HTML Steganography Algorithms and Detection Methods

Thesis submitted in accordance with the requirements of
the University of Liverpool for the degree of Doctor in Philosophy by

Iman Thannoon Sedeeq

May 2018

Dedication

To The Memory of My Father Thannoon Sedeeq Omer

Glossary

Binary classification
The task of classifying the elements of a given set into two groups.

Carrier object
A cover media used to host the hidden message.

Channel
A pathway of the transmitted message.

Data mining
The process of discovering patterns in large data sets.

Feature vector
An n-dimensional vector of numerical features that represent some object.

HTML Tag
The hidden keywords within a web page that define how your web browser must format.

Hyper Text Markup Language
The standard markup language for creating web pages.

Maximum Embedding Capacity
The maximum hidden message length in bytes a carrier object can carry.

Message encoding/decoding
A method used to embed/encode and extract/decode the hidden message.

Standard Deviation
A measure used to quantify the despersion of a set of data values with respect to their mean.

Stego object
A carrier object after having a hidden message.

Stego-key
A piece of information used to embed and extract the hidden message.

List of Abbriviations

Acc	Accuracy
APCC	Attribute Position Change Counts
APS	Attribute Permutation Steganography
AUC	Area Under The Curve
CD	Critical Difference
DIC	Detect Invisible Characters
FN	False Negatives
FP	False Positives
HTML	Hyper Text Markup Language
ICS	Invisible Characters Steganography
MEC	Maximum Embedding Capacity
MLP	Multilayer Perceptrons
NB	Nave Bayes
SD	Standard Deviation
SVM	Support Vector Machines
TLCSS	Tag Letters Case Switching Steganography
TN	True Negatives
TP	True Positives
FP	False Positives
TV	Tag Variance

Acknowledgements

I would like to thank my both supervisors Dr. Alexei Lisitsa and Dr. Frans Coenen for their constant support, guidance and patience throughout my PhD journey. My deepest gratitude for all the time and the overwhelming experience I gained while working under their supervision. This experience will be fruitful with respect to the rest of my professional life. Again, profound gratitude for their constant support.
My gratitude also extends to the Iraqi government/Ministry of Higher Education for awarding me this opportunity so that my dream of getting a PhD can come true. I also would like to thank the Vice President of Mosul University, Prof. Obay Saeed Aldewachi, for his support to overcome the financial hardship that I experienced throughout my PhD study.
My sincere thanks to my husband and my children for their prayers and encouragement and I am profoundly sorry about what you experienced while I was away.
A special thanks also to my mother, brothers, sisters and everybody else who hoped that this journey would be ended in success and peace.

Abstract

The work presented in this thesis is directed at steganography detection in HTML files. The most common HTML steganography methods from the literature are considered in the thesis: (i) Attribute Permutation Steganography (APS) (ii) Invisible characters Steganography (ICS) and (iii) Tag Letters Case Switching Steganography (TLCSS). Steganography detection in HTML files entails two challenges. The first challenge is the identification of the steganography features that eventually would facilitate detection. Three features are thus proposed depending on the nature of HTML steganography method: (i) in case of APS the change of an attribute positions in an HTML tags was used (ii) in case of ICS the frequency distributions of white space character segments was used and (iii) in case of TLCSS the smoothness of tag letters was used. The second challenge is how to represent webpage content after steganography features have been identified. Four representations were proposed in this thesis again depending on the nature of HTML steganography method. In case of APS detection two representations were used (i) standard deviation of a webpage attributes' positions (ii) a feature vector of an attribute position changes count. In case of ICS detection a feature vector of frequency distributions of white space character segments was used. In case of TLCSS detection a webpage is represented by two sets of tags "Regular" and "Singular" according to a defined tag smoothness statistics. The aforementioned HTML Steganography methods are considered in the thesis in the context of both: the dynamic (monitoring) context and the (non-monitoring) static context. With respect to the dynamic (monitoring) context the proposed Statistical Detection (SD) approach directed at APS detection was considered. With respect to the static (non-monitoring) context all three identified forms of HTML steganography were considered. More specifically the Attribute Position Changes Count (APCC) approach directed at APS detection, the Detect Invisible characters (DIC) approach directed at ICS, and the Tag Variance (TV) approach directed at TLCSS. The reported evaluation indicated that good results were obtained in most cases. In the dynamic context the results indicated that proposed SD approach was effective. In the static context both the proposed APCC and the proposed DIC approaches, coupled with appropriate classification methods, resulted in an effective classification of stego and non-stego webpages. Also in the same context the proposed Tag Variance was effective at detecting very short messages that were embedded using TLCSS.

Contents

Illustrations

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

The work presented in this thesis is concerned with methods for HTML steganography detection. Steganography is an information hiding technique intended to provide covert communication in such a way that no one but the sender and receiver are aware of the communication's existence. Steganography requires a carrier, a container or a cover, used by the steganographer to carry the hidden message. The main requirement for a carrier is that its usage for steganographic purposes should not attract any third party attention, in other words any hidden message should not be visible to a casual observer. Steganography has been used in many forms since ancient times, however the advent of the Internet in modern times has significantly increased the scope for its usage.

Internet services provided many benefits, services that include, but are not limited to, abundant information and resources, communication without borders, entertainment and eCommerce. The Internet, despite its many benefits, has also provided a mechanism for unethical and criminal behavior and activity, including providing a conduit for steganography. Digital media, available through Internet services, for example images, video, audio files and documents, all provide a carrier for steganography. This digital media includes redundancy in various forms that can be used for the purpose of message hiding. For example image files can be easily distorted so as to hide a message whilst, so far as an unsuspecting observer is concerned, the image remains unchanged.

Network protocols provide another form of a carrier for steganography, as do webpages. Webpages are accessed frequently on daily bases, therefore they are ideal for usage as a cover for steganography because accessing of webpages will not cause any suspicion. Webpages are written using HTML (Hyper Text Markup Language) some of whose features provide an excellent opportunity for steganography, so called HTML steganography. HTML steganography can be achieved in a variety of ways, for example the addition of white space characters, switching letter case and alternating the ordering of tag attributes, can all be used for the purpose of steganography without changing the way that WWW pages are rendered by a web browser. There are many freely available

tools to support HTML steganography [42, 49]. The work presented in this thesis is directed at the detection of HTML steganography.

Steganography does have legitimate applications. For example the copyrighting of intellectual property by including (say) serial numbers in digital films, audio recordings and books [7]. Other legitimate applications include: feature tagging and time stamping of images of various kinds, and military and security applications where unobtrusive communication is required. Examples of non-legitimate usage of steganography include: criminal communication, data smuggling and industrial espionage [100].

Closely related to the legitimate use of steganography is watermarking [17, 31]. Watermarking is frequently used with respect to images available for purchase over the Internet so as to prevent their illicit usage. Once purchased the image is sent without the watermark. The distinction between watermarking and steganography is that watermarking is not intended to be unobtrusive as in the case of steganography. The emphasis is on robustness. In the case of watermarked images, the watermark must be resistant to various forms of transformations such as rotation, compression and cropping.

Another way of sending information that can only be deciphered by a sender and receiver, although much more obtrusive, is data encryption [13, 71]. The distinction with steganography is that steganography is used to hide the existence of a message, while cryptography is used to hide the content. To provide an extra layer of security the stenographer can of course also encrypt the hidden messages. The work presented in this thesis is directed at steganography detection, and is not concerned with cryptography or watermarking (although a relationship clearly exists between cryptography and water marking, and the process of steganography).

Given the above, the advantages offered by steganography, to the steganographer, can be summarized as follows:

- Secrecy of both the sender and receiver's identities.
- Avoidance of the need to use encryption; some countries prohibit the transmission of encrypted messages [5, 92].
- The hiding of messages very existence (unlike in the case of cryptography).

Steganography detection is concerned with identifying the presence of steganography; and, in some cases, by extension, also the extraction of hidden messages [15] [63]. The work presented in this thesis is directed at HTML steganography detection. In the context of steganography detection the term *steganalysis* is also used in the literature. Both terms, are used in this thesis and are assumed to be synonymous. In the literature the term steganalysis is sometimes used to describe processes whereby the security and robustness of steganography algorithms are assessed [11, 40]. So as to prevent confusion, in this thesis, the term steganalysis will only be used in the context of steganography detection.

The structure of the reminder of this introductory chapter is as follows: In Section 1.2 the motivation for the research is presented in more detail. The research question

and associated research issues are then discussed in Section 1.3. Section 1.4 outlines the research methodology adopted to address the research question and the associated issues, followed in Section 1.5 with an itemization of the research contributions. Section 1.6 presents details of published work produced as a result of the presented research, followed in Section 1.7 with an outline of the structure of the remainder of this thesis.

1.2 Research Motivation

As already noted above, the Internet is an open public access network which provides many benefits while also facilitating some undesirable activities, one of which is the provision of an illicit communication channel whereby hidden messages can be sent and received. One mechanism for doing this is by using steganography, the process of hiding messages within some cover messages in such a way that the cover message, at least to an unsuspecting observer, remains unchanged.

Although, as also noted above, steganography does have legitimated usages, it also has undesirable usages. Steganography has been used in the context of malware, malware such as Duqu and Alureon, both directed at capturing information from the infected system and unobtrusively transferring this information back using innocent images as the carrier [100]. The usage of steganography by terrorists was suspected during the lead up to the 9/11 attack in the USA [56, 57]. There is also an evidence of a Russian spy ring using digital images as a cover to pass classified information from the USA to Moscow [86]. Simply by monitoring the number of times, running to hundreds of thousands, that steganography tools are downloaded, is an indicator of the scale of stegongraphy usage [84, 85](for example 59737 downloads of openPuff 4.00 tool for steganography and watermarking and 9825 downloads of SteganoG 1.130.0 tool for steganography using BMP files/ last updated Dec. 2017).

Given the above there is a need for techniques to detect the presence of steganography. The motivation for these techniques is to uncover unauthorized covert messaging. As noted above, webpages are an ideal carrier for steganography, specifically HTML steganography. However, steganography detection (steganalysis) in the context of HTML steganography has received very little attention in the research literature.

The motivation for the work presented in this thesis can therefore be summarized as follows:

1. The increasing prevalence of steganography as witnessed by the number of times steganography tools have been downloaded from the Internet.

2. Following on from (1) the corresponding requirement for comprehensive steganography detection tools and methods.

3. The lack of steganography detection tools directed at HTML steganography, despite the suitability of HTML encoded webpages as a carrier for steganography.

1.3 Research Question

Given the above motivation the work presented in this thesis is therefore directed at techniques for detecting the presence of steganography in HTML files (HTML steganography). The overriding research question is thus:

"What are the most appropriate detection approaches, dynamic and static, required to efficiently and effectively detect hidden messages in HTML files?"

The term dynamic used here refers to the process of continuously monitoring webpages, while the term static refers to the process of applying steganography detection techniques to webpages in an offline manner.
To provide an answer to the above research question the following research sub-questions need to be considered:

1. What are the features of steganography that can best be adopted to facilitate detection?
2. How can steganography detection effectiveness be measured so that different techniques can be assessed and compared to one another?
3. Given a set of webpages how should the content of those webpages be represented so as to facilitate, dynamic or static, steganography detection?
4. Given a solution to the above, how can the performance of such techniques be best improved so as to maximize effectiveness?
5. What is the most appropriate mechanism for webpage monitoring in the context of dynamic steganography detection?
6. What is the effect on detection of the nature of messages that are hidden (English language or random)?
7. What is the effect on detection of the length of messages that are hidden?

1.4 Research Methodology

To provide an answer to the above listed research issues, and overriding research questions, the start point for the work presented in this thesis, was to review established methods for HTML steganography. From this review three HTML steganography techniques were identified: (i) Attribute Permutation Steganography (APS) [45, 80, 81] (ii) Invisible characters Steganography (ICS) [60, 91] and (iii) Tag Letters Case Switching Steganography (TLCSS) [87, 96]. Each of these is to be considered in turn and improved detection techniques (over existing techniques) identified. At the same time the proposed detection techniques will be considered in both the static (non-monitoring) and the dynamic (monitoring) contexts. To identify appropriate detection mechanisms the

broad strategy adopted is to consider mechanisms founded on the concept of machine learning [2, 41]. More specifically, classification mechanisms, whereby binary classifiers (stegnography exists versus does not exist) can be learnt from labeled training data where data records (webpages) had been seeded with hidden messages using identified HTML steganography techniques. Once learnt, the classifier could be applied to new, previously unseen webpages, and these webpages are classified according to whether steganography is suspected or not. There are a number of benefits that the classification approach offered which led to its adoption. Firstly, the approach is well established and well documented. Secondly, classification approaches have well founded mechanisms for establishing their effectiveness which could easily be adopted (including statistical significance testing). Thirdly, that there was very little previously reported work where classification has been used for HTML steganography detection. For the building of the classification models three classification algorithms are considered: (i) Neural Network (ii) Support Vector Machines and (iii) Naive Bayes; these were all taken from the WEKA machine learning toolbox [32].

Classification algorithms typically operate using a feature vector representation as the input. This means that webpages needed to be represented in this format so that classification algorithms can be applied. Thus work is required on how such representations could be generated given the different HTML steganography techniques available. The idea was to test the classification models with and without feature selection, although not in all cases. The "CfsubsetEval" attribute evaluation algorithm [3] together with a best-first search strategy are used.

For evaluation purposes Ten-fold Cross Validation (TCV) is typically used throughout. The evaluation measures used are average accuracy and AUC (Area Under the receiver operator characteristic Curve). To determine whether the results obtained are statistically significant a comprehensive statistical analysis of the results is conducted using Friedman test (followed by Nemenyi post-hoc test) for comparing several classification models (more than two) and Wilcoxon-signed rank test for comparing two related samples [23]. In this statistical analysis the effect on steganography detection of both the type and length of embedded hidden messages was also considered. Note that the "type" of a hidden message, as used here, refers to whether the message is in English language or some other form.

As already noted above, it is considered desirable to investigate steganography detection both in the dynamic (monitoring) and the static (non-monitoring) contexts by considering two different scenarios. In more detail:

- **Dynamic (Monitoring) scenario**: For the dynamic scenario a number of "copies" of a webpage is collected over sample intervals of time τ over a period of time T. A sequence of "webpage snapshots" is thus obtained. The scenario is designed to simulate the process whereby webpages might be monitored for the presence of steganography.

- **Static (Spot Check) (Non-Monitoring) scenario**: In this case only one snapshot per webpage is used for determining whether steganography is present or not. This scenario is designed to simulate the situation where a "spot check" is conducted.

To support the analysis a number of algorithms needed to be developed, whereby HTML steganography can be conducted, based on existing HTML steganography methods and techniques reported in the literature. These algorithms needed to be able to embed both English language and random text into selected HTML files, and messages of different length, so that a comprehensive analysis of the proposed steganography detection techniques could be conducted.

1.5 Research Contributions

The main research contribution of the thesis is a number of HTML steganography detection mechanisms and algorithms directed at different categories of HTML steganography. Figure 1.1 presents an overview of the identified HTML steganography detection techniques. In the figure intermediate nodes (square cornered rectangles) represent the identified HTML steganography categories and leaf nodes (round cornered rectangles) represent the individual HTML steganography detection approaches proposed in this thesis. In Attribute Permutation Steganography (APS) the embedding of hidden messages is performed through rearranging the attributes in HTML tags. For the attribute permutation detection category two detection algorithms were proposed: (i) Statistical Detection (SD) and (ii) Attribute Position Changes Count (APCC). For the Invisible characters Steganography (ICS) category, also known as open space steganography (OSS), the hidden message is inserted using invisible characters such as white space characters. In this case the frequency distributions of white space character segments was considered and the Detect Invisible characters (DIC) algorithm was proposed. In Tag Letters Case Switching Steganography (TLCSS) the embedding is performed by switching the tag letters case from uppercase to lowercase and vice versa. In this case the Tag Variance (TV) algorithm was proposed, an enhancement on a mechanism first presented in [44]. A further contribution is that this set of detection methods has been encoded and is available for download[1].

For reference the following list summarizes the steganography detection algorithms proposed in this thesis (the main contributions of the thesis). In each case the relevant chapter number, where each contribution is discussed in further derail, is included in parenthesis.

1. SD algorithm directed at APS (Chapter 3).
2. APCC algorithm also directed at APS (Chapter 4).

[1] https://sandbox.zenodo.org/record/167143

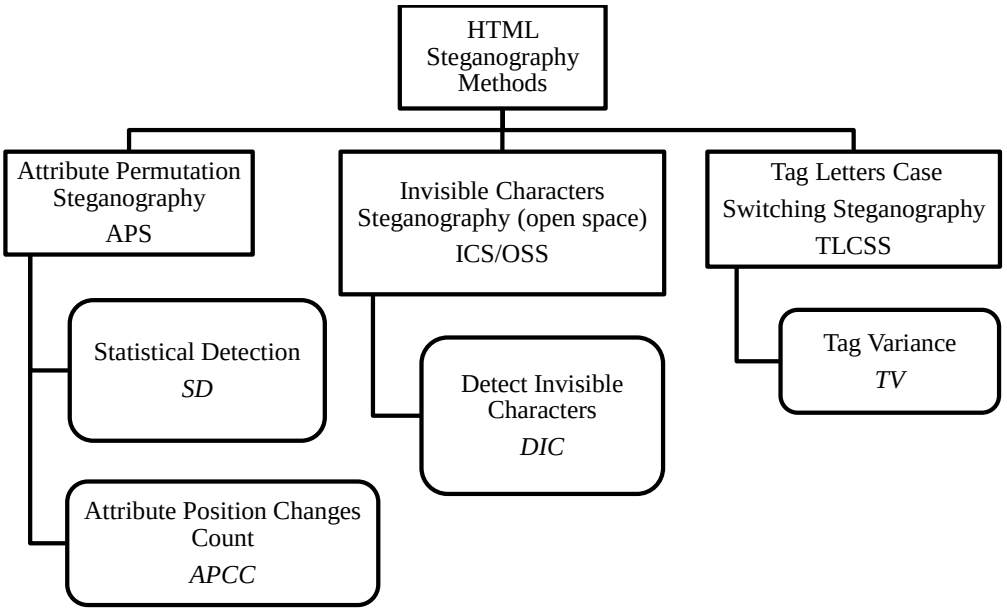


FIGURE 1.1: Categorization of HTML steganography methods and the associated detection approaches proposed in this thesis

3. The Detect Invisible characters (DIC) algorithm directed at ICS (Chapter 5).

4. The Tag Variance TV algorithm directed at TLCSS (Chapter 6).

As demonstrated later in the thesis, the proposed steganography detection approaches outperformed all known HTML steganography detection approaches (from the literature) in most cases. Note also that the proposed steganography detection methods can be used in combination to detect all three forms of identified HTML steganography.

1.6 Publications

Three peer reviewed publications have arisen out of the work presented in this thesis. These are listed below together with a brief description of each.

1. **Sedeeq Iman, Coenen Frans and Lisitsa Alexei (2016). "A Statistical Approach to the Detection of HTML Attribute Permutation Steganography", In Proceedings of 2nd International Conference on Information Security Systems and Privacy (ICISSP 2016), SCITEPRESS-Science and Technology Publications, pp. 522-527**. This paper proposed the first detection approach presented in this thesis, the SD approach directed at APS. The idea presented was to consider the standard deviation of the position of tag attributes in a webpage as an indicator of the presence (or otherwise) of steganography. The paper suggested that by monitoring a webpage, standard deviation values could be compared with a threshold value, and if they were over that threshold HTML steganography was identified. The content of this paper is included in Chapter 3.

2. **Iman Sedeeq, Frans Coenen and Alexei Lisitsa (2017). "Attribute Permutation Steganography Detection using Attribute Position Changes Count", In Proceedings of 3rd International Conference on Information Security Systems and Privacy (ICISSP 2017), SCITEPRESS-Science and Technology Publications, pp. 95-100, (winner of Best student paper prize)**. This paper proposed the APCC algorithm, the second technique, also directed at APS. The approach was found on the idea of an attribute position changes count measure. This count was used to generate a feature vector webpage training set that could be used to generate a classification model which could then be used to distinguish between normal webpages and stego webpages. Three classification models were considered: (i) Naive Beyes (NB), (ii) Support Vector Machine (SVM) and (iii) MutiPerceptron Neural Network (MLP). The content of this paper is covered in the context of Chapter 4.

