Steganalysis in Overlapping Images

James M. Whitaker III

Oriel College

University of Oxford

A discertation submitted for the degree of

Master of Science in Computer Science

Trinity 2014

DEDICATED TO:

My mother,
Nancy Whitaker
whose prayers and support were
abundant and yearned after
throughout my life.

AND

My supervisor,
Dr. Andrew D. Ker,
whose insight and excitement were
both illuminating and inspiring.

"Whether, then, you eat or drink or whatever you do, do all to the glory of God." - I Corinthians 10:31

Abstract

Steganalysis aims to detect the presence of hidden information, known as steganography, in digital images. Modern detectors rely on machine learning techniques, using features from digital images to classify them as covers or stegos. Often the ability to detect steganography can be improved through a process called calibration. Calibration attempts to establish a reference for cover features independent of an image's content. This reference can then be used to aid in the detection of steganography, which violates the feature characteristics of a cover through the presence its stego signal. This paper tests the novel idea that using independent images containing overlapping content for calibration can improve the detectability of steganographic images. To this end, a dataset including cover and stego objects of overlapping, uncompressed images was created and tested with state of the art steganalysis methods. In laboratory conditions I show that existing steganalysis can be increased by a factor between 2 and 10 when using cover images with overlapping content for calibration. The discovery of this new method of calibration suggests that state of the art steganalysis can be improved when overlapping images are present, however the extent is subject to further research.

Contents

Chapter 1

Introduction

The need for secure and secret communications has existed for thousands of years. Early forms of steganography can be traced back to ancient Greece [8]. The desire to detect secure communications has existed just as long. Today almost all modern societies have adopted digital communications as the default method of transferring data. The use of steganography to secretly transfer data in digital communications is becoming more widespread, increasingly for undesirable purposes such as malware [18]. Consequently, the need to detect steganography is becoming increasingly important.

1.1 Steganography

The aim of Steganography is to transfer information covertly by hiding it in another medium. This differs from cryptography, which aims to transfer a message securely by encrypting it, rendering it unreadable. Steganography, however, attempts to disguise its very existence, transmitting messages that appear completely benign to an intruder or eavesdropper.

Steganography has typically been explained through the Prisoners' Problem [17].[1] In the Prisoners' Problem there are two inmates, referred to as Alice and Bob. Alice and Bob will soon be separated, but they will still be able to send messages to each other. They want to coordinate an escape strategy; however, all of their messages will be inspected by the Warden. If the Warden suspects foul play it will result in consequences, such as the loss of communications. Thus, the goal of the inmates is for their secret communications to go unnoticed by the Warden.

In the Prisoners' Problem we assume that Alice and Bob have had enough time to coordinate their strategy of sending messages and even select a key before they are separated. The inmates therefore choose to send messages by hiding them in an unsuspicious medium. The messages are commonly referred to as payloads. An object that can carry a payload is referred to as a cover object, or cover. By extension, a steganographic object, or stego, is a cover object that has a payload embedded in it. The goal of steganography is, therefore, to send payloads through stego objects, with

[1]The original intention of the Prisoners' Problem was to describe authentication in the presence of impersonation. However, it has been adapted to provide an appropriate analogy for steganography.

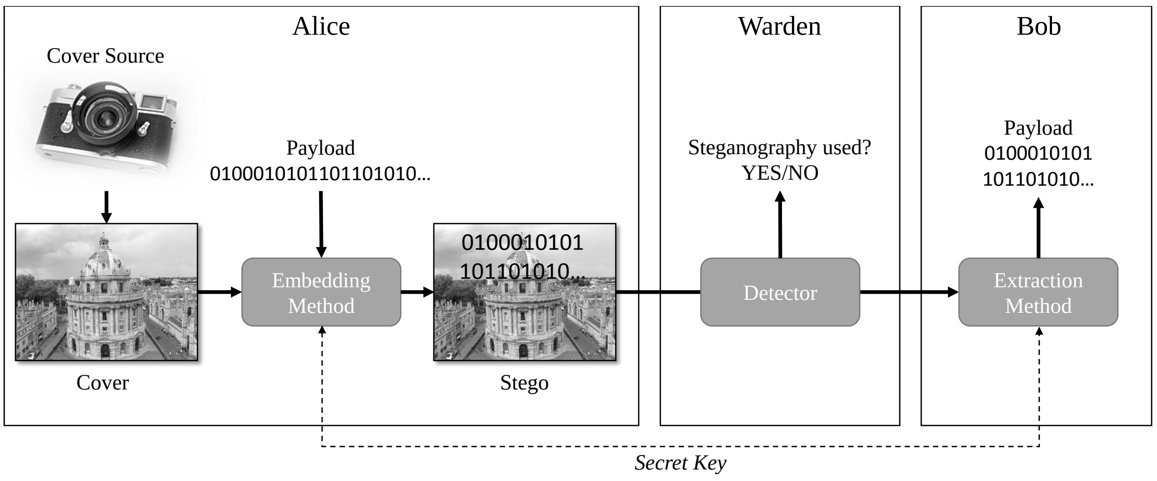

Figure 1.1: Diagram of the general steganographic and steganalytic process as described in the Prisoner's Problem. Alice uses the cover source to create a cover object. She then creates a stego object by using an embedding method to hide a payload in the cover and sends it to Bob. Alice and Bob share a secret key that is used for the embedding and extracting methods. When Bob receives the stego object, he is able to extract the payload using the secret key. The Warden carries/intercepts the communications between Alice and Bob and uses a detector in an attempt to find out if steganography is being used. If a payload is detected, the Warden takes actions against Alice and Bob.

little to no detectable difference between the covers and stegos. An illustration of the Prisoners' Problem is shown in Figure 1.1.

Early forms of steganography relied on hiding information in unexpected places [15]. However, modern technology allows the modification of digital cover objects, such as digital images, to carry a payload. One is able to make minor changes to the pixels of a digital image in order to hide a payload. Digital media files are the most common mediums used for steganography. Their payload capacity, overwhelming presence, and susceptibility to minor changes makes them ideal for steganography. Currently, digital images are the most commonly studied mediums due to their simple properties and widespread use. This paper focuses specifically on uncompressed digital images, therefore other image types and respective embedding methods are not described.

An early and still current embedding method for uncompressed digital images is Least Significant Bit (LSB) steganography. In LSB Replacement, a specific version of LSB steganography, the payload is converted into a stream of bits. The cover is then traversed and the least significant bits of the pixels are modified to contain the payload. Typically, a secret key is shared between Alice and Bob to create a pseudo-random order to visit the pixels creating a random distribution for the changes in the cover image. Furthermore, a payload is often encrypted before it is embedded, resulting in a randomly distributed payload as well. Payloads are normally described in terms of how much information is hidden per unit of the cover image. In the case of LSBR steganography, a grayscale image could have a payload of 1 bit per pixel (bpp). This would be the maximum LSBR payload for an image, because each pixel of the stego image contains a payload bit. A payload of 1 bpp is easily detectable with modern steganalysis, thus payload sizes are often decreased and other techniques implemented to provide additional security.

The LSBR embedding strategy illustrates how a cover image can be modified with little to no visual differences in a way that is relatively easy to implement and lightweight to use. However, the modifications to the cover introduce statistical differences in the noise characteristic of the image, which can be used to distinguish it from cover objects, leading to the development of other LSB embedding methods. LSB steganographic algorithms have continued to improve by minimizing embedding impacts with the introduction of coding methods and minimization of distortion functions; however, detection methods have become more sophisticated as well.

1.2 Steganalysis

Steganalysis (also shown in Figure 1.1) takes the perspective of the Warden in the Prisoners' Problem. The Warden is trying to detect with high reliability and few errors whether or not particular messages (digital images) sent between Alice and Bob contain steganography.

Early methods of steganalysis designed for LSB embedding relied on specific detectors designed to test how often adjacent pixels changed, revealing inconsistencies between covers and stegos. These methods of steganalysis proved quite reliable, producing high accuracy with even small payloads [9]. However, modern embedding techniques are designed to foil these detectors by reducing the embedding impact by maintaining histograms or adjacencies, using coding methods, and/or avoiding more detectable parts of the cover images. These advances make it challenging to create new detectors in the same fashion.

The difficulties of writing specific detectors for each new steganographic method led to modern steganalysis which relies on machine learning strategies. Steganalysis is treated as a supervised learning problem. By providing a learning algorithm with labeled cover and stego examples, the learner attempts to find the best statistical separator between the images in order to classify them accurately. Classification of images with pixel values would be unreasonably difficult due to the domain size as well as the dependency on the content of the image. Instead, the values of all possible covers are reduced to a fixed number of features. These features can then be compared to highlight the differences between covers and stegos.

An early feature set for spatial domain images was the SPAM (Subtractive Pixel Adjacency Matrices) feature set which contained 686 features [19]. The SPAM features use higher-order Markov chains to create a high-pass filter on an image to remove the content and reveal the noise characteristic. This noise characteristic can therefore be used to model the patterns of cover images and reveal symmetries introduced by stego images. The SPAM features were very reliable at categorizing LSB implementations with much higher accuracy than the specific detectors. [19]

In addition to machine learning-based steganalysis, a method known as “calibration” was also introduced to improve the accuracy of detectors. Calibration was first used to estimate the cover histogram from an image [2], which could then be used to predict the number of changes made to the image and determine whether or not steganography was present. The aim of calibration is to create a reference image whose features characterize cover images irrespective of content. The reference cover can then be used to provide baseline feature values that can be combined with cover and stego features to suppress variations in the content of an image and increase the sensitivity to embedding. [12]

Current feature sets avoid targeting specific embedding algorithms. Instead, they develop general models of images in order to detect anomalies introduced by steganography. The hope in obtaining a general model of images is that irregularities introduced by steganography will violate the generalized cover model, making it detectable. One such feature set, the SRM features, takes this approach. It uses 12,753 features that model the noise characteristics of an image, which are normally affected by steganography, and is effective at detecting many advanced steganographic algorithms. The SRM feature set, which is used in the experiments in this paper, is discussed more in Chapter 3.

1.3 Structure of Paper

In this chapter a brief and general overview of steganography and steganalysis is given. The steganographer is attempting to send information secretly by hiding it in a cover object while the steganalyst attempts to detect the presence of hidden information. Throughout the rest of the paper, we take the perspective of the Warden, or steganalyst, in an attempt to improve the detection accuracy of state of the art steganalysis for uncompressed images. The rest of the paper is structured in the following way:

- The second chapter presents an ideal, hypothetical situation that motivated this research and the research hypothesis.
- The third chapter describes the setup of the experiments. This includes a description of the creation and construction of the dataset, the embedding methods used, the steganalysis method, the calibration strategy, and the computational requirements for the experiments conducted.
- The fourth chapter contains the results of the experiments as well as an analysis of their outcomes. The scope of research is then extended by further experiments.
- The fifth chapter concludes the research with a summary of the work done, an evaluation of the work performed, and proposed future work in the field.

