Annual ADFSL Conference on Digital Forensics, Security and Law

2010
Proceedings

May 20th, 1:00 PM

CANVASS - A Steganalysis Forensic Tool for JPEG Images

Jennifer L. Davidson
Department of Mathematics, Iowa State University, Ames, IA, davidson@iastate.edu

Jaikishan Jalan
Department of Computer Science, Iowa State University, Ames, IA, jai.ism@gmail.com

CANVASS - A Steganalysis Forensic Tool for JPEG Images

Jennifer L. Davidson
Department of Mathematics
Iowa State University, Ames, IA 50011
Phone: (515) 294-0302
Fax: (515) 294-5454
davidson@iastate.edu

Jaikishan Jalan
Department of Computer Science
Iowa State University, Ames, IA 50011
jai.ism@gmail.com

ABSTRACT

Steganography is a way to communicate a message such that no one except the sender and recipient suspects the existence of the message. This type of covert communication lends itself to a variety of different purposes such as spy-to-spy communication, exchange of pornographic material hidden in innocuous image files, and other illicit acts. Computer forensic personnel have an interest in testing for possible steganographic files, but often do not have access to the technical and financial resources required to perform steganalysis in an effective manner. This paper describes the results of a funded effort by a grant from the National Institutes of Justice to develop a user friendly and practical software program that has been designed to meet the steganalysis needs of the Iowa Division of Criminal Investigation in Ankeny, Iowa. The software performs steganalysis on JPEG image files in an efficient and effective way. JPEG images are popular and used by a great many people, and thus are naturally exploited for steganography. The commercial software that is available for detection of hidden messages is often expensive and does not fit the need of smaller police forensic labs. Our software checks for the presence of hidden payloads for five different JPEG-embedding steganography algorithms with the potential of identifying stego images generated by other (possibly unknown) embedding algorithm.

Keywords: steganography, steganalysis, JPEG images, GUI software

INTRODUCTION

Steganography is the practice of communicating a hidden message in such a way that no one, apart from the sender and intended recipient, suspects the existence of the message. The goal of steganography is to embed a payload into a cover object to obtain a stego object in such a way that the presence of hidden information cannot be detected by either perceptual or statistical analysis of the stego object. The counterpart of steganography is steganalysis. The main goal of steganalysis is to identify whether a given object has a payload embedded in it. Other information about the payload is often sought, including identification of the steganography algorithm, estimation of payload length, recovery of the payload, or obliteration of the payload.

With the advent of digital media and the Internet, multimedia objects such as still images and videos have become popular and are shared easily. Image and video data make a good choice for hiding payload. These objects are readily available and their broad presence on the Internet makes it difficult to check each one for hidden payload and thus difficult to detect the use of steganography. A single image can hold a reasonable amount of information, and a video file can hold more. In addition, there is a plethora of freeware available for hiding secret information, as can be seen by visiting the site stegoarchive.com (Stegoarchive, 1997). MSU StegoVideo is a video steganographic tool that is freely available online (MSU, 2004). In this paper, we restrict steganalysis of image data to Joint Photographic Experts Group (JPEG) format because of its wide use in consumer cameras and on the Internet. It also has the advantage of low bandwidth for storage and transmission, unlike raw or other uncompressed formats.

There is a growing concern within the community that steganography is being used for illicit purposes. The USA Today (Kelly, 2001), the New York Times (Kolata, 2001), and the United States Institute of Peace (Weimann, 2004) have reported that terrorists may be using steganography and cryptography on the web as a means of covert communication. A recent report from National Institute of Justice encourages investigators to look for steganographic information while dealing with child abuse or exploitation and terrorism cases (NIJ, 2008). A more recent online report from the New York Times (Kerbaj, 2008) reported that a raid on computers from a terrorist group produced child pornographic images that contained secret messages hidden in them. These reports have lead local police departments to be concerned about the use of steganography for crimes committed within their local jurisdictions. For example, the Iowa Crimes Against Children Taskforce (ICAC) in Ankeny, Iowa, has expressed such concerns. This paper presents results of the authors collaborating with ICAC to address their steganography detection needs. We found that while steganalysis algorithms abound in the academic literature, there are few software programs that address the needs of local police departments who perform computer forensic functions for steganalysis. Here we describe Canvass, a software package that has been developed to make our research accessible to the Iowa Department of Criminal Investigation forensic lab.

