Searching For Hidden Messages: Automatic Detection of Steganography

George Berg, Ian Davidson, Ming-Yuan Duan and Goutam Paul

Computer Science Department
University at Albany, SUNY
1400 Washington Ave
Albany, NY 12222
{berg. davidson, myduan, goutam}@cs.albany.edu

Abstract

Steganography is the field of hiding messages in apparently innocuous media (*e.g.* images), and steganalysis is the field of detecting these covert messages. Almost all steganalysis consists of hand-crafted tests or human visual inspection to detect whether a file contains a message hidden by a specific steganography algorithm. These approaches are very fragile – trivial changes in a steganography algorithm will often render a steganalysis approach useless, and human inspection does not scale. We propose a machine learning (ML) approach to steganalysis. First, a media file is represented as a *canvas* – the available space within the file to hide a message. Those features that can distinguish clean from stego-bearing files are then selected. We use ML algorithms to distinguish clean and stego-bearing files. The results reported here show that ML algorithms work in both content- and compression-based image formats, outperforming at least one current hand crafted steganalysis technique in the latter. Our current work can detect previously seen (trained on) steganography techniques, and we discuss extensions that we believe will be able to detect steganography using more sophisticated algorithms, as well as the use of previously unseen steganography algorithms.

Introduction

Steganography literally means "covered message" and involves transmitting secret messages through seemingly innocuous files. The goal is that not only does the message remain hidden, but also that a hidden message was even sent goes undetected (Johnson and Jajodia, 1998). There are many tools available (Steganograpy Software Web Page) that can hide messages in images, audio files and video, and steganography is now in common use (Johnson, et al., 2001). Whereas cryptography has been the preferred tool for sending secret messages, relying on complex ciphers to prevent detection, the huge bandwidth of the Internet now offers an alternative or complementary approach. Steganography supports hiding messages amongst the huge volume of Internet traffic, in media files where the addition of a hidden message is difficult to detect with the human eye even if the file is viewed.

The process of detecting steganographic messages is known as *steganalysis* and a particular steganalysis technique is called an *attack*. The current state of the art involves manually identifying a particular signature associated with a steganographic technique or non-infected file type and devising a statistical test to identify this signature. This handcrafted approach, though having some success (Westfeld and Pfitzmann, 1999) suffers from a high false positive rate and is vulnerable to steganographic approaches that hide messages in such a way as to preserve an expected property (Provos, 2001). An "arms race" situation has developed where a steganography technique's creator and the designer of a steganalysis attack iteratively and incrementally improve their approaches. For example, the F3 steganography algorithm has evolved into the F4 and most recently F5 algorithms (Westfeld, 1999) with further small variations to overcome its vulnerability to specific attacks.

We propose using pattern recognition to learn attacks on steganography techniques by automatically identifying what differentiates clean files and files containing hidden messages (*stego*-files). This is a difficult problem for a number of reasons. It involves learning an evolving and changing phenomenon that directly violates the fundamental assumption in learning algorithms – that the test and training set are drawn from the same distribution -– the so called stationary distribution assumption (Kearns and Vazirani, 1994). In addition, hidden messages are typically embedded in media files such as images, audio, and video, whose size and varied formats make applying machine learning and data mining difficult. Representing the many types of multi-media and the different formats in a common form to present to a learning algorithm is a challenging task. In particular, some formats are based on lossy compression, such as JPEG/MPEG, and others use efficient indexing for lossless representation, such as GIF. This difference makes a common representation difficult.

Our approach, which is applicable to many media and file formats, begins with the notion of a *canvas*, which is the data representations that a steganographic algorithm has available to write a hidden message. The notion of the

canvas is a natural representation and has the benefit of being applicable to both indexing (GIF, BMP) and compression based representation schemes (JPEG, MPEG, QuickTime). We measure general properties of the canvas such as conditional entropies and transition probabilities. Standard machine learning algorithms are then applied to detect steganography in both GIF and JPEG formats. Our results show that ML techniques can detect messages hidden in both formats, and for JPEG images our approach outperforms one of the state of the art steganalysis techniques (no GIF format method was available for a corresponding comparison). Applying automated pattern recognition tools to a difficult domain such as steganalysis offers many challenges. We illustrate this by the failure of our current approach on a particular steganography algorithm for each of GIF and JPEG formats. We indicate how our approach may be generalized to detect the use of these algorithms, as well as how to detect the use of previously unseen steganographic algorithms.

Background and Relationship to Previous Work

Images are the most widely used media to transmit hidden messages. The raster data that defines the image can be stored either by using efficient indexing with simple compression (GIF) or by performing transformations on the raster data that make it more amenable to compression (JPEG).