Chapter 2

Problem Domain

2.1 Motivating Example

Imagine a hypothetical situation in which a steganalyst has been given a set of uncompressed images, which may or may not contain steganography. The set of images is composed of photos that contain overlapping content (i.e. images of the same scene or partially overlapping scenes). The steganalyst also has access to the cover source for the creation of training data. The steganalyst's job is to determine with high accuracy which, if any, of the images in the set contain steganography.

It seems very reasonable to assume a set of photos, perhaps from an individual's image library, would contain overlapping content. With the increasing capacities of digital media devices people are able to take many photos without a need to be selective or delete any. Therefore, it seems reasonable to expect that a set of images, perhaps obtained from an individual's image library, would contain images with overlapping content. Some situations where one could expect multiple images of the same or similar scene are:

- Capturing multiple images of the same scene with the intention of choosing the best one.
- Photographing the same scene with different alignments.
- Attempting to capture a panorama through multiple photographs of different parts of a scene.
- Retaking an image of people where someone was not looking or smiling.
- Recapturing a scene where changes occur in the background.

These situations are very common when taking photographs, and many of these images would contain the same frame with minor differences. Thus, the steganalyst would like to find a way to maximize the accuracy of the detector using the contents of the training and testing sets, which in this specific case is overlapping images.

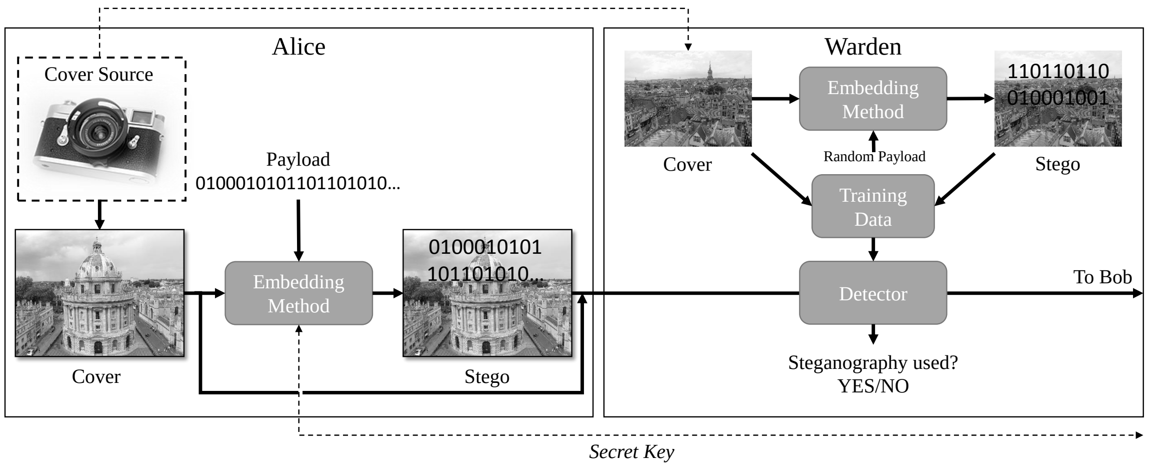

Figure 2.1: Illustration of the steganalyst described in the motivating example. The warden has access to the cover source and is able to develop training data that can be used to improve the detector.

2.2 Hypothesis

Calibration has proven to be a useful method for improving steganalysis by creating reference images. This paper extends the notion of calibration to independent images with overlapping content. If two images are taken of a scene with the same cover source and settings, we would expect the images to be very similar. Additionally, we would expect their noise characteristics and features to be similar. When a payload is embedded in an image, it introduces a stego signal into the noise characteristic of that image. Therefore, if there are two independent images with overlapping content, one which contains a payload and one which does not, we would expect the stego signal to disrupt the similarities in their features. In this case it would be reasonable to assume an image of a scene could be used to calibrate, or provide a reference for, a separate image of that same scene, assuming they were produced from the same cover source under the same settings.

In order formulate the problem we assume an ideal situation for the motivating example, under which to conduct our experiments. The following assumptions are used to describe what is know and presented to the steganalyst:

(i) The steganalyst is able to access the cover source in order to create training data.

(ii) The payload size and embedding method are known.

(iii) Pairs of images in the set have overlapping content.

(iv) The overlapping images have been taken using the same settings (ISO, exposure, white balance, etc.).

(v) In the pairs of overlapping images one image is known to be a cover, but the class of the other image is unknown.

An illustration of this situation is described in Figure 2.1.

The assumptions provide a limited yet simple starting point, however, most of the assumptions described could be overcome by future research. Assumption (iii) could potentially be automated to determine if images have overlapping content. Assumption (iv) could be determined by examining the EXIF data of the image to determine what settings were used to take the image. Assumption (v) could be overcome by testing the image pairings with the images swapped to determine if a payload is present. Some of these assumptions are visited again in Chapters 4 and 5, but as an initial study we will focus on testing the concept of using overlapping images for calibration.

Stated simply, the objectives of this paper are to determine: Can using independent images with overlapping content for calibration decrease the classification error of steganography in uncompressed images? Determining if the hypothesis is feasible requires constructing a dataset of overlapping images, creating stegos using current embedding methods, and using state of the art steganalysis, such as the feature set and classifier. Thus, the aims of this paper are to improve upon leading-edge steganalysis under certain conditions.

Chapter 3

Experiment Design

3.1 Laboratory Conditions

The assumptions described in Section 2.2 are extended to “laboratory conditions” used for the experiments in this paper. These conditions were created specifically to test the hypothesis and provide a reasonable starting place for future research, and therefore do not specifically contain “real world” application at this stage. The assumptions in Section 2.2 are extended to the following:

- A dataset is created containing:
  - covers and stegos and
  - independent, overlapping images created with the same cover source and settings.
- Stegos are created from the dataset with:
  - different embedding methods and
  - different payloads.
- Leading steganalysis methods are used (feature set and classifier).
- Various calibration formulae are used to determine the effects of calibration using overlapping covers.

Each element will be described in detail, however the overall work flow of the experimental process is shown in Figure 3.1.

Although practical application is not the goal of this paper, many decisions were made with realistic circumstances in mind. The dataset constructed, for example, maintains properties that could be similar in many regards to a typical individual’s image library. The embedding methods are current techniques that could be used in practice, and the classifier and features used to build detectors are current, state of the art implementations. Future work is described in Chapter 5.

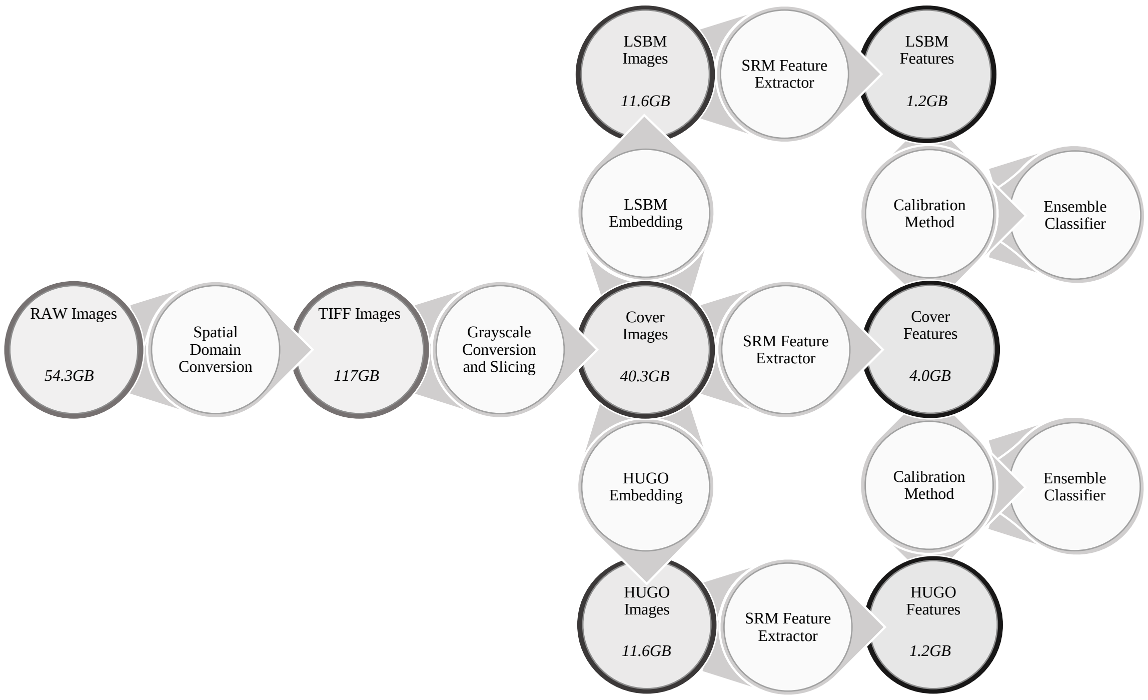

Figure 3.1: Flow chart of the experiment process. As discussed in Section 3.2 the cover sizes (images and features) include all 7 aspects. The LSBM and HUGO embedding was performed on one aspect with two different payloads, creating the image and feature size difference visible in the chart.

3.2 Dataset and Terminology

The dataset constructed for the experiments is composed of images that have specific amounts of overlap. These images are divided into cover images and stego images created from the covers.

3.2.1 Structure Of Dataset

The cover images in the dataset make up a corpus with 7 sets of images called *aspects*. An aspect is a set of images that has a defined amount of overlap with other aspects. Aspects overlap by amounts of 100%, 75%, 50%, 25%, and 0%. Additionally, aspects are referred to as letters A-G or by the amount that they overlap. A set of 7 images (one from each aspect) that were taken at the same location and relative time are referred to as a *scene*. Figure 3.2 shows an image from each aspect from the same scene. The first aspect, aspect A, is referred to as the *origin* aspect.

