feature extractors is presented. The main idea of the method is to exploit the threshold T used by some feature extractors, like SPAM, and hide information beyond this threshold. The method uses the pairs of neighboring pixels for which their difference is greater than a given threshold T to hide data. In this way, steganalythic methods that use a threshold to avoid high dimensionality can be deceived only by adjusting the threshold of the proposed steganographyc algorithm.

We have tested the proposed method using both the SPAM and the PPD feature extractors. In both cases, the proposed method is not detected when the threshold is properly selected.

4.1.4 Artificial training sets

A novel unsupervised steganalysis method is presented. The method is based on the idea that, by hiding information in an element of the class of cover images, we can transform it into an element of the class of stego images. This fact allows building an unsupervised method, that is, a method that does not need training information.

We show how to build artificial training sets (ATS) performing subsequent additional embeddings into the testing images. Using these artificial training sets, we can address the classification problem in steganalysis by using supervised classification. Since we need to know the algorithm and the embedding bit rate used by the steganographer, the proposed method can be considered targeted.

Since the method does not need training information, it can bypass the CSM problem. Removing the need of a training data set in the machine learning problem is the major contribution of this work. The ATS method has been tested against three steganographic methods: LSB matching, HUGO and WOW. The proposed method not only bypasses the CSM problem, but also achieves a better classification accuracy than the methods in the state of the art.

We also present different techniques to deal with problems that appear in the real world, such as an unbalanced number of stego and cover images, the use of testing sets with mixed databases, the use of testing sets with mixed embedding bit rates, the use of testing sets with an unknown bit rate, and the construction of the testing set in real time.

4.1.5 Manifold alignment

A new approach to the CSM problem based on *manifold alignment* (a well-known technique in machine learning) is presented. We show that the CSM problem can be represented as a problem of a missing alignment between data sets. If the data sets are aligned (domain adaptation), the accuracy results can be improved.

We propose a technique based on a pre-classification step to measure the correlation between some samples of the source domain and some samples in the destination domain. Once this correlation is available, it can be used to project the samples in a new aligned space. We have applied this technique using seven different databases showing a remarkable improvement compared to the non adapted case.

4.1.6 General conclusions

One common technique in our work is the embedding of additional information into the testing images using the same algorithm we want to detect. We have shown that this additional embedding is a convenient tool to be used in supervised steganalaysis (Lerch-Hostalot and Megías, 2013) but also in our novel unsupervised steganalysis method (Lerch-Hostalot and Megías, 2016b). The idea of measuring the effect of (additional) embedding is also applied in (Lerch-Hostalot and Megías, 2012) to compare the volatility of the histogram of differences before and after random embedding, which helps identify cover and stego images according to the different behavior of volatility in both situations.

These results suggest that introducing (additional) embedding(s) in the detection phase is a convenient and powerful tool to incorporate in steganalysis.

4.2 Possible directions for future research

Different possibilities for future research in the different contributions of this thesis are discussed in the following sections.

4.2.1 Histogram shifting steganalysis

For future research, it would be interesting to analyze the use of other histograms to estimate volatility. On the other hand, the applicability of the proposed technique to other data hiding schemes based on histogram shifting may be worth investigating.

4.2.2 Feature extraction

As a future work, the inclusion of the PPD features into the recent rich model techniques is a possible continuation line. In the same way, it would be interesting to build a complete rich model using PPD-based features. In addition, the difference between the test image before and after random data embedding may be used with other feature extraction methods.

4.2.3 Steganography beyond the detection threshold

For future research, it would be interesting to use different embedding strategies exploiting the same idea of the threshold. A theoretical study of the optimal value of the threshold T is another interesting research topic.

4.2.4 Artificial training sets

Although some techniques to deal with real world challenges have been presented, these techniques need to be improved. For example, we need to deal with situations in which the embedding bit rate or the algorithm used is unknown. Currently, the proposed method can not be applied when the splitting function is not known. It would be interesting to investigate how to proceed in these cases, although this would mean entering the field of universal steganalysis.

Furthermore, we propose applying the proposed method to other machine learning scenarios beyond steganalysis. The idea of transforming samples of one class to another class using a splitting function might be applied to other classifications problems that exhibit properties similar to those of steganalysis.

