

# Markov Process Based Steganalysis by Using Second-Order Transition Probability Matrix

Ainuiddin Wahid Abdul Wahab<sup>1,2</sup>, Hans Georg Schaathun<sup>1</sup> and Anthony TS Ho<sup>1</sup>

<sup>1</sup>Department of Computing, University of Surrey, UK

<sup>2</sup>Department of Computer System and Technology, University of Malaya, Malaysia

[a.abdulwahab@surrey.ac.uk](mailto:a.abdulwahab@surrey.ac.uk)

[h.schaathun@surrey.ac.uk](mailto:h.schaathun@surrey.ac.uk)

[a.ho@surrey.ac.uk](mailto:a.ho@surrey.ac.uk)

**Abstract:** In this paper, we analyse and extend the steganalysis technique employed by Shi et al. (2006), where they used a first-order Markov model to generate second order statistics from the JPEG 2-D array. They have suggested that a second order Markov model should improve the performance, but did not proceed with the implementation due to its computational complexities. We have implemented a steganalysis based on the proposed second-order model, and found that the classification result was significantly improved at the expense of computational cost (2916 features). Interestingly, we find improved detection of embedding by the F5 software, but this seems to be entirely due to the detection of double compression and not due to the embedding itself.

**Keywords:** Steganalysis, F5, SVM.

## 1. Introduction

Steganography allows one to hide covert messages in such a way that an adversary will not be able to detect the existence of the secret. From *The Histories of Herodotus*, steganography can be dated back to 440 BC. One example is the tale of Demaratus sending a warning by writing directly on the wooden back of a wax tablet before applying its beeswax surface to conceal that warning (Petitcolas 1999). Another example is when Histiaeus shaved the head of his most trusted slave and tattooed a message on it. The message is then hidden when the hair had grown.

Over the last decade a wide range of steganography techniques have appeared in the literature. Similarly, a wide range of steganalysis techniques have also been made available, intended to let an adversary determine whether an intercepted image contains any embedded message. In particular, a number of steganalysis techniques based on machine learning have also emerged (Lyu 2002, Shi 2006). Such techniques tend to be blind, in the sense that it do not assume any particular steganography algorithm and can normally break a variety of algorithms. Other methods that are specific only to certain steganography techniques, such as proposed by Fridrich (2002), are categorised as non-blind techniques.

Our purpose is to analyse and extend the JPEG steganalysis technique by Shi et al. (2006). We evaluate the system critically in view of the double compression bugs in common steganography software, which largely been ignored by other authors. This is of particular concern since the performance of the steganalysis may be due to the detection of changes caused by double compression, and not on the embedded process artefacts. With that in mind, this research is of potential benefit in other areas of steganalysis.

## 2. Markov Process Based Steganalysis

Proposed by Shi et al. (2006), this steganalysis technique works on JPEG 2-D arrays formed from the magnitudes of quantised block DCT coefficient. To utilize the second order statistics for steganalysis, a Markov process is applied to model the difference of JPEG 2-D arrays along horizontal, vertical and diagonal directions. To reduce the feature vector dimensionality, thresholding is applied in this technique. Thresholding greatly reduce the dimensionality of transition probability matrices which then contribute to smaller dimensions of feature vectors and reduce the computational complexity in the approach.

## 2.1 JPEG 2-D Array

In their approach, Shi et al. first define the JPEG 2-D array. They first consider the 2-D array consisting of all the JPEG coefficients which have been quantised with the JPEG quantisation table but have not been zig-zag scanned, run-length coded and Huffman coded. This 2-D array, which is the same size as the original image filled up with the corresponding quantised block DCT (BDCT) coefficients for each 8X8 block. Only then, the absolute value for each coefficient is considered.