The images were taken in portrait mode using a single camera[1]. The camera was set up to take 12 megapixel images (3000×4000 pixels) in RAW format. The focus, F-stop, exposure, and ISO were set manually for each scene and were not changed

[1]Canon PowerShot G16

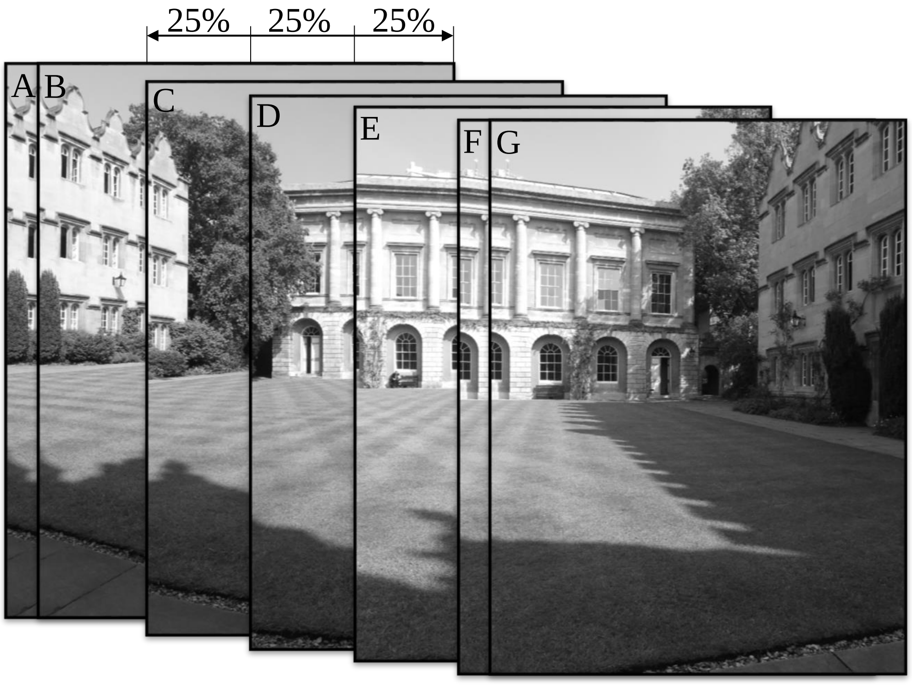

Figure 3.2: These images illustrate the overlaps of the different aspects of a scene. Images A (the origin) and B overlap 100% with each other. Aspect C overlaps 75% with A (and B). Aspect D overlaps 50% with A. Aspect E overlaps 25% with A. Aspects F and G overlap 100% with each other, but have 0% overlap with A.

between aspects. A tripod was used to limit the amount of movement in the camera between aspects. The aspects of a scene were created by using a point of reference in the origin image, such as an object in the frame. The camera was then panned to move the point of reference 25% of the frame to capture the next image in the scene.[2] The overlaps cannot be expected to be exact due to manual estimation. Therefore, a 25% overlap is approximately 25% but could be between 20 and 30%. Furthermore, these approximations can be exaggerated further when considering optical and perspective distortions in photography. However, these imperfections seem to add to the potential of practical implementation by introducing realistic, natural differences in overlapping images.

The dataset is composed of 500 portrait image scenes of 3000×4000 pixel images, mostly of outdoor photographs in parks and cities. Each image was then divided horizontally into 5 sub-images and converted to grayscale to both increase the number of images in each aspect for additional training and testing data for the classifier and decrease the size of the images for the embedding and feature extraction process.

[2]Variations in zoom prevent the the use a defined degree rotation to obtain an exact 25% offset between aspects.

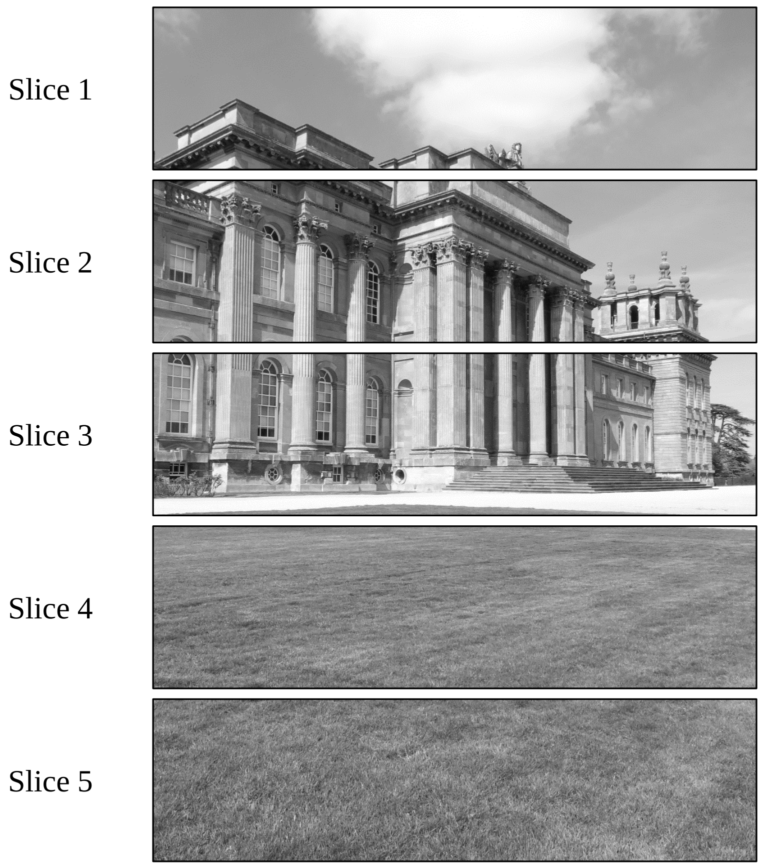

Figure 3.3: Sub-images, or slices, created from the original image after it is converted to grayscale.

An example of the image slicing is shown in Figure 3.3. Horizontal slicing creates images with sky in the top slice, which is normally relatively smooth, and ground in the bottom slice, which is normally noisy. This greatly diversifies the content of the dataset which can reduce detector anomalies introduced by similar images. Once the images are "sliced," there are 2500 sub-images per aspect of size 3000×800 pixels. The stego subsets are then created by embedding in the sub-images of the cover subset.

The images provide a realistic example of overlapping photographs that someone may take, even though there are consistent overlapping patterns for the individual images. They contain mild changes in the images between aspects that would occur in normal photography, such as slight changes in trees due to the wind, movement of clouds, and people moving in the background. This introduces a "natural" amount of noise into the images that would be expected in the "real world" overlapping images.

3.2.2 Dataset Construction

Before the images could be used as covers and stegos, they must be converted and processed. First, the raw images are converted to the spatial domain 8-bit TIFF format by using the Digital Photo Professional software supported by Canon. During the conversion the white balance setting is maintained for each aspect in a scene. The images are then converted to grayscale and cropped to create the sub-images for each aspect, using the ImageMagick software. The sub-images are then ready to be used as cover objects and stego objects.

It is important to mention the software used to develop and process the images, because different softwares may lead to differences in the images. Adobe Photoshop, for example may produce dissimilar images due to differences in its grayscale conversion algorithms.

3.3 Steganographic Methods

Two embedding methods were used for the experiments in this paper: LSBM (Least Significant Bit Manipulation, also known as ± 1 embedding) and HUGO (Highly Undetectable steGO). These embedding methods are applied to the uncompressed, 8-bit, grayscale images. Thus, a pixel contains one byte of image information, and all changes are made to the least significant bit of the pixel value. In these grayscale images the maximum payload is 1bpp for LSBM and varies for HUGO according to the image size and content. The decisions for payload sizes were based on the initial detectability of the steganography in the baseline test. In this test a detector was built on the covers and stegos of a single aspect without additional information to give a baseline accuracy. If the baseline accuracy were too high (e.g. 99%), then no noticeable gains can be observed when performing calibration with overlapping images. If the baseline accuracy were too low (e.g. 51%), then no quantitative comparison for the detector could be established.[3] Payload sizes for the embedding methods were obtained experimentally.

3.3.1 LSBM

LSB steganography is the most widely distributed and used method of steganography in uncompressed images [3]. While LSB Replacement (LSBR) is a completely flawed method of steganography, LSBM is still considered secure if a small enough payload is used. LSBM embedding also hides the payload in the pixels of the images by modifying the value of its least significant bit, however the algorithm is slightly modified to avoid the pitfals of LSBR. In LSBM the cover is traversed just as it would be with LSBR. If the payload value is already contained in the pixel, then the algorithm continues to look at the next payload bit and the next cover byte. If the pixel value does not contain the payload bit, then the pixel is randomly incremented or decremented by 1. The two special cases for this method are when the pixel is

[3]The lowest detector accuracy is 50%, indicating a random cover-stego classification.

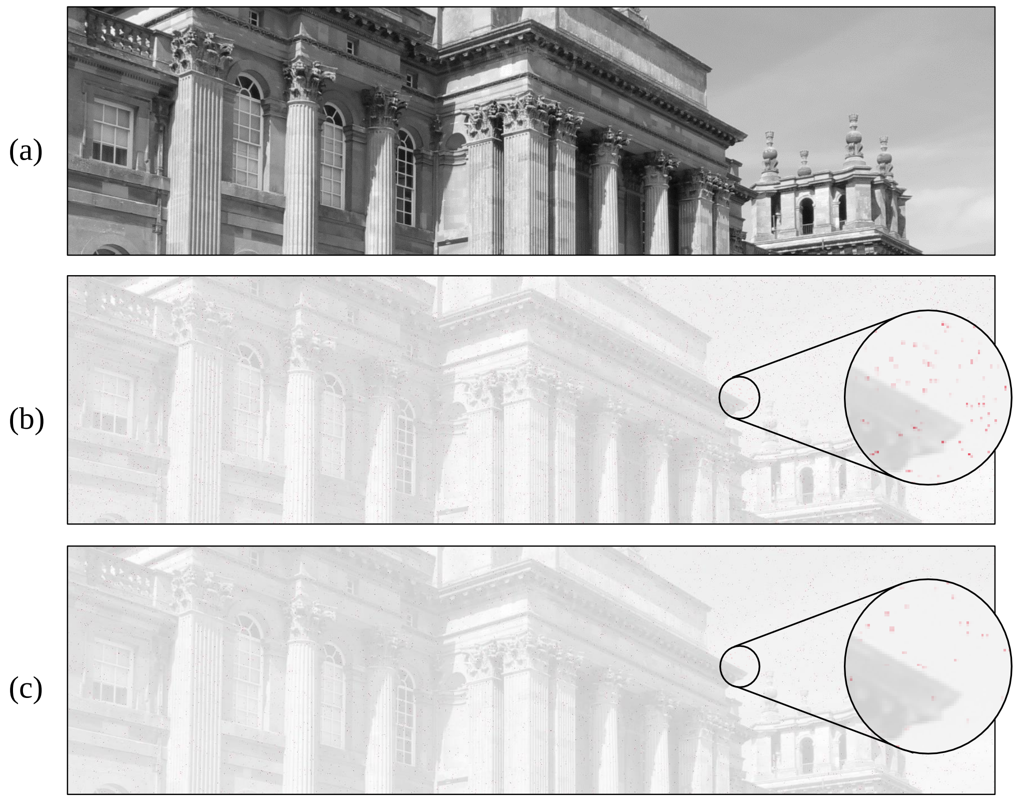

Figure 3.4: Comparisons of cover and stego images highlighting the LSB pixel changes made to the cover. (a) Cover image (b) LSBM embedding with a payload of 0.01 bpp (c) LSBM embedding with a payload of 0.005 bpp

the maximum value, in which case it is always decremented, or if the pixel has the minimum value in which case it is always incremented. Therefore, assuming a random cover and payload, LSBM only changes the value of a cover pixel 50% of the time.