The GIF image format uses a table-based representation to save space while still storing the original raster data. Since, in many types of images, the same colors appear in many pixels, the RGB colors of all of the pixels are stored in a single color table, the *palette*. Each pixel in the image representation is simply an index into the palette.

The JPEG representation (Wallace, 1991) allows compression of the raster data to varying degrees. The compression is *lossy* – that is, the original image cannot be exactly recreated since the compression algorithm loses some of the information. Raster data is converted to JPEG by first extracting 8 x 8 blocks of pixels. A Discrete Cosine Transformation (DCT), related to a fast Fourier transformation, is then calculated for each pixel in the block. Each block is then compressed using Huffman encoding. Varying the severity of the DCT transformation can provide different levels of compression.

The different representation schemes for GIF and JPEG formats lend themselves to different strategies for hiding messages in images. While the message may be text, image, *etc.*, in digital steganography it is ultimately represented as bits. The hidden message may be compressed or even encrypted before it is hidden, to reduce the amount of information or hide its content.

The underlying basis of steganographic techniques is hiding the bits of the message among the bits of the image, so that 1) the message is undetectable, but 2) that it is recoverable by the intended recipients. One common steganography approach is to embed the hidden message by altering least significant bits (LSBs) of the DCT coefficients. Altering the LSB results in a change that is not visually noticeable in JPEG files (see Figure 1). The LSB-based Jsteg steganography algorithm (Upham) embeds the 0 and 1 values of the message in the LSB of the coefficients after the quantization step. However, our own preliminary work, described here, and that of others, *e.g.* (Lyu and Farid, 2002, Westfeld and Pfitzmann, 1999) has shown such changes are discernable.

The LSB techniques are not directly applicable to GIF images, as bit changes would point to a different entry in the color table, resulting in visually noticeable changes in the image. One GIF-specific steganography algorithm is GIFShuffle (Kwan), which depends on the fact that the order of the colors in the palette is not specified. In GIFShuffle, the order of the entries in the palette is manipulated to convey information.

Because of the relative ease with which Jsteg steganography can be detected, improved methods of embedding information in images, such as F5 (Westfeld, 1999) and Spread Spectrum Image Steganography (Marvel, et al., 1999) have been developed. This has led, in turn, to more advanced methods of steganalysis (Fridrich, et al., 2002, Lyu and Farid, 2002). This is just the type of arms race, mentioned earlier, that is akin to the ongoing battle between viruses and immune systems. One assumes that this escalation will continue into the foreseeable future.

Figure 1: The cover picture on the top and the stego picture on the bottom that contain a hidden text message of 2 kilobytes.

To our knowledge, the work of Farid (Lyu and Farid, 2002) is the only steganalysis besides our own that uses machine learning. However, their emphasis is different than our own. They build special purpose second order statistical models of the images using a type of wavelet decomposition. These are used as the input to a machine learning technique, in this case support vector machines that learn to distinguish stego-bearing images. Their technique, while powerful, is focused on the complex second order model built of the image (Lyu and Farid, 2002). In contrast, our work begins an effort to use machine learning to build a general framework for steganalysis in many formats, for multiple media and a variety of content.

Experiments and Results

As mentioned earlier, manually crafted statistical attacks on steganography techniques must be continually updated, and, since they are created for specific steganography techniques, do not generalize beyond those techniques. Automated learning and data mining techniques can potentially create models that successfully attack a variety of steganography techniques, including previously unseen variations of existing techniques. The first step towards this goal is to determine if data mining and machine learning techniques can learn to identify messages hidden using a specific steganography technique. To establish this, we tried three common data mining and learning techniques (Duda, et al., 2001): decision trees, error back-propagation artificial neural networks and the naïve Bayes classifier, to identify messages hidden in compression- (JPEG) and content-based (GIF) images.

We tested the ability to detect the JPEG steganography technique Jsteg Version 4 (Upham) by creating a database of 150 natural images, 50 each of flowers, mountains and trees. Each image is represented by the unconditional entropy, positional conditional entropy values, and transition probabilities of the DCT coefficient's LSB. There are 51 features – the mean entropy for the entire image (1 feature), the mean and standard deviation of the entropy across each block in the image (2), the mean and standard deviation across each block of the transition probabilities $0\rightarrow0$, $0\rightarrow1$, $1\rightarrow0$, and $1\rightarrow1$ (8), the average probability across the entire image of each of the transitions $0\rightarrow0$, $0\rightarrow1$, $1\rightarrow0$, and $1\rightarrow1$ (4), and the conditional entropy for each non-boundary position in the 8 x 8 DCT coefficient grid, calculated for the entire image (36). Table 1 shows our results and the results of applying the StegDetect handcrafted statistical attack (Provos and Honeyman, 2002) when the number of stego and clean images is equal in the dataset. The results for the learning algorithms are five fold cross-validated results while the StegDetect attack is the result for the entire data set. For each of the databases, at least one of the machine learning algorithms outperforms StegDetect. In particular, the error-backpropagation artificial neural network algorithm (Rumelhart, et al., 1986) always outperforms StegDetect in these tests.