*Definition 1: In thresholding technique, the values smaller than threshold value  $-T$ , will be represent by  $-T$  and values bigger than  $T$  will be present by  $T$ .*

## 2.2 Difference in JPEG 2-D Array Elements

In (Shi 2006), it is expected that the disturbance caused by the steganographic methods in JPEG images can be enlarged by observing the difference between an element and one of its neighbours in the JPEG 2-D array. Denoting the JPEG 2-D array as  $F(u, v)(u \in [0, S_u - 1], v \in [0, S_v - 1])$ , where  $S_u$  and  $S_v$  are the size of JPEG 2-D array in horizontal and vertical directions, the difference between the arrays are generated by:

$$\begin{aligned}F_h(u, v) &= F(u, v) - F(u + 1, v), \\F_v(u, v) &= F(u, v) - F(u, v + 1), \\F_d(u, v) &= F(u, v) - F(u + 1, v + 1), \\F_m(u, v) &= F(u + 1, v) - F(u, v + 1),\end{aligned}$$

where  $u \in [0, S_u - 2], v \in [0, S_v - 2]$  and  $F_h(u, v)$ ,  $F_v(u, v)$ ,  $F_d(u, v)$  and  $F_m(u, v)$  denote the difference between the arrays in horizontal, vertical, main diagonal and minor diagonal directions respectively.

## 2.3 Transition Probability Matrix

The generated difference between the JPEG 2-D arrays are then modelled by a Markov random process. According to the theory of random processes, the Markov process can be characterised by a transition probability matrix. The one-step transition probability matrix was used in their experiment in order to have a suitable balance between high steganalysis capability and manageable computational complexity. Thresholding technique is then applied to further reduce the computational complexity.

By selecting a threshold with value  $T$ , only values which fall into  $\{-T, \dots, -1, 0, 1, \dots, T\}$  from JPEG 2-D arrays are considered. The result will be a transition probability matrix with a dimensionality of  $(2T + 1) \times (2T + 1)$ . In summary,  $(2T + 1) \times (2T + 1)$  elements for each of the four transition probability matrices contribute to  $(2T + 1) \times (2T + 1) \times 4$  dimensional features for steganalysis. In their experiment, by setting the threshold value to 4, the resultant transition probability matrix is  $9 \times 9$  for each of the differences in four directions of the 2-D arrays. In total,  $(9 \times 9) \times 4 = 324$  feature elements were constructed.

## 3. Second-Order Transition Probability Matrix

For our study, we try to look at the disturbance caused by the steganographic methods in JPEG images by observing the difference between an element and two of its previous neighbours in the JPEG 2-D array (Figure 2).