There are several major stego-detection tools in existence today. StegoSuite is a commercially available software program developed by WetStone Technologies for the U.S. Air Force (Wetstone, 2010). StegoSuite performs a variety of steganography detection actions including previously installed software, image filters, and other features. The cost for a single user license is approximately $1495. Another group of commercial software is available through Steganography Analysis and Research Center in West Virginia. For example, StegAlyzerRTS is an advanced data leak protection software product that is capable of detecting the use of digital steganography in real time (SARC, 2010). It scans files entering and leaving a network for signs of steganography applications. It is also very expensive: as of March, 2009, the price was listed as $14,995.00 (PublishersNewswire, 2009). Neither of these software products presents the product’s false alarm rates of detection. A false alarm occurs when an innocent image (without hidden content) is flagged as a stego image erroneously. High false alarm rates can lead to manual inspection by a computer forensic analyst, which can require large amounts of time by the human. Thus, a low false alarm rate is necessary for practical use in forensic labs. A third stego-detection software is the freeware StegDetect, developed in 2001 by Neil Provos (Provos, 2001) to perform steganalysis on suspected stego images. His experiment was to steganalyze millions of JPEG images from sites like eBay and USENET (Provos & Honeyman, 2002; Provos & Honeyman, 2003) to determine if his program could detect hidden content. Not a single image with hidden data in it was detected. False alarms rates for the experiment performed with Stegdetect ranged from less than 10 percent to more than 20 percent. In short, commercial software is often too expensive for local police departments while at the same time, reliable false alarm rates of the software are not available for evaluating practical use.

It is of course desirable to have high detection rates of stego images (high true positive rates) while keeping the false alarm rate low, and this is the area of performance that CANVASS is focused to addressed. In this paper, we present true positive rates for the Canvass steganalyzer using a public steganography database. A true positive rate equals one minus the false positive alarm rate, discussed in more detail below.

The main goal of this research was to develop a software package that addressed the needs of local police departments who perform computer forensic functions for steganalysis. The authors collaborated with personnel at the state of Iowa’s Division of Criminal Investigation lab in Ankeny, Iowa, to develop a user-friendly software package. The lab required software that was easy for computer forensic personnel who are not experts in steganalysis to use. After numerous meetings with the Internet Crime Against Children (ICAC) workforce members, we developed Canvass as a cross-platform software that specifically addressed the image forensic application of steganalysis for the lab. It is designed with an intuitive graphical user interface that implements a state-of-the-art steganalyzer, discussed below briefly and in more detail in (Davidson & Jalan, 2010). The "brain" of the steganalyzer is a classifier designed to have low false alarm rate on known testing data; indeed, most of the research time was spent on developing a state-of-the-art pattern classifier that produced simultaneously higher true positive rates and low false alarm rates. The testing data was drawn from a collection of standard image databases (see (Davidson et al., 2010)) used by the steganlyzer community for steganalysis development. A table of false alarm rates calculated from more than 115,000 different images is available in Canvass, where many classifications have a low false positive rate.

The remainder of this paper is organized as follows. In "Steganography and Detection," we give a short description of how steganography operates and how the detection in JPEG images is performed. In "Canvass Steganalyzer," a description of the classifier that distinguishes between stego and cover images is given. In the section, "The Graphical User Interface (GUI) of Canvass," the software package is described from a user's perspective. We then discuss our findings and point out future directions of our research in the Conclusions section. The last three sections give Acknowledgements, Author Biographies, and the References.