3. **Iman Sedeeq, Frans Coenen and Alexei Lisitsa (2017). "A Prediction Model Based Approach to Open Space Steganography Detection in HTML webpages", Proceedings 16th International Workshop on Digital Forensics and Watermarking (IWDW17), Springer International Publishing IWDW 2017, LNCS 10431, pp. 235-247**. This paper proposed the Detect Invisible characters (DIC) algorithm; the third approach presented in this thesis and directed at the detection ICS. The idea presented was to use the frequency distributions of white space segments to define feature vectors similar to those considered in the previous paper. A training set was developed in this manner and classifiers were generated. The same three classification algorithms were considered as in the previous paper: (i) Naive Beyes (NB), (ii) Support Vector Machine (SVM) and (iii) MutiPerceptron Neural Network (MLP). Previously unseen webpages could then be classified as being either normal or stego webpages. The content of this paper is covered in Chapter 5.

The TV approach presented in Chapter 6 is as yet unpublished.

1.7 Structure of Thesis

The rest of this thesis is organized as follows:

- **Chapter 2**: Presents a review of the relevant literature, together with definitions of the terminology used in the domain of streganography. The chapter also provides a review of HTML steganography algorithms and current detection methods.
- **Chapter 3**: Provides a detailed review of the SD detection approach for APS in the dynamic (monitoring) context. This is where the concept of Standard Deviation of HTML attributes is introduced.

- **Chapter 4**: Presents the APCC detection approach also directed at APS in the static (non-monitoring) context. A range of classification model generators were considered and evaluated and the results reported and discussed in the chapter. Statistical significance analysis is also provided.

- **Chapter 5**: Considers the DIC detection approach. As in the case of the forgoing chapter evaluation was conducted using a number of classification models, with/without feature selection; the results are reported on and discussed in the chapter. Statistical significance analysis is again provided.

- **Chapter 6**: Presents the proposed TV algorithm to detect TLCSS. Note that the TV algorithm is a more effective variation of the tag letter case switching steganography detection algorithm presented in [44]. A comparison was thus conducted; the outcomes of this comparison are reported on in the chapter. Statistical significance analysis is also provided.

- **Chapter 7**: The concluding chapter of the thesis which presents a summary of the research presented, the main research findings and some suggested directions for the future work.

Chapter 2

Steganography Background and Previous Work

2.1 Introduction

In the previous chapter, Chapter 1, it was established that the work presented in this thesis is directed at investigating and proposing a number of HTML steganography detection methods. A review of the previous work related to the research presented in this thesis is thus given in this chapter. The chapter commences, Section 2.2, with a review of the historical evolution of steganography, since its foundation in the ancient era till today. This section also includes a classification of modern digital steganography: (i) Digital media steganography (ii) File system steganography and (iii) Network steganography. The chapter then continues, in Section 2.3, with a more detailed look at modern steganography and especially the terminology used. Section 2.4 then reviews the concept of steganalysis, its potential goals, methodologies and types of steganlysis. This is then followed by Section 2.5 which considers steganography in digital media (digital images and text files). Section 2.6 then reviews steganalysis in digital media (digital images and text files).

In some cases the HTML steganography detection techniques presented in this thesis are founded on a data mining (machine learning) approach. The data mining process is thus reviewed in Section 2.7 together with how a steganalysis model can be built for classifying webpages as either normal webpages or stego webpages. In Section 2.8 a comprehensive review of HTML steganography methods are presented (attribute permutation, invisible characters and tag letters case switching) in more details. Section 2.9 presents a comprehensive overview of detection approaches that have been presented in the literature directed at HTML steganography. The significance is that the operation of the steganography detection approaches presented in this thesis was compared with the operation of these approaches. The chapter then ends with a summary in Section 2.10.

2.2 History of Steganography

In ancient Greek steganography literally means "covered writing"; the more modern computer science interpretation is data concealed in other data. The usage of steganography, as a method for secret communication, dates back to antiquity. The Greek historian Herodotus (486-425 B.C.) reported examples of how steganography was used in ancient times. Figure 2.1 shows how the usage of steganography has evolved with time [100]. From the figure it can be seen that the earliest examples are those that Herodotus reported on, such as: (i) shaving a slave's hair and tattooing it with a message which became hidden when the hair had regrown (ii) hiding letters in the soles of shoes worn by "messengers" or in women's earrings and (iii) sending notes by carrier pigeons [31]. Arguably the most sophisticated method of that time was the practice of carving a message on a wood tablet and then covering it by a thick layer of wax to avoid arousing suspicion. The message could then be revealed simply by melting the wax [54].

As a result of humankind's endless desire for communication secrecy, new opportunities for covert communication arose with the widespread availability of parchment (made from animal matter) and the invention of invisible inks which become hidden upon drying. The message could be revealed using a heating process. Such inks were originally made from organic matter but progresses in chemistry provided for more advanced inks [17].

The invention of paper, by the Chinese, brought more opportunities for steganpgraphy. For example watermarking as an anti-counterfeiting and/or copyrighting technique to distinguish manufactures' products. Digital watermarking is now considered to be a separate discipline within the domain of information hiding. Another consequence of the invention of paper was the emergence and popularization of what is called linguistic steganography where document text was arranged in such a way that a hidden message was included. One example of linguistic steganography is where the first letter of each line in a piece of writing is used to convey a message [52]. Geometric drawings have also been used to hide messages. Examples also exist with respect to music scores where each note represents a letter, or the number of occurrences of individual notes is used as a letter indicator.

The European Enlightenment movement of the late 17th and 18th centuries and the later industrial revolution facilitated further opportunities for steganography; of particular note was the widespread readership of newspapers. Secret messages could be embedded using space over or under letters and changing the height of letter strokes (a further form of linguistic steganography).

In the 20th century the need for secrecy during periods of conflict (two World Wars) necessitated further mechanisms for secret communication as a consequence of which steganography began to truly flourish. That period witnessed the return to many forms of invisible ink, and textual or linguistic steganography methods but more importantly the usage of "spread spectrum" techniques in telecommunication and radio communication. In the case of the latter the spread of frequencies was used to embed information

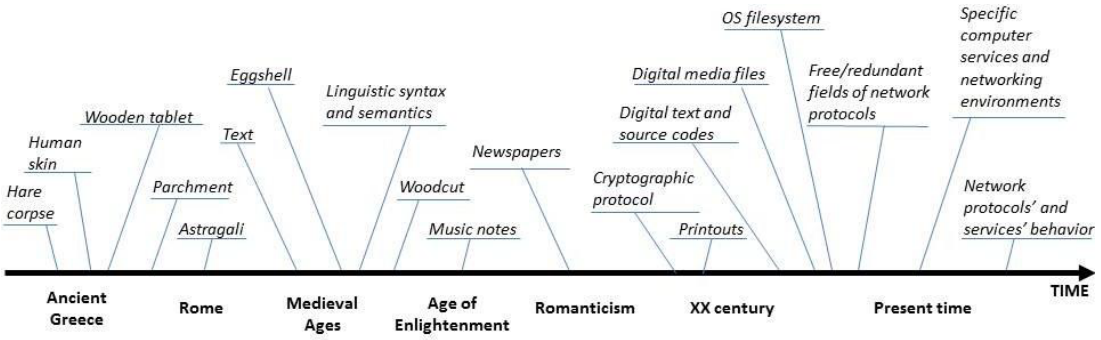


FIGURE 2.1: Steganography carrier evolution over time [100]

and scatter the signal so as to avoid detection. The advent of computer technology in the 20th century lead to further innovation in steganography techniques.

Modern (digital) steganography can be classified as follows:

- **Digital media steganography**: This category is concerned with steganography applied to digital media, such as: images, video, audio and text files. More details will be given later on this category in Section2.5.

- **File system steganography**: File System Steganography is steganography directed at the way that operating systems store files. For example using some operating systems (such as Windows 95) if the "cluster" size is 32KB and the size of a file is 1KB the remaining unused 31KB can be used to hide data [50]. Another form of file system steganography is by creating hidden partitions. These partitions can then be used to hide files that only a person who knows the name of the file and password can access [6, 72].

- **Network steganography**: This is the most recent form of modern digital steganography whereby network protocols are utilized to convey secret communications. Network protocols provide the two essential characteristics that a carrier of a hidden message should feature. Firstly these protocols are common and secondly they permit alterations with no noticeable visual behavior. Moreover using communication channel delays, damaged packets and retransmission for some packets are natural features that support hidden message passing without raising any alarm [67, 75, 89].

The work presented in this thesis falls into the Digital Media Steganography category. The remainder of the discussion presented in this chapter is therefore directed at this form of Steganography.

2.3 Steganography In The Modern Context

This section presents an overview of steganography in terms of the modern context. A steganography model is illustrated by "prisoners problem" proposed by Simmons and

shown in Figure 2.2 [82]. In which two prisoners, Alice and Bob, are kept in separated cells and wish to plan an escape, but are allowed to communicate via a (communication) channel Wendy the warden using some carrier object and what is known as a stego key. Alice and Bob therefore require a means of communication that will not draw Wendy's attention. In other words a form of steganography is required where Wendy is the channel.

In her attempts to discover whether steganography is taking place Wendy is also a steganalyst. Wendy could either be a "passive" or "active" steganalyser. Passive steganalysis is concerned with only detecting the presence of a hidden message, whilst active steganalysis is concerned not only with detecting the presence of steganography but also with extracting the message content and either destroying it or replacing it with something else.

The work presented in this thesis is concerned with passive steganalysis. The model proposed by Simmons, as described, is the fundamental model on which the work presented in the rest of this thesis is based.

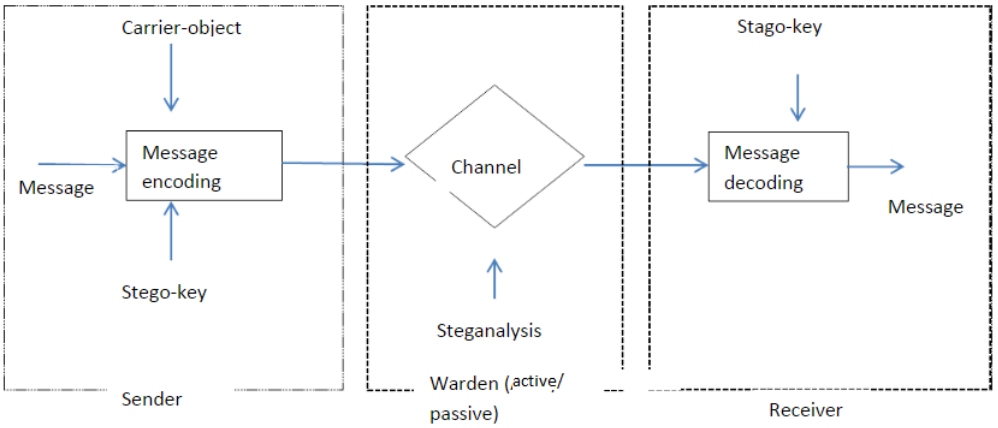


FIGURE 2.2: Steganography model [46]

The message which Alice wants to send is known also as the payload. Alice uses an encoding method to hide the message in the carrier object. The possible carrier object where a payload has already been hidden is a stego object. Alice will be considered the steganographer. Finally Bob uses the stego-key and a decoding method to extract the message. The usage of the stego-key in a steganography model can be thought of as being analogous to the use of encryption keys in cryptography.

Kertchof's principle [54], which holds that the secrecy of a hidden message lies on the choice of the stego-key shared between the sender and receiver, is therefore applicable in steganography. According to that there must be a prior steganography protocol that both Alice and Bob agree with. Whatever the case, steganography protocols, can be categorized according to whether a stego-key is used or not, and if so the nature of this key, as follows [5, 19, 27, 54]:

- **Pure steganography**: steganography where no stego-key is used, in other words nothing, apart from the embedding and extracting algorithms, is required to start

communication. The security of the steganography system in this case depends only on its secrecy. This is the case with respect to most modern steganography techniques including HTML steganography.

- **Secret key steganography**: steganography where a secret stego-key (known only to the sender and receiver) is used for embedding and extracting the hidden message. In this case the steganography system is similar to a cipher text system.
- **Public key steganography**: steganography where two keys one public and the other private are required. The public key is used to embed the hidden message while the private key is used to reconstruct the hidden message.

2.4 Steganalysis

As noted in Chapter 1, the term steganalysis and the phrase steganography detection, are both concerned with the process of detecting the presence of steganography in a carrier. Steganalysis is analogous to cryptanalysis as applied in cryptography. The main distinction between steganalysis and cryptanalysis is that the steganalysis process starts with suspicion but uncertainty as to whether (say) a file has a hidden message, whilst in cryptanalysis there is a prior evidence that a file contains an encrypted message. Referring back again to Figure 2.2 Wendy the warden's primary goal, as the steganalyist, is to provide an answer to the question "Is a hidden message presenting or not in this media?".

The central theme of steganalysis is the observation that the process of replacing redundancy in a carrier-object with an embedded message will change the statistical properties of that carrier object. Steganography detection is conducted by analyzing these statistical changes so as to uncover the presence of steganography [50].

When faced with a steganalysis task the attributes of the steganography algorithm (embedding capacity, invisibility, undetectability and robustness) should be considered first and weakness identified by the attacker.

The steganographer should also consider the nature of a potential attacker, human or computer? A steganography method that operates by introducing grammatical errors would be more susceptible to a human attacker than a computer attacker [11].

In this thesis steganography detection (steganalysis) for some cases is formulated as a supervised learning problem. More specifically, given that the thesis is directed at HTML steganography, where a labeled training set comprising a collection of stego and non-stego pages is provided with which to train a binary classification model. The labeled training set is used by a supervised learning algorithm to find a mapping function between the attributes that feature in the webpages and the class attributes. This mapping is then used to categorize previously unseen webpages as either stego or non-stego webpages. What the relevant attributes are, is a part of the research, and will be expanded upon later in the thesis.

The remainder of this section comprises two subsections. Firstly Sub-section 2.4.1 considers the challenges of steganalysis in terms of both passive and active steganalysis; particularly in the context of detecting HTML steganography. The huge number of WWW pages that are available is the first, and most obvious, challenge for the steganalyzer. Sub-section 2.4.2 then considers various categorizations of available steganalysis methodologies and how the proposed HTML steganography detection approaches presented in this thesis fit into this categorization.

2.4.1 Challenges of Steganalysis

This subsection considers the challenges of steganalysis. The challenges encountered in the context of passive steganalysis may be summarized as follows:

- In the case of HTML staganalysis scanning all webpages available over the Internet is not a realistic proposition.
- Hidden data can be either place in sequential (specific) locations or scattered throughout a carrier-object.
- Hidden data can be embedded in a chain over a number of carrier-objects.
- Steganalysis is a resource intensive and consequently time consuming.

The work presented in this thesis is broadly directed at all of these challenges, in the context of HTML steganography, by proposing a number of computer automated steganalysis techniques.

2.4.2 Categorization of Steganalysis Methodologies

In the foregoing steganalysis was categorized as being either passive or active according to whether the objective. In this section three alternative categorizations, taken form the literature, are considered. The first of these is according to whether the steganalysis methodologies of interest are targeted or universal as defined as follows [15]:

- **Targeted or Specific Steganalysis**: Steganalysis directed at the detection hidden messages that are embedded using a particular steganography technique. When the steganography algorithm is known, the presence of hidden data can be identified using knowledge of the way the information is embedded.
- **Universal (blind)**: Steganalysis conducted where the nature of the steganography algorithm is unknown. Thus these are general purpose stegnography detection methods [35].

Clearly the steganography detection rate in the case of targeted steganalysis is much higher than the detection rate using universal steganalysis methods because of the additional knowledge available of the targeted steganographic methods. The work presented in this thesis is directed at targeted (passive) steganalysis.

A second alternative steganalysis categorization found in the literature [11, 31, 53, 54] is according to what knowledge the attacker has, such as: the nature of stego object, original cover object, hidden message and the steganography algorithm.

Accordingly we can identify six types of attacks as listed in Table in 2.1. The table lists the attack type in column 1 and in the following columns what, in each case, the steganalyser (the attacker) is expected to know about the items (indicated by check marks).

TABLE 2.1: Attack types according to what available to the attacker

	Stego object	Original cover object	Hidden message	Stego algorithm or tool
`Stego only`	✓			
`Known cover`	✓	✓		
`Known message`	✓		✓	
`Chosen stego`	✓			✓
`Chosen message`	✓	See below for detail concerning available knowledge		
`Known stego`	✓	✓		✓

From the table it can be noted that "stego only" attack is the hardest while "known stego" attack is the easiest. In case of "chosen message " attack the attacker can generate the stego objects using different hidden messages and different embedding algorithms. This attack is used to recognize signatures that help the detection of other stego objects.

The final categorization considered in this subsection, and one of particular relevance with respect to the work presented in this thesis, is that of steganalysis methods founded on machine learning approaches and those that are not. Because of the relevance with respect to the work presented in this thesis this categorization is considered in more detail later in this chapter.

Thus in conclusion the steganalysis methodology adopted with respect to the stag-analysis approaches presented in this thesis can be said to be passive, targeted, chosen stego machine learning based methods.

2.5 Digital Media Steganography

As mentioned in Section 2.2 the category of digital media steganography will be presented in more details in this section. This category of steganography is applied to digital media. The redundancy in these digital media types can easily be replaced by hidden information. This category can also include watermarking although watermarking is not strictly a steganography technique. Moreover these media types are popular on the Internet so using them as a cover communication channel will not raise a suspicion.

The remainder of this section comprises two subsections. Firstly Sub-section 2.5.1 presents digital image steganography in particular because it is the most widespread research area. Secondly Sub-section 2.5.2 presents steganography in text file, the significance is that HTML steganography can be argued to be a special form of text steganography.

2.5.1 Digital Image Steganography

Digital images are the most popular multimedia type that attracted steganographers. Some of reasons that motivated researchers to utilize digital images for steganography are:

- The abundance of digital images on the Internet.
- Some of Human Visual System HVS attributes motivate researchers to exploit these attributes in data hiding systems design.
- This media type provides enough redundancy to manipulate steganography.

Some of digital image steganography applications are: (i) Copyright (ii) Image integrity (fraud detection) and (iii) Adding captions and (iiii) Fingerprinting (trailer-tracing). In literature the common classification of image steganography methods [17, 20, 51, 63] is based on the category of the domain that embedding process is taken place in as follows:

- `Spatial domain steganography methods:` in these methods, embedding hidden messages in a cover image will occur in spatial domain. The Least Significant Bits LSB_s of image pixels are involved with encoding of a secret message [37, 62, 74, 98]. This category also includes applications for watermarking in binary document images [14, 66].
- `Frequency domain steganography methods:` modification of LSB_s in spatial domain techniques did not succeed in resistance to attacks and deceiving human visual system. To overcome aforementioned shortcoming, modification of LSB_s in frequency domain is adopted [47, 64, 76, 94, 97].

2.5.2 Steganography in Text Files

Text is one of the oldest choice of media for performing steganography. The abundance of textual information on the Internet has encouraged steganographers to use this media as a carrier for hidden messages.

There are many data hiding applications related to text files. In the context of copyright protection, as already noted, text watermarking might be used [4]. Steganography can be used as well to add a "hash" to a text file to protect the file from tampering [28]. The lack of redundancy in text files, compared to other forms of digital media makes steganography using text files a challenge. There are many different kinds of text steganography: a categorization of common text steganography methods is given in both [11] and [21] where the following three broad categories are identified: (i) format based methods that utilize existing text characteristics to hide data, (ii) random or statistical generation methods where steganographers create their own text file, in which to hide data, in such a way that it is statistically "natural" (frequencies of letters and words), and (iii) linguistic methods that include syntactic and semantic methods.

In [58] the following alternative text steganography categorization is suggested:

- `Technical steganography`: Methods that used technical processes to hide data such as invisible ink.