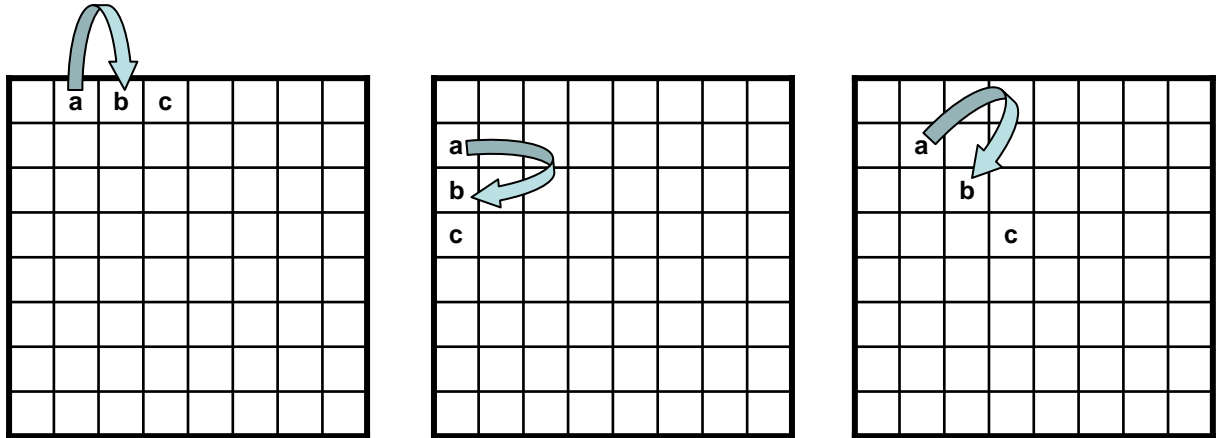


Figure 1: First-order transition probability

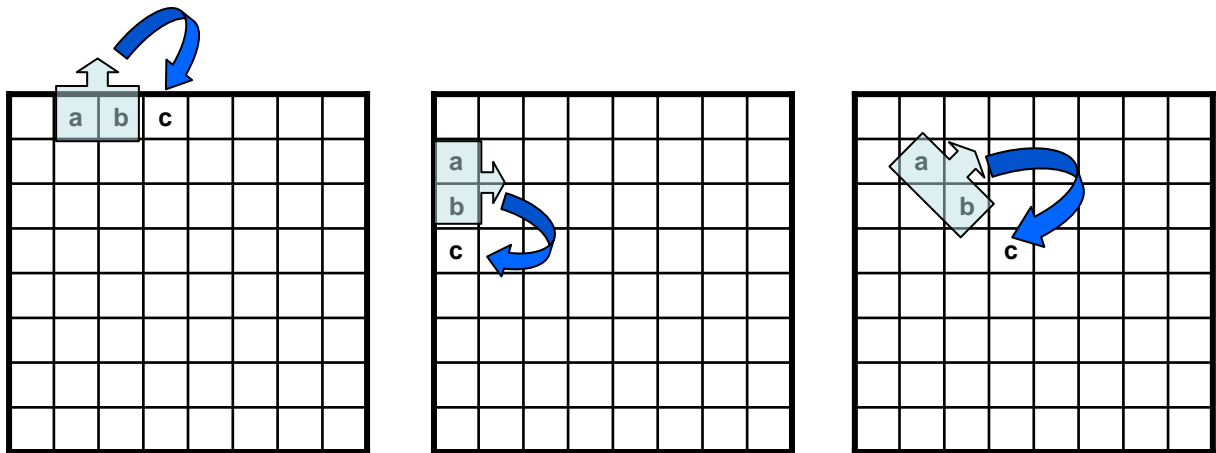


Figure 2: Second-order transition probability

All the experimental tasks are the same as what have been discussed above. The only different is on the transition probability matrix used. In previous work by Shi et al., transition probability matrix that used to characterize a Markov process is the first-order transition probability matrix (Figure 3) where it refers to the transition probabilities between two immediately neighbouring elements in the difference 2-D array (Figure 1).

	a	b	c
a	$p(x=a,y=a)$	$p(x=a,y=b)$	$p(x=a,y=c)$
b	$p(x=b,y=a)$	$p(x=b,y=b)$	$p(y=b,x=c)$
c	$p(x=c,y=a)$	$p(x=c,y=b)$	$p(x=c,y=c)$

Figure 3: First-order transition probability matrix

	a	b	c
aa	$p(x=a,y=a   z=a)$	$p(x=a,y=a   z=b)$	$p(x=a,y=a   z=c)$
ab	$p(x=a,y=b   z=a)$	$p(x=a,y=b   z=b)$	$p(x=a,y=b   z=c)$
ac	$p(x=a,y=c   z=a)$	$p(x=a,y=c   z=b)$	$p(x=a,y=c   z=c)$
ba	$p(x=b,y=a   z=a)$	$p(x=b,y=a   z=b)$	$p(x=b,y=a   z=c)$
bb	$p(x=b,y=b   z=a)$	$p(x=b,y=b   z=b)$	$p(x=b,y=b   z=c)$
bc	$p(x=b,y=c   z=a)$	$p(x=b,y=c   z=b)$	$p(x=b,y=c   z=c)$
ca	$p(x=c,y=a   z=a)$	$p(x=c,y=a   z=b)$	$p(x=c,y=a   z=c)$
cb	$p(x=c,y=b   z=a)$	$p(x=c,y=b   z=b)$	$p(x=c,y=b   z=c)$
cc	$p(x=c,y=c   z=a)$	$p(x=c,y=c   z=b)$	$p(x=c,y=c   z=c)$