STEGANOGRAPHY AND DETECTION

Written records of humans communicating covertly in plain sight go back over 2400 years ago, to the Greek Herodotus. His written records titled "*The Histories*" (de Selincourt, 1996) document the use of slaves to send messages secretly tattooed under scalp hair grown out to cover evidence of the tattoo. The tattoo was shaved off to reveal the hidden message. More recently, microdots and DNA have been used for covert communications (White, 1992; Clelland, Risca, & Bancroft, 1999). The explosive use of steganography in digital media in the past 15 years has occurred because of the ease with which it is possible to hide files in images. Hiding in audio files such as .mp3 and .wav is also relatively easy. In particular, JPEG formatted image data is easy to get and easy to embed in, using readily available freeware. For this reason, our efforts are focused on JPEG image data, and our software processes only JPEG image data.

A steganography algorithm consists of two parts: the embedding algorithm and the extraction algorithm. The embedding algorithm takes the message and alters a "cover" image in such a way that the message is part of the image file but invisible to the eye. This produces a stego image. The extraction algorithm takes the stego image, and inverting the process of embedding, recovers the hidden message. Knowledge of the original image is not typically necessary for proper extraction of the message. If the alterations to the original cover image are relatively small, it is impossible to notice visually that changes have been made to an image. In practice, the sender destroys the cover image after generating the stego image and therefore it is not possible to get access to the cover image. Thus, if a third party such as a computer forensic investigator would like to determine if a possible image contains a hidden message, other more sophisticated techniques than visual inspection must be used.

There are several different approaches for detection of hidden data. One way would be to inspect a computer disk for steganography freeware-related files, the presence of which may lead to more thorough searches for stego images. This in fact can be a very good indicator of the use of steganography freeware (Zax, 2009). Another way is to produce a hash file for a known stego image, and store the hash values in a database along with other hash values of known stego images. This will work if specific images are being passed from user to user, but will not work if a user creates a new stego image whose hash value is not in the database. A third way is to perform *blind* detection on an image. A blind detector takes an image as an input and produces a YES (stego) or NO (innocent) classification of the image. Most successful blind detectors use sophisticated pattern classifiers that extract statistical measures from known examples of stego and cover image data to generate “signatures” able to distinguish between the two classes, stego and cover images. The steganalyzer in Canvass has such a classifier, based on a state-of-the-art set of statistical measure (Davidson et al. 2010). The use of the classifier is transparent to the user, and thus the user does not have to know the inner workings of the steganalyzer.

How does steganography embedding work? Most steganographic algorithms embed bit values into an image. Thus, the digital file to embed must be represented as a bit stream of zeros and ones. Since all digital data on a computer is stored in this way, any file in theory can be embedded into an image. The basic process is as follows. An image consists of *gray values* at pixel locations. The gray values are integers represented by a certain number of bits. Many images use 8 bit integers to represent their colors. The first bit of the message is compared with the least significant bit (LSB) of the first gray value at a specified pixel location in the image. If the bit values match, then nothing is done and the second bit of the message is compared with the LSB of the gray value at the next specified pixel location. If any message bit does not match the LSB of the current gray value, then the gray value’s LSB is changed to match the message bit. Each message bit is thus “embedded” in this manner until there are no more message bits or there are no unused pixel locations left. Extraction involves the reverse process: the scanning order of the embedding process is followed and the LSB of each gray value is inspected and written down. The bit insertion process is pictorially represented in Figure 1.

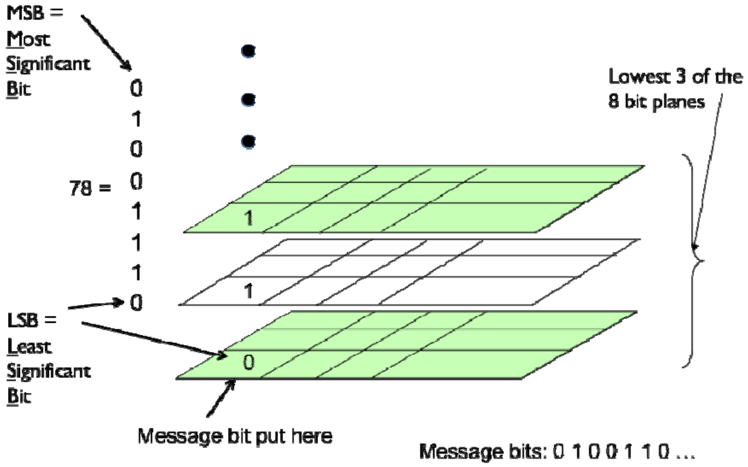


Figure 1. How message bits are inserted into the LSBs of image.