4.2.5 Manifold alignment

Manifold alignment is a novel approach in steganalysis and, therefore, there is a large room for improvement. First, we need to develop new strategies to chose the dimension d of the aligned space. Our preliminary results show that the optimal choice could be in a very low dimensional space, but there is no theoretical support for this. In addition, The choice of the parameter k used in the k-nearest neighbors algorithm also needs some analysis. We have used some values proposed in machine learning literature, but we do not know if other choices could lead to a better alignment in our particular case.

A problem with the proposed algorithm is when adaptation of the data sets is applied when no adaptation is needed, i.e, when no there is no CSM. This problem could be circumvented exploring the degree of similarity between domains before using adaptation.

Finally, this methodology needs to be tested using different feature extractors and classifiers to analyze if the accuracy of the results can be further improved.

4.2.6 Other future research lines

The feature extraction in the state of the art is based on Rich Models (Fridrich and Kodovský, 2012) and produces a set of tailor-made features for steganalysis. Nevertheless, in other areas of machine learning like object recognition, the use of convolutional neural networks (Cun et al., 1990; Krizhevsky et al., 2012) is very effective for learning features automatically. There is no much research about deep learning in steganalysis yet, but some recent publications indicate that this could be the next step in the field (Qian et al., 2015, 2016; Xu et al., 2016). Therefore, future research in this area is mandatory.

Bibliography

N. Ahmed, T. Natarajan, and K. R. Rao. Discrete Cosine Transform. *IEEE Transactions on Computers*, C-23(1):90–93, Jan. 1974.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics Series. Springer, 2006.

G. Cancelli, G. Doërr, M. Barni, and I. J. Cox. A Comparative Study of ±1 Steganalyzers. In *Multimedia Signal Processing, 2008 IEEE 10th Workshop on*, pages 791–796, 2008.

C.-C. Chang and C.-J. Lin. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. (Software available: `http://www.csie.ntu.edu.tw/~cjlin/libsvm`, accessed on February 3, 2017).

C.-C. Chang, W.-L. Tai, and C.-C. Lin. A Reversible Data Hiding Scheme Based on Side Match Vector Quantization. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(10):1301–1308, Oct. 2006.

M. Chaumont and S. Kouider. Steganalysis by Ensemble Classifiers with Boosting by Regression, and Post-Selection of Features. In *Image Processing (ICIP), 2012 19th IEEE International Conference on*, pages 1133–1136, 2012.

I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker. *Digital Watermarking and Steganography*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2nd edition, 2008.

A. Croisier, D. Esteban, and C. Galand. Perfect Channel Splitting by use of Interpolation Decimation, Tree Decomposition Techniques. In *Proceedings of the First International Conference on Information Sciences and Systems*, pages 443–446, 1976.

Y. L. Cun, B. Boser, J. S. Denker, R. E. Howard, W. Habbard, L. D. Jackel, and D. Henderson. Handwritten Digit Recognition with a Back-propagation Network. In *Advances in Neural Information Processing Systems 2*, pages 396–404, 1990.

S. Dumitrescu, X. Wu, and N. D. Memon. On Steganalysis of Random LSB Embedding in Continuous-tone Images. In *Proceedings of the International Conference on Image Processing, ICIP 2002*, pages 324–339, Rochester, NY, USA, 2002. IEEE.

T. Filler, T. Pevný, and P. Bas. Break our Steganographic System (BOSS). `http://exile.felk.cvut.cz/boss/`, 2010. (Accessed on September 21, 2012).

J. Fridrich. Feature-based Steganalysis for JPEG Images and its Implications for Future Design of Steganographic Schemes. In *International Workshop on Information Hiding*, pages 67–81, 2004.

J. Fridrich. *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge University Press, 2010.

J. Fridrich and J. Kodovský. Rich Models for Steganalysis of Digital Images. *IEEE Transactions on Information Forensics and Security*, 7:868–882, 2012.

J. Fridrich, M. Goljan, and R. Du. Detecting LSB Steganography in Color and Grayscale Images. In *Proceddings of the ACM Workshop on Multimedia and Security*, pages 22–28, Ottawa, Canada, 2001.

J. Fridrich, J. Kodovský, V. Holub, and M. Goljan. Breaking HUGO: The Process Discovery. In *Proceedings of the 13th International Conference on Information Hiding*, IH'11, pages 85–101, 2011a.

J. Fridrich, J. Kodovský, V. Holub, and M. Goljan. Steganalysis of content-adaptive steganography in spatial domain. In *Information Hiding*, pages 102–117, 2011b.