	Decision tree	**Naïve Bayes**	**Neural Net**	**Steg-Detect**
Flower	78%	69%	81%	68%
Mountain	54%	35%	76%	55%
Tree	57%	33%	51%	37%

Table 1: Comparing the accuracy of hand-crafted attacks and automatically learned attacks on the Jsteg-Jpeg steganographic technique.

We also learned to detect steganography used in content-based (GIF) image formats. The steganographic technique evaluated, GIFShuffle (Kwan), shuffles the GIF palette to store a message up to $\log_2(256!)$ bits in length. We used unconditional and conditional entropies of the color indices to represent each GIF. We did not use transition probabilities, as it would have resulted in an excessively large number of features. The conditional entropies used are slightly different than in JPEG as there is no notion of a cell in content-based formats. The conditional probability distribution used in the conditional entropy calculations is the probability that a pixel, k, has the color index j, given the number of neighborhood pixels having the color index j.

Our results at detecting this approach are shown in Table 2. As with the other results, these are the product of a five fold cross-validation.

	Accuracy
Decision Tree	69.1%
Naïve Bayes	58.4%
Neural Network	85.6%

Table 2: Comparing the accuracy of automatically learned attacks on the GIFShuffle steganographic technique.

All of the algorithms perform better than random choice, and the artificial neural network has better than 85% accuracy at detecting messages hidden using GIFShuffle.

The success of our initial results establishes the key points about this approach to steganalysis:

1. The feasibility of using a steganographic canvas metaphor,
2. The viability of using standard properties of the canvas, and
3. The feasibility of using machine learning and data mining algorithms on these properties to create a model that can differentiate between clean and stego files.

In addition, we found that the three learning techniques tried (decision trees, naïve Bayes and artificial neural network classifiers) performed significantly better than random guessing in a variety of situations (results not shown). These situations include when the target event occurred in different proportions in the training and test sets. This result is significant, as it is often not known *a priori* how often the target event occurs.

From Prototype to the Full System

The results above show the success of the canvas representation and the use of machine learning to detect the use of steganography. However, the present system is a prototype. The full system will have two additional features, 1) it will use unsupervised learning to select those features of the canvas that are useful in discriminating clean from stego-bearing files, and 2) it will employ several different machine learning algorithms, as well as a second-level algorithm to combine their results into an overall, refined prediction. We motivate these extensions by showing the shortcomings of the prototype and how the full system is designed to overcome them. The goal is a system that successfully detects the use of steganography, and can detect the use of new steganography methods by their deviation from the norms of a canvas. Even in situations where this is not the case, the system is flexible enough that it can easily be re-trained to detect and attack new steganographic algorithms.

20	5	-3	1	3	-2	1	0
-3	-2	1	2	1	0	0	0
-1	-1	1	1	1	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

19	4	-3	1	3	-2	1	0
-3	-1	1	2	1	0	0	0
-2	-1	1	1	2	0	0	0
0	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0
0	0	0	0	1	0	1	0
0	1	0	0	0	1	0	0
0	0	0	0	0	0	1	1

Figure 2: Original (top) and corrupted (bottom) DCT cells. Note that the entropy of the LSB is greatest in the top left hand corner and decreases towards the lower right hand corner as shown by the arrow.

Compression Based Formats (JPEG)

While the prototype system detects Jsteg steganography at high accuracy levels, it detects steganography using the F5 algorithm (Westfeld, 1999) no better than random chance (detailed results not shown). We hypothesize that, similar to hand crafted steganalysis, our prototype is fragile because it only uses a few, specific features, and the changes in the F5 algorithm render the features successfully used in Jsteg useless. Our overall system overcomes this fragility in two ways. First, the canvas representation can use any number of features, and can be expanded to use additional ones in response to new algorithms or directions in detection. For the F5 example, the canvas space can be expanded to represent the 8 x 8 DCT cells that form the basis of a JPEG representation. The cells have distinct signatures created by the DCT, a feature that promotes their subsequent compression. It is our belief that detecting messages in JPEG images using their cells, rather than the simpler representations used in our prototype, is promising. The 8 x 8 pattern of the cells can then be used as input to the prototype's ML algorithms, as well as approaches shown to useful at identifying signatures such as Hidden Markov Models (Rabiner, 1989).