Figure 4: Second-order transition probability matrix

What we considered in our study is the second-order transition probability matrix (Figure 4), it refers to the transition probabilities between three consecutive elements (Figure 2). In summary,  $(2T + 1) \times (2T + 1) \times (2T + 1)$  elements for each of the four transition probability matrices contribute to  $(2T + 1) \times (2T + 1) \times (2T + 1) \times 4$  dimensional features for steganalysis. In our study, by setting the threshold value to 4, the resultant transition probability matrix is  $9 \times 9 \times 9$  for each of the differences in four directions of the 2-D arrays. In total,  $(9 \times 9 \times 9) \times 4$  feature elements were constructed. We assume that having more computational complexity by considering second-step transition probability matrix, will contribute for a better result as what suggested in (Shi 2006).

#### 4. Support Vector Machine

Support Vectors Machine (SVM) is employed to classify an image as either clean or stego based on its features vector. SVM has been proved to provide good classification results for a universal steganalyzer, as proposed by Lyu and Farid in (Lyu 2002). It also contributed to the good classification accuracy for a novel steganalysis technique based on Markov process that was proposed by Shi et al. (2006).

A classification requires training and testing data instances (Figure 5). In any given training set, each individual instance has a class label value (for example 1 or -1) and several attributes or features (feature vector). SVM will produce a classification model (mapping from instances to predicted classes) based on the training set data. The optimal hyperplane (the separating hyperplane which presents the largest margin or the distance of the closest points to the hyperplane) will be determined in the process. Classification model can then be used to predict the class of testing set based only on their attributes.

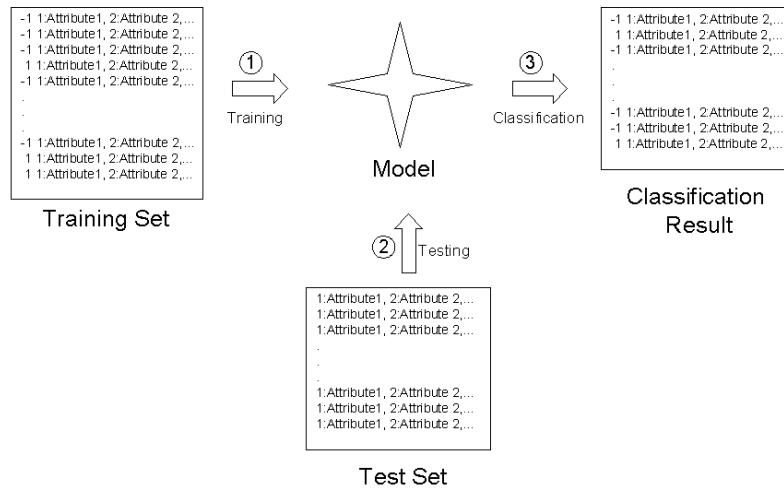


Figure 5: SVM classification process

With more features extracted in most of the recent steganalysis technique, SVM has become a popular choice for a classifier with its capability to view input data as two sets of vectors in a multidimensional space. In addition, its capability to perform multi class classification also makes SVM a good classifier (Pevny 2006).

The latest version of LibSVM (version 2.88) was used in our study. This package was developed by Chih-Chung Chang and Chih-Jen Lin (Chang 2008). In (Lyu 2002), linear non-separable and non-linear SVM (using radial basis kernel function) are used from LibSVM. A different kernel, polynomial kernel with degree of 2, was used in (Shi 2006). In addition, LibSVM was also used in (Avcibas 2005), (Kharrazi 2005) and (Kharrazi 2006) for steganalysis purposes.