The LSBM simulator created to produce stego objects in the experiments simulates a random payload of a defined length with a random key. This is realistic for steganalysts who do not know the key and where the payload is normally encrypted before embedding. For the experiments in this paper, payloads of 0.01bpp and 0.005bpp were used.

One drawback of LSBM is that the model of images it uses is too general. It assumes that the pixels are independently and identically distributed throughout the image. Embedding is performed without considering relationships between pixels. Thus, the effects of embedding with LSBM appears like the introduction of random noise as shown in Figure 3.4. However, pixel dependencies are readily present in natural images, making it detectable with large enough payloads. The LSBM model overgeneralizes the structure of an image, but with a small enough payload it is considered secure.

3.3.2 HUGO

HUGO is a more advanced spatial domain embedding method for uncompressed images. It is reportedly capable of embedding messages 7 times longer than LSBM with the same level of security [20]. The HUGO embedding method is separated into two components: coding schemes and defining a better cover model. These methods allow HUGO to hide information in the "noisier" parts of the image, as shown in Figure 3.5. When comparing Figure 3.5 to Figure 3.4, one can see the difference in the two embedding methods. The changes made by the LSBM algorithm are randomly distributed across the cover, whereas the changes made by the HUGO algorithm are concentrated in the noisy areas of the image. Fewer changes to the smooth areas of the image where large amounts of noise are seldom present reduces detectable anomalies which increases the security of the stego.[4]

The first component of the HUGO embedding algorithm is to use coding schemes to embed a payload. Coding schemes aim to represent a given payload with more bits but using few changes. HUGO uses syndrome-trellis codes to minimize the impact to the cover image. By reducing the number of changes to the original image, the probability of detection decreases [1]. Using coding schemes is a common practice with LSB implementations and is not novel to HUGO. Rather, it is seen as a best practice approach to steganography.

The second component of the HUGO embedding method aims to gain a better model of cover images. By considering inter-image pixel dependencies, the HUGO algorithm quantifies the impact of embedding in each location. This is done by assigning a value, or cost, to each pixel before attempting to embed the payload. These costs are constructed by using pixel adjacencies through the use of SPAM features to construct high-dimensional models of the embedding impact at each location. By minimizing the changes that are detectable by these features and using coding schemes, the embedding algorithm is able to achieve a much safer embedding method. [20]

The implementation of HUGO used for the experiments and described in this paper is the Binghamtom HUGO simulator written in Matlab [5]. The algorithm has not been made publicly available to minimize security risks, however, the simulator allows an accurate implementation of a random key and payload, given a payload size, to facilitate the testing of steganalysis methods. The payload values used in the experiments in this paper for HUGO were 0.1bpp and 0.05bpp.

[4]Some of the images in the dataset had to be replaced, when the HUGO implementation was unable to embed the payload. This was likely due to large overexposed areas of the image. When an image was unable to be used, the entire scene was replaced.

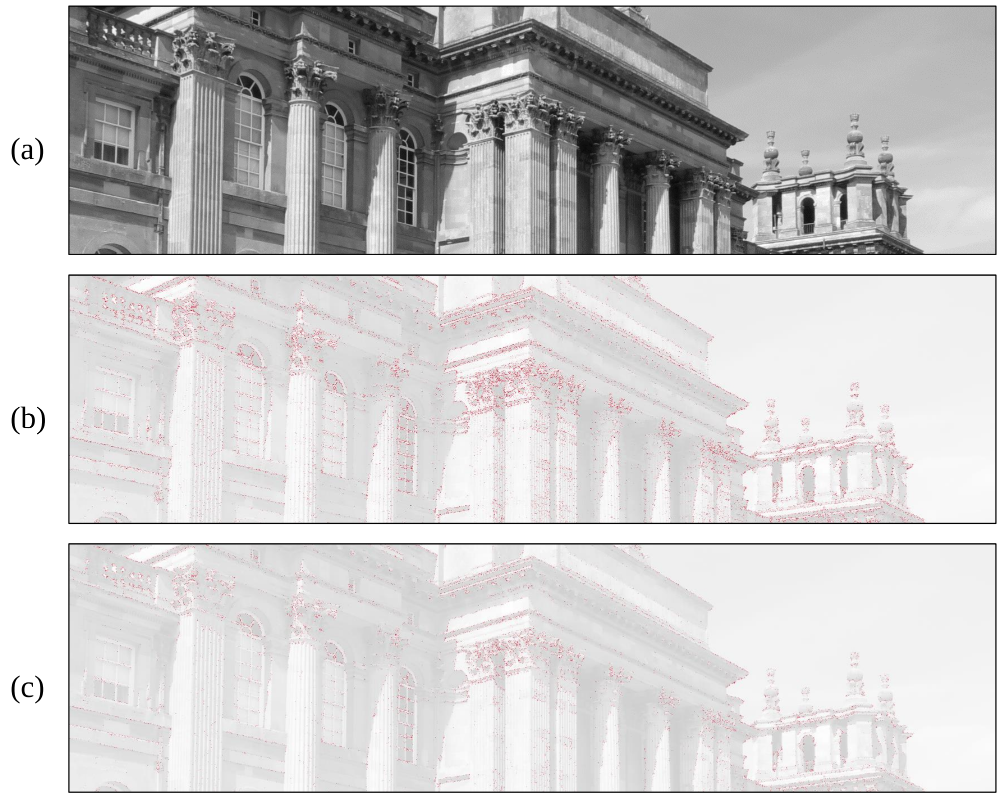

Figure 3.5: Comparisons of cover and stego images highlighting the LSB pixel changes made to the cover. (a) Cover image (b) HUGO embedding with a payload of 0.1 bpp (c) HUGO embedding with a payload of 0.05 bpp

3.4 Steganalysis Method

Most current steganalysis methods utilize machine learning techniques. The research contained in this paper uses the SRMQ1 features (a version of the SRM features) with an ensemble classifier to create steganographic detectors. The SRM features are a set of values that characterize the noise in an image. The ensemble classifier is an off the shelf classifier that has been designed specifically for steganalysis. Furthermore, it is currently the state of the art tool for steganalysis due to its accuracy and low time complexity.

3.4.1 SRMQ1 Features

The SRM (Spatial domain Rich Model) features are a current steganalysis feature set that aims to generalize the cover model [4]. This feature set was the winning entry in the BOSS (Break Our Steganography System) contest in 2011 [14]. By combining a variety of submodels into a larger general model, the features are able detect a wide range of steganographic implementations, instead of being limited to a particular embedding method. Each submodel is created through three steps: computing residuals, truncation and quantization, and construction of co-occurrences. The aim of the SRM feature set is to create "a combination of submodels that achieves a good trade-off between model dimensionality and detection accuracy" [4]. The submodels are used to construct a model of cover image dependencies that, when broken through steganography, are detectable.

The first step in the submodel creation is to compute the residuals. A residual is a measurement difference between the value of a statistical model and the measured. Thus, the residuals described in SRM show how much observed pixels differ from a theoretical model, describing the typical noise of an image. By subtracting a denoised version of the image from itself, the model is able to reduce the impact of variations in content of the image. The residuals are essentially the results from applying linear and non-linear filters to characterize the noise of the image. This is particularly helpful in steganalysis, because an embedded payload is a signal hiding in the noise of the image.

Various types of residuals make up the different submodels. The residuals are calculated in two ways: locally and using the "minmax" [4]. The locally-supported features are similar to the SPAM features. They are predicted by the neighboring pixels and are computed by applying a linear high-pass filter. The minmax residuals are obtained by applying more than one linear filter to a pixel to obtain the value. Computing the minmax residuals enables the model to obtain non-linear models of the noise in the image.

The second step in SRM submodel creation is the truncation and quantization step. The purpose of this step is to use quantization to reduce the domain of the the residuals. The quantization is computed with different quantization steps to better detect embedding changes in smooth and textured areas. The algorithm optimizes the quantization step for each submodel.[5] Quantization allows the domain of real numbers to be put into bins, allowing histograms to be computed from the residuals.

After the residuals are quantized, co-occurrence matrices are constructed. The co-occurrence matrices are multi-dimensional histograms that show how often a value occurs in a given context. The SRM algorithm uses these four-dimensional co-occurrence matrices of the quantized residuals to make up the submodels for the feature set.

This process is used to create 45 submodels. There are 12 submodels of 169 features composed from SPAM-like residuals and 33 submodels of 325 features of the minmax type, giving a total of 12,753 features for the SRM feature set. The SRM model is successful in detecting many uncompressed image embedding algorithms, even HUGO which was designed to avoid changes detectable by the SPAM features.

The SRM implementation used in the experiments contained in this paper is an off the shelf implementation of SRMQ1 written in Matlab [6]. The SRMQ1 features use a fixed quantization step rather than optimizing for the best quantization step. The decision to use SRMQ1 over SRM was made to decrease the computation time.

[5]This is not done with SRMQ1.

Feature extraction time for SRMQ1 is approximately 1/3 that of SRM which makes a significant difference in the computational requirements as shown in Section 3.6. Experimental results show a relatively small loss in accuracy and approximately no loss for the LSBM. [4]

Once the features are extracted for each image in an aspect, they are combined into a matrix representing a cover or stego aspect as a whole. Each row contains the features for a different image and each column holds the values for the same feature. For example, cover features for aspect A are represented in the feature matrix C_A which has a size of 2,500x12,753. Furthermore, rows with same value from different aspects or embedding methods correspond to the same scene. In other words, if S_A represents the stego matrix for aspect A, then the a row from S_A has its corresponding cover in the same row of C_A. This concept also extends to the other, overlapping aspects, e.g., a row in C_C corresponds to an image that has 75% overlap with the image used for the same row in C_A. Structuring the cover and stego matrices in this way allows them to be used with the ensemble classifier for classification. Additionally, the matrix structure allows for calibration formulae to be implemented easily with overlapping aspects using addition, subtraction, and element-wise operations between matrices. Calibration methods are discussed in detail in Section 3.5.