We believe that representing an image as a series of non-sequential propositions (attribute-value pairs) does not fully exploit our *a priori* knowledge of the signature. Consider the picture of the original DCT and altered DCT cells Figure 2. We expect for clean files that the LSB conditional entropy for cells along the diagonal to be very large in the upper-left hand corner and progressively become less towards the lower right hand corner. The speed of change for these entropies will differ from image to image, but the overall signature should occur. We propose to represent each DCT cell as a series of eight sequential entropy measurements, one for each position along the diagonal. We then intend to use learning techniques that can differentiate between signatures. Hidden Markov Models have been very successful at learning process signatures such as those occurring in speech (Rabiner, 1989) and biology (Krogh, et al., 1994). We can then identify the number of cells in an image that potentially contain parts of a hidden message.

In addition, by using unsupervised learning to cluster clean and stego-bearing files based on various representations in the canvas, and by using a second stage to determine which of the machine learning algorithms is giving the best results, our full system will have the flexibility to automatically determine the features and learning methods for successful steganalysis.

	% Accuracy
Decision Tree	34.57
Naïve Bayes	40.96
Neural Network	34.57

Table 3: Comparing the accuracy of automatically learned attacks on the HideAndSeek steganographic technique. Classifier results are for 5 fold cross validation.

Content Based Formats (GIF)

An analogous situation exists for GIF format images. While the prototype can detect steganography done using GIFShuffle with high accuracy, it performs very poorly on the HideAndSeek (Maroney) algorithm. HideAndSeek changes the index value of specific pixels to hide the ASCII values of the message. Our results, shown in Table 3, indicate that attacks using the supervised learning algorithms in our prototype could not identify this technique.

Most steganographic techniques for content-based file formats randomly place the hidden message in the image, which is obvious to the human eye (Figure 3) but is surprisingly difficult for a machine to detect. This is to be expected as our properties are for the entire canvas and it is impossible that a standard signature exists for all images for the entire image. However, by looking at the image we can see that an implicit signature exists for most clean images, namely that pixels in a region all have the same color. We believe that by dividing an image into regions and then measuring the number of outlier pixels in each region we can differentiate stego images from clean images. We intend to try to divide an image into regions using techniques such as spatial clustering and self organized maps (Kohonen, 1982). We will try to determine the number of outliers using traditional data mining and machine learning, but this may be difficult due to the spatial nature of the data. We intend to apply Gibbs restoration techniques (Geman and Geman, 1984) to determine the number of restorations to perform for each section. For unaltered images we expect that background regions will have few if any restorations. We can colloquially consider a restoration as being where the eye notices an anomaly.

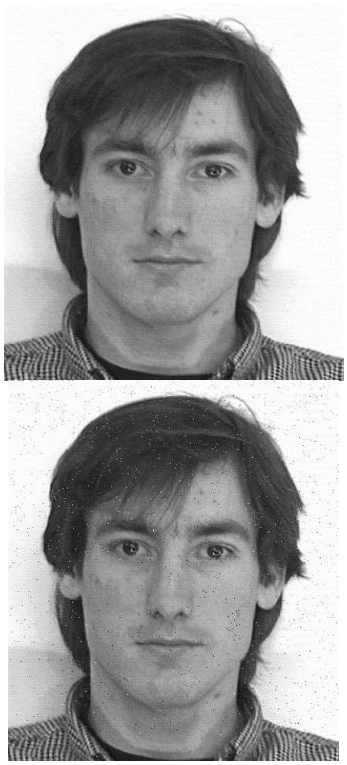

Figure 3. The upper image is clean, and the lower one contains a steganographically hidden message. Note the sporadic black pixels of the stego image, resulting from hiding a message with the HideAndSeek tool.

Conclusion

We have shown the feasibility of using a machine learning and data mining (ML/DM) approach to automatically build steganography attacks. For both content-based (GIF) and compression-based (JPEG) image formats, ML/DM techniques are very successfully able to distinguish stego-files from clean ones. This work is based on a canvas representation of the media format that makes explicit all of the features that can be used for steganographic embedding. We have shown how this can be combined with a set of features selected from the canvas representation. In the current work, this includes value occurrence probabilities, and both unconditional and conditional entropies. These features were successfully used, for both GIF and JEPG formats, and by several different learning algorithms, to find hidden message bearing files. For JPEG format images, this approach outperforms one of the current state of the art steganalysis techniques.