## 5. Task

To evaluate the performance of the suggested technique on F5 (Westfeld 2008), we used a total of 2000 images from online databases (USC-SIPS 2009), (GroundTruth 2009) and our own captured images to have a sufficient number of training and testing images. All images were decompress, cropped to the centre of 640X480 pixels, and then compressed with a quality factor of 75 in JPEG image format. The cropping technique can help to ensure that the image dimensions is not correlated with spatial characteristics, such as noise or local energy (Bohme 2005)

After the image preparation process, all the images went through the F5 encoding process to produce sets of stego images with message size of 4096 bytes, 1848 bytes and 618 bytes embedded. With the cover and stego images ready, the steganalysis technique conducted to produce the features for the subsequent classification process. The freely available LibSVM (Chang 2008) was then used as the classifier. For SVM, the optimal soft-margin 'c' and kernel's parameter  $\gamma$  was determined using 'grid.py' utilities in LibSVM package.

## 6. Results and Discussion

As what used in (Shi 2006), (Farid 2002) and (Lyu 2002), classification accuracy was used to measure the performance of the proposed technique.

*Definition 3: The classification accuracy is the percentage of correct classification for cover and stego images in test set.*

$$Accuracy = \frac{TruePositive + TrueNegative}{TotalCover + TotalStego}$$

By using confidence interval estimation (Bhattacharyya 1977), from the results in Table 1, we have computed with 95.0% confidence intervals of the accuracy for first-order transition probability matrix and second-order transition probability matrix approaches. The confidence intervals were calculated using the following formula

$$p = \left( \hat{p} - z_{\alpha/2} \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}, \hat{p} + z_{\alpha/2} \sqrt{\frac{\hat{p}(1-\hat{p})}{n}} \right)$$

where  $\hat{p}$  is calculated correct classification,  $\alpha$  is a significance level,  $Z_{\alpha/2}$  is the  $Z$  value from standard normal distribution and  $n$  is the number of images in the test set. For example, the calculation of confidence interval for first-order transition probability matrix with message size of 618 bytes is

$$p = \left( \hat{p} - 1.96 \sqrt{\frac{0.98 \times 0.02}{200}}, \hat{p} + 1.96 \sqrt{\frac{0.98 \times 0.02}{200}} \right)$$

where  $z_{\alpha/2} = 1.96$  (from normal table) and  $n = 200$ .

Table 1: Classification accuracy and confidence interval

	Message Size (bytes)		
	618	1848	4096
<b>First-order transition probability</b>	98.0% 96.1%, 99.9%	93.5% 90.0%, 96.9%	89.5% 85.3%, 93.8%
<b>Second-order transition probability</b>	99.1% 97.8%, 100%	99.1% 97.8%, 100%	98.6% 97.0%, 100%