Knowing the scanning order for embedding plus any keys used to encrypt the bit stream allows that user to extract and decrypt the message. However, this information is not usually available to anyone except the sender and receiver. Changing enough bits can, however, render the stego image statistically different from the original cover image, and it is this property that permits pattern classifiers to be developed that can differentiate between stego and cover images. The interested reader is guided to (Provos et al. 2003) for more detailed information on this topic.

In the case of JPEG images, the situation is slightly more complicated. The values that are changed to match the message bits are the *quantized discrete cosine transform (DCT) coefficients,* the important information representing the visual content of the image. They are integers that reside in a transform domain (of the discrete cosine transform) and are subsequently Huffman encoded in a lossless format to represent the image data in a compressed format. The representation of the image data using the quantized DCT coefficients allows much of the redundant information to be discarded, and thus allows a *compressed* representation of the original data to be stored typically with many fewer bits. The interested reader is directed to (Bhaskaran & Konstantinides, 1995) for further details of the use of the DCT in image compression. The pertinent information for steganalysis is that statistical analysis of the distributions of these quantized DCT coefficients can offer very good steganalysis detection and the steganalyzer in Canvass uses this approach.

CANVASS STEGANALYZER

As mentioned previously, Canvass uses a pattern classifier as the “brain” of the software. This is a sophisticated software program that is trained on known examples of stego-images and innocent or cover images. The classifier used in Canvass is the Support Vector Machine (SVM) (Vapnik, 1995) and provides very good classification results in steganalysis. There are two stages to using an SVM: 1. Training and 2. Testing. In the training stage, input values in the form of features are processed by the SVM algorithm iteratively until an acceptable solution is found. In the testing phase, a possibly unknown image is input to the classifier and very quickly it is classified as one of two classes, stego or innocent. The training stage takes several hours to several days of computing time to find a solution and is typically compute-intensive.

The features that are used to train the SVM are crucial to the performance of the classifier. If the features extracted from the image are not representative of the class then the performance of the classifier will be poor in both phases. Thus, selection of good features is essential to good performance of the classifier. The features in Canvass are based on conditional probability density functions derived from modeling the image as a partially ordered Markov model (POMM) (Davidson, Cressie, and Hua, 1999). Using the difference of quantized Discrete Cosine Transform (DCT) coefficients in four pixel directions, the differences are modeled as POMMs and the empirical values of their probability density functions are used as input to the pattern classifier. More details can be found in (Davidson et al, 2010). The number of features used is 98. Another state-of-the-art steganalyzer in (Pevny and Fridrich, 2007) used the quantized DCT coefficients directly plus values from a Markov transition matrix representing differences of the coefficients. Their classifier required 274 features. The paper (Davidson et al, 2010) describes detection rates and false positive rates of both steganalyzers, with false alarm rates were typically between 0 and 10 percent for both Pevny’s and Davidson’s steganalyzers. Thus, the sophisticated and accurate POMM steganalyzer developed over many months of research effort was used in the core of Canvass for classification. As mentioned earlier, it is important not only to have high detection rates of stego images, but low false alarm rates of innocent images.