- `Linguistic steganography`: Methods that hide data in non obvious ways. The category can be subdivided into two sub-categories: (i) semagrams and (ii) open codes. The semagrams sub-category includes text steganography methods whereby data hiding is performed using the text itself, such as adding extra space characters, line shifting, word shifting, deliberate misspelling and font resizing. The relevance is that this is also applicable with respect to HTML stegnography. The open codes sub-category is where data hiding is performed either using Jargon code, a language understood by a select group and meaningless to others, or using covered ciphers to hide data so that only someone who knows how the message was concealed can recover it. The simplest method is null cipher method whereby (say) a prearranged rule must be followed in order to embed and extract the hidden message. If the rule is "read every first letter" then the set of first letters of each word, in an innocuous piece of writing will represent the hidden message.

Another important branch of text steganography involves Natural Language Processing (NLP) for the purpose of identifying synonym substitution [16], word abbreviation and paraphrasing [28] as a means of conveying hidden messages. Also a text file can be treated as an image and image steganography methods can be applied for watermarking purposes as in [70, 99].

Figure 2.3 presents a total of 15 text steganographic tools, some of them are used to embed secret messages in HTML files. Of these tools only wbStego and SNOW were freely available to be used for evaluation of the proposed approach DIC for detecting invisible characters steganography ICS, described in Chapter 5.

2.6 Steganalysis over Digital Media

In this section steganalysis over digital images and text files will be present. This is as mentioned before that digital images are the most widespread research and HTML steganography can be argued to be a special form of text steganography. The remainder of this section comprises two subsections. Firstly Sub-section 2.6.1 presents steganalysis over digital images. Secondly Sub-section 2.6.2 presents steganalysis over text files.

2.6.1 Steganalysis over Digital Images

As mentioned earlier that steganalysis aims to determine whether a testing object is a clean object (has no hidden message) or a stego-object (has a hidden message). Accordingly steganalysis can be considered as a two-class pattern classification problem. Also it is mentioned in Sub-Section 2.4.2 that steganalysis is classified into targeted/specific and universal/blind steganography. Some applications of these two categories in literature are [63]:

Text Steganographic Tools	Plain Text	Other	Source Code	License	Production
PGPn123		Yes		Shareware	Yes
Nicetext	Yes		Yes	Open Source	Yes
Snow	Yes		Yes	Open Source	Yes
Texto	Yes		Yes	Open Source	Yes
Sam's Big Play Maker	Yes		Yes	Open Source	Yes
Steganosaurus	Yes		Yes	Open Source	Yes
FFEncode	Yes			Open Source	Yes
Mimic	Yes			Open Source	Yes
wbStego	Yes	HTML, PDF	Yes	Open Source	Yes
Spam Mimic	Yes			Not Specified	Yes
Secret Space	Yes			Not Specified	Yes
WitnesSoft	Yes	Yes		No longer in production	
MergeStreams		Hides excel file in word		Freeware	Yes
Steganos	Yes	HTML		Commercial	Yes
Invisible Secrets		HTML		Commercial	Yes

FIGURE 2.3: Text steganographic tools [42]

- For targeted/specific steganalysis: These methods take advantage of the insecure aspect of a steganographic algorithm such as (i) Attacking LSB steganography in spatial domain as in [34, 77, 93] (ii) Attacking LSB steganography in frequency domain as in [36].

- For universal/blind steganalysis: These methods have less or even no such prior information about steganography algorithm. These methods are based on two stages training and testing. During the process, a feature extraction step is used. The aim of the training stage is to obtain a trained classifier. Many models can be used such as support vector machines SVM and neural network (NN), etc. In literature many applications can be found depending on the features used (i) Image Quality Feature [8] (ii) Calibration based feature [33] (iii) Moment based feature [83] (iiii) Correlation based feature [88].

2.6.2 Steganalysis over Text files

In text files a tokenization process is required and the punctuation marks are removed to get a stream of words as a preprocessing step for steganalysis. Some applications need a dictionary preparation that includes all words of a collection. This dictionary might be reduced to the words of high entropy which refer to the words that are important in a given domain. Keywords that are identified the document content of a particular class can be used as features. These keywords are the words that are repeated frequently in the document [18, 101].

2.7 Data Mining

As already established earlier in this categorization, the fundamental approach to steganography analysis adopted in this thesis is one founded on data mining; a branch of machine learning. Data mining can be defined as the extraction of interesting and useful information by exploration and analysis of large quantities of data. It is an essential step of the Knowledge Discovery in Databases (KDD) process for extracting useful, but hidden, knowledge from datasets of all kinds. Machine learning has been a computer science domain of research since the 1970s, data mining, as a domain of research, was established more recently in the 1990s [2, 41].

The KDD process is traditionally considered to comprise the following steps:

1. **Domain knowledge acquisition**: In order to know how to utilize data mining results a proper understanding is required of the problem domain. The domain of interest with respect to this thesis is HTML steganography.

2. **Data selection**: In this step the data miner decides the nature of the data to be used. In the context of the thesis the data of interest are HTML webpages, which may or may not contain hidden messages.

3. **Data preprocessing**: This step typically comprises several tasks directed at the identified data from Step 2, including: data cleaning, data pruning and data transformation. The data preprocessing stage may also include data integration and feature selection. Data integration is where the selected data has been collected from more than one resource and consequently needs to be combined. Feature selection is where a subset of the available features (attributes) are selected and retained; the aim being to work with only the features "best" suited to producing a good data mining result. Feature selection has a number of claimed advantages [2, 12]; with respect to classification these may be itemized as follows:

    - Reducing the dimentionality of the feature space, thus reducing storage requirements and training time.
    - Selecting the most relevant features for building the desired classification model.
    - Improving the classification model accuracy.
    - Reducing overfitting.

    In this thesis the proposed approach DIC for detecting invisible characters steganography ICS, described in Chapter 5 used the wrapper feature selection method (Appendix C). This form of feature selection adopts a two stages approach: (i) choosing the evaluation method by which the feature subsets are to be assessed and (ii) choosing the search method by which the feature subset space is searched. For proposed DIC approach the selected evaluation method was "CfsubsetEval".

For the search method a bestFirst search approach was adopted where the search is conducted using a greedy hill-climbing search method with backtracking. This search method either starts with an empty features set and searches forward, or with full features set and searches backward.

4. **Data Mining**. This is the central element of the KDD process where the useful knowledge hidden in the data is extracted. In the context of the thesis we are interested in identifying patterns and relationships that can be used to determine the presence of HTML steganography. There are a variety of data mining models (descriptive and predictive) [48] that can be adopted depending on the desired outcome.

5. **Evaluation of patterns discovery**: In this step an evaluation of the identified patterns and/or relationships from step 4 is conducted. The aim is to gain confidence in the discovered patterns and relationships. The evaluation mechanism considered in the context of this thesis is discussed in Sub-section 2.7.2 below.

6. **Discovered knowledge deployment**: In this final step the discovered knowledge is utilized. In the case of the thesis this is HTML steganography detection (steganalysis).

The work presented in this thesis falls into the predictive, supervised, learning category where a pre-labeled training set to build a predictor is used. More specifically the work is directed at building binary classifiers that can be used to "predict" whether a given webpage features steganography or not. The process of building, evaluating and using a classifier is therefore discussed in further detail in the following Sub-sections.

2.7.1 Classification

A classification problem is where we wish to identify the membership of an object with respect to a given set of categories called classes. If we have only two classes we have a binary classification problem. Thus the classification problem can be formulated in the form of a tuple $\langle X, Y \rangle$, where X is a set of attributes and Y is a set of classes (also known as the target attributes). While the attribute set might be discrete and/or continuous the classes must be discrete. Regression can be used where the classes are not discrete.

Classification has many diverse applications, examples include: the classification of human tissue cells as being malignant or benign and the classification of galaxies according to their type. Classification operates by first generating a classifier. In the case of supervised classification this is done using a pre-labeled training set. The nature of the training set is therefore important. Evaluation of the classifier is conducted using a pre-labeled test set; the training and test set should not be the same.

There are many algorithms available that can be employed to build the desired classifier. In this thesis the following are used: (i) Naive Bayes (NB) (ii) Support Vector Machines SVM and (iii) Neural Network (NN) in WEKA machine learning software.

These were selected because they are among the most commonly used in the literature and because they operate in very different ways.

2.7.2 Classifier Evaluation

In order to evaluate the outcome of a binary classifier (a two classes problem) there are a number of performance measures that can be adopted. The majority can be generated using what is known as a *confusion matrix* of the form shown in Figure 2.4. A confusion matrix is a table that compares class values predicted by a classifier and true observed values. Using the counts True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) held in the confusion matrix we can determine a variety of classifier performance measurements. The most frequently used are:

		Bench Mark		
		X	Not X	
Classifier	X (Positive)	True Positives (TP), records of class X that were correctly classified as X	False Positives (FP), records of class Not X that were incorrectly classified as X	Total Positive classifications
	Not X (Negative)	False Negatives (FN), records of class X that were incorrectly classified as class Not X	True Negatives (TN), records of class Not X that were correctly classified as class Not X	Total Negative classifications
		Total number of records belonging to class X	Total number of records belonging to class Not X	Total number of records

FIGURE 2.4: Confusion matrix for a binary classifier

- **Accuracy**: The number of correct predictions divided by the total number of predictions. It is an overall measure of a classifiers performance. The higher the accuracy the better. Accuracy, using a confusion matrix, is calculated as follows:

$$Accuracy(Acc) = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.1}$$

- **Sensitivity**: Also known as recall or the true positive rate (TPR) of the positive class label (class X in Figure 2.4), is calculated as follows:

$$Sensitivity(Sens) = \frac{TP}{TP + FN} \tag{2.2}$$

- **Specificity**: Also known as the false positive rate of the negative class label (class not X in Figure 2.4), is calculated:

$$Specificity(Spc) = \frac{FP}{FP + TN} \tag{2.3}$$

- **Area Under Curve (AUC)**: AUC is a combination of sensitivity and specificity in one single metric. Different thresholds ($\{0.0, 0.01, 0.02, \dots\}$) are used to calculate and plot sensitivity and specificity on a graph with specificity along the X-axis

and sensitivity along the Y-axis. The resulting curve is known as a ROC curve. AUC is then the area under the ROC curve as shown in Figure 2.5. The closer the ROC curve is to the northwest corner (in the figure) the higher the AUC. A classifier with an AUC of 1.0 will be a perfect classifier.

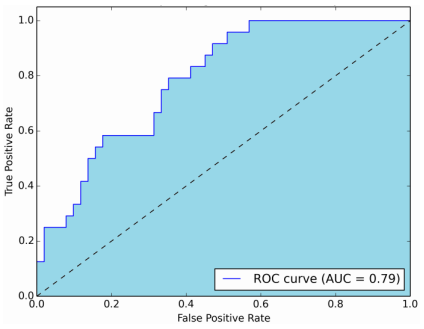


FIGURE 2.5: ROC example with $AUC = 0.79$ [24]

2.8 Methods of HTML Steganography

The work presented in this thesis is directed at three forms of HTML steganography:

1. Attribute Permutation Steganography (APS) [45, 80, 81].
2. Invisible characters Steganography (ICS) [60, 91].
3. Tag Letters Case Switching Steganography (TLCSS) [87, 96].

In each case a variety of methods are available to allow their implementation. Those used with respect to the work presented in this thesis are discussed in the following three subsections, Subsections 2.8.1 to 2.8.3.

2.8.1 Attribute Permutation Steganography APS

Attribute Permutation Steganography (APS) uses the fact that HTML is insensitive to how tag attributes are ordered. Thus in the following example the HTML "img" tag has four attributes: src, alt, width and height.

<img src="w3schools.jpg" alt="W3Schools.com" width="104" height="142">

Thus there are 24 ($4 \times 3 \times 2 \times 1$) different ways in which the attributes can be ordered, enough to encode the most common letters in the English alphabet. The browser however will render all the images in the same way regardless of the attribute ordering. Also the rearranging of the attribute ordering will not cause any change to the overall size of the HTML file, it will be the same regardless of the attribute ordering. Three different APS mechanisms for implementing APS are considered in this thesis: (i) Deogol [80], (ii) Huang et al. [45] and (iii) Shen et al. [81]. These were selected because they are referenced in the literature. Each is discussed in further detail below. With respect to the discussion the following definitions should first be noted:

- H: An (HTML encoded) webpage.
- M: A text message to be hidden in H.
- N: A single (large) number made of the ASCII codes representing M.
- T: A tag with its set of attributes in H such that $T = \{a_1, a_2, a_3, \ldots a_m\}$, where m is the number of the attributes in it ($|T| = m$).
- Q: A set of tags in H such that $|T| \geq 2$.

The mechanisms of Deogol and Huang et al. are approximately the same, the only difference is in how the attribute permutation is conducted. Algorithm 1 shows the pseudo code for the APS method of Deogol and Huang et al. The input to the algorithm is a webpage H and a message M to be embedded in the webpage. The output is a stego webpage H'. In Huang et al. step 10 is performed using a lexical ordering method (A-Z) to generate the permutations. In the case of Deogol the specifics of the permutation generation were not given and thus, in the context of the thesis, reverse lexical ordering was used (Z-A).

Algorithm 1: Attribute Permutation Steganography [45, 80]

```
1: Input a webpage H
2: Input a message M
3: Output= a stego webpage H'
4: N=large number made of the ASCII codes representing M.
5: M=N
6: Find Q set
7: while M ≠ 0 do
8:    for each tag T ∈ Q do
9:       M'=M div m!       (m is the number of T attributes)
10:      p=M mod m!        (p is a number between 0 and m!-1)
11:      Transform p to a permutation
12:      Replace T with the new permutation
13:      M=M'
14:   end for
15: end while
```

The mechanism of Shen et al. [81] used an alternative approach to APS to that utilized in Deogol. and Huang et al. Using the approach of Shen et al. a binary relation between the tag attributes is translated to a binary string. Before describing the approach in further detail the following definitions should be noted, in addition to the previously presented definitions:

- SubMessage: A function used to find a sub-message sm in M such that the length of $|sm| = |T| - 1$.
- BinaryString: A function to transform $|T| - 1$ attributes into a binary string BS.

- G: A set of BS permutations.
- Trunc: a function that "cuts" sm from M.

The pseudo code for the APS method proposed of Shen et al. is given in Algorithm 2. As before the input to the algorithm is a webpage H and a message M to be embedded in H. The output is a stego webpage H'. The algorithm operates by first fetching sm and generating BS to represent the relationship between tag attributes (line 8). Note that BS is generated using a dictionary ordering of the attributes first letters such that if the first letter of attribute a_1 is alphabetically before the first letter of attribute a_2 or they are the same then the bit in BS is 0, otherwise the bit is 1. In line 9 the set G, that includes all the permutations of BS, is generated. Any permutation in G that equals sm will be used to replace T.

All three APS methods were used with respect to the work presented in this thesis. Although not included in Algorithms 1 and 2 a check for the Maximum Embedding Capacity (MEC) of a webpage is required before embedding can take place. The MEC is the maximum hidden message length, in bits or bytes (assuming a one byte character encoding), that a webpage H can carry. This is an essential step to test that the webpage H can be a container for the hidden message or not. For Algorithm 1 the MEC, MEC_1, can be calculated using Equation 2.4; whilst for Algorithm 2 the MEC, MEC_2, can be calculated using Equation 2.5. Note that the MEC values resulting from equations 1 and 2 are in bytes. According to Equation 2.4 and 2.5 MEC provided by approaches of Deogol and Huang et al. is greater than MEC provided by the approach of Shen et al.

Algorithm 2: Attribute Permutation Steganography [81]

```
1:  Input a webpage H
2:  Input a message M
3:  Output= a stego webpage H'
4:  Find Q set
5:  while M ≠ 0 do
6:    for each tag T ∈ Q do
7:      sm=SubMessage(m, |T| − 1)
8:      BinaryString=transform |T| − 1 attributes to a binary string BS
9:      Find G a set of BS permutations
10:     p=permutation in G that equals to sm
11:     Replace T with p.
12:     M=Trunc(M,sm)
13:   end for
14: end while
```

$$MEC_1(H) = \frac{1}{8}\lfloor \log_2 \prod_{T_j \in Q} (|T_j|!) \rfloor \tag{2.4}$$

$$MEC_2(H) = \frac{1}{8} \sum_{T_j \in Q} (|T_j| - 1) \tag{2.5}$$

2.8.2 Invisible Characters Steganography ICS

When a web browser renders an HTML webpage it does not pay attention to invisible characters such as: white space (ASCII 0x20) and tab (ASCII 0x19) characters. The Invisible Characters Steganography (ICS) approach makes use of this feature to hide messages in HTML files with no visual alteration in the rendered webpage. This kind of steganography is known also as Open Space Steganography (OSS). The idea was first proposed in the mid-1990s [10]. Although embedding white space characters in a text file will increase the file size, these methods offer the advantage that in any document white space characters will appear frequently, more than any other character, therefore the existence of additional white space characters is unlikely to cause suspicion. In addition, and subject to how the ICS is applied and how the cover text file is viewed, in many cases the inclusion of additional white space characters will not result in any noticeable change in the look of the file from the viewer's perspective. Even where additional white space characters are visible, an observer is unlikely to pay significant attention to their presence and is unlikely to consider these spaces to represent a hidden message. ICS uses one or more of the following i.e to hide messages: (i) inter-word spacing i.e spacing between two successive words; (ii) inter-sentence spacing i.e spacing between two successive sentences, (iii) additional spaces at the end of lines; and (iv) inter-paragraph spacing, spacing between two consecutive paragraphs. An example of using white space characters to hide a message is shown in the following:

<tag>, </tag> sending "0"

<tag >, </tag > sending "1"

In this example when tags are written without inserting a space character this sends a 0, while inserting a space character sends a 1 (assuming that the messages to be hidden is in binary form).

As in the case of APS there are many tools available for ICS. Examples include: SNOW [60], Spacemimic [73], wbStego4open [91] and WhiteSteg [69]. The two selected for use with respect to the work presented in the thesis were SNOW and wbStego4open; these were selected because they are freely available, frequently referenced in the steganography literature and because of their ease of use.

Using SNOW the message is embedded using white space characters at the end of lines. When the host file is too short to accommodate the message SNOW embeds the remaining message at the end of the host file. Before hiding, each character in the message is encoded and written as a triplet (3 bits at a time) by introducing 0 to 7 spaces, the tab character is used as a delimiter for these triplets.

Using the wbStego4open steganography tool, white space characters are inserted between words and sentences. Using wbStego4open a space character is used to embed a 1 and a tab character to embed a 0. Unlike SNOW, before hiding the message wbStego4open checks if the cover file is large enough to accommodate the message.

2.8.3 Tag Letters Case Switching Steganography TLCSS

As mentioned before that HTML tags and attributes are case insensitive. This feature is used in Tag Letters Case Switching Steganography (TLCSS) to hide messages in HTML files. An example is given in Figure 2.6.

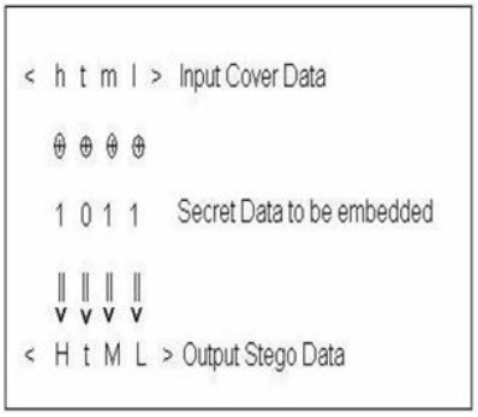


FIGURE 2.6: HTML steganography using tag letters case switching [25]

Both algorithms of Sui and Luo in [87] and Shen Y.[96] present approaches for conducting TLCSS. The following definitions should be noted before the pseudo code for the aforementioned steganography approaches are given:

- H: An (HTML encoded) webpage.
- M: A binary encoded text message to be hidden in H. $M = \{m_1, m_2, m_3, \ldots m_n\}$.
- Q: A set of tag names in H.