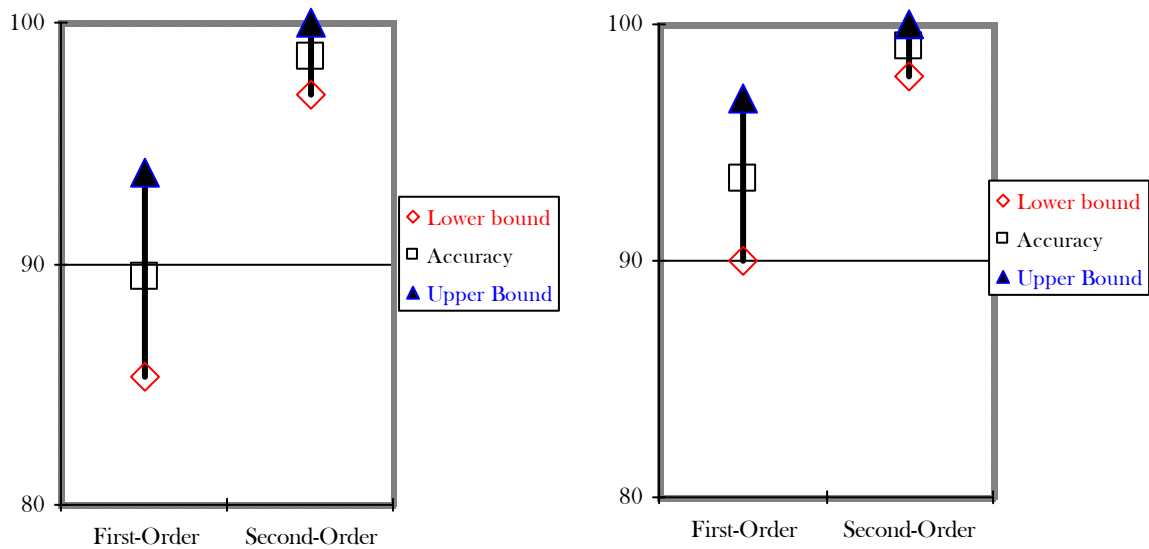


Figure 6: Confidence interval comparison

In Table 1, for message size of 1848 bytes and 4096 bytes, we note that the 95% confident interval do not overlap (Figure 6), which clearly means that the second-order transition probability matrix approach is significantly more accurate in this two cases. However, there is an overlap on confidence interval for message size of 618 bytes. Although there is no statistically significant conclusion in this case, the result still shows a better performance for the calculated accuracy.

## 7. Double Compression

Using a cover image in JPEG format for this study means that the image used has been compressed at least once. By working on these compressed cover images, F5 software then decompress and recompress the images with a specific quality factor before performing the embedding process. The resulting output of this process is the doubly compressed stego images.

Based on these processes, we realised that it is possible that the steganalysis techniques used can also produce an error by not considering the double compression effect. In a typical or common approach (Figure 7), the steganalyser will use the original cover image (single compress) and the corresponding stego image (doubly compress) for the steganalysis process as what conducted by Lyu (2002), Avcibas (2005), Kharrazi (2006) and Shi (2006). In this approach, there is a possibility that the stego image will also contains the artefacts caused by the double compression during the embedding process in addition to the embedded message.

Based on that assumption, we proposed a new approach where the steganalyser will use a cover image that has been compressed using the same steganography technique without any message embedded (Figure 7). By doing this, we assumed that the artefacts caused by double compression during the embedding process are equal for both cover and stego images. The differences between the two images are now based only on the artefacts caused by the embedded message.

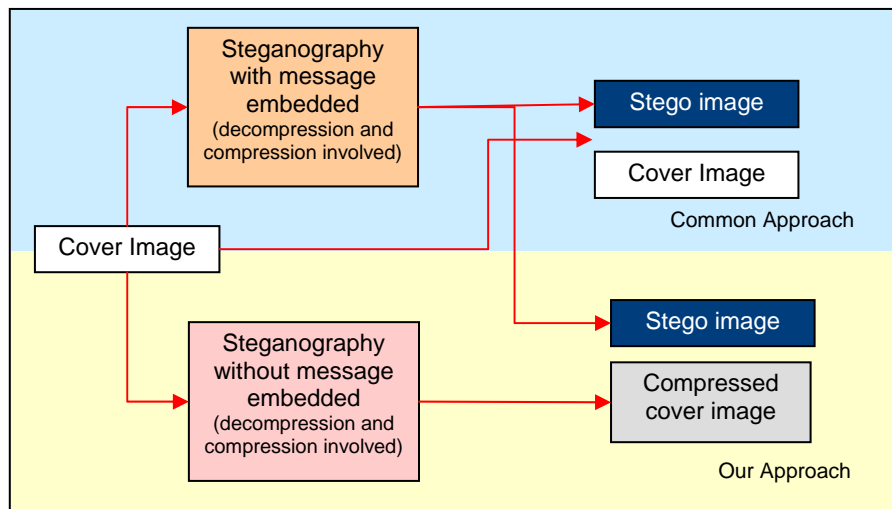


Figure 7: Steganalysis with JPEG compression consideration

Using the same steganalysis technique and methodology as in the previous section, our initial findings showed the possibility of the expected problem. The steganalysis performance was visibly affected when double compressed cover images were considered for both the first-order and second-order transition probability matrix methods (Table 2).