Once trained, Canvass detects the presence of five different embedding algorithms: Jsteg-jpeg (Upham, 1995), Outguess (Provos, 2001), F5 (Westfeld, 2001), Steghide (Hetzl, 2003), and JPHide (Latham, 1999). The general detection process is described pictorially in Figure 2. The suspect image is input to a steganalyzer. The steganalyzer extracts the statistical features called a feature profile, and inputs the features to Canvass. The feature values are used by the classifier to produce a YES (stego) or NO answer. Figure 2 shows a sample statistical profile which consists of feature number and feature value extracted by a steganalyzer. In the case of Canvass, an embedding algorithm that was likely used is also identified in the case that the classification of the image is stego.

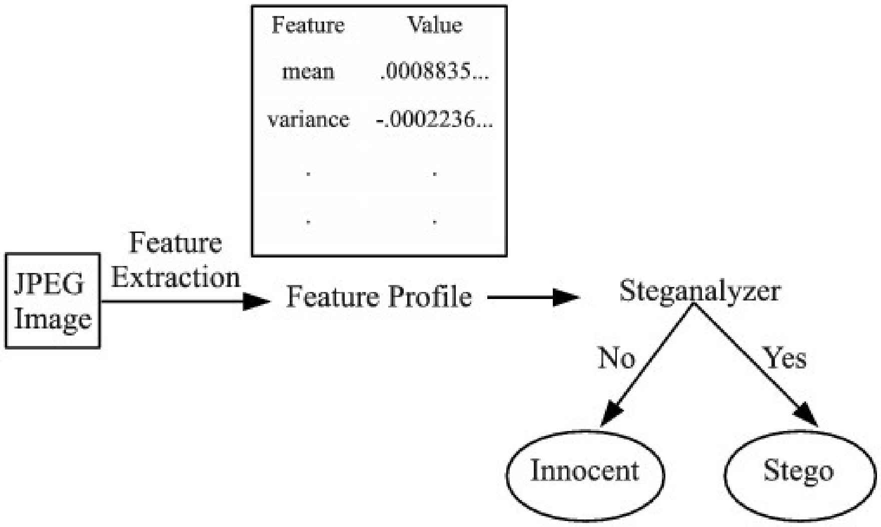
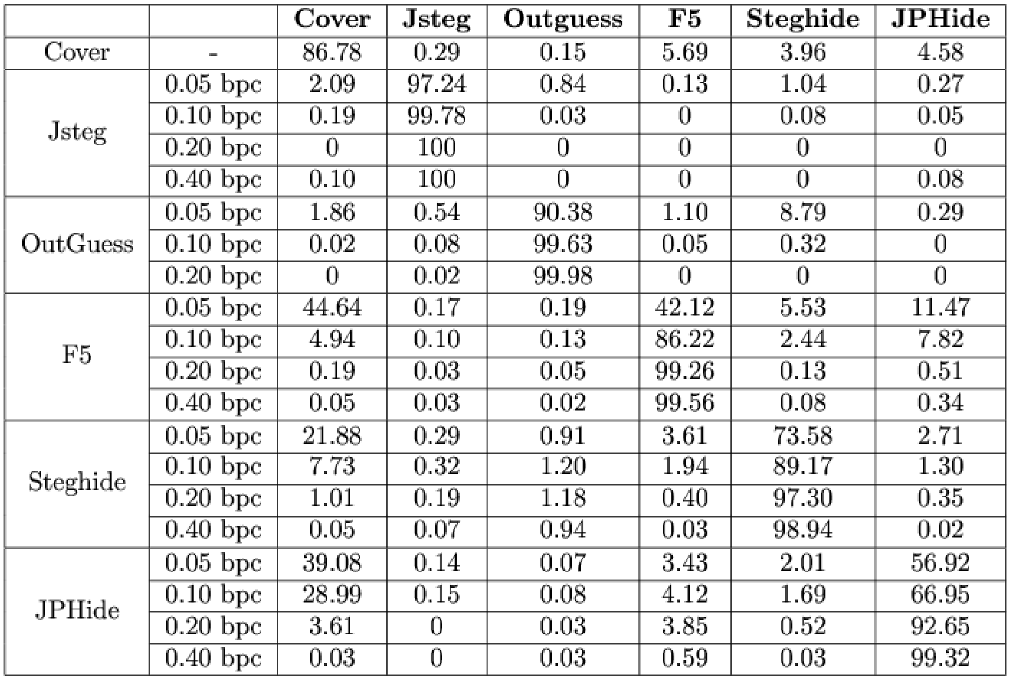


Figure 2: Processing of image through steganalyzer to detect stego or innocent image.