The pseudo code for the method adopted in Sui and Luo [87] is shown in Algorithm 3. The input to the algorithm is a webpage H and a message M to be hidden, the output is a stego webpage H'. In the algorithm, if a message bit m_i is equal to 1 then the tag letter is switched to uppercase otherwise nothing is done. All tag letters are utilized to hide the message. The approach presented in Shen Y. [96] is very similar to that presented in Algorithm 3 except that only the first letter of each tag is used.

Again, although not included in Algorithm 3 a Maximum Embedding Capacity (MEC) check is required before embedding can take place. For the approach of Sui and Luo in Algorithm 3 this can be calculated using Equation 2.6, while for the approach of Shen Y.[96] it can be calculated using Equation 2.7. Note that the MEC values resulting from equations 2.6 and 2.7 are in bytes. It should also be noted that Q is a set that contains the tags that are included in the given webpage H, and that $|T_j|$ here refers to the number of English alphabet letters in tag T_j.

$$MEC_1(H) = \frac{1}{8} \sum_{T_j \in Q} |T_j| \tag{2.6}$$

Algorithm 3: Tag letters case switching of Sui and Luo

```
Input a webpage H
Input a message M
Output= a stego webpage H'
Find Q set
while M ≠ 0 do
  for each tag T ∈ Q do
    if m_i = 1 then
      Switch a tag letter to uppercase
    else
      No switching
    end if
    M = M − m_i
  end for
end while
```

$$MEC_2(H) = \frac{1}{8} * |Q| \tag{2.7}$$

Unlike in the case of ICS, and like APS, the usage of TLCSS will not cause any HTML file size to increase. However, the irregular use of upper and lower case used in the tags can be readily detected from the raw HTML source code and therefore arouse suspicion.

2.9 HTML Steganography Detection

HTML steganograohy detection approaches can be categorized according to whether they are used in a dynamic (monitoring) context or a static (non-monitoring) context. The two scenarios were briefly reviewed in Chapter 1. An alternative, and compatible, categorization is according to the nature of the HTML steganography. As noted in the previous section three types of HTML steganography are considered in this thesis: (i) Attribute Permutation Steganography (APS), (ii) Invisible characters Steganography (ICS) and Tag Letters Case Switching Steganography (TLCSS). In this penultimate section of the literature review chapter the previous work on HTML steganography detection is reviewed. The section is divided into three subsections corresponding to the three types of HTML steganography considered in this thesis.

2.9.1 Attribute Permutation HTML Steganography Detection

In the literature there are two notable APS detection techniques that have been proposed, that of L.Polak and Z. Kotulski [65] and that of W.Jian-feng et al. [95]. Both are considered here, and both techniques are considered for the purpose of evaluating the APS detection techniques proposed later in this thesis in Chapters 3 and 4.

The work presented of L.Polak and Z. Kotulski falls into the static APS detection category and is based on the idea of identifying a predominant attribute pair ordering.

For each pair of attributes the detection algorithm counts how many times the first attribute appears before the second and vice-versa; the assumption is that a non-stego webpage will have a more consistent ordering than in the case of a stego-webpage. The ordering distribution is encapsulated using a value of W. More formally, given a set of m webpage tags $T = \{t_1, t_2, t_3, \dots t_m\}$ and the knowledge that each tag t_i has k_i occurrences. Each tag occurrence $t_{i,j}$ has a set of attributes associated with it $A_{i,j} = \{a_1, a_2, \dots\}$ where i is the tag identifier and j is the occurrence number of the tag ($j \leq k$). If tag t_i has a pair of attributes a_x and a_y ordered in such a way that a_x proceeds a_y in most instances of t_i then the value $R_{ij}(a_x, a_y) = 1$ otherwise the R value is 0. If it so happens that the distribution of both attribute pairs a_x, a_y and a_y, a_x is equal, $R_{ij}(a_x, a_y) = 1$ as well. The W value is then calculated using Equation 2.8 which calculates the fraction of the a_x and a_y attribute pairs that occurs in different orderings from the total number of pairings C.

$$\forall x \forall y \; x < y : W = 1 - \frac{\sum_{n=1}^{m} \sum_{i=1}^{k} R_{n,i}(a_x, a_y)}{C} \tag{2.8}$$

When the value of W is high this indicates that the attribute pair a_y and a_x occurred more often than the attributes pair a_x and a_y which is different from the predominant order. This denotes a changing of webpage structure. Thus W can be used as an indicator of APS given some threshold value.

The work presented of W.Jian-feng et al. in [95] falls into the dynamic APS detection category. This work, as in the case of some of the work presented later in this thesis, also falls into the supervised learning category. Similarly the work presented in [95] is based on the idea of learning a classification model that can be used to predict the presence or absence of APS. SVM classification was adopted, although any other form of classification model generator could equally well have been used. Feature vectors were generated using two statistics: (i) the distance between the attribute mean position benchmark and the sample attribute mean position of the suspicious webpage and (ii) the variance of attribute positions. In [95] three parameters were used with respect to the selected attributes: (i) the number of attributes to be considered (BA), (ii) the number of attribute appearances in terms of mean distance (TD) and (iii) the number of attribute appearance in terms of variance (TV). The following parameter settings were recommended in [95]: (i) BA = 3, (ii) TD = 3 and (iii) TV = 3.

The approaches of both L.Polak and Z. Kotulski [65] and W.Jian-feng et al. were used for evaluation purposes with respect to the work presented later in this thesis in Chapters 3 and 4.

2.9.2 Invisible Characters HTML Steganography Detection

As in the case of APS detection, there are a variety of mechanisms that have been proposed in the literature for the purpose of ICS. Two mechanisms found in literature

of Sui and Luo[68] and Huang et al. [43]; both are directed at static detection since they used a single webpage snapshot to determine if steganography exists.

The idea presented by Sui and Luo in [68] is based on the usage of a probabilistic model to detect whether additional invisible characters have been inserted into an HTML file or not. Two occurrence probability values were used for this purpose: (i) the probability of a white space character occurring (p_{tsco}) and (ii) the probability of white space character sequence occurrence (p_{scso}). The first is calculated using Equation 2.9 where W is a webpage, while the second is calculated using Equation 2.10.

$$p_{tsco}(W) = \frac{N_{ws}}{N_{allchar}} \tag{2.9}$$

$$p_{scso}(W) = \frac{N_{wss}}{N_{ws}} \tag{2.10}$$

where:

- N_{ws}: Is the number of white space characters in a webpage W.
- $N_{allchar}$: Is the number of all characters in a webpage W.
- N_{wss}: Is the number of space character sequences in a webpage W.

These probabilities were compared with predetermined thresholds to decide whether a given webpage was a normal webpage or a stego-webpage. The thresholds in this case were identified using Zipf's and Heaps' law, and a Finite-State Model [78]. The authors estimated that the probability of occurrence of a white space character (p_{tsco}) in a text file was approximately 0.2 ± 0.1 (varying range), a threshold of 0.3 was thus proposed. The authors also estimated that the total white space character sequence occurrence probability (p_{scso}), in a text file, was 0.2.

The work presented by Huang et al. in [43] was founded on the concept of "embedding rate" (e_{rate}). This is the ratio between the length of a hidden message M and the size of a given WWW page ($W_{allchar}$). M = size of a given webpage ($W_{allchar}$) - size of a given webpage with invisible characters removed ($W_{without}$). Embedding rate of a given webpage ($e_{rate}(W)$) can be calculated using Equation 2.11. The authors defined the normal distribution of the embedding rate using the mean (μ) and standard deviation (δ) of the e_{rate} distribution to define a threshold with which to distinguish normal webpages from stego-webpages.

$$e_{rate}(W) = \frac{M}{W_{allchar}} \tag{2.11}$$

The approaches proposed by Sui and Luo [68] and Huang et al. [43] were both utilized for evaluation purposes with respect to the proposed ICS technique presented later in this thesis (see Chapter 5).

2.9.3 Tag Letters Case Switching Steganography Detection

In the case of TLCSS the work of Huang et al.[44] is another static detection approach found in literature considered in this thesis. The idea presented by Huang et al. is based on the idea presented in [35] where image steganography is considered. More specifically the idea of considering the Least Significant Bits LSB_s in grayscale images to detect the presence of hidden messages. The fundamental approach to TLCSS detection in [44] is founded on the observation that switching the tag letters case will break the inherent "smoothness" of the tag content. Tag letters will be a mixture of lowercase letters (the set $c = \{26\ lowercase\ ASCII\ characters\}$) and uppercase letters (the set $C = \{26\ uppercase\ ASCII\ characters\}$). The detection approach of Huang et al. used the Tag Offset concept. This is the absolute value of the summation of the distance between two adjacent letters in a tag. This Tag Offset is computed using the following function F:

$$F(T(x_1, x_2, ..., x_n)) = \sum_{i=1}^{n-1} |A(x_{i+1}) - A(x_i)|, \tag{2.12}$$

Where $T(x_1, x_2, ...x_n)$ is an HTML tag. $A(x_i) \in (c' \cup C')$ such that:
set $c' = \{lowercase\ ASCII\ codes\}$ and set $C' = \{uppercase\ ASCII\ codes\}$. Hiding data by switching tag letters will increase the Tag Offset, in other words the discrimination F value (from Equation 2.12) will increase.
Switching tag letters can be performed using the following two flipping functions and the mask M :

- Positive flipping f_{+1} :
  - $f_{+1}(x)$ switches the lowercase→ uppercase.
- Negative flipping f_{-1} :
  - $f_{-1}(x)$ switches the uppercase → lowercase.

Where M is a vector of length n comprised of the values $+1$, -1 and 0. There will be $M+$ mask for positive flipping and $M-$ mask for negative flipping. Note that for TLCSS detection the locations of 1 and -1 values in both the $M+$ and $M-$ masks are the same. A third function $f_0(x)$ maintains the letter case as it is.

Applying these flipping functions to all letters in $T(x_1, x_2, ..., x_n)$ using a mask M results in an altered tag $T'_M(x_1, x_2, ..., x_n)$ such that:

$$T'_M(x_1, x_2....x_n) = T(f_{M(1)}(x_1), f_{M(2)}(x_2), \ldots f_{M(n)}(x_n)) \tag{2.13}$$

The offset of $T'_M(x_1, x_2, ...x_n)$ is computed using the function F in Equation 2.12. Accordingly the tags in a webpage are partitioned into three sets of tags Regular (R), Singular (S), Unchanged (U); these sets are referred to as the RS statistics in the work of Fridrich et.al [35] such that :

$$R = \{T(x_1, x_2, ..., x_n)|F(T'_M(x_1, x_2, ..., x_n)) > F(T(x_1, x_2, ..., x_n))\}$$
$$S = \{T(x_1, x_2, ..., x_n)|F(T'_M(x_1, x_2, ..., x_n)) < F(T(x_1, x_2, ..., x_n))\}$$
$$U = \{T(x_1, x_2, ..., x_n)|F(T'_M(x_1, x_2, ..., x_n)) = F(T(x_1, x_2, ..., x_n))\}$$

According to the above, when applying positive flipping, the statistics are R_{M+} and S_{M+} using mask $M+ = [+1, 0]$; and R_{M-}, and S_{M-} when applying negative flipping using mask $M- = [-1, 0]$.

An alternative discrimination function is proposed in this thesis in Chapter 6 as an alternative to the Tag Offset function. A comparison of the TLCSS detection approach presented by Huang et al. and that proposed in this thesis is presented in Chapter 6.

2.10 Summary

This chapter has provided the reader with a review of the existing work that underpins the work presented in the thesis. The chapter commenced with a review of the history of steganography from ancient times to the modern day. Definitions for the terminology used in steganography were then presented, together with discussion of both the challenges of steganalysis and the methodologies that may be adopted. The central theme of the work presented in this thesis is to address steganalysis using data mining approaches. Data mining is a central step in the Knowledge Discovery in Data (KDD) process, this process was thus also reviewed. There are a variety of data mining techniques that fall within the domain of KDD, the technique adopted in this thesis was classification. An overview of classification and the process of evaluating the performance of classification models was thus included in the review. The chapter then went on to consider the nature of HTML steganography, the type of steganography which the work presented in this thesis is directed; especially the HTML steganalysis methods that have been proposed in the literature. The significance of the later was that some of these mechanisms were used for evaluation purposes with respect to the steganalysis methods proposed later in the thesis. The next chapter describes the proposed SD detection approach for APS in the monitoring context.

Chapter 3

HTML Attribute Position Standard Deviation for APS Detection

3.1 Introduction

This chapter presents the Statistical Detection (SD) proposed approach for detecting Attribute Permutation Steganography (APS). Recall that the fundamental idea of APS, as presented in Chapter 2, is the utilization of the HTML feature of insensitivity to tag attribute ordering to hide messages. For example the two following tags are interpreted identically by an HTML browser, without noticeable alteration in the rendered webpage:

<img src="w3schools.jpg" alt="W3Schools.com" > sending "0"

<img alt="W3Schools.com" src="w3schools.jpg" > sending "1"

Both of the above express the same information but the different attributes ordering can be used to send a hidden message. Note also that any reordering does not result in any increase in the HTML file size after hidden message embedding has taken place. This makes it difficult to identify the possible presence of any steganography by inspection of the HTML source code alone.

As also noted in Chapter 2, the APS algorithms used with respect to this thesis were: (i) Huang et. al [45], (ii) Deogol [80] and (iii) Shen et. al [81]. The nature of APS is governed by the number of tags, in a given carrier WWW page, with two or more attributes. The fundamental idea underpinning the proposed SD APS detection approach is to monitor the position changes of a webpage attributes in terms of the position standard deviations ($St.D$s) and to use these to identify the presence of a hidden message or otherwise use some specified threshold. The utilization of this concept for steganography detection entails two significant challenges:

- The $St.D$ computed for different HTML encoded webpages, without any embedded messages, may vary significantly; hence it will be difficult to establish a global APS detection threshold.
- In the case of dynamic webpages the $St.D$ computed at time $t1$ may not be the same as time $t2$.

Some experimental results evidencing the above are given in Section 3.3.

As noted in Chapter 2, previous work on APS detection can be found in L.Polak and Z. Kotulski [65] and W.Jiang-feng et al. [95]. The first addressed APS in the dynamic (monitoring) context, whilst the second considered the static (non-monitoring) context. Recall also that in L.Polak and Z. Kotulski [65] a threshold W was used to describe the ratio of HTML attribute pairs that occurs in different orderings to the total number of HTML attribute pairings. It is suggested that W should be a constant when considering webpages where the hidden message is transmitted continuously and a variable if the hidden message is transmitted only over an appointed period of time, however, no specific values for W were suggested. Similarly, the most suitable parameter settings for: (i) total training time T, (ii) the interval between webpage snapshot captures τ and (iii) the nature of an embedded message L, were not provided. Consequently, in the evaluation reported on in this chapter the parameters used were those selected by the author.

Regarding the work presented by W.Jiang-feng et al. [95] this was directed at the static (non-monitoring) context. In this work the following settengs were used: (i) a feature vector representation was used as input to a SVM classifier model and that two features were considered, distance between the attribute benchmark mean position to the mean of the same attribute of the suspicious webpage and the variance of the attribute positions; (ii) the detection rate of the reported experimental results varied between 72.4% and 84.6% and (iii) the length L, or the nature (natural English language text or random text) of the hidden message was not considered in the reported evaluation.

The structure of the remainder of this chapter is as follows: Section 3.2 presents the results of an investigation into the effect of APS on the attribute position statistics in a webpage. Section 3.3 introduces the webpage SD concept. Section 3.4 then presents the SD monitoring process. Section 3.5 presents the evaluation of the SD concept and finally Section 3.6 presents a summary of the chapter and the main findings.

3.2 Attribute Positions Modification Resulting From APS

The process of using the standard deviation of attribute positions with respect to the mean position in HTML files is founded on the observation that, generally speaking, without any hidden message embedding, any given attribute tends to be located in the same position in most of the tags. By reordering tag attributes to hide a message, the *coherency of the attribute positions* changes.

An example is presented in Table 3.1 using the `rel` [1] and the `src` [2] tag attributes. The table lists the attribute positions before and after embedding an English language message using Shen APS algorithm. From the table it can be seen that the attribute position number changes as a result of message hiding. In the table the `rel` attribute appears ten times at position "0" until message hiding takes place when the position fluctuates between position "0" and position "1". In the case of the case of the `src` attribute the attribute appears 11 times, usually at position "3" and sometimes at other positions before embedding. Again positions are changed after message hiding has taken place.

TABLE 3.1: Attribute positions before and after embedding process

attribute	attribute positions before embedding	attribute positions after embedding
rel	0000000000	0001010111
src	33333332001	32322312103

As shown in Table 3.1 the dispersion of attribute positions will increase if message hiding is taking place, in other words the variance of attribute positions will increase and consequently the standard deviation will increases as well. This in turn can be used as an indicator of the presence of APS (as described in Chapter 2). Recall that in statistics Standard Deviation is a measure used to quantify the dispersion of a set of data values with respect to their mean. A low $St.D$ value indicates that the distribution of the data values is close to the mean, while a high $St.D$ value indicates that the distribution of the data values is much more spread. For any dataset S a standard deviation can be calculated using the following equation:

$$St.D = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \mu)^2}{n}} \tag{3.1}$$

Where x_i is a data value in set S, n is the number of data values in S and μ is the mean of the data values in S.

3.3 Standard Deviation of an HTML File Calculation

After presenting the effect of APS on the distribution of attribute locations within HTML tags in the previous section, the conjecture was that this can be usefully employed to distinguish a stego webpage from a clean webpage. The main idea was to identify the presence of APS by monitoring the $St.D$ of attribute positions within webpages.

Before presenting the proposed APS detection algorithm the following definitions should be noted:

[1] The `rel` is used with the `a` HTML tag to indicate the relationship between the current document and the linked resource.

[2] The `src` is used with the `img` HTML tag to specify the URL of the image.

- H: An input webpage.
- S: A set of tags T in H that have two attributes or more, $S = \{T_1, T_2, \dots\}$.
- $T_j = \{a_1, a_2, \dots, a_n\}$ denotes the jth tag in S with its attributes.
- attPositions(a_i): The array of positions in S for attribute a_i, $attPositions(a_i) = [p_1, p_2, \dots p_m]$. Each attribute has m occurrences such that: length(attPositions(a_i))=m.
- VattPositions(a_i) is the variance of attribute a_i positions in attPositions(a_i) array and is calculated as follows:

$$VattPositions(a_i) = \frac{\sum_{k=1}^{m} (p_k - av)^2}{m} \tag{3.2}$$

$$av = \frac{\sum_{k=1}^{m} p_k}{m} \tag{3.3}$$

- VS is the total variance of attributes positions for all tags in S. The total variance is equal to the summation of their variances as shown in equation 3.4 below:

$$VS = \sum_{i=1}^{n} VattPositions(a_i) \tag{3.4}$$

  Where n is the total number of all tags attributes in S.

- $St.D$ is the standard deviation of VS used to quantify the amount of variation in webpage attribute positions, and is calculated using:

$$St.D = \sqrt{VS} \tag{3.5}$$

The process for calculating the $St.D$ of a webpage attributes locations is given in Algorithm 4. The input to the algorithm is a webpage H, whilst the output is the $St.D$ of the attributes positions in the set of tags S. The algorithm commences by first finding S, the set of tags with two or more attributes in the given webpage. The attribute positions are collected in the the array attPositions(a_i) using collectPositions(a_i function. The total variance of attributes positions, VS, is the calculated using the sequence of an attribute positions in VattPositions(a_i) for all attributes one by one. Finally the associated value for $St.D$ is obtained.

Table 3.2 gives the $St.D$ values, computed using algorithm 4, for 10 webpages of Monitoring dataset in Appendix A. This $St.D$ values were computed before any embedding of messages had taken place. From the table it can be seen that the calculated $St.D$ values vary significantly between the selected webpages illustrating the challenge of identifying a suitable threshold for the purpose of detecting APS.