3.4.2 Ensemble Classifier

The machine learning algorithm used to learn and classify the images as cover objects or stego objects is an ensemble classifier utilizing a random forest. This classifier is a state of the art (for steganalysis), off the shelf, steganalysis classifier written in Matlab [7]. The advantage of using the ensemble classifier over a traditionally favored classifier, such as the Support Vector Machine (SVM), is the training time. The training time of the ensemble classifier is approximately eight times faster than the SVM if a linear kernel is used and even more so if a Gaussian kernel is used. [13]

The ensemble classifier works by training a number of random forest learners. Classification is then performed by combining the final decisions of all the learners in a majority vote. The base learners of the random forest are FLDs. These learners search for optimal ways to separate the cover and stego classes by creating a linear classifier for a subset of the SRM features. The subset of features chosen to train the learners is significantly smaller than the full dimensionality of the SRM features. This greatly decreases the computational complexity of the learners. The random forests are trained on a random sample with replacement of the training tuples of cover and stego images. The FLDs then maximize the "separation," which is the ratio of the variance between covers and stegos to the variance within covers and stegos. Although the individual learners are very poor, the classifier improves with the number of learners used. Both the number of dimensions used for each learner and the number of learners is optimized in the classifier. [13, 16]

In the experiments performed, false positives and false negatives are considered equally undesirable, thus no bias is given. The metric used for the accuracy of the classifier and displayed in results in Chapter 4 is the average classification error. A 10-fold cross validation was used so that the majority of the data could be used for training. In other words, ten runs of the classifier were performed, using 9/10 of the of the cover and stego data as the training set and 1/10 as the testing sets, ensuring that each testing set is different for each run. The average testing error is the average of the testing errors for all of these tests. The implementation of the ensemble classifier used in the experiments was written to allow only the same number of cover and stego objects, as well as pairing features between the covers and the stegos. Minor modifications were made to the implementation for different experiments such as paring features for concatenation calibrations (described in Section 3.5), randomizing the rows of the cover and stego matrices before training, and allowing for unequal amounts of cover and stego data for later tests.

3.5 Calibration Method

The calibrations used in the experiments act as element-wise formulae on the feature matrices before they are passed to the classifier. The calibration methods are described as functions of x and y. In all cases, unless otherwise defined, x represents overlapping cover features of a defined amount and y represents the origin cover aspect and corresponding stego aspect.

For example, the origin feature matrices from a cover and stego, $C_A$and S_A are going to be classified after being calibrated by an overlapping cover reference aspect C_B. Using a calibration formula such as subtraction, represented by $x - y$, we are able to calibrate the features before introducing them into the classifier. Therefore, training and testing the ensemble classifier on the subtraction calibration conducts the experiment: $C_B - C_A$ *vs.* $C_B - S_A$.

Another calibration method used is the concatenations operation, which retains two image feature vectors in a single row of the feature matrix. The concatenation operation is represented by a $\bullet$. This calibration could be particularly useful to combine two calibration methods in hopes to further improve upon the error rate by using the features of both sub-methods. When the concatenation operation is used, it is important to maintain the selection of the features from both halves of the new feature vector in the ensemble classifier. The training algorithm of the ensemble classifier was modified to pair the features when vectors are concatenated.

By using concatenation as well as a variety of calibrations allows, we are able to compare the average testing errors from the experiments and determine which calibration performs the best. All calibration operations described in the results tables presented in Chapter 4 are element-wise operations. There are some experiments that involve element division, which will produce undefined values for some of the calibrated features. In these cases, the undefined values are changed to 0 after the operation has been performed.

3.6 Computational Requirements

One of the major difficulties to overcome in this research was the computational requirements needed for the experiments. While some of the steps required minimal

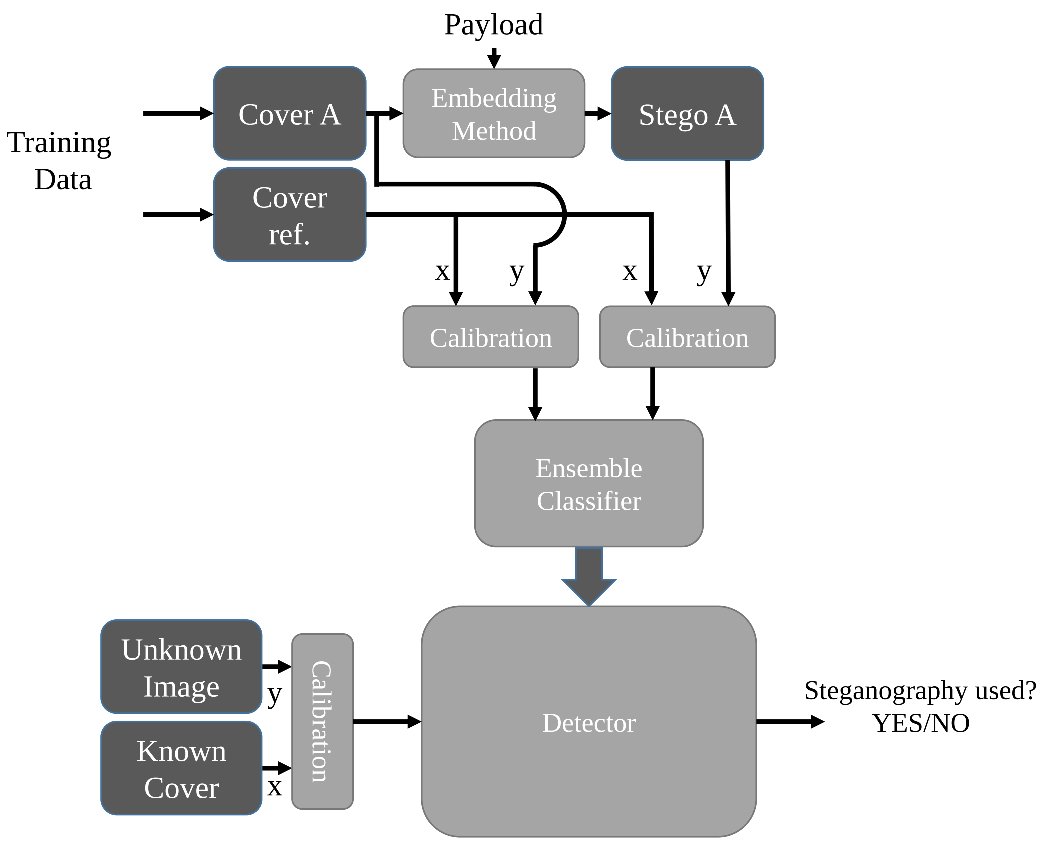

Figure 3.6: Diagram of the training and testing process with calibration for the motivating example described in Section 2.1. The cover reference (ref.) images are used to calibrate the covers and stegos before training the ensemble classifier. The ensemble classifier produces a detector which is then used on the calibrated testing data to determine whether steganography is present or not.

computational effort, such as the LSBM embedding algorithm which operates in linear time, others were more significant. The HUGO embedding method, for example, required an average of 8 seconds of computational time per sub-image, resulting in 5 hours 33 minutes per aspect.[6] The most significant computational constraint, however, was presented by the extraction of the SRMQ1 features. The average time to extract the features for a single sub-image was approximately 4 minutes and 35 seconds and required approximately 3GB of memory during computation. Thus, it would require 191 hours or almost 8 days to compute a single aspect. Additionally, the need for usable baseline values described, as described in Section 3.3, resulted in the calculation of four additional stego aspects (two for LSBM and two for HUGO) containing different payloads. The payloads of 0.05 bpp and 0.02 bpp used for LSBM

[6]All computations were performed with an 3.47GHz Intel Xeon X5690 Processor (6 cores).

were too detectable as were the payloads of 0.4 bpp and 0.2 bpp for HUGO. These four aspects required embedding and feature extraction, although they were unable to be used with the calibration tests.

Additional embedding and extraction was needed for aspects to correct errors, verify features, and confirm results. The total amount of computational time used for the feature extraction process for the work contained in this paper was computed to be 743 core days.[7] In order to complete these experiments in a reasonable time, the work was parsed and distributed to a cluster of computers with a job management system that my supervisor possessed. Learning to use the job management system on the cluster setup allowed the computational time for feature extraction to be reduced to just under 18 hours per aspect compared to the 191 hours it would have taken otherwise.

The ensemble classifier used in the experiments was inconsistent with the computational time needed to complete the experiments with a 10-fold cross validation due the optimization of the parameters. Some of the experiments took up to 10 hours, however many were less than 10 minutes. The total computation time is unknown, but approximately 390 different experiments were conducted to obtain or lead to the results contained in this paper.

[7]Cropped images used for additional experiments described in Chapter 4 required an average of 3 minutes of computation time when the image was 75% of the original and 2 minutes and 20 seconds when the image was 50% of the original. These result in approximately 5.2 and 4 days per aspect respectively. The total computational time figure includes these computations as well as the Fixed Scene extractions used Section 4.3.2.

Chapter 4

Results and Analysis

4.1 Baseline for Detectors

The first step in determining whether or not calibration is possible with overlapping images, we must obtain a baseline. The baseline figure for an embedding method and corresponding payload is computed by training and testing the classifier on covers and stegos from the same aspect. The baseline test can be described as the cover vs. stego test which provides an ideal situation for the classifier. The stego was created from the cover image and any differences in the feature values between the cover and stego matrices are purely from the embedded payload.

Table 4.1 shows the average testing errors for the different embedding methods and payloads in the baseline test. The testing errors are representative of current, state of the art steganalysis for the respective embedding methods and payloads. Therefore, if calibrating with overlapping images achieves a lower testing error than the baseline values computed here, the improvements in the detection accuracy are improving on the state of the art techniques.

Embedding Method	Payload	Average Testing Error
LSBM	0.01 bpp	0.107
LSBM	0.005 bpp	0.204
HUGO	0.1 bpp	0.065
HUGO	0.05 bpp	0.216

Table 4.1: Average testing error baseline for all embedding methods and payloads. Payloads are in bits per pixel.

4.2 Overlapping Images

After establishing the baseline values, calibration using the overlapping images can now be tested. This section describes the same process for both the LSBM embedding method and the HUGO embedding method. The similarities and differences in the results between the two embedding methods are highlighted.

4.2.1 Calibrating LSBM

The average testing error for LSBM embedding method with a 0.01 bpp payload for various calibrations and overlaps is shown in Table 4.2. In the best performing calibration, where there is 100% overlap between the images, the testing error is approximately 9.7× smaller than the baseline value. Furthermore, the baseline testing error is improved in the other overlaps by factors of 3.8×, 2.9×, 2.1×, and 1.6×, respectively. While some of the calibrations in this table make the average testing error increase, the majority of the calibrations decrease the testing error, improving the accuracy of the detector. Additionally, we can see that the improvements become smaller when as the overlap decreases. We would expect this result, considering the statistical differences in the overlapping images would tend to increase as the amount of overlap decreases. Towards the bottom of the table there are a number of concatenations that are combinations of the best performing calibrations. Some slight decreases in test error can be obtained by these concatenations.