A classifier has an associated false alarm rate, which is calculated on known classes of data. A *false positive* occurs when the steganalyzer classifies a cover image as a stego image. A *true positive* occurs when a stego image gets classified as a stego image. Obviously, the lower the false positives the better off the forensic analyst is. After using 106,571 images for training the SVM classifier for Canvass, a set of 115,603 different test images was used to produce the Confusion matrix shown in Table 1. Each pair of classes has an SVM that chooses between the two classes. The six classes are: cover, Jsteg, Steghide, Outguess, F5, and JPHide. There are a total of 15 such binary classifiers, and the best answer is chosen by majority vote of the 15 answers. This approach gives reasonable results and is applied to the Canvass steganalyzer.

Table 1. Confusion matrix for Canvass's true positive rates expressed in percentages.



		Cover	**Jsteg**	**Outguess**	**F5**	**Steghide**	**JPHide**
Cover	-	86.78	0.29	0.15	5.69	3.96	4.58
Jsteg	0.05 bpc	2.09	97.24	0.84	0.13	1.04	0.27
	0.10 bpc	0.19	99.78	0.03	0	0.08	0.05
	0.20 bpc	0	100	0	0	0	0
	0.40 bpc	0.10	100	0	0	0	0.08
OutGuess	0.05 bpc	1.86	0.54	90.38	1.10	8.79	0.29
	0.10 bpc	0.02	0.08	99.63	0.05	0.32	0
	0.20 bpc	0	0.02	99.98	0	0	0
F5	0.05 bpc	44.64	0.17	0.19	42.12	5.53	11.47
	0.10 bpc	4.94	0.10	0.13	86.22	2.44	7.82
	0.20 bpc	0.19	0.03	0.05	99.26	0.13	0.51
	0.40 bpc	0.05	0.03	0.02	99.56	0.08	0.34
Steghide	0.05 bpc	21.88	0.29	0.91	3.61	73.58	2.71
	0.10 bpc	7.73	0.32	1.20	1.94	89.17	1.30
	0.20 bpc	1.01	0.19	1.18	0.40	97.30	0.35
	0.40 bpc	0.05	0.07	0.94	0.03	98.94	0.02
JPHide	0.05 bpc	39.08	0.14	0.07	3.43	2.01	56.92
	0.10 bpc	28.99	0.15	0.08	4.12	1.69	66.95
	0.20 bpc	3.61	0	0.03	3.85	0.52	92.65
	0.40 bpc	0.03	0	0.03	0.59	0.03	99.32

Different amounts of payload (hidden content) were embedded into the images, expressed in *bits per nonzero coefficient*, or bpc. This is an average of amount of bits embedded over all the possible bits allowing embedding. The confusion matrix gives the percentages of images that were correctly classified into the proper category: Cover, Jsteg-jpeg, Outguess, StegHide, and JPHide. If an image was misclassified into a different category then it appears as part of the percentage of all images that were misclassified in that category improperly. For example, 0.19 percent of the 5,936 images embedded with 0.05 bpc using the F5 algorithm were improperly classified as being embedded using Outguess. These numbers are as good as or better than the state-of-the-art steganalyzer in (Pevny et al., 2007). The false positive rate for Outguess embedded at 0.10 bpc is 100 – 99.63 = 0.37%. Thus, many of the image classes have low false positive rates. Note that although many classes have low false positive rates, a classifier cannot be trained on the universe of jpeg data and so there will always be some error. The goal is to use as large a database of training images as possible to find the “best” solution to the classification problem.