Algorithm 4: Standard Deviation ($St.D$) calculation of a webpage attributes' positions

1: Input: H
2: Output: $St.D$ standard deviation of H
3: Find S The set of tags T with two attributes or more
4: VS=0
5: **for** each *tag* $T \in S$ **do**
6: i=1
7: **for** each attribute $a_i \in T$ **do**
8: **if** notSeenBefore(a_i) **then**
9: attPositions(a_i)=collectPositions(a_i)
10: m= length(attPositions(a_i))
11: **for** each position p in attPositions(a_i) **do**
12: $av = \frac{\sum_{j=1}^{m}(p_j)}{m}$
13: $VattPositions(a_i) = \frac{\sum_{j=1}^{m}(p_j - av)^2}{m}$
14: **end for**
15: $VS = VS + VattPositions(a_i)$
16: i=i+1
17: **end if**
18: **end for**
19: **end for**
20: $St.D = \sqrt{VS}$

TABLE 3.2: Standard Deviation ($St.D$) values calculated for 10 webpages of Monitoring dataset

A webpage	$St.D$ before embedding
www.bbc.co.uk	3.01
www.nytimes.com	6.12
www.wikipedia.org	2.91
www.stackoverflow.com	2.41
www.sony.com	4.19
www.liverpool.ac.uk	4.43
www.ieee.org	5.84
www.webmd.com	4.83
www.microsoft.com	4.94
www.amazon.com	6.77

3.4 APS Monitoring Process

Using the above approach we can determine the attribute position $St.D$ value for a given WWW webpage H. By monitoring the $St.D$ over a period of time the conjecture is that we can identify attribute permutation steganography APS whenever unusual $St.D$ values are detected. This requires the establishment of a webpage steganography detection threshold σ. The idea here is that the value for σ can be learnt by deliberately seeding a training sequence of "snapshots" $C = \{s_1, s_2, \ldots, _n\}$ of H with hidden messages (using APS). In other words the monitoring commences with a "training process" during

which the value for σ is learnt. More specifically snapshots of the webpage of interest are collected periodically at intervals of time τ over a period of time T. For each snapshot a message of length L is generated $L \leq MEC(H)$ (recall that MEC is a Maximum Embedding Capacity as defined in Chapter 2 and the $St.D$ before and after message embedding is calculated. The value for σ is then calculated using the following equation:

$$\sigma = (\alpha \times St.D_{avbefore}) + ((1 - \alpha) \times St.D_{avafter}) \tag{3.6}$$

Where: (i) α is a user specified sensitivity factor of between 0 and 1 (ii) $St.D_{avbefore}$ is the average $St.D$ before message hiding and (iii) $St.D_{avafter}$ is the average $St.D$ after message hiding.
The inputs of the monitoring process are as follows:

- H: An (HTML encoded) webpage with dynamic content.
- α: A sensitivity factor, $0 < \alpha < 1$.
- T: The total training time.
- τ: The interval between webpage snapshot captures, for example every two hours.
- L: The length of the embedded message.

The actual monitoring process then comprises three steps:

1. **Training Sequence Collection**: Collect a series of snapshots of H with interval τ over time period T to give the set $C = \{s_1, s_2, \ldots, s_n\}$
2. **Threshold Calculation**: Calculate the threshold σ as follows:
   (a) For each $s_i \in C$ compute the $St.D$ of snapshot s_i, then embed a message using some appropriate APS algorithm (in the evaluation presented later in this chapter Shen et al. APS was used. the reason behind this is to investigate other than Deogol APS algorithm which was used in both approaches L.Polak and Z. Kotulski [65] and W.Jiang-feng et al. [95]).
   (b) Compute the average of the $St.D$s before embedding ($St.D_{avbefore} = \frac{\sum_{i=1}^{n} St.D_i}{n}$) and the average of the $St.D$s after embedding ($St.D_{avafter} = \frac{\sum_{i=1}^{n} St.D_i}{n}$); Where n represents the number of snapshots
   (c) The σ value is then calculated using equation 3.6.
3. **Detecting**: Compute the $St.D$ value of each further collected instance of H and compare it with σ; if $St.D \geq \sigma$ then the page is a stego webpage, otherwise the webpage is clean.

3.5 Evaluation of SD Concept

The results obtained from the evaluation of the proposed SD approach are presented in this section. The objectives of the evaluation were as follows:

1. To confirm the conjecture that increases in the attribute position $St.D$ of webpages is indeed an indicator of APS regardless of the APS technique used.
2. To compare naturally occurring changes in $St.D$ values caused by the changing content of dynamic webpages with the changes caused by message embedding.
3. To test how different kinds of embedded messages might affect the proposed APS detection mechanism.
4. To confirm that the attribute position $St.D$ concept can be effectively used to monitor webpages for APS.

The results obtained from experiments to establish the above objectives are presented in Subsections 3.5.1 to 3.5.4 respectively.

3.5.1 Standard Deviation of HTML Files as an Indicator of APS

The experiments designed to establish that the attribute position $St.D$ of webpages could indeed be used to detect APS were founded on three "snapshots" of the landing page of three well known websites: (i) Sony (ii) BBC and (iii) Wikipedia. This sample of three webpages was selected from Monitoring dataset in Appendix A. In each case attribute permutation steganography was used to embed hidden message using the algorithm presented in Shen et al. [81] (as described previously in Chapter 2). Messages of different length L were embedded from between 10% to 100% of the MEC, incrementing in steps of 10%. The MEC for the BBC, Wikipedia, Sony pages were 70 bytes, 74 bytes and 117 bytes respectively.

The results are shown in Figure 3.1. In the figure the X-axis lists the message size as a percentage of the MEC for each landing page, and the Y-axis the recorded $St.D$ before and after embedding. From the figure it can be observed that the $St.D$ starts to increase, with respect to the $St.D$ for the landing page without any hidden message embedding, as the size of the embedded message increases (obviously when the size of the embedded message is 0% of the MEC the recorded $St.D$ values will be the same). When the size of the hidden message was equivalent to the MEC for each given landing page the recorded increase in $St.D$ equated to 13.1%, 14.6% and 21% respectively. Thus, from the above, it was conclude that attribute position $St.D$ can indeed be usefully employed for the purpose of identifying attribute permutation steganography.

3.5.1.1 Standard Deviation Behavior Using a Range of APS Methods

This subsection presents the evaluation results obtained with respect to the investigation conducted as to whether the fundamental idea of using attribute positions $St.D$ as an

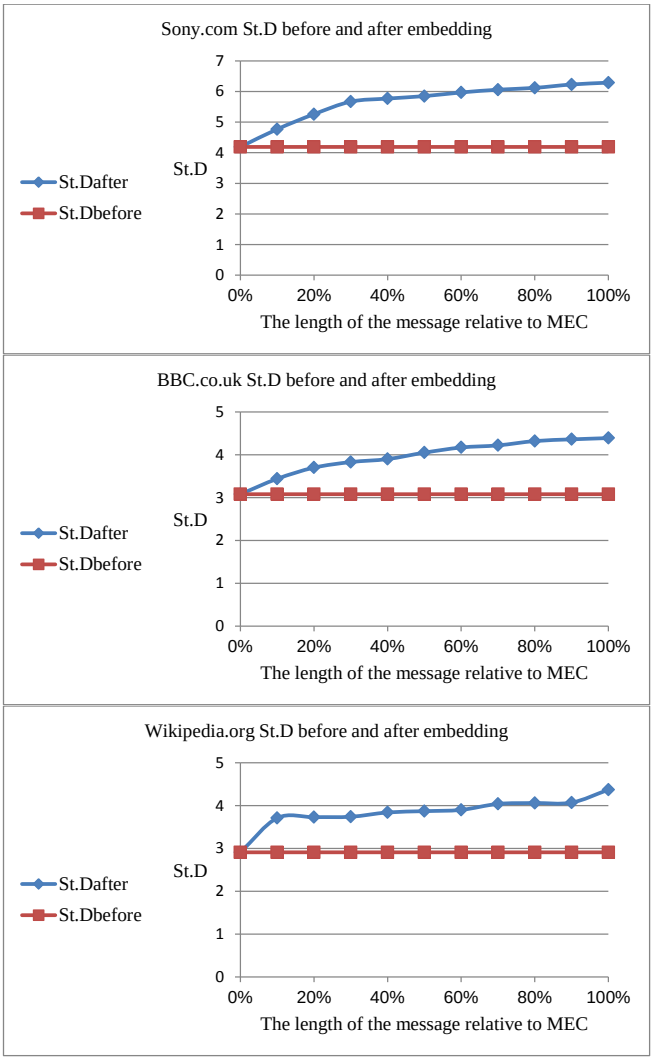


FIGURE 3.1: Standard Deviation ($St.D$) of attribute positions with respect to messages of different length

indicator of the presence or APS absence. The same landing pages and message length (L) settings were used as in the case of the experiments reported on in the previous section. Thus snapshots of the Sony, BBC and Wikipedia landing pages, and values of L ranging 10% to 100% of the MEC incrementing in steps of 10%. APS message embedding was conducted using both Huang et al.[45] and Deogol [80]. The results are shown in Figure 3.2. On the left-hand side of the figure the results obtained with respect to Huang et al. are given, and the right-hand side the results with respect to Deogol.

From Figure 3.2 it can be seen that in both cases the St.D values changed when APS was applied and that the $St.D$ difference increases as the hidden message length

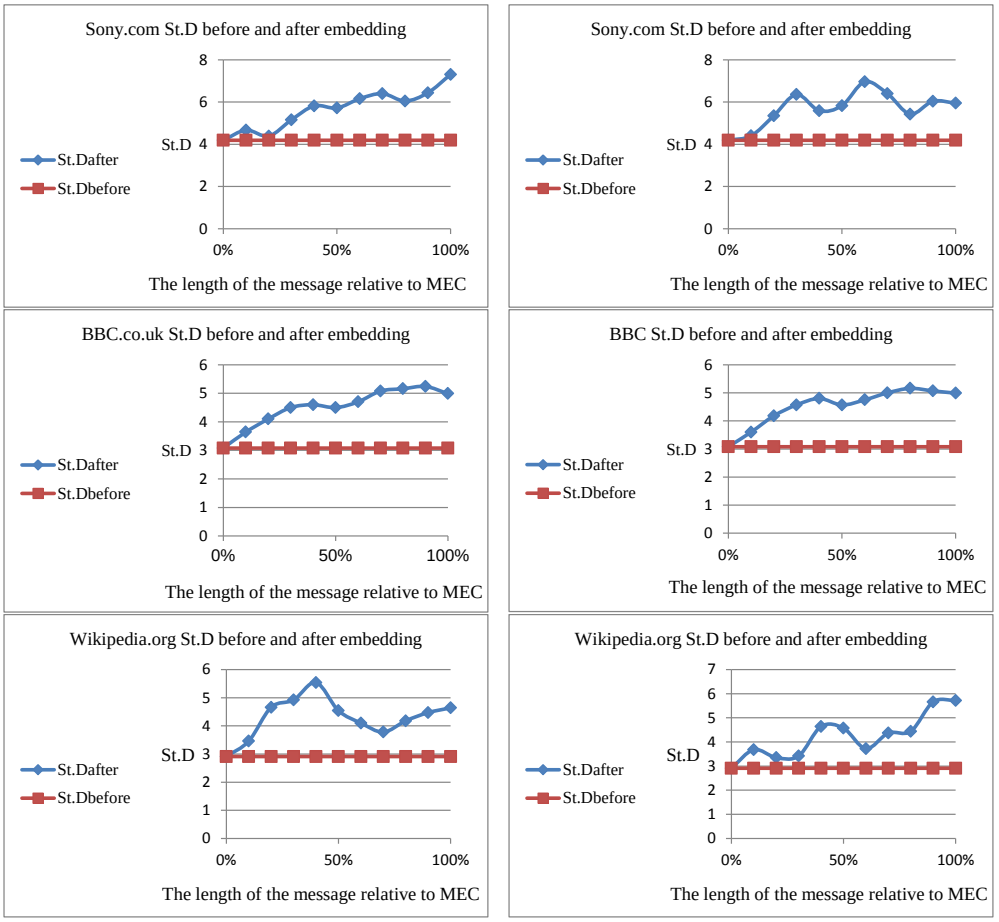


FIGURE 3.2: *St.D* values before and after the application of APS using: Huang et al. (left) and Deogol (right)

increases. These results endorse the outcome from the previous set of experiments. Supporting the conjecture that change in *St.D* can be usefully employed as an indicator of APS.

3.5.2 Natural Changes in Standard Deviation vs Changes Caused by Embedding

This Sub-section reports on the experiments conducted to compare naturally occurring changes in *St.D* values, caused by the nature of the ever-changing content of dynamic webpages, with *St.D* value changes caused by APS. For the experiments, Monitoring dataset in Appendix A was used in its entirety. Recall that the dataset was constructed using the following parameters: (i) T = one week and (ii) τ = every two hours. The length of embedded message was $L = 40\%$ of webpage MEC. The *St.D* value was calculated before and after message hiding using the APS algorithm of Shen et al.

The results obtained with respect to a sample of eight of the Monitoring dataset in Appendix A sequences are presented in Figure 3.3. In the Figure the X-axis gives the index number (ID) for each snapshot in the given sequence, whilst the Y-axis gives the recorded $St.D$ value in each case. From the figure it can be seen that in every case changes in the $St.D$ values caused by message embedding, and the changes occurring naturally, are significantly different. Thus conforming the utility of the proposed APS detection mechanism, founded on attribute position change $St.D$, in the context of monitoring dynamic webpage content.

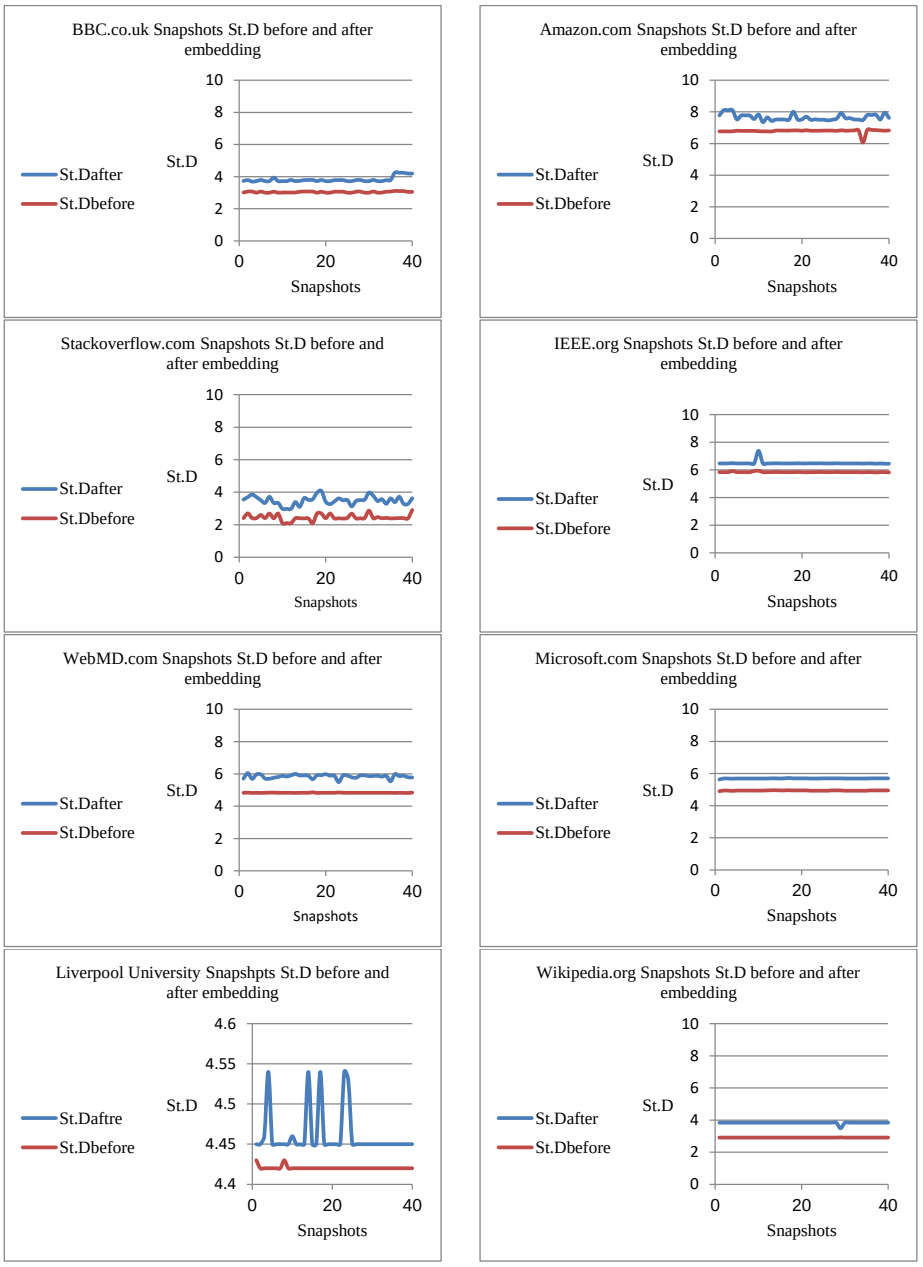


FIGURE 3.3: *St.D* values before (down) and after (up) message embedding for eight selected webpages sequences from Monitoring dataset

3.5.3 Effect of Different Types of Message Embedding

In the previous reported experiments, in each case, the same message type was embedded (natural lowercase English text). The goal of the next series of experiments was to test the effect of using a variety of different messages on APS detection performance using the proposed attribute position SD mechanism. The Monitoring dataset in Appendix A was again used, however, in this case the embedding was conducted using a variety of different messages. In total $29 \times 10 = 290$ different messages were used; a mixture of 29 natural English language messages and random messages (uppercase, lowercase, numbers, symbols) and 10 different lengths ranging 10% to 100% of the MEC increasing in steps of 10%. The results are presented in Table 3.3. In the table the "mean" row gives the webpage averaged $St.Ds$ after embedding, whilst the $Sdeviation$ row gives the standard deviation value for averaged $St.Ds$ after embedding at each incrementing of the hidden message. From the results presented in the table it can clearly be seen that the nature of the hidden message does not have a adverse effect on APS detection.

TABLE 3.3: Before and after averaged $St.D$ values using a variety of embedded messages

Length of embedded messages relative to a webpage MEC										
	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
mean	6.05	6.14	6.34	6.52	6.65	6.81	6.83	6.91	6.99	7.16
Sdeviation	0.04	0.06	0.10	0.12	0.11	0.12	0.12	0.16	0.15	0.17

3.5.4 Evaluation of the Monitoring Process

The final set of experiments reported on in this chapter was used to evaluate the usage of the proposed SD mechanism in the context of webpage monitoring. This is the situation where we wish to continuously check dynamic webpage content for the potential of APS.

Recall that the idea here was to first learn an appropriate detection threshold value σ. The Monitoring dataset in Appendix A was again used but with the first 30 allocated to training and the last 10 to testing. Message embedding, for evaluation purposes, was conducted using Shen et al. APS. The embedded message was a natural English language message. Once training was completed the effectiveness and sensitivity of the monitoring process to both the length of the embedded message L and α could be tested.

The testing was conducted by embedding hidden English language messages using APS, in the test set. Experiments were again conducted using different values for L from $L = 10\%$ and increasing to $L = 100\%$ in steps of 10%.

Table 3.4 shows the values of St.D before ($St.Db$) and after ($St.Da$) embedding a message of length 40% of a webpage MEC. Table 3.4 also shows how incrementing α values from 0.1 to 0.9 will reduce the value of detection threshold σ for the webpages of the Monitoring dataset in Appendix A.