In Table 2, for message sizes of 1848 and 4096 bytes, we note that the 95% confident interval do not overlap (Figure 8), which means that the second-order transition probability matrix approach was significantly less accurate in these two cases. Result from this new approach clearly contradicts the claim made by Shi (2006). In our study, the accuracy for the second-order is less than the first-order when comparing doubly compressed images with and without F5 embedding.

Table 2: Classification accuracy and confidence interval with double compression consideration

	Message Size (bytes)		
	618	1848	4096
<b>First-order transition probability</b>	50.2% 43.3%, 67.1%	84.3% 79.3%, 89.3%	97.9% 95.9%, 99.9%
<b>Second-order transition probability</b>	50.0% 43.1%, 57.0%	55.6% 48.7%, 62.3%	70.6% 64.3%, 76.9%

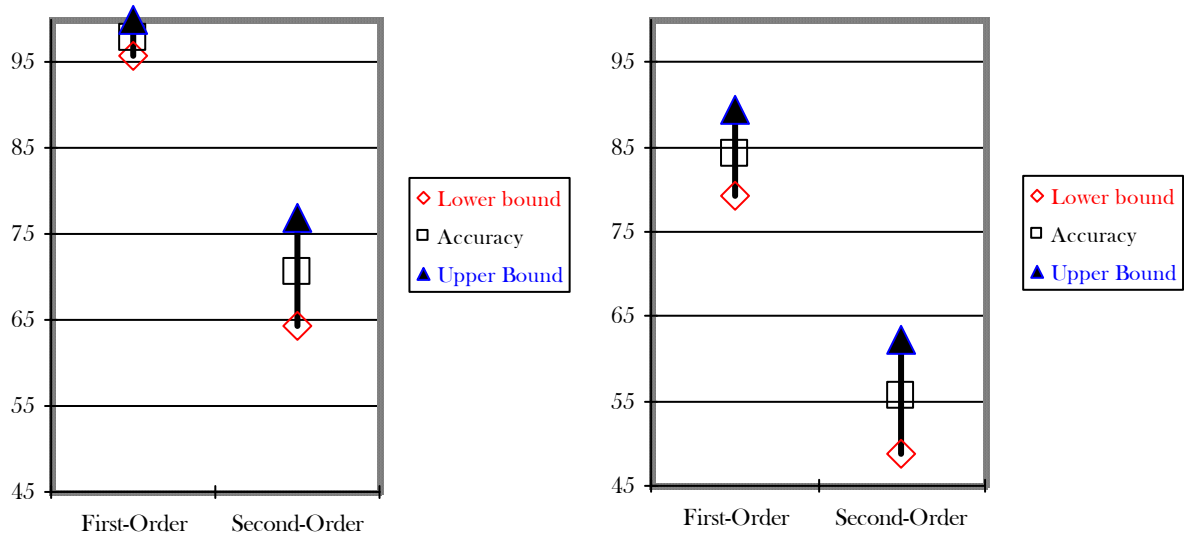


Figure 8: Confidence interval comparison

Based on these results, for first-step and second-step transition probability matrices, the steganalysis technique used is apparently more effective at detecting double compression artefacts rather than the changes caused by the embedding process itself. This problem will be further investigated in a future research by considering other JPEG steganography algorithms and steganalysis techniques. The finding from this research will enable us to evaluate the effectiveness of current steganalysis techniques and hopefully will help pave the way for better design and implementation of future steganalysis techniques.