Interestingly, the average testing error is 1.6× smaller than the baseline when there is no overlap between the images at all. This result implies that an image taken with the same settings still provides a good enough reference for calibration even if no overlapping content is present (at least with this embedding method an payload).

After obtaining successful results with LSBM embedding with a payload of 0.01 bpp, the same experiment is performed with LSBM embedding with a different payload (0.005 bpp). Table 4.3 shows the results from this test. In this table we focus on the calibration methods that performed best in the 0.01 bpp LSBM test.

The results from the 0.005 bpp LSBM experiment show similarities to the 0.01 bpp experiment. The calibrations in this table with 100% overlap all produce testing errors lower than the baseline value, with the best calibration providing a testing error 9.3× smaller than the baseline. We see that, again, the performance decreases as the amount of overlap decreases, and the same calibration from both Tables 4.2 and 4.3 produced the lowest testing error. We also notice that when the payload is 0.005 bpp no gain in performance over the baseline is achieved when the amount of overlap is less than 50%. Furthermore, calibration begins to perform worse than the baseline when there is no overlap or 25% overlap between the images and in some cases when there is 50% overlap.

Cropped Images

The results obtained in the initial LSBM tests support the hypothesis, and the lowest testing errors with both payloads were achieved when the images overlapped 100%

Calibration		Overlap				
		100%	75%	50%	25%	none
payload 0.01 bpp						
(none) baseline	0.107					
$x \bullet y$		0.023	0.063	0.086	0.101	0.111
$x - y$		0.015	0.037	0.044	0.072	0.066
$\frac{x-y}{x+y}$		0.019	0.043	0.078	0.088	0.114
$\left\lvert\frac{x-y}{x+y}\right\rvert$		0.092	0.295	0.363	0.422	0.456
$(x-y)^2$		0.204	0.3672	0.402	0.411	0.432
$\frac{(x-y)^2}{x+y}$		0.095	0.299	0.364	0.396	0.408
$\frac{x^2-y^2}{x^2+y^2}$		0.019	0.051	0.070	0.119	0.122
$x \bullet x - y$		0.021	0.061	0.074	0.096	0.096
$y \bullet x - y$		0.023	0.060	0.072	0.100	0.098
$x \bullet y \bullet x - y$		0.022	0.070	0.090	0.104	0.107
$x - y \bullet \frac{x-y}{x+y}$		0.012	0.024	0.041	**0.051**	0.067
$x - y \bullet \frac{x^2-y^2}{x^2+y^2}$		**0.011**	**0.028**	**0.037**	0.052	**0.065**

Table 4.2: Average testing error for detecting LSBM embedding with a payload of 0.01 bits per pixel for various calibration methods. Bold numbers indicate the best-in-column value if it is better than the baseline.

Calibration		Overlap				
		100%	75%	50%	25%	none
payload 0.005 bpp						
(none) baseline	0.204					
$x \bullet y$		0.076	0.161	**0.192**	0.206	0.222
$x - y$		0.044	0.169	0.204	0.230	0.240
$\frac{x-y}{x+y}$		0.062	0.174	0.218	0.232	0.246
$\frac{x^2-y^2}{x^2+y^2}$		0.039	0.176	0.215	0.238	0.253
$x - y \bullet \frac{x^2-y^2}{x^2+y^2}$		**0.022**	**0.096**	**0.192**	0.220	0.228

Table 4.3: Average testing error for detecting LSBM embedding with a payload of 0.005 bits per pixel for various calibration methods. Bold numbers indicate the best-in-column value if it is better than the baseline.

Calibration		Overlap (Cropped)			Overlap (Cropped)	
		75%	50%		75%	50%
payload 0.01 bpp				*0.005 bpp*		
(none) baseline	0.107	0.125	0.149	0.204	0.145	0.192
$x \bullet y$		0.058	0.089		0.145	0.192
$x - y$		0.029	0.069		0.147	0.200
$\frac{x-y}{x+y}$		0.044	0.086		0.151	0.204
$\frac{x^2-y^2}{x^2+y^2}$		0.044	0.077		0.156	0.201
$x - y \bullet \frac{x^2-y^2}{x^2+y^2}$		**0.027**	**0.034**		**0.099**	**0.177**

Table 4.4: Average testing error for detecting LSBM embedding with payloads of 0.01 and 0.005 bits per pixel for various calibrations methods in overlapping images cropped to have 100% overlap. Bold numbers indicate the best-in-column value if it is better than the original baseline.

(although there still improvements with less overlap). Initially it was believed that calibration with overlaps less than 100% could be improved by removing the portions of the image that did not overlap. In the case of a 75% overlapping reference, the 25% that does not overlap is hindering the calibration. In order to test this assumption, the original covers, stegos, and overlapping covers were cropped to create smaller images with 100% overlap. After cropping the images, the features were extracted, and the calibration tests were repeated, focusing on a subset of the original calibrations. The results for both LSBM payloads are displayed in Table 4.4.

Some of the error rates were lower in the cropped images; however, overall no significant improvements are observed. It seems that too much of the stego signal is being removed by cropping the images making it less detectable. This is explained by the Square Root Law of steganographic capacity (SRL) [11]. In the case of LSBM we would assume that the payload is evenly distributed throughout the image, therefore removing 25% of the image is also removing approximately 25% of the payload. According to the Square Root Law for the IID cover model, which shows that the secure capacity of a stego object shrinks with the square root of its cover size, this is making the payload less detectable. So, by linearly reducing the cover size and payload size, detectability is decreasing. Therefore, the marginal gains are likely due to noise.

4.2.2 Calibration HUGO

The experiments conducted in the previous section were repeated with the HUGO embedding algorithm using payloads of 0.1 bpp and 0.05 bpp. The results from the calibration tests are shown in Tables 4.5 and 4.6 for the payloads of 0.1 bpp and 0.05bpp, respectively.

In Table 4.5, which shows HUGO with a payload of 0.1 bpp, the best performing calibration is again the 100% overlap which is approximately 5.9× smaller than the baseline. Similar to the 0.01 bpp LSBM tests in Table 4.2, as the amount of overlap increases the error rate increases, and gains are still observed even when there is no overlap between the aspects.

The 0.05 bpp HUGO test results are shown in Table 4.6. Once again the best performing calibration is the 100% overlap which is 2.6× smaller than the baseline. The average error also increases as the overlap decreases, however it performs worse than the baseline if the overlap is less than 75%.

The results from HUGO experiments are similar in many respects to the results obtained in the LSBM experiments. Testing error increases as the amount of overlap decreases, and improvements in the accuracy of the detectors are achieved with calibration although the greatest improvements are achieved with different formulae. However, many of the HUGO tests performed poorly, increasing the error rate above the baseline. Also, no additional benefit was achieved with the concatenation of the calibrations. Therefore, we can deduce that HUGO noise is better at modeling cover noise than the LSBM embedding method.

In both Tables 4.5 and 4.6 the $x - y$ calibration performs the best which differs from the LSBM method. And although Table 4.5 improves even when there in no overlap between the images (similar to Table 4.2), Table 4.6 shows that there is no

Calibration		Overlap				
		100%	75%	50%	25%	none
payload 0.1 bpp						
(none) baseline	0.065					
$x \bullet y$		0.036	0.059	0.068	0.073	0.072
$x - y$		**0.011**	**0.021**	**0.034**	**0.041**	**0.050**
$\frac{x-y}{x+y}$		0.127	0.172	0.198	0.206	0.210
$\left\lvert\frac{x-y}{x+y}\right\rvert$		0.198	0.382	0.445	0.466	0.478
$(x-y)^2$		0.210	0.432	0.482	0.511	0.499
$\frac{(x-y)^2}{x+y}$		0.145	0.336	0.401	0.431	0.452
$\frac{x^2-y^2}{x^2+y^2}$		0.120	0.163	0.196	0.202	0.217
$x \bullet x - y$		0.023	0.050	0.069	0.058	0.069
$y \bullet x - y$		0.026	0.037	0.049	0.060	0.061
$x \bullet y \bullet x - y$		0.038	0.057	0.087	0.082	0.083

Table 4.5: Average testing error for detecting HUGO embedding with a payload of 0.1 bits per pixel for various calibration methods. Bold numbers indicate the best-in-column value if it is better than the baseline.

Calibration		Overlap				
		100%	75%	50%	25%	none
payload 0.05 bpp						
(none) baseline	0.216					
$x \bullet y$		0.141	0.216	0.249	0.255	0.262
$x - y$		**0.082**	**0.189**	0.239	0.251	0.251
$\frac{x - y}{x + y}$		0.241	0.303	0.321	0.329	0.336
$x \bullet x - y$		0.138	0.217	0.249	0.254	0.263
$x \bullet y \bullet x - y$		0.138	0.218	0.248	0.250	0.263

Table 4.6: Average testing error for detecting HUGO embedding with a payload of 0.05 bits per pixel for various calibration methods. Bold numbers indicate the best-in-column value if it is better than the baseline.

Calibration		Overlap (Cropped)				Overlap (Cropped)	
		75%	50%			75%	50%
payload 0.1 bpp				*0.05 bpp*			
(none) baseline	0.065	0.107	0.154		0.216	0.259	0.311
$x \bullet y$		0.094	0.151			0.235	0.294
$x - y$		0.066	0.137			**0.215**	0.275
$\frac{x - y}{x + y}$		0.183	0.221			0.309	0.349
$x \bullet x - y$		0.088	0.151			0.234	0.288
$x \bullet y \bullet x - y$		0.098	0.152			0.236	0.288

Table 4.7: Average testing error for detecting HUGO embedding with payloads of 0.1 and 0.05 bits per pixel for various calibrations methods in overlapping images cropped to have 100% overlap. Bold numbers indicate the best-in-column value if it is better than the original baseline.

gain with less than 75% overlap between the image sets.

Cropped Images

As with the LSBM tests, the cropping image tests were also performed with the HUGO embedding method. The same operation was performed: The images with 75% and 50% overlap were cropped with the origin image and its corresponding stego to have approximately 100% overlap with one another, albeit with a smaller size. The features were extracted, and the calibration experiments were performed. The results are displayed in Table 4.7.

In the results obtained only one of the calibrations performed better than the baseline; however, it is only a 0.5% difference which could be attributed to noise. It seems that the cropped image experiments performed with HUGO supports the observation made for the cropped LSBM experiments. While the HUGO embedding method does not support the IID cover model of images, it seems that a significant amount of the stego signal is lost when the images are cropped, making it more difficult to detect.