Our current system is certainly no panacea. We have shown examples of steganography algorithms for both GIF and JPEG formats that it cannot detect. While the current work uses straightforward features of the canvases and well-known learning methods, we indicate how it can

be extended to more powerful representations and ML/DM methods. With these we anticipate being able to extend this work to use unsupervised anomaly detection approaches to steganalysis. These approaches should be able to detect the canvas features of clear media files, and hence should be able to distinguish those from stego-bearing files, regardless of the steganography method used. This should be able to detect steganography hidden using more advanced algorithms.

In addition, by establishing general signatures of clean files, deviations from these signatures are a possible sign of steganographic embedding. While specific steganography algorithms would have specific deviations, any deviation raises the possibility of a hidden message. This holds the very exciting potential to transcend the current fragile nature of modern steganalysis – it may be possible to identify that a file has a hidden message, even if it is hidden using a new, previously unseen steganography algorithm.

References

Duda, R.O., Hart, P.E. and Stork, D.G. 2001 *Pattern Classification*. John Wiley and Sons, Inc., New York City.

Fridrich, J., Goljan, M. and Hogea, D. 2002. Steganalysis of JPEG Images: Breaking the F5 Algorithm. In *Fifth International Workshop on Information Hiding*, (Noordwijkerhout, Netherlands), Springer Verlag, 310-323.

Geman, S. and Geman, D. 1984. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 6* (6). 721-741.

Johnson, N.F., Duric, Z. and Jajodia, S. 2001 *Information Hiding: Steganography and Watermarking - Attacks and Countermeasures*. Kluwer Academic, Dordrecht, The Netherlands.

Johnson, N.F. and Jajodia, S. 1998. Steganalysis: The Investigation of Hidden Information. In *IEEE Conference on Information technology*.

Kearns, M.J. and Vazirani, U.V. 1994 *An Introduction to Computational Learning Theory*. MIT Press, Cambridge, MA.

Kohonen, T. 1982. Self-organized Formation of Topologically Correct Feature Maps. *Biological Cybernetics, 43*. 59-69.

Krogh, A., Brown, M., Mian, I.S., Sjolander, K. and Haussler, D. 1994. Hidden Markov Models in Computational Biology. Applications to Protein Modeling. *J Mol Biol, 235* (5). 1501-1531.

Kwan, M., The GIF-Shuffle Homepage. http://www.darkside.com.au/gifshuffle/.

Lyu, S. and Farid, H. 2002. Detecting Hidden Messages Using Higher-Order Statistics and Support Vector Machines. In *Fifth International Workshop on Information Hiding*, (Noordwijkerhout, Netherlands), Springer Verlag, 340-354.

Maroney, C., Hide and Seek. ftp://ftp.csua.berkeley.edu/pub/cypherpunks/steganography/hdsk41b.zip.

Marvel, L.M., Boncelet Jr., C.G. and Retter, C.T. 1999. Spread Spectrum Image Steganography. *IEEE Transactions on Image Processing, 8*. 1075-1083.

Provos, N. 2001. Defending against Statistical Steganalysis. In *Tenth USENIX Security Symposium*.

Provos, N. and Honeyman, P. 2002. Detecting Steganographic Content on the Internet. In *Internet Society Symposium on Network and Distributed System Security*, (San Diego).

Rabiner, L.R. 1989. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proc. IEEE, 77* (2). 257-286.

Rumelhart, D.E., Hinton, G. and Williams, R.J. 1986. Learning Internal Representations by Error Propagation. In Rumelhart, D.E. and McClelland, J.L. eds. *Parallel Distributed Processing*, MIT Press, Cambridge, MA, 318-362.

Steganograpy Software Web Page. http://members.tripod.com/steganography/stego/software.html.

Upham, D., Jpeg-Jsteg: Modification of the Independent JPEG Group's JPEG Software (Release 4) for 1-bit Steganography in JFIF Output Files. ftp://ftp.funet.fi/pub/crypt/steganography.

Wallace, G.W. 1991. The JPEG Still Picture Compression Standard. *Communications of the ACM, 34* (4). 30-44.

Westfeld, A. 1999. F5 - A Steganographic Algorithm. In *Third International Workshop on Information Hiding*, (Dresden, Germany), Springer Verlag, 289-302.

Westfeld, A. and Pfitzmann, A. 1999. Attacks on Steganographic Systems. In *Third International Workshop on Information Hiding*, (Dresden, Germany), Springer Verlag.