## 8. Conclusion

In this study, we have analysed the performance of steganalysis technique proposed by Shi et al. (2006) by using the second-order Markov model. At the expense of computational cost (2916 features), we found an improved detection of embedding by the F5 software. This extraneous advantage however seems to be entirely due to the detection of double compression and not due to the embedding itself.

In the common approach, the steganalyzer will use the original cover image and the corresponding stego image from Westfeld's implementation of F5 when the classifier is trained. This is, for instance the case in (Shi 2006). Westfeld's implementation has one serious flaw, where the image is decompress and recompress before embedding.

Using purely F5 software for steganalysis is clearly unreliable due to the inherent double compression issue. The best solution probably would be to implement an F5 directly on an image without any process of decompress and recompress. Unfortunately there is still no API easily available for that purpose. Having a solution for this task is important in order to have a fair evaluation of steganalysis performance. This applies not only for F5 but also other JPEG based steganography, where decompression and recompression usually involved in their implementation.

## References

- Avcibas, I. Kharrazi, M. Memon, N. and Sankur B. (2005) "Image steganalysis with binary similarity measures", *EURASIP Journal on Applied Signal Processing 2005:17*
- Bhattacharyya, G.K. and Johnson, R.A. (1977) *Statistical Concepts and Methods*, Wiley
- Bohme, R. (2005) "Assessment of steganalytic methods using multiple regression models", *Information Hiding (IH05)*, Barcelona, Spain, pp 278-295
- Chang, C.C. and Lin, C.J. (2008) "LIBSVM: a library for support vector machines", [online], <http://www.csie.ntu.edu.tw/~cjlin/libsvm>
- Farid, H. (2002) "Detecting hidden messages using higher-order statistical models", *International Conference on Image Processing*, Rochester, NY
- Fridrich, J. Goljan, M and Hogeia, D. (2002) "Steganalysis of jpeg images: breaking the F5 algorithm", *Proc. 5th International Workshop on Information Hiding (IH 02)*, Noordwijkerhout, The Netherlands, pp 310-323
- Ground Truth Database (2009) "Department of Computer Science and Engineering, University of Washington", [online], <http://www.cs.washington.edu/research/imagedatabase/groundtruth/>
- Katzenbeisser, S. and Fabien, A.P. (2000) *Information Hiding Techniques for Steganography and Digital Watermarking*. Artech House
- Kharrazi, M, Sencar, H.T. and Memon, N. (2005) "Benchmarking steganographic and steganalysis techniques", *Security, Steganography, and Watermarking of Multimedia Contents VII*, San Jose, California, USA, pp 252-263
- Kharrazi, M, Sencar, H.T. and Memon, N. (2006) "Improving steganalysis by fusion techniques: A case study with image steganography", *LNCS Transactions on Data Hiding and Multimedia Security I, 4300*, pp 123-137
- Lyu, S. and Farid, H. (2002) "Detecting hidden message using higher-order statistics and support vector machines", *Proc. 5th International Workshop on Information Hiding (IH 02)*, Noordwijkerhout, The Netherlands, pp 340-354
- Petitcolas, F.A.P Anderson, R.J. and Kuhn M.G. (1999) "Information hiding: A survey", *Proceedings of the IEEE, special issue on protection of multimedia content, 87(7)*, pp 1062-1078
- Pevny, T. and Fridrich, J. (2006) "Multi-class blind steganalysis for JPEG images", *Proceedings of SPIE Vol. 6072*, San Jose, CA, pp 257-269
- Shi, Y.Q. Chen, C. and Chen, W. (2006) "A markov process based approach to effective attacking JPEG steganography", *8th International Workshop on Information Hiding (IH 2006)*, Alexandria, VA, USA , pp 249-264
- USC-SIPI Image Database (2009), "The University of Southern California – Signal and Image Processing Institute Image Database", [online], <http://sipi.usc.edu/database/>
- Westfeld, A. (2008) "F5", [online], <http://www.inf.tu-dresden.de/~westfeld/f5>