THE GRAPHICAL USER INTERFACE (GUI) OF CANVASS

Canvass was written in Java to provide complete portability to different platforms. It provides the following features to the user:

1. Ability to process multiple images with one command either on a computer or at a specified website.
2. Display of processing information in real time. It shows a variety of information such as which steganography algorithm was likely used for embedding.
3. An option to save the processing information at any time.
4. It displays the current image for visual inspection.
5. It has the ability to run on multiple platforms.

A Model-View-Controller (MVC) architecture has been used to design this software. Because of this, it can be easily extended using a different steganalyzer from the backend, for example, if additional binary classifiers are added to extend the ability to classify other embedding algorithms or even using other feature sets as inputs. This software will be made available from the Midwest Forensic Resource Center, Ames Laboratory for limited distribution to recognized police departments, after June 1, 2010.

CONCLUSIONS AND FUTURE DIRECTIONS

The software package Canvass has been developed to address the limited steganalysis needs of the Iowa Division of Criminal Investigation. The GUI is easy to use and effective in processing many image data in batch mode. The steganalyzer itself is designed to detect innocent images and stego-embedded images from five different algorithms - Jsteg-jpeg, Outguess, F5, Steghide, and JPHide. Since it uses a blind steganalyzer, Canvass has the possibility of detecting other similar embedding algorithms that use the JPEG domain. There are several directions for improvements. One is to add the detection of more embedding algorithms by training more binary SVMs and including them as part of Canvass’s steganalyzer. Another extension of Canvass’s capabilities would be to include a system scan of the computer to look for file artifacts residing on the computer, which indicate stego software was used for embedding. Including capabilities for password attack on identified stego images from within Canvass could also extend the practical use of Canvass. Identifying double-compressed images accurately and then passing to appropriate classifiers would also be very useful.

Another area that could use improvement is the accuracy of low embedding rate. The detection accuracies are quite high when an appreciable fraction of the largest possible message is embedded, such as at 0.2 bpc and 0.4 bpc. However, for the lower rates of 0.05 bpc and 0.1 bpc, detection rates particularly for F5 and Steghide are quite low. Increasing the accuracy of the steganalyzer will require research into creating a better pattern classifier, and will include looking for better features, and developing better classifiers models. Complementing the current POMM features with other different features could increase the classifier accuracies at the lower embedding rates. Addition of other features sets, such as those in (Pevny et al., 2007), might give better complementary detection. Another approach might be to estimate message length using the POMM. Since the POMM provides a closed form for calculating the joint probability density function, new techniques could be investigated by assuming a parameterized model of POMM to estimate the message length and maximizing the joint probability density function conditioned on parameters of the model. This could provide an indirect measure of message length, which in turn could be used to provide detection of embedding above a low threshold representing the message length. Another way to estimate the message length might be to use the current features and length of message embedded along with SVM regression, a variant of the support vector machine, to predict the length of messages in unknown images.

ACKNOWLEDGEMENTS

This work was funded by the National Institute of Justice, through the Midwest Forensics Resource Center at Ames Laboratory under Interagency Agreement number 2008-DN-R-038. The Ames Laboratory is operated for the U.S. Department of Energy by Iowa State University, under contract No. DE-AC02-07CH11358. We would also like to thank Dr. Gwaneal Doerr for providing easy access to a rich database of JPEG images that facilitated the success of this research.

AUTHOR BIOGRAPHIES

Dr. Jennifer Davidson is the Associate Chair of the Department of Mathematics at Iowa State University. She has been performing research in image processing for the past 25 years and steganalysis for the past six years. She is a member of SPIE and is a coordinating editor for the Journal of Mathematical Imaging.

Jaikishan Jalan holds a Bachelor's degree in Computer Science and an M.S. degree in Computer Science from Iowa State University. His research interest lies in image processing, multimedia and information security.

REFERENCES

Stegoarchive (1997). Available online at http://www.stegoarchive.com/ Accessed 4/11/2010.

MSU Stego Video (2004). Available online at http://www.compression.ru/video/stego_video/index_en.html Accessed 4/11/2010.