TABLE 3.4: Reducing APS detection threshold values σ when α increases

A webpage	St.Db	St.Da	$\alpha = 0.1$	$\alpha = 0.2$	$\alpha = 0.3$	$\alpha = 0.4$	$\alpha = 0.5$	$\alpha = 0.6$	$\alpha = 0.7$	$\alpha = 0.8$	$\alpha = 0.9$
www.bbc.co.uk	3.030	3.700	3.633	3.566	3.499	3.432	3.365	3.298	3.231	3.164	3.097
www.amazon.com	6.800	7.650	7.565	7.480	7.395	7.310	7.225	7.140	7.055	6.970	6.885
www.stackoverflow.com	2.400	3.400	3.300	3.200	3.100	3.000	2.900	2.800	2.700	2.600	2.500
www.ieee.org	5.840	6.500	6.434	6.368	6.302	6.236	6.170	6.104	6.038	5.972	5.906
www.wbmed.com	4.800	5.800	5.700	5.600	5.500	5.400	5.300	5.200	5.100	5.000	4.900
www.microsoft.com	4.900	5.600	5.530	5.460	5.390	5.320	5.250	5.180	5.110	5.040	4.970
www.liverpool.ac.uk	4.420	4.460	4.456	4.452	4.448	4.444	4.440	4.436	4.432	4.428	4.424
www.wikipedia.org	3.820	2.910	3.729	3.638	3.547	3.456	3.365	3.274	3.183	3.090	3.001
www.sony.com	5.300	4.100	5.180	5.060	4.940	4.820	4.700	4.580	4.460	4.340	4.220
www.ntimes	7.100	6.200	7.010	6.920	6.830	6.740	6.650	6.650	6.470	6.380	6.290

Table 3.5 shows the obtained results for one example of these webpages included, namely the IEEE webpage. In Table 3.5 the testing results for 10 webpages of IEEE.org using different obtained σ are presented. The table also shows that α values increases from 0.1 to 0.9. The selected testing set was half seeded with APS with hidden messages of length between 10% to 50% of a webpage MEC. From the table it can be noted that when $\alpha = 0.1$ and $L = 40\%$ the computed σ is 6.434 this is also shown in Table 3.4. Using this thresold for testing faild to identify webpages with $ID_{(6,7,8)}$ as stego-webpages so that False Negatives FN=3. When $\alpha = 0.9$ the computed σ is 5.906 identifiedy webpages with $ID_{(2,3)}$ as stego webpages while they are not so that False Positives FP=2.

			α								
			0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
			Steganography detection threshold δ								
Webpage ID	Webpage status	St.D for tested webpage	6.434 FP=0 FN=3	6.368 FP=0 FN=3	6.302 FP=0 FN=2	6.236 FP=0 FN=2	6.170 FP=0 FN=1	6.104 FP=0 FN=1	6.038 FP=0 FN=0	5.972 FP=0 FN=0	5.906 FP=2 FN=0
1	clean	5.840									
2	clean	5.920									X
3	clean	5.940									X
4	clean	5.830									
5	clean	5.840									
6	Stego L=10%	6.050	X	X	X	X	X	X			
7	Stego L=20%	6.220	X	X	X	X					
8	Stego L=30%	6.330	X	X							
9	Stego L=40%	6.470									
10	Stego L=50%	6.560									

TABLE 3.5: False Positives FP and False Negatives FN for 10 tested webpages when different values of σ and $L = 40\%$ of a webpage MEC

These results in Table 3.5 are also illustrated in Figure 3.4. The Figure shows how the computed σ for $L = 40\%$ of a webpage MEC fails to identify stego-webpages with hidden messages of length less than L (FN goes high). Also using high values of α will cause identifying clean webpages as stego-webpages (FP goes high).

Again the testing is repeated but this time for a hidden message length of L=60%. Table 3.6 shows the obtained results for 10 webpages of IEEE.org using different σ. The table also shows that α values increases from 0.1 to 0.9. Again the same previous selected testing set was used. From the table it can be noted that how increasing of L produces high σ which impacts the detection process (FN metric) when α is fixed. Figure 3.5 depicts also this conclusion.

			α								
			0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
			Steganography detection threshold δ								
Webpage ID	Webpage status	St.D for tested webpage	6.796 FP=0 FN=5	6.690 FP=0 FN=5	6.584 FP=0 FN=5	6.477 FP=0 FN=4	6.371 FP=0 FN=3	6.265 FP=0 FN=2	6.158 FP=0 FN=1	6.042 FP=0 FN=0	5.946 FP=0 FN=0
1	clean	5.840									
2	clean	5.920									
3	clean	5.940									
4	clean	5.830									
5	clean	5.840									
6	Stego L=10%	6.050	X	X	X	X	X	X	X		
7	Stego L=20%	6.220	X	X	X	X	X	X			
8	Stego L=30%	6.330	X	X	X	X	X				
9	Stego L=40%	6.470	X	X	X	X					
10	Stego L=50%	6.560	X	X	X						

TABLE 3.6: False Positives FP and False Negatives FN for 10 tested webpages when different values of σ and $L = 60\%$ of a webpage MEC

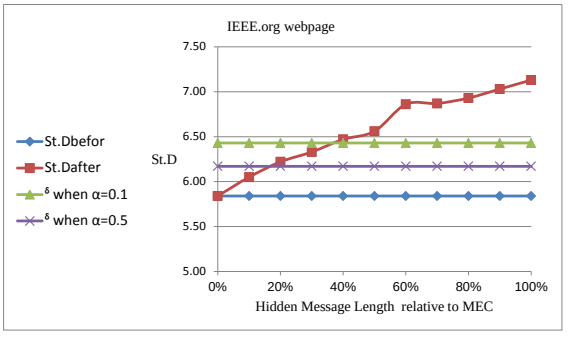


FIGURE 3.4: Derived detection threshold σ using different values for α when $L = 40\%$ of the webpage MEC

3.6 Summary

In this chapter the proposed SD approach for detecting APS has been presented. The approach is directed at the monitoring of dynamic webpages for APS. The fundamental

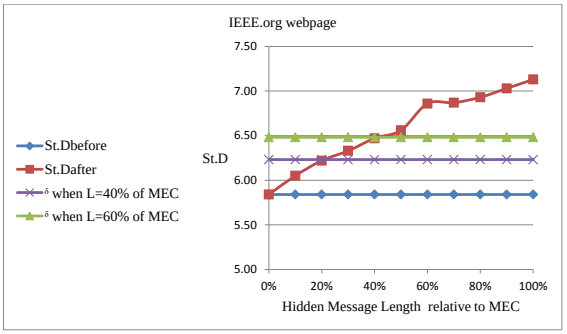


FIGURE 3.5: Detection threshold σ sensitivity of different values of L while $\alpha = 0.5$

idea was that a change in the standard deviation ($St.D$) value for a given webpage would be an indicator of the existence of APS. More specifically that if the $St.D$ value increased above some specified threshold σ, APS could be said to exist. A particular challenge was determining what the value for the σ threshold, to facilitate APS detection, should be; a "one-size-fits-all threshold value was found to be inappropriate. The idea was to identify a mechanism whereby the σ value for a particular WWW page could be learnt. This process was fully described together with the evaluation conducted to determine whether the conjecture that the monitoring $St.D$ values can be used to detect the presence of APS was correct or not. The reported evaluation results indicated that the conjecture was correct; the Standard Deviation of a webpage could indeed be usefully employed for the detection of APS. The next chapter presents the proposed APCC approach for APS detection; an alternative to the SD mechanism present in this chapter for use in the non-monitoring context.

Chapter 4

APS Detection Using Attribute Position Changes Count

4.1 Introduction

In this chapter the proposed Attribute Position Changes Count (APCC) APS detection mechanism is presented. APCC is directed at detecting APS in the static context (whereas the SD approach presented in the previous chapter was directed at dynamic context). The APCC mechanism adopts a classification approach to APS detection.

The main idea is to train a classification model which can then be used to differentiate between WWW pages as being either "stego pages" or "normal pages". This approach is inspired by the work of W.Jian-feng et al. in [95] who also proposed a classification based approach to APS detection in the static context. In [95] a collection of webpages were used for evaluation purposes with APS applied using Deogol [80]. The classification model used was a Support Vector Machines (SVM) model.

Classification is a well understood branch of Machine learning [9], the challenge is how best to transform the application focused data into a format appropriate for a classifier generation. Most generators take, as input, a feature vector representation of some sort. In the context of APS detection the requirement is thus to represent attribute data in a feature vector format. The feature vector used in by W.Jian-feng et al. in [95] comprised: (i) distance between the mean of the attribute positions in a benchmark webpage to the mean of the same attribute positions in a suspicious webpage and (ii) the attribute positions variance. However, it is argued here that using average position values ignores the spread of position changes, which may in turn be important in the context of APS detection; in other words important information may be lost when using average values. Instead, the proposed APCC mechanism, as the name suggests, uses the concept of attribute position changes count. The idea is motivated by the observation that APS entails frequent attribute position changes which can thus be used to distinguish APS WWW pages from non-APS WWW pages. The APCC metric is fully described later

in this chapter together with how it was incorporated into the APCC APS detection approach.

The proposed approach was evaluated using the Non-Monitoring dataset in Appendix A and three different classifier generation models. The dataset was seeded using the three most commonly referenced APS approaches, namely: (i) Deogol [80], (ii) Huang et al. [45] and (iii) Shen et al. [81] (see Chapter 2). As a consequence three variations of the dataset were produced labeled using the name of the APS algorithm used for message hiding: (i) Deogol, (ii) Huang et al. and (iii) Shen et al. The three classification paradigms used were: (i) Neural Networks, (ii) SVM and (iii) Naive Bayes. In each case the operation of the proposed APCC approach was compared with three alternative APS detection algorithms: (i) L.Polak and Z. Kotulski [65] , (ii) Sedeeq et al. [79] (see Chapter 2) and (iii) W.Jian-feng et al. [95].

The remainder of this chapter is organised as follows: Section 4.2 demonstrates the APS effect on the attribute positions in an HTML file and how this can be utilized to introduce APCC metric. Section 4.3 presents the proposed classification model for APS steganalysis using the APCC approach. Section 4.4 then presents the evaluation of proposed APCC approach. Finally Section 4.5 gives a summary of the chapter.

4.2 The Attribute Position Change Count Concept

The concept of APS was introduced in earlier chapters where it was also noted that in normal circumstances (no steganography present) attributes within HTML tags tend to feature a consistent ordering, but once APS has taken place a consistent ordering is no longer the case. To formalize the Attribute Position Change Count (APCC) concept we allocate a position number to each attribute within a tag, the first position is given the number 0, the second position 1 and so on. Thus, given an attribute a_i belonging to the set of attributes A featured in a HTML page this will have a set positions $P_{a_i} = \{p_1, p_2, \dots\}$ associated with it. In the absence of APS we would expect the values in P to be more constant. Thus it is conjectured that by counting the number of attribute position changes we have an indicator of the presence (or otherwise) of APS. We calculate the position change count for an attribute as follows by comparing consecutive positions; if there is a change in position we increment the change count for that attribute by 1. Thus, given two consecutive positions for an attribute, p_j and p_{j+1}, we increment the position change count so far by 1 if $p_j \neq p_{j+1}$.

The APCC concept is illustrated in Table 4.1 with respect to a fictitious website, and two attributes: (i) `content`[1] and (ii) `rel`[2]. The second column in Table 4.1 shows the positions of the attributes in the HTML document that might exist if no APS has taken place. Note that the first attribute features 9 times and the second 15 times (hence

[1]The `content` attribute is used with the `meta` HTML tag to provide additional information that can be used by, for example, www browsers.

[2]The `rel` attribute specifies the relationship between the current document and the linked document in a link HTM tag

TABLE 4.1: Attribute position changes before and after APS

Attribute	Att. pos. before APS	Apcc before APS	Att. pos. after APS	Apcc after APS
`content`	111111111	0	101001010	7
`rel`	000000000001111	1	101000110104242	11

9 and 15 positions respectively). The third column then lists the associated position changes count. The forth column shows the positions of the attributes in the HTML document that might exist if APS has taken place, whilst the fifth column shows the associated position changes count. Note that a clear distinction can be observed before and after APS (at least in this example).
Table 4.2 presents the attribute position changes count for the first three attributes in ten selected webpages before and after ASP embedding.

TABLE 4.2: First three APCC values in selected webpages before and after APS embedding

Webpage	Apcc before embedding	Apcc after applying Huang et al. [45])	Apcc after applying Deogol [80]	Apcc after applying Shen et al. [81]
www.wikipedia.org	2, 2, 6	12, 20, 15	14, 25, 16	17, 12, 11
www.stackoverflow.com	2, 12, 13	14, 14, 16	18, 16, 23	20, 8, 23
www.ieee.com	9, 2, 0	18, 15, 7	15, 15, 6	19, 18, 10
www.sony.com	4, 4, 3	14, 5, 1	1, 5, 15	16, 13, 17
www.bbc.com	14, 6, 3	12, 15, 19	19, 15, 16	17, 12, 11
www.dhl.com	7, 7, 7	9, 6, 11	7, 7, 13	13, 20, 18
www.linkedin.com	14, 4, 0	12, 3, 15	17, 8, 15	14, 9, 16
www.microsoft.com	4, 6, 16	5, 20, 10	5, 21, 10	9, 11, 17
www.sears.com	11, 4, 0	16, 9, 0	7, 16, 0	13, 18, 16
www.ox.ac.uk	7, 6, 14	10, 19, 18	11, 19, 16	16, 8, 19

The webpages were selected from the Non-Monitoring dataset in Appendix A used for the evaluation of the proposed APCC approach discussed later in this chapter. A short English language message of 17 characters was embedded in each case using the selected APS algorithms: Huang et al., Deogol and Shen et al. Column 2 in the table gives the APCC in each case before embedding, whilst columns 3, 4 and 5 give the APCC values after embedding using each APS algorithm. From the Table it can be clearly seen, in most cases, that the APCC values after embedding a message are higher than the value before embedding. This idea is thus to use the APCC concept as a mechanism

for detecting the presence (or otherwise) of APS. From the table it can also be noted that in some cases the APCC value decreased after embedding had taken place, this was attributed to the different implementations of the individual attribute permutation APS algorithms that were used.

4.3 Proposed APCC Classification Approach for APS Steganalysis

Given the above, the idea is to use attribute position changes count information to generate feature vectors and, given an appropriately defined training set featuring both stego and non-stego WWW pages, to train a classification model for APS detection.

There are many algorithms available that can be used to build classification models (also known as prediction models). Common examples include: (i) Neural Networks, (ii) Support Vector Machines and (iii) Naive Bayes. What these algorithms have in common is that they take as input a set of feature vectors of length n generated from an n-dimensional feature space. In our case the feature space represents the set of attributes A that we wish to consider. Each dimension represents an attribute whose values range from 0 to some maximum number of possible position changes count.

The required training data comprises an $n \times m$ matrix where n is the number of webpages in the training set and m is the number of attributes to be considered. The value associated with each attribute is the position changes count generated as described above. Before presenting the proposed algorithm used for counting the attribute position changes, the following definitions should be noted:

- H: An input webpage.
- S: A set of tags T in H that have two attributes or more, $S = \{T_1, T_2, \dots\}$.
- FV: The generated feature vector of all tag attribute position changes count in S.
- attPositions(a_i): The array of positions in S for attribute a_i, attPositions(a_i) $=[p_1, p_2, \dots p_m]$.
- $apcc(a_i)$: The number of position changes for an attribute (a_i), incremented when $p_j \neq p_{j+1}$.

The pseudo code for generating APCC feature vector is presented in Algorithm 5. The input to the algorithm is a WWW page H. The output is FV, a feature vector recording the number of attributes position changes with respect to all the tags within S, the set of tags with two attributes or more. The algorithm commences by finding S (line 3). Next (lines 5 to 9) the algorithm loops through each tag T in S, and each attribute a_i in each tag, and records the attribute a_i positions in attPositions(a_i) array using collectPositions(a_i) function. Next (lines 10 to 16) the algorithm loops through each attPositions(a_i) array to calculate the attribute a_i position changes count *apcc* according

to the positions value associated with it. Each calculated *apcc* value is stored in FV to generate the desired feature vector for the webpage. Note also that for the training data each feature vector has a class label associated with it: class 1 represents a stego webpage while class 2 represents a normal webpage.

Algorithm 5: APCC feature vector generation algorithm

```
1: Input: H
2: Output: FV a feature vector holding position changes count for attributes in H
3: Find S The set of tags T with two attributes or more
4: k=0
5: for each tag T ∈ S do
6:    i=1
7:    for each attribute a_i ∈ T do
8:       if notSeenBefore(a_i) then
9:          apcc=0
10:         attPositions(a_i)=collectPositions(a_i)
11:         j=1
12:         for each position p_j in attPositions(a_i) do
13:            if p_j ≠ p_{j+1} then
14:               apcc = apcc + 1
15:               j=j+1
16:            end if
17:         end for
18:         FV[k] = apcc
19:         k = k + 1
20:      end if
21:      i = i + 1
22:   end for
23: end for
```

4.4 Evaluation of Proposed APCC Approach

To evaluate the proposed APCC approach the Non-Monitoring dataset in Appendix A was used. The conjectured advantage of the proposed approach was that it would operate well regardless of how the APS was conducted. To demonstrate this the three most commonly used and referenced APS algorithms in literature, identified in Chapter 2, were again used to generate three evaluation datasets: (i) Deogol, (ii) Huang and (iii) Shen. In each case half of the selected WWW pages were seeded using APS. In this case the hidden message was a natural text message of 35 characters in length.

For the purpose of incorporating APS five tag attributes were selected. Consequently the feature vectors had five "slots" each holding a position changes count. Thus the entire data matrix in each case measured 150×5 (150 because this was the number of webpages included in the non-monitoring dataset).

For the evaluation Ten-fold Cross Validation (TCV) was used throughout whereby the input data was divided into 10 folds and the classifier generation process conducted

and tested 10 times, each time using a different fold for the testing. The evaluation metrics used were those recommended in [23], namely:

- Accuracy (Acc); the percentage of correctly classified samples.
- The Area Under the receiver operator characteristic Curve (AUC); a measure of the area under the graphical plot of the true positives rate versus the false positives rate.

The objectives of the evaluation were as follows:

1. To analyze the effectiveness of the proposed APCC mechanism in terms of a number of common APS algorithms and a number of classification model generators.
2. To determine the effect of the nature of the embedded message type, and its length, on the classification models.
3. To determine whether the results obtained, with respect to the experiments conducted to address the previous objective, were statistically significant or not.
4. To compare the operation of the proposed approach with alternative monitoring approaches proposed in the literature, namely the approaches of L.Polak and Z. Kotulski [65] and Sedeeq et al. [79] (presented in Chapter 3) and with alternative non-monitoring approaches proposed in the literature, namely that of W.Jian-feng et al.[95].

The conducted evaluation with respect to each objective is considered in further details in Sub-sections 4.4.1 to 4.4.4 below.

4.4.1 Effectiveness of the APCC Approach

For the first of the above objectives three classifier generation models were considered, as implemented in the Weka machine learning environment [32]: (i) Multi-Layer Perceptron (MLP) Neural Network, (ii) SVM and (iii) Naive Bayes NB. Default settings for the MLP and SVM classifiers in WEKA software were used and as follows: for MLP classifier the parameters were as follows: one hidden layer, learning rate=0.3, momentum=0.2 and number of epoches=500, while SVM classifier parameters were: complexity parameter c=1 and tolerance parameter=0.001. Note that SVM was used because this was the classification model used with respect to the APS detection approach proposed by W.Jian-feng et al. [95]. Recall that the significance of the later is that, to the best knowledge of the author, this is the only other previous work that adopts a classification model approach to APS detection; however, as already noted, using a very different feature vector representation. The results are summarized in Tables 4.3 and 4.4 (best results highlighted in bold font). Table 4.3 shows the results obtained in terms of average Accuracy (Acc) while Table 4.4 shows the results obtained in terms of the AUC

measure. The averaged standard deviation of both Acc and AUC for each fold of 10-TCV is in parentheses. From the tables it can be seen that the proposed APCC feature vector representation can be successfully used to train classifiers to distinguish between normal webpages and stego webpages, regardless of the adopted APS algorithm. The best performance was obtained with respect to the APS algorithm of Shen et al. From the tables it can also be seen that there is little difference in operation between the selected classifier generators.