G. Gul and F. Kurugollu. A New Methodology in Steganalysis: Breaking Highly Undetectable Steganography (HUGO). In *Proceedings of the 13th International Conference on Information Hiding*, pages 71–84, 2011.

V. Holub and J. Fridrich. Designing Steganographic Distortion Using Directional Filters. In *International Workshop on Information Forensics and Security (WIFS)*, pages 234–239, 2012.

V. Holub, J. Fridrich, and T. Denemark. Universal distortion function for steganography in an arbitrary domain. *EURASIP Journal on Information Security*, 2014(1):1–13, 2014.

W. Hong, T. S. Chen, and C. W. Shiu. Reversible Data Hiding Based on Histogram Shifting of Prediction Errors. In *Intelligent Information Technology Application Workshops, 2008. IITAW '08. International Symposium*, pages 292–295, 2008.

C. W. Honsinger, P. Jones, M. Rabbani, and J. C. Stoffel. Lossless Recovery of an Original Image Containing Embedded Data. United States Patent N.: 6,278,791 B1, 2001.

J. Hwang, J. Kim, and J. Choi. A reversible watermarking based on histogram shifting. In Y. Shi and B. Jeon, editors, *Digital Watermarking*, volume 4283 of *Lecture Notes in Computer Science*, pages 348–361. Springer-Verlag, Berlin-Heidelberg, Germany, 2006.

A. D. Ker and T. Pevný. A Mishmash of Methods for Mitigating the Model Mismatch Mess. In *Proceedings of SPIE - The International Society for Optical Engineering*, volume 9028, pages 90280I–90280I–15, 2014.

A. D. Ker, P. Bas, R. Böhme, R. Cogranne, S. Craver, T. Filler, J. Fridrich, and T. Pevný. Moving Steganography and Steganalysis from the Laboratory into the Real World. In *Proceedings of the First ACM Workshop on Information Hiding and Multimedia Security*, pages 45–58, 2013.

A. Kerckhoffs. La Cryptographie Militaire. *Journal des Sciences Militaires*, IX:5–83 & 161–191, 1883.

J. Kodovský, J. J. Fridrich, and V. Holub. Ensemble Classifiers for Steganalysis of Digital Media. *IEEE Transactions on Information Forensics and Security*, 7(2):432–444, 2012.

J. Kodovský, V. Sedighi, and J. Fridrich. Study of Cover Source Mismatch in Steganalysis and Ways to Mitigate its Impact. In *Proceedings of SPIE - The International Society for Optical Engineering*, volume 9028, 2014.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

W.-C. Kuo and Y.-H. Lin. On the Security of Reversible Data Hiding Based-on Histogram Shift. In *Proceedings of the 2008 3rd International Conference on Innovative Computing Information and Control*, ICICIC '08, pages 174–174, 2008.

S.-K. Lee, Y.-H. Suh, and Y.-S. Ho. Lossless Data Hiding Based on Histogram Modification of Difference Images. In *Proceedings of the 5th Pacific Rim Conference on Advances in Multimedia Information Processing*, pages 340–347, 2004.

D. Lerch-Hostalot and D. Megías. Steganalytic Methods for the Detection of Histogram Shifting Data Hiding Schemes. In *XII Reunión Española de Criptología y Seguridad de la Información*, pages 381–386, 2012.

D. Lerch-Hostalot and D. Megías. LSB Matching Steganalysis based on Patterns of Pixel Differences and Random Embedding. *Computers & Security*, 32:192–206, 2013.

D. Lerch-Hostalot and D. Megías. Esteganografía en Zonas Ruidosas de la Imagen. In *XIII Reunión Española de Criptología y Seguridad de la Información*, pages 173–178, 2014. (In Spanish).

D. Lerch-Hostalot and D. Megías. Unsupervised steganalysis based on artificial training sets. *Engineering Applications of Artificial Intelligence*, 50:45–59, 2016a.

D. Lerch-Hostalot and D. Megías. Manifold Alignment Approach to Cover Source Mismatch in Steganalysis. In *XIV Reunión Española de Criptología y Seguridad de la Información*, 2016b. (To appear).

I. Lubenko and A. D. Ker. Going from Small to Large Data in Steganalysis. In *Media Watermarking, Security, and Forensics 2012*, volume 8303 of *Proceedings of SPIE - The International Society for Optical Engineering*, pages 0M01–0M10, 2012.