Kelley, J. (2001)0. “Terrorist instructions hidden online,” available online at http://www.usatoday.com/tech/news/2001-02-05-binladen-side.htm,” USA Today (February 2001).

Kolata, G. (2001). “Veiled messages of terrorists may lurk in cyberspace,” available online at http://www.nytimes.com/2001/10/30/science/physical/30STEG.html. New York Times (October 2001).

Weimann, G. (2004). “How modern terrorism uses the internet. Special report 116, available online at http://www.usip.org/files/resources/sr116.pdf,” United States Institute of Peace (March 2004).

“Electronic crime scene investigation: A guide for first responders,” (2008). Second edition, available online at http://www.ojp.usdoj.gov/nij/publications/ecrime-guide-219941/chapter7-219941.pdf. National Institute of Justice (April 2008). Accessed 4/11/2010.

Kerbaj, R. and Kennedy, D. (2008). “Link between child porn and muslim terrorists discovered in police raids, available online at http://www.timesonline.co.uk/tol/news/uk/crime/article4959002.ece,” (October 2008). Accessed 4/11/2010.

“Stego suite- discover the hidden.” (2010). Available online at http://www.wetstonetech.com/cgi-bin/shop.cgi?view,1 Accessed 4/11/2010.

Provos, N. (2001). Stegdetect software. Available online at http://www.outguess.org/detection.php

Provos, N. and Honeyman, P. (2002). “Detecting steganographic content on the internet,” In Proceedings of the Internet Society, Network and Distributed System Security Symposium (2002).

Provos, N.; Honeyman, P. (2003). "Hide and seek: an introduction to steganography," IEEE Security & Privacy, vol.1, no.3, pp. 32- 44.

SARC (2010). ” Information on StegAlyzerRTS available online at http://www.sarc-wv.com/products/stegalyzerrts.aspx Accessed 4/11/10.

PublishersNewswire (2009). http://publishersnewswire.com/2009/05/28/PNW1141_110950.php Accessed 4/11/10.

Davidson, J. and Jalan, J. (2010). Steganalysis using Partially Ordered Markov Models, submitted to 12th Information Hiding Conference, June 28 - 30, 2010, Calgary, Alberta, Canada.

de Selincourt, A. (1996). Herodotus: The Histories, Penguin Books.

White, W. (1992). The microdot: History and application, Phillips Publications.

Clelland, C. T., Risca, V., and Bancroft, C. (1999). “Hiding messages in DNA microdots,” Nature , 533–534.

Zax, R. and Adelstein, F. (2009). “Faust: Forensic artifacts of uninstalled steganography tools,” Digital Investigation 6(1-2), pp. 25 – 38.

Bhaskaran, V. and Konstantinides, K. (1997). Image and Video Compression Standards: Algorithms and Architectures, Kluwer Academic Publishers, Norwell, MA, USA.

Upham, D. (1995). Jsteg-Jpeg http://www.funet.fi/pub/crypt/steganography/Accessed online on 4/11/10.

Westfeld, A. (2001). “F5: A steganographic algorithm,” in Proceedings of Information Hiding, Lecture Notes in Computer Science 2137, 289–302, Springer Berlin / Heidelberg.

Hetzl, S. (2003). Steghide. http://steghide.sourceforge.net/ Accessed 4/13/2010.

Latham, A. (1999). Jphide&seek. http://linux01.gwdg.de/~alatham/stego.html Accessed 4/11/2010.

Vapnik, V. (1995). The Nature of Statistical Learning Theory. Springer-Verlag.

Davidson, J., Cressie., and Hua, X. (1999). “Texture synthesis and pattern recognition for partially ordered Markov models,” Pattern Recognition, Vol. 32, pp. 1475-1505.

Pevny, T. and Fridrich, J. (2007). “Merging Markov and DCT features for multi-class jpeg steganalysis,” Security, Steganography, and Watermarking of Multimedia Contents IX Vol. 6505, Society for Photo-optical Engineers (SPIE).