TABLE 4.3: Average accuracy (Acc) using APCC (best results highlighted in bold font)

APS Algorithm	**MLP**	**SVM**	**NB**
Deogol [80]	93.39% (9.71)	90.71% (10.54)	90.89% (10.46)
Haung et al. [45]	90.36% (11.17)	93.21% (7.18)	88.75% (13.10)
Shen et al. [81]	**93.57%** (10.75)	**96.07%** (6.34)	**93.57%** (10.75)

TABLE 4.4: Average AUC using APCC (best results highlighted in bold font)

APS Algorithm	**MLP**	**SVM**	**NB**
Deogol [80]	0.97 (0.06)	0.90 (0.11)	0.98 (0.05)
Haung et al. [45]	0.94 (0.11)	0.91 (0.08)	0.96 (0.07)
Shen et al. [81]	**0.99** (0.03)	**0.95** (0.06)	**0.99** (0.03)

4.4.2 The Effect of Different Message Type and Length Embedding

To investigate the effect of the nature of the hidden message, and its length, on the APS detection process two sets of experiments were conducted, the first using English language embedded messages and the second using random messages (a mixture of uppercase, lowercase letters and numbers). In each case the length of the embedded message was varied ranging from between 10% to 100% of Maximum Embedding Capacity (MEC) incrementing in steps of 10%, although only results for lengths of 10% to 50% of the MEC are presented here because beyond this no change in the results was detected. At each incrementing step ten different hidden messages were considered. The embedding algorithms used, as in the case of earlier experiments, were: (i) Deogol, (ii) Haung et al. and (iii) Shen et al.

The same classifier generators as considered with respect to the previously reported experiment were also used: (i) Multi-Layer Perceptron (MLP) Neural Network, (ii) SVM and (iii) Naive Bayes NB. For each APS algorithm two datasets were generated, one using

natural English language text embedding and one using random text embedding. Each dataset comprised 5 sub-datasets each containing message of different length as it is shown in Table 4.5 column 1 in Sub-section 4.4.3.

The results obtained, using MLP, SVM and NB, are summarized in Figures 4.1, 4.2 and 4.3 respectively. In each case the graphs on the left were generated using English language texts (the first set of experiments) and the graphs on the right using random text (the second set of experiments). As in the case of the graphs used to report the outcomes of the experiments reported on earlier in this chapter, the X-axis represents the message length (expressed as a percentage of MEC) and the Y-axis represents the accuracy and AUC. Inspection of the figures firstly indicates that the results were consistent with the results presented previously in Tables 4.3 and 4.4. More specifically the results confirmed the following:

- The proposed APCC approach can successfully be used to train classifiers to distinguish between stego and non-stego webpages, regardless of the APS algorithm used.
- The nature of the embedded messages, random or otherwise, has no significant effect on the performance of the classifiers.
- With respect to the length of the embedded message this has a significant impact on classifier performance when the APS algorithm of Shen et al. is used. This is considered as an advantage of using the APS algorithm of Shen et al. because it it is hard to be detected with short messages.

4.4.3 Statistical Evaluation

This subsection reports on the experiments conducted to determine whether the obtained results presented in Figures 4.1, 4.2 and 4.3 were indeed statistically significant, and not purely a matter of chance. There are several available statistical tests available to compare the operation of classifiers. Demsar, in his comprehensive study [23] concluded that non-parametric statistical tests are the most appropriate for comparing classification algorithms over multiple datasets.

With respect to the work presented in this sub-section the Friedman non-parametric statistical test was adopted. In this test each classifiers performance, with respect to each dataset, is ranked. The best performing classifier is assigned a rank (r) of 1, the second is a rank of $r = 2$ and so. The Friedman test statistic is calculated using Equation 4.1[90] [26].

$$X_F^2 = \frac{12N}{K(K+1)}\left[\sum_{j=1}^{k} R_j - \frac{K(K+1)^2}{4}\right] \tag{4.1}$$

Where: (i) $R_j = \frac{1}{N}\sum_{i=1}^{N} r_i$ is the average rank of the classifier over the used datasets, (ii) N is the number of used datasets, (iii) K is the number of classifiers and (vi) r_i is

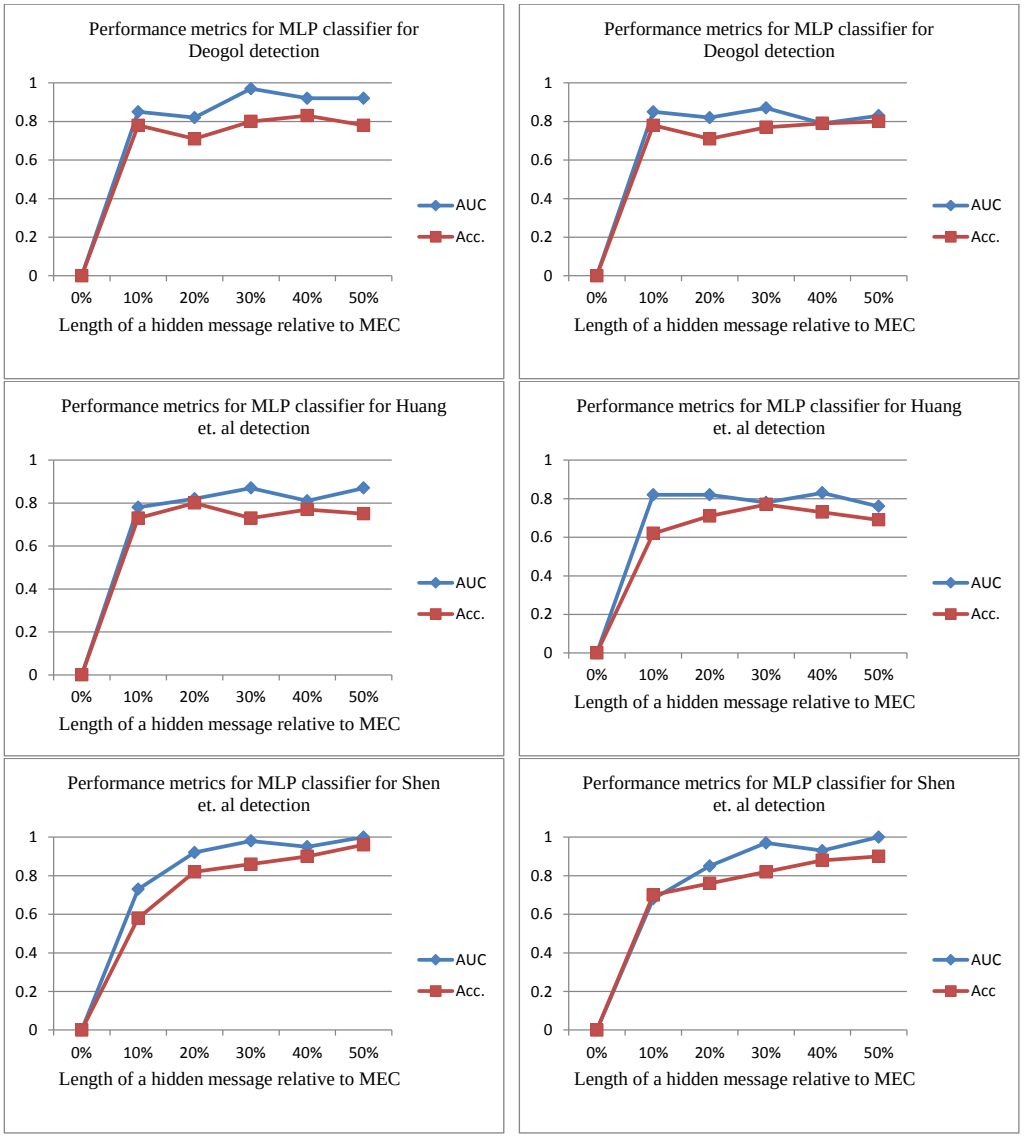


FIGURE 4.1: Performance of MLP classifier for detecting APS. Left: natural English language messages embedding. Right: random messages embedding

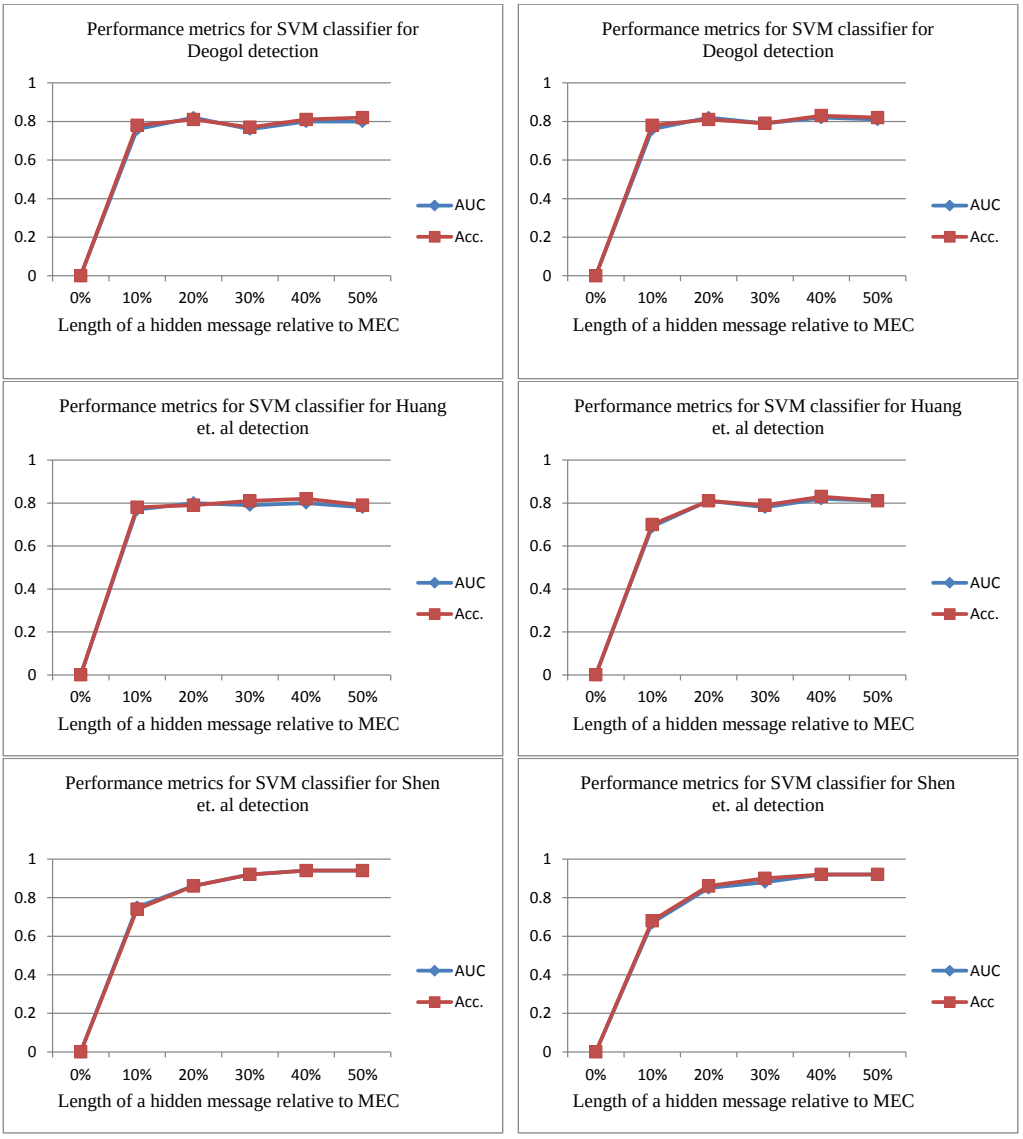


FIGURE 4.2: Performance of SVM classifier for detecting APS. Left: natural English language messages embedding. Right: random messages embedding

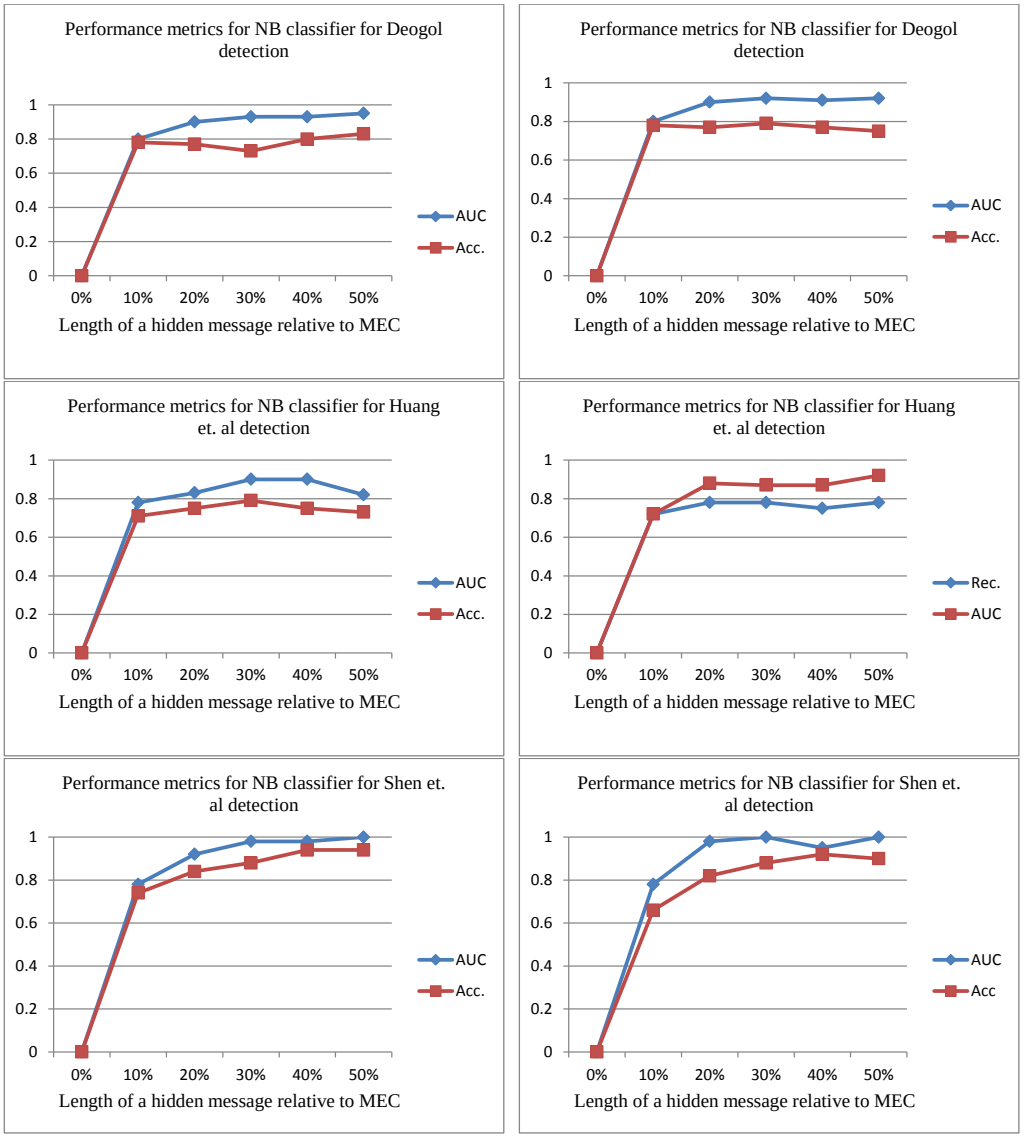


FIGURE 4.3: Performance of NB classifier for detecting APS. Left: natural English language messages embedding. Right: random messages embedding

the rank of the j_{th} classifier with respect to the i_{th} dataset. The Friedman statistic X_F^2 is distributed according to the Chi-Squared distribution with $K-1$ degrees of freedom (df). The test is used to evaluate classifiers performance so that the null-hypothesis, $H0$, that there is no significant difference between classifiers can be accepted or rejected. The null hypothesis is rejected if the X_F^2 value is greater than the "null distribution". This is a pre-determined distribution with a most commonly used cut-off (significance level) value of $\alpha = 0.05$ [22, 29].

Also, for null hypothesis testing a p-value is used to weigh the strength of the evidence against the null hypothesis. Therefore low values of $p \leq \alpha$ provide strong evidence against the null hypothesis, in which case $H0$ can be rejected; while high values of $p > \alpha$ indicate weak evidence against the null hypothesis, in which case $H0$ is accepted. When $H0$ is rejected an alternative hypothesis $H1$, that there is a significant difference between classifiers, is established and a post-hoc test is applied to determine which classifier is significantly better than the others.

Several post-hoc tests can be adopted for this purpose [23], however the Nemenyi post-hoc test was adopted. The Nemenyi test holds that there is a statistically significance difference between the performance of classifiers if the difference between classifier average ranks is equal to or greater than a "Critical Difference" (CD) between them. The critical difference is computed using the following equation [90] [26].

$$CD = q_\alpha \sqrt{\frac{K(K+1)}{6N}} \tag{4.2}$$

Where the value for critical value $q\alpha$ is taken from the table given in Figure 4.4. This table shows the critical value q_α at both significance levels $\alpha = 0.05$ and $\alpha = 0.01$ according to the number of used classifiers.

For the purpose of the significance testing the AUC results from the experiments reported in Sub-section 4.4.2 were used (see also Figures 4.1, 4.2 and 4.3). The resulting Friedman test outcomes are given in Table 4.5 which gives the Friedman test statistics when embedding both natural English and random messages using the three considered APS techniques. Column 1 in Table 4.5 lists the datasets used, recall that each dataset consisted of five sub-datasets as noted in column 2. The APS algorithm used to embed the messages and the nature of the embedded messages are shown in columns 3 and 4 respectively. The previously obtained AUC values, using MLP, SVM and NB, are given in columns 5, 6 and 7. In each case the Chi-Square Friedman test statistics and the associated p-values are given in columns 7 and 8 respectively.

From Table 4.5 it can be seen that all p-values are less than or equal to the significance level of $\alpha = 0.05$; therefore the null-hypothesis ($H0$), that there is no significant difference in the operation of the classifiers, can be rejected. It can be also noted from Table 4.5 that for all generated datasets the highest average rank was obtained using the NB classifier, followed by MLP and then SVM. Then, according to Friedman test the rank given to NB is 1, MLP is 2 and SVM is 3. Since as mentioned that ($H0$) can be rejected accordingly the Nemenyi post-hoc test was applied. The critical difference

#classifiers	2	3	4	5	6	7	8	9	10
$q_{0.05}$	1.960	2.343	2.569	2.728	2.850	2.949	3.031	3.102	3.164
$q_{0.10}$	1.645	2.052	2.291	2.459	2.589	2.693	2.780	2.855	2.920

FIGURE 4.4: Critical values for Nemenyi test [23]

(CD) between the classifiers was calculated using Equation 4.2. Applying this equation with the following parameters: ($K = 3$ the number of classifiers, $N = 5$ the number of sub-datasets and $q = 2.343$ from Figure 4.4) will give $CD = 1.4$.

TABLE 4.5: Friedman test results using AUC values

Main Datasets	Sub-Data Sets	APS Algorithm	Embedded Messages	MLP	SVM	NB	Chi-Square	p-value
				AUC				
huangNatural	$D_{s10\%}$	Huang et.al	Natural English text	0.78	0.77	0.78	8.316	0.016
	$D_{s20\%}$			0.82	0.80	0.83		
	$D_{s30\%}$			0.87	0.79	0.90		
	$D_{s40\%}$			0.81	0.80	0.90		
	$D_{s50\%}$			0.87	0.78	0.82		
Average Rank				0.83	0.78	0.84		
huangRandom	$D_{s10\%}$	Huang et.al	Random text	0.82	0.69	0.72	6.000	0.050
	$D_{s20\%}$			0.82	0.81	0.88		
	$D_{s30\%}$			0.78	0.78	0.87		
	$D_{s40\%}$			0.83	0.82	0.87		
	$D_{s50\%}$			0.76	0.81	0.92		
Average Rank				0.80	0.78	0.85		
deogolNatural	$D_{s10\%}$	Deogol	Natural English text	0.85	0.76	0.80	6.632	0.036
	$D_{s20\%}$			0.82	0.82	0.90		
	$D_{s30\%}$			0.97	0.76	0.93		
	$D_{s40\%}$			0.92	0.80	0.93		
	$D_{s50\%}$			0.92	0.80	0.95		
Average Rank				0.89	0.78	0.90		
deogolRandom	$D_{s10\%}$	Deogol	Random text	0.85	0.76	0.80	6.000	0.050
	$D_{s20\%}$			0.82	0.82	0.90		
	$D_{s30\%}$			0.87	0.79	0.92		
	$D_{s40\%}$			0.79	0.82	0.91		
	$D_{s50\%}$			0.83	0.81	0.92		
Average Rank				0.83	0.80	0.89		
shenNatural	$D_{s10\%}$	Shen et. al	Natural English text	0.73	0.75	0.78	6.706	0.035
	$D_{s20\%}$			0.92	0.86	0.92		
	$D_{s30\%}$			0.98	0.92	0.98		
	$D_{s40\%}$			0.95	0.94	0.98		
	$D_{s50\%}$			1.00	0.94	1.00		
Average Rank				0.91	0.88	0.93		
shenRandom	$D_{s10\%}$	Shen et.al	Random text	0.68	0.67	0.78	9.000	0.011
	$D_{s20\%}$			0.85	0.85	0.98		
	$D_{s30\%}$			0.97	0.88	1.00		
	$D_{s40\%}$			0.93	0.92	0.95		
	$D_{s50\%}$			1.00	0.92	1.00		
Average Rank				0.88	0.84	0.94		

A visualization of Nemenyi test for results in Table 4.5 is given in Figure 4.5. The figure displays the ranked classifiers along with the critical difference. From the figure it can be clearly seen that NB is the best performing classifier, however this is statistically significant only with respect to SVM and not significant to MLP (the CD tails overlap).