S. Lyu and H. Farid. Detecting Hidden Messages Using Higher-Order Statistics and Support Vector Machines. In *Revised Papers from the 5th International Workshop on Information Hiding*, pages 340–354, 2002.

J. Mielikäinen. LSB Matching Revisited. *Signal Processing Letters*, 13:285–287, 2006.

T. M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1997.

Y. Mohsenzadeh, J. Mohajeri, and S. Ghaemmaghami. Histogram Shift Steganography: A Technique to Thwart Histogram Based Steganalysis. In *Proceedings of the 2009 Second International Workshop on Computer Science and Engineering*, pages 166–170, 2009.

Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su. Reversible Data Hiding. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(3):354–362, 2006.

J. Pasquet, S. Bringay, and M. Chaumont. Steganalysis with Cover-Source Mismatch and a Small Learning Database. In *Proceedings of the 22nd European Signal Processing Conference (EUSIPCO)*, pages 2425–2429, 2014.

T. Pevný and J. Fridrich. Merging Markov and DCT Features for Multi-class JPEG Steganalysis. In *Proceedings of SPIE - The International Society for Optical Engineering*, volume 6505, pages 650503–650503–13, 2007.

T. Pevný and A. D. Ker. The Challenges of Rich Features in Universal Steganalysis. In *Proceedings of SPIE - The International Society for Optical Engineering*, volume 8665, pages 86650M–86650M–15, 2013.

T. Pevný, P. Bas, and J. Fridrich. Steganalysis by Subtractive Pixel Adjacency Matrix. *IEEE Transactions on Information Forensics and Security*, 5:215–224, 2010a.

T. Pevný, T. Filler, and P. Bas. Using High-Dimensional Image Models to Perform Highly Undetectable Steganography. In *Information Hiding - 12th International Conference*, pages 161–177, 2010b.

L. Pibre, J. Pasquet, D. Dienco, and M. Chaumont. Deep Learning for Steganalysis is Better than a Rich Model with an Ensemble Classifier and is Natively Robust to the Cover Source-Mismatch. In *Proceedings of Media Watermarking, Security, and Forensics*, 2016.

Y. Qian, J. Dong, W. Wang, and T. Tan. Deep Learning for Steganalysis via Convolutional Neural Networks. In *Proceedings of SPIE - The International Society for Optical Engineering*, volume 9409, pages 94090J–94090J-10, 2015.

Y. Qian, J. Dong, W. Wang, and T. Tan. Learning and Transferring Representations for Image Steganalysis using Convolutional Neural Network. In *IEEE International Conference on Image Processing (ICIP)*, pages 2752–2756, 2016.

T. Sharp. An Implementation of Key-Based Digital Signal Steganography. In *Information Hiding*, volume 2137 of *Lecture Notes in Computer Science*, pages 13–26. Springer, 2001.

G. J. Simmons. The Prisoner's Problem and the Subliminal Channel. In *Advances in Criptology*, CRYPTO 83, pages 51–67, 1983.

K. Sullivan, U. Madhow, S. Chandrasekaran, and B. S. Manjunath. Steganalysis of spread spectrum data hiding exploiting cover memory. In *Security, Steganography, and Watermarking of Multimedia Contents VII, Proceedings of SPIE - The International Society for Optical Engineering*, volume 5681, pages 38–46, 2005.

C. T. H. Thom, H. V. Canh, and T. N. Tien. Steganalysis for Reversible Data Hiding. *International Journal of Database Theory and Application*, 3(2):21–30, 2010.

V. Vapnik. *Estimation of Dependences Based on Empirical Data: Springer Series in Statistics (Springer Series in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1982.

V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

A. Westfeld. F5-A Steganographic Algorithm. In *Proceedings of the 4th International Workshop on Information Hiding*, pages 289–302, 2001.

A. Westfeld and A. Pfitzmann. Attacks on steganographic systems. In *Information Hiding*, volume 1768 of *Lecture Notes in Computer Science*, pages 61–76. Springer-Verlag, Berlin-Heidelberg, Germany, 2000.

G. Xu, H. Z. Wu, and Y. Q. Shi. Structural Design of Convolutional Neural Networks for Steganalysis. *IEEE Signal Processing Letters*, 23(5):708–712, 2016.

D. Zou, Y. Q. Shi, W. Su, and G. Xuan. Steganalysis based on Markov model of Thresholded Prediction-error Image. In *IEEE International Conference on Multimedia and Expo*, pages 1365–1368, 2006.