4.3 Further Exploration of Results

The results obtained for both the LSBM and HUGO embedding methods support the hypothesis that using overlapping images for calibration can increase the detection of steganography. While a variety of calibrations outperformed the baseline, the $x - y \bullet \dfrac{x^2 - y^2}{x^2 + y^2}$ method was most successful for calibrating LSBM, and $x - y$ was best for calibrating HUGO. This implies that there is not a best calibration for all embedding methods, rather embedding methods have different effects on features that can be emphasized through different formulae. Let us now turn our attention to some additional experiments to further our understanding of steganalysis with overlapping images.

With the general consistency of the results in the previous section, further experiments will focus on the more advanced embedding scheme, HUGO. In the following experiments, the HUGO embedding method and 0.05 bpp payload are fixed, unless otherwise indicated, in order to focus on possibilities and discrepancies presented with overlapping images.

4.3.1 Potential Objections

An initial argument that could be made is that the calibration experiments are provided with more cover data for training and testing than the baseline test, which could be causing the improvements in the performance of the detector. In order to test this quandary, the baseline experiment was conducted two more times with double the amount of cover images: firstly, with additional covers from the 100% overlapping aspect and secondly with additional covers from the 0% overlapping aspect. This can be represented symbolically as C_A, C_B vs S_A which produced an average test error of

0.213 and C_A, C_G vs S_A which produced an average test error of 0.212 compared to the original baseline average test error of 0.216. The testing errors with the additional cover aspects are marginally better (perhaps due to noise), but can be considered insignificant when compared to the gains achieved with the $x - y$ calibration. So, while an additional cover aspect is used in the calibration experiments, the improvements in detection are due to calibration and not the presence of more data.

Although doubling the amount of cover data for training did not meaningfully improve the testing error, it is reasonable to believe that the presence of more training data could improve the baseline. An additional baseline test, using half the training data (1,250 covers and 1,250 stegos) produced a higher testing error of 0.230 and using a quarter of the training data produced a still higher testing error of 0.242. The baseline, therefore, seems to be improving with the addition of training data, but the difficulty is knowing how much training data is needed. A rule of thumb for machine learning is that you need roughly 10 times as many training examples as you have features. In the case SRM features, this would require 127,000 cover and stego images. Developing such a dataset is not practical when considering the motivating example proposed in Section 2.1 and the computational time required to extract the features in Section 3.6. Therefore, it is possible that with enough training data the baseline test could produce similar results to those achieved by calibration. However, this paper shows that we are able to achieve significant improvements in testing error by using calibration with a much smaller dataset.

4.3.2 Covers and Stegos in Feature Space

The successful classification of the cover and stego data shows that they are separable in feature space. Furthermore, calibrating with overlapping covers was very successful, implying that an overlapping cover provides a good noise characteristic reference for the origin image. We would expect the stego signal to disturb the noise characteristic of the cover image, however is the difference between the noise characteristic of a cover and its corresponding stego greater than that between a cover and an independent overlapping cover?

The noise characteristic of the images is being represented in the SRM feature space. The Mahalanobis distance, given by the following formula

$$D(\mathbf{a}, \mathbf{b}) = \sqrt{(\mathbf{a} - \mathbf{b})^T \Sigma^{-1} (\mathbf{a} - \mathbf{b})}$$

was used to compute the distance in feature space between the overlapping covers and stego (HUGO with a 0.05 bpp payload). Where **a** and **b** are feature vectors for the images and Σ is the covariance matrix of the origin cover aspect. This process was performed for each image in the dataset. In order to determine how significant the effects of embedding are on the noise characteristic in feature space, the ratio of the average cover-cover distance to the average cover-stego distance was computed, yielding a value of 3.685. This means that difference in the noise characteristic represented by the SRM features is much greater between covers than it is between a cover and stego.

In the previous experiment there was a significant amount of variation in the content of the images which could cause the covariance matrix to impact the result. Therefore, an additional test was performed to comfirm this result with a set of images that are very similar. The same camera and settings as described in Section 3.2.1 were used to capture 500 new images of a single scene. The development of the images and extraction of the features process described in Figure 3.1 was performed on the new images (only HUGO embedding with a 0.05 bpp payload was used), and then the distances between the covers and stego aspect were computed. The ratio of the average distances for this new dataset was 3.701, which is very similar the first test.

These tests provide insight into what is happening to overlapping covers and cover-stego pairs in feature space. The results from these two experiments shows that the differences in the noise characteristics of overlapping covers is approximately 3.7× greater than the differences between covers and stegos. Therefore, for the stego signal to be detectable it must be moving the noise characteristic of the cover in a consistent "direction" in feature space that is different from the direction of an overlapping cover. In other words, using HUGO to embed a payload into an image has a consistent effect in feature space that is different from the noise characteristic of an overlapping cover image. This is why the reference cover can be useful for calibration, because the reference image is removing noise introduced by the various changes in the environment without removing the stego signal. Ambitiously, we could imagine that if an embedding method is moving the noise characteristic of an image in a consistent direction, the size of the payload could impact the distance moved in the stego direction.

To illustrate this idea, feature reduction techniques are used to visualize the data in feature space. A Principal Component Analysis (PCA) was used on the origin cover aspect to reduce the dimensions and illustrate the impact of the stego signal on the noise characteristic of the cover. Additionally, Fisher's linear discriminant (FLD) was used to obtain a vector for a linear separator between the origin cover and stego (HUGO embedding with a payload of 0.1 bpp) data, defined by the formula:

$$\mathbf{w} = (\Sigma_{Cover} + \Sigma_{Stego} + \epsilon I)^{-1}(\mu_{Stego} - \mu_{Cover})$$

Where $\mathbf{w}$ is the FLD vector, Σs are covariance matrices, ϵI is a small multiple of the identity matrix to ensure that the matrix is invertible, and μs are mean vectors. The FLD provides a vector that best separates the origin covers and HUGO stegos with a 0.1 bpp payload in feature space. This method should perfectly separate these two sets of data in order to show the effects of HUGO embedding.

Using the vectors obtained from the PCA and FLD, we can project cover and stego data down to two dimensions. Figure 4.1 shows the projection of the data on these two vectors. The figure shows that the direction HUGO embedding moves a cover and this direction is generally maintained even when a smaller payload of 0.05 bpp is used.[1] This trend means that it may be possible to train the classifier with

[1]While this figure presents an important visual aid, it is purely for illustrative purposes. It would perform very poorly for classification, because it is "over-trained" on the data. This is demonstrated by the distribution of the 100% overlapping cover which would often be classified incorrectly if this were used as a classifier.

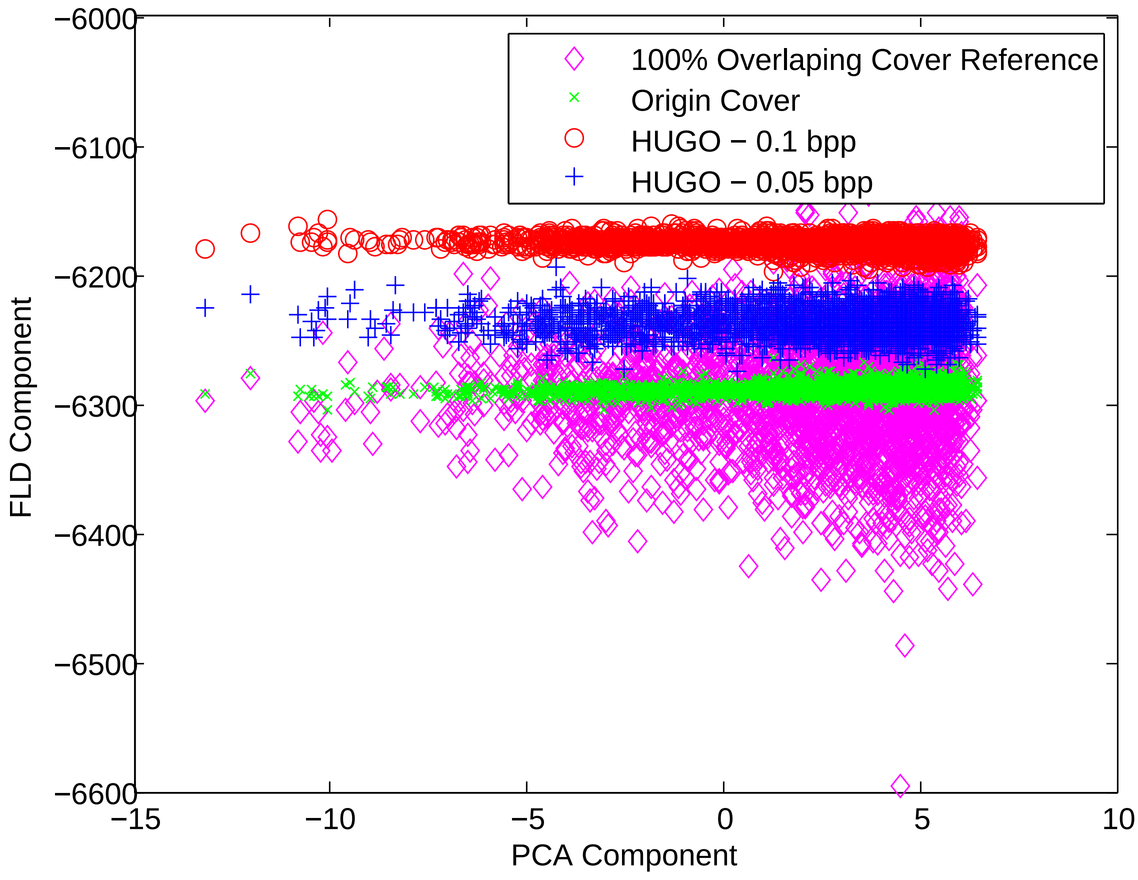

Figure 4.1: Illustration of cover and stego (HUGO) data in feature space. The FLD component uses the HUGO with a 0.1 bpp payload data and the origin cover data to separate the covers and stegos, while the PCA Component is generated from the origin cover. When plotting the 0.05 bpp stego images on the graph, it is in the same direction as the 0.1 bpp images, suggesting that there is a consistent direction with a varied magnitude in feature space.

one payload and still be able to detect another.

In order to test this idea, the ensemble classifier was trained with the HUGO stego with a 0.05 bpp payload and then tested on the HUGO stego with a 0.1 payload to determine if it is feasible. [2] Symbolically this can be described as: training on $C_B - C_A$ *vs.* $C_B - S_{A(0.05)}$ and testing on $C_B - C_A$ *vs.* $C_B - S_{A(0.1)}$. This experiment produced an average testing error of 0.011 which is equivalent to the best performing case in the calibration tests performed purely on HUGO with a payload of 0.1 bpp presented in Table 4.5. Thus, it is feasible to train the classifier with a smaller payload and still detect stego objects with a larger payload.

[2]The $x - y$ calibration is used for best performance.

4.4 Moving Toward Real World Application

Let us now return to the assumptions of the hypothesis given in Section 2.1. With the original hypothesis confirmed in laboratory conditions, we can begin to relax some of the assumptions to further the research presented in this paper. The tests that follow provide interesting results for a scenarios which may inspire directions for further research.

The first assumption that we revisit is assumption (ii). Assumption (ii) states that the steganalyst knows the payload size and embedding method are known. With the understanding that embedding methods, specifically HUGO, moves in a general direction in feature space and that larger payloads move further in the same direction. Perhaps the assumption could be revised to: the embedding method and a bound for the payload size were known (e.g. the HUGO embedding method is used and the payload is greater than 0.05 bpp).

Additionally, the second assumption that could be revisited is assumption (v). As previously stated in Section 2.1 it must be known that the first image of the pair of overlapping images is the a cover while the other is to be determined. It was also noted that this stipulation could be overcome by trying the different paring orders, which is now attempted. If we also include the modification to the first condition, allowing for two separate payloads with the HUGO embedding (a more thorough analysis would include a range of payloads), there are 9 possible cover-stego pairs that could be presented. For this experiment we again perform the training and testing of the classifier separately: training on $C_B - C_A$ $vs.$ $C_B - S_{A(0.05)}$ and testing on $C_B - C_A$ $vs.$ $x - y$ parings. When the order of the parings are reversed, swapping the labels would be sufficient. [3] Table 4.8 contains the average testing error for each of the possible parings.

In Table 4.8, we see that although the classifier is trained on the HUGO embedding method with a 0.05 bpp payload, it is able to correctly classify the 0.1 bpp payload when the 0.05 bpp payload is used for calibration. Also, when the pairings are in the opposite order, reversing the labels produces accurate classification. As we would expect the diagonal elements in this table show that when calibration is performed with the same type of image, classification is unsuccessful. This is intuitive when considering the cover-cover calibration, because it should be difficult to classify covers into different groups. It is also intuitive when considering using a stego to calibrate another stego with the same payload. The direction that the stego signal moves the cover image in feature space after embedding would be removed or greatly reduced by calibrating with a stego image with the same payload. However, when an image with a smaller payload is used for calibration, the stego signal after calibration may be reduced, but it is still detectable because the magnitude of the stego signal in feature space was greater. This approach provides a means of detecting whether or not two overlapping images have different payloads (above a certain payload threshold) for a given embedding method. This method could be very useful in practice, like in the motivating example where the steganalyst is trying to determine whether or not

[3]Multiplying each calibrated feature by -1 would invert the effect of the $x - y$ calibration

Type	Cover	HUGO (0.05)	HUGO (0.1)
Cover	0.5	0.082	0.011
HUGO (0.05)	0.052*	0.5	0.027
HUGO (0.1)	0.012*	0.036*	0.5

*Calibration labels were swapped

Table 4.8: Average testing error for the different parings of overlapping images with HUGO embedding with payloads of 0.1 and 0.05 bits per pixel. The $x - y$ calibration method is used in all experiments. A 0.5 average testing error shows performance is comparable to a random guess.

steganography is being used in overlapping images, because determining a difference in the payloads of two images indicates that steganography is being used.

While specific conclusions have be drawn from these results, general conclusions and applications would require further experimentation beyond the scope of this paper. However, these findings do indicate that practical application is not unrealistic, but plausible through further development.

Chapter 5

Conclusions

5.1 Summary

Calibration is a commonly used method to improve steganalysis by providing an image reference to characterize the noise characteristic of a cover image. This paper presents a new approach to calibration, using independent overlapping images created from the same cover source and settings. The experiments performed on uncompressed images using LSBM and HUGO embedding methods in laboratory conditions show that this is true. When calibration was used with the LSBM embedding method, the average testing error was 9.7× smaller than the baseline with a payload of 0.01 bpp and 9.3× smaller than the baseline with a payload of 0.005 bpp. Similarly, the calibration with HUGO embedding method achieved an average testing error 5.9× smaller than the baseline with a payload of 0.1 bpp and 2.6× smaller than the baseline value with a payload of 0.05 bpp. We can summarize the initial results obtained in this paper as follows:

- Independent overlapping images provide a good reference image for calibration, improving the testing error of the detector.
  - Improvements to the baseline are more significant when there is more overlap in the images.
  - No significant improvement to the baseline is obtained when images are cropped to contain a 100% overlap.
  - Improvements to the baseline are more significant when the baseline accuracy starts out higher.
- Improvements in testing error are not due to the presence of more cover data.
- Calibration with an independent, overlapping stego image with a smaller payload also improves the testing error.

These results can be seen as improvements to state of the art steganalysis methods (assuming certain conditions about the data).

After confirming the hypothesis, additional experiments were performed to provide insight into the results obtained, revealing that:

- The distance between the noise characteristic of independent, overlapping covers is more approximately 3.7× larger than the distance between a cover and its corresponding stego.
- Embedding a payload with the HUGO algorithm makes consistent changes to an image in feature space.
- A larger payload is a larger distance from the cover in feature space.
- It is possible to determine whether or not independent, overlapping images contain different payload sizes (above a threshold) for a particular embedding method (HUGO).

These conclusions allow the hypothetical assumptions needed for the laboratory conditions to be relaxed, providing direction for future research.

5.2 Evaluation

Calibration has commonly been used to improve steganalysis; however, to the author's knowledge, overlapping images have never before been used for calibration. The results obtained in this paper show that using overlapping images can provide significant improvements to the testing error, greatly improving upon state of the art steganalysis in some situations. This is a significant discovery when considering how likely the presence of overlapping images would be in practice. However, with the early state of this research the impact that it could have on the future of steganalysis is still unknown. The results obtained are dependent on the correct type of training data, i.e., same cover source and cover settings. Therefore, before this research could be used in practice, a number of hurdles must be overcome.

5.3 Future Work

The presence of steganography is growing rapidly [18] and steganalysis will likely grow with it. Much of the research in this paper deals with the discover that overlapping images can be successfully used for calibration. The extent to which this is possible is still relatively unknown. This paper presents successful attempts under laboratory conditions for uncompressed images, but it is unknown whether or not this is possible in compressed images, which are far more prevalent today. Each of the tests previously performed could also be done on the compressed version to determine if similar results are obtained.

As described in Section 2.1 and 4.4 there are a number of steps that could be taken to actually to provide a real world solution to the motivating example discussed. In Section 4.4 we briefly explored some potential possibilities to see if they were plausible. These assumptions would need to be readdressed before a real world application is possible.

A far-reaching future work would be in the field of mass detection of steganography using images from social media. We can imagine using geo-tagged images with the EXIF date and time properties to find images from various users that overlap. These images could be used for calibration to increase the accuracy of mass steganalysis. Work in batch steganography [10] is a very recent field, so there are a number of significant advancements that would need to be made before this would be feasible.

Bibliography

[1] Jessica Fridrich. *Steganography in Digital Media: Principles, Algorithms, and Applications.* Cambridge University Press, New York, NY, USA, 1st edition, 2009.

[2] Jessica Fridrich, Miroslav Goljan, and Dorin Hogea. Steganalysis of JPEG images: Breaking the F5 algorithm. In *Information Hiding*, pages 310–323. Springer, 2003.

[3] Jessica J. Fridrich and Jan Kodovsk. Steganalysis of lsb replacement using parity-aware features. In Matthias Kirchner and Dipak Ghosal, editors, *Information Hiding*, volume 7692 of *Lecture Notes in Computer Science*, pages 31–45. Springer, 2012.

[4] Jessica J. Fridrich and Jan Kodovský. Rich Models for Steganalysis of Digital Images. *IEEE Transactions on Information Forensics and Security*, pages 868–882, 2012.

[5] Vojtech Holub. HUGO, 2012. Available at `http://dde.binghamton.edu/download/stego_algorithms/`. Last accessed September 21, 2014.

[6] Jan Kodovský and Jessica Fridrich. SRMQ1, 2011. Available at `http://dde.binghamton.edu/download/feature_extractors/`. Last accessed September 21, 2014.

[7] Jan Kodovský and Jessica Fridrich. Ensemble Classification Version 2.0, 2013. Available at `http://dde.binghamton.edu/download/ensemble/`. Last accessed September 21, 2014.

[8] David Kahn. *The Codebreakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet.* Scribner, 1996.

[9] Andrew D. Ker. Improved Detection of LSB Steganography in Grayscale Images. In *In Proc. 6th Information Hiding Workshop, volume 3200 of Springer LNCS*, pages 97–115, 2004.

[10] Andrew D. Ker and Tomás Pevný. Batch steganography in the real world. In *MM&Sec'12*, pages 1–10, 2012.

[11] Andrew D. Ker, Tomáš Pevný, Jan Kodovský, and Jessica Fridrich. The square root law of steganographic capacity. In *Proceedings of the 10th ACM Workshop on Multimedia and Security*, MM&Sec'08, pages 107–116, New York, NY, USA, 2008. ACM.

[12] Jan Kodovský and Jessica Fridrich. Calibration revisited. In *Proceedings of the 11th ACM Multimedia & Security Workshop*, pages 63–74, 2009.

[13] Jan Kodovský, Jessica J. Fridrich, and Vojtech Holub. Ensemble Classifiers for Steganalysis of Digital Media. *IEEE Transactions on Information Forensics and Security*, 7(2):432–444, 2012.

[14] Tomáš Filler Patrick Bas and Tomáš Pevnỳ. Break Our Steganographic System: The Ins and Outs of Organizing BOSS. In *Information Hiding*, pages 59–70. Springer, 2011.

[15] Fabien A. P. Petitcolas, Ross J. Anderson, and Markus G. Kuhn. Information Hiding – A Survey, 1999.

[16] Peter E. Hart Richard O. Duda and David G. Stork. *Pattern Classification*. Wiley, 2001.

[17] Gustavus J. Simmons. The Prisoners' Problem and the Subliminal Channel. In *CRYPTO*, pages 51–67, 1983.

[18] Luca Caviglione Steffen Wendzel, Wojciech Mazurczyk and Michael Meier. Hidden and uncontrolled - on the emergence of network steganographic threats.

[19] Patrick Bas Tomás Pevný and Jessica J. Fridrich. Steganalysis by Subtractive Pixel Adjacency Matrix. *IEEE Transactions on Information Forensics and Security*, pages 215–224, 2010.

[20] Tomás Filler Tomás Pevný and Patrick Bas. Using High-Dimensional Image Models to Perform Highly Undetectable Steganography. In *Information Hiding*, pages 161–177, 2010.