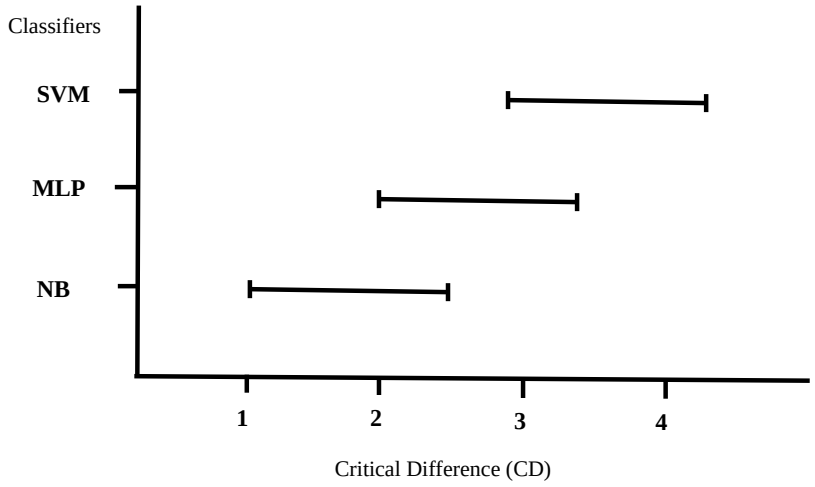


FIGURE 4.5: Visualization of Nemenyi test results from Table 4.5

4.4.4 Comparisons with Other Detection Approaches

With respect to the fourth objective, comparisons with the operation of alternative APS detection mechanisms in the context of the dynamic (monitoring) context with respect to the operation of: (i) L.Polak and Z. Kotulski [65] and (ii) Sedeeq et al. [79] (details in Chapter 3) and with the operation of the approach of W.Jian-feng et al. [95] in the static (non-monitoring) context were conducted. The results are presented and discussed in this sub-section. The Non-Monitoring dataset in Appendix A was used as before. For each APS algorithm half of the dataset was seeded with hidden messages. In the case of L.Polak and Z. Kotulski [65], the detection method was adapted so that feature vectors were generated comprised of W values. Recall that W is the ratio of HTML attribute pairs that occurs in different orderings from the total number of HTML attribute pairings (see Chapter 2). Each webpage was thus represented by a single statistical feature W calculated as described in Chapter 2. In the case of Sedeeq et al.[79], the detection method was adapted so that feature vectors were generated comprised of the standard deviation St.D values as described in Chapter 3. This resulted in each webpage being represented by one feature, the standard deviation of attribute positions.

With respect to the approach of W.Jian-feng et al. [95], this detection approach, as described in Chapter 2, adopted a classification model to classify single webpage snapshots as either clean or stego webpages in the non-monitoring context. Recall that the distinction between the monitoring and non-monitoring context is that in the monitoring context sequence of webpage snapshots are considered while in the non-monitoring context single webpage snapshots are considered. For comparison with W.Jian-feng et al. feature vectors were generated in the same manner as described in Chapter 2 using two statistics: (i) the distance between the attribute mean position benchmark and the sample attribute mean position of the webpage in question and (ii) the variance of the

TABLE 4.6: Operation of L.Polak and Z. Kotulski [65] and (ii) Sedeeq et al. [79] without classifiers

APS detection approach	Deogol [80]		Huang et al. [45]		Shen et al. [81]	
	Acc	AUC	Acc	AUC	Acc	AUC
L.Polak and Z. Kotulski [65]	75.00%	0.74	51.00%	0.51	56.00%	0.56
Sedeeq et al. [79]	81.00%	0.81	51.00%	0.50	56.00%	0.56

attribute positions. Note that with respect to the reimplementation of the detection approach in [95] the same three parameters for attributes selection were used as mentioned in Chapter 2: (i) the number of attributes to be considered (BA), (ii) the number of attribute appearances in terms of mean distance (TD) and (iii) the number of attribute appearance in terms of variance (TV). The values used for these parameters were those recommended in [95]: (i) BA = 3, (ii) TD = 3 and (iii) TV = 3.

Two experiments were conducted the first experiment was to investigate the operation of alternative APS detection mechanisms (i) L.Polak and Z. Kotulski [65] and (ii) Sedeeq et al. [79] (details in Chapter 3) without using the classifiers and the results are shown in Table 4.6. The used detection threshold to distinguish between clean webpages and stego-webpges is computed using the average of the ($St.D_s$) and (W_s) of the stego webpages. The Table shows that the proposed approach of Sedeeq et al. (Chapter 3) performed better than the approach of L.Polak and Z. Kotulski in terms of detection accuracy. Both detection approaches performed poorly in terms of AUC metric.

The second experiment was conducted to compare the operation of APCC with the other detection approaches L.Polak and Z. Kotulski [65], Sedeeq et al. [79] and W.Jian-feng et al.[95], also to investigate the operation of both approaches in L.Polak and Z. Kotulski [65], Sedeeq et al. using the classifiers. For the evaluation, the three APS algorithms considered previously (Deogol, Huang et al. and Shen et al.) and the three classifier generation paradigms considered previously (Neural Network MLP, SVM and Naive Bayes NB) were again used. Thus nine different combinations were considered. The results are presented in Table 4.7.

TABLE 4.7: Comparison of APCC results with other detection approaches (best results highlighted in bold font)

Classifier	**Detection Approach in APS Algorithm**	**Deogol** [80]		**Huang et al.** [45]		**Shen et al.** [81]	
		Acc	AUC	Acc	AUC	Acc	AUC
MLP	L. Polak and Z.Kotulski [65]	74.64%	0.78	47.32%	0.48	44.46%	0.33
	Sedeeq et al.[79]	75.36%	0.88	68.10%	0.84	61.37%	0.65
	W. Jian-feng et al. [95]	86.61%	0.93	78.93%	0.91	77.32%	0.88
	APCC	**93.39%**	**0.97**	**90.36%**	**0.94**	**93.57%**	**0.99**
SVM	L.Polak and Z.Kotulski [65]	61.96%	0.65	48.57%	0.50	47.32%	0.46
	Sedeeq et al. [79]	76.79%	0.77	74.70%	0.74	58.10%	0.59
	W. Jian-feng et al.[95]	88.04%	0.88	81.43%	0.82	75.54%	0.75
	APCC	**90.71%**	**0.90**	**93.21%**	**0.91**	**96.07%**	**0.95**
NB	L.Polak and Z.Kotulski [65]	70.63%	0.75	51.43%	0.47	51.61%	0.53
	Sedeeq et al. [79]	78.04%	0.89	74.70%	0.84	61.01%	0.65
	W. Jian-feng et al.[95]	**91.96%**	**0.99**	**89.64%**	0.91	79.82%	0.89
	APCC	90.89%	0.98	88.75%	**0.96**	**93.57%**	**0.99**

Inspection of Table 4.7 indicates that, with respect to accuracy Acc, the APCC approach produced best results in seven of nine cases; and, with respect to AUC, the best result in eight of the nine cases. It is interesting to note that the proposed approach of Sedeeq et al. (Chapter 3) performed better than the approach of L.Polak and Z. Kotulski (however both detection approaches performed better without classifiers see Table 4.6. From Table 4.7 it is also interesting to note that, although the authors of the W.Jian-feng et al. detection approach considered only SVM classification [95], their detection approach produced a better performance (in some cases better than the proposed APCC algorithm) when Naive Bayes classification was applied.

4.5 Summary

In this chapter the Attribute Position Changes Count (APCC) APS detection approach has been presented. This approach offers the dual advantages that: (i) it serves to capture more detail concerning APS than methods that use average statistical values and (ii) it can be readily used to generate feature vectors with which to train an APS classification model. The evaluation was conducted by considering three alternative APS methods and three classifier generation paradigms (thus three-by-three combinations). Comparisons were presented between the proposed APCC approach and two alternative APS detection approaches in the monitoring context, and between the proposed APCC approach and an alternative APS detection approach in the non-monitoring context. In each case the proposed APCC APS detection approach produced the best results, with respect to accuracy (Acc.) and AUC, in the majority of cases thus indicating the viability of the proposed approach. The next chapter presents the Detect Invisible Characters (DIC) detection approach for Invisible Characters Steganography (ICS) detection.

Chapter 5

A Prediction Model for Invisible Characters Detection

5.1 Introduction

In Chapters 3 and 4 the proposed SD and APCC algorithms were presented, both were directed at Attribute Permutation Steganography (APS). This chapter considers a different form of HTML steganography, Invisible Characters Steganography (ICS) or Open Space Steganography (OSS) both introduced in Chapter 2. More specifically this chapter presents the Detect Invisible Characters (DIC) detection approach. The mode of operation of this approach is similar to the APCC non-monitoring approach presented in the previous chapter. The idea is to build a classifier that uses frequency distribution of continuous sequences (segments) of white space characters, of different lengths, to distinguish between normal webpages and stego webpages.

Recall from Chapter 2 that HTML browsers ignore space characters when rendering webpages. ICS utilizes this feature of HTML browsers for message (data) hiding. The proposed DIC approach, presented in this chapter, is based on an idea first suggested in Sui and Luo [68], namely that the embedding of a message using white space characters will affect the frequency distributions of continuous white space characters used in sequences; an observation that holds regardless of the adopted ICS method used to hide data. The main challenge of ICS detection in WWW pages is the large number of white space characters that will normally exists, regardless of whether embedding has taken place or not.

Two previously proposed alternative ICS detection approaches, that of Sui and Luo [68] and Huang et al.[43] were described in Chapter 2. Both approaches used probabilistic models to identify HTML ICS. Both were used for comparison purposes with respect to the evaluation of the proposed DIC approach (reported on later in this chapter).

As also noted previously in Chapter 2, there are several steganography tools that can be used to hide messages in HTML files using ICS. Of these wbStego4open and SNOW are freely available for download and hence were used in the context of the

evaluation presented later in this chapter. SNOW utilizes end-of-line spacing to hide data, whilst wbStego4open makes use of both inter-word spacing and inter-sentence-spacing to embed data. Unlike SNOW, the wbStego4open tool also checks that the cover file is large enough to accommodate the desired hidden message.

The proposed DIC approach used the Non-Monitoring dataset in Appendix A to generate two WWW datasets. Two evaluation datasets were derived from this dataset, the first seeded with hidden messages using wbStego4open and the second seeded with hidden messages using SNOW. These two datasets were used to evaluate the proposed DIC approach detailed later in this chapter. As in the case of the APCC approach described in the foregoing chapter, the operation of the proposed DIC approach was also considered in the context of three different classifier generation models: (i) Neural Networks, (ii) SVM and (iii) Naive Bayes with the same settings in WEKA software (for MLP classifier the parameters were as follows: one hidden layer, learning rate=0.3, momentum=0.2 and number of epoches=500, while SVM classifier parameters were: complexity parameter c=1 and tolerance parameter=0.001).

The rest of this chapter is organized as follows: Section 5.2 presents an analysis of the alternative ICS detection approaches considered later in this chapter with respect to the evaluation of the DIC approach. Section 5.3 provides a full description of the idea behind the DIC method. Section 5.4 then presents the evaluation of the proposed approach. Finally Section 5.5 concludes the chapter with a brief summary.

5.2 Analysis of Existing Methods

As noted in Chapter 2, previous work on ICS detection can be found in Sui and Luo [68] and Huang et al. [43]. The work presented in this section commences with an analysis of these existing approaches in the context of whether or not their operation could be improved upon, and if so how this might be achieved. To analyze the two approaches ICS was applied using both SNOW and wbStego4open to half of the the Non-Monitoring dataset in Appendix A using a 72 characters English language text. In this manner two datasets were generated, D_{ws} and D_{snow}, of 150 webpages each.

Recall, from Chapter 2, that Sui and Lou used two occurrence probabilities: (i) p_{tsco}, the probability of a white space character occurrence; and (ii) p_{scso}, the probability of a white space character sequence occurrence. The value of p_{tsco} and p_{scso} is calculated using Equations 5.1 and 5.2 respectively (repeated from Chapter 2).

$$p_{tsco}(W) = \frac{N_{ws}}{N_{allchar}} \tag{5.1}$$

$$p_{scso}(W) = \frac{N_{wss}}{N_{ws}} \tag{5.2}$$

where:

- N_{ws}: Is the number of white space characters in W webpage.
- $N_{allchar}$: Is the number of all characters in W webpage.
- N_{wss}: Is the number of space character sequences in W webpage.

The idea thus was to calculate these probabilities before and after suspected embedding of a hidden message. If no embedding had taken place the before and after values would of course be the same. Table 5.1 and Table 5.2 show the before and after embedding values of p_{tsco} and p_{scso} obtained with respect to six of the sample webpages in D_{ws} and D_{snow} where message hiding had taken place. Column 1 in 5.1 gives the white space character occurrence probability p_{tsco} before embedding while Column 1 in 5.2 gives the white space character sequence occurrences probability p_{scso}. Both tables give the same probabilities in columns 3 and 4 after embedding had taken place using ICS tools wbStego4open and SNOW, respectively. From the tables it can clearly be seen that all probabilities increased after embedding had taken place and there is a wide variation in the range of p_{tsco} and p_{scso} values obtained. A full analysis with respect to both datasets confirmed this result. A summary of this analysis, with respect to the ICS seeded WWW pages, is given in Figure 5.1 and Figure 5.2. Each figure shows two "box plots" one for each set of ICS seeded WWW pages. Note that a box plot is a diagram used for displaying the distribution of data in terms of five limits [1]: (i) minimum, (ii) first quartile, (iii) median, (iv) third quartile and (v) maximum. Inspection of both figures confirms that the range of p_{tsco} and p_{scso} values is substantial. Thus it would seem that using a p_{tsco} and p_{scso} static threshold values to detect ICS, as proposed by Sui and Luo[68], is unlikely to provide good ICS detection results because of this variability.

TABLE 5.1: The probability of white space character occurrence probability p_{tsco} in selected webpages before and after embedding

Webpage	p_{tsco} before embedding	D_{ws} p_{tsco}	D_{SNOW} p_{tsco}
www.bbc.co.uk	0.090	0.094	0.092
www.bbc.co.uk /weather	0.171	0.189	0.173
www.linkedin.com	0.032	0.078	0.046
www.cnn.com	0.046	0.053	0.048
www.liverpool.ac.uk	0.118	0.156	0.135
www.wikipedia.com	0.041	0.087	0.057

In the case of the approach proposed in Huang et al. [43], as described in Chapter 2, used the concept of "embedding rate" (e_{rate}). This is the ratio between the length of a hidden message M and the size of a given WWW page $W_{allchar}$. M = size of a given webpage $W_{allchar}$ - size of a given webpage with invisible characters removed ($W_{without}$).

TABLE 5.2: The probability of white space character sequence occurrences p_{scso} in selected webpages before and after embedding

Webpage	p_{scso} before embedding	D_{ws} p_{scso}	D_{SNOW} p_{scso}
www.bbc.co.uk	0.031	0.035	0.032
www.bbc.co.uk /weather	0.055	0.057	0.056
www.linkedin.com	0.002	0.031	0.015
www.cnn.com	0.033	0.036	0.034
www.liverpool.ac.uk	0.078	0.087	0.079
www.wikipedia.com	0.002	0.028	0.014

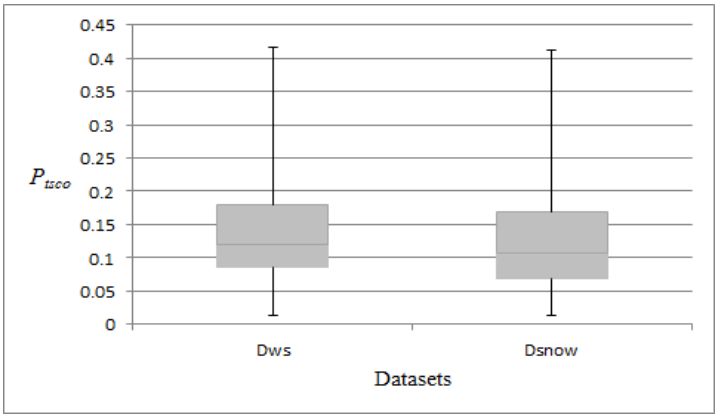


FIGURE 5.1: Box plots for p_{tsco}

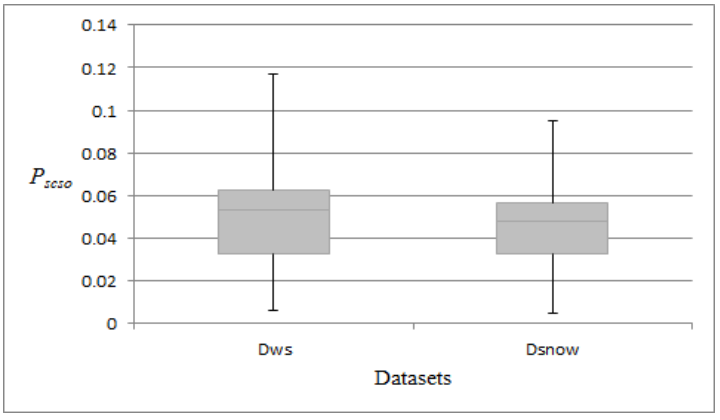


FIGURE 5.2: Box plots for p_{scso}

Embedding rate of a given webpage $e_{rate}(W)$ can be calculated using Equation 5.3.

$$e_{rate}(W) = \frac{M}{W_{allchar}} \tag{5.3}$$

Table 5.3 shows the before and after embedding values of the embedding rate e_{rate} obtained with respect to six of the sample webpages in D_{ws} and D_{snow} where message hiding had taken place. As in Table 5.1 and Table 5.2 Column 1 in Table 5.3 gives

the e_{rate} before embedding. The e_{rate} after embedding had taken place using ICS tools wbStego4open and SNOW is given in Columns 3 and 4 respectively. From the table it also can clearly be seen that all e_{rate} values increased after embedding had taken place and there is a wide variation in the range of e_{rate} values obtained. A full analysis with respect to both datasets confirmed this result. A summary of this analysis, with respect to the ICS seeded WWW pages, is given in Figure 5.3. From the figure it can be seen that the application of a static e_{rate} threshold for detecting ICS, as promoted in Huang et al.[43], is also not ideal.

TABLE 5.3: The embedding rate e_{rate} of selected webpages before and after embedding

Webpage	e_{rate} before embedding	D_{ws} e_{rate}	D_{SNOW} e_{rate}
www.bbc.co.uk	0.091	0.100	0.095
www.bbc.co.uk /weather	0.179	0.224	0.183
www.linkedin.com	0.046	0.138	0.062
www.cnn.com	0.049	0.063	0.053
www.liverpool.ac.uk	0.133	0.217	0.157
www.wikipedia.com	0.055	0.063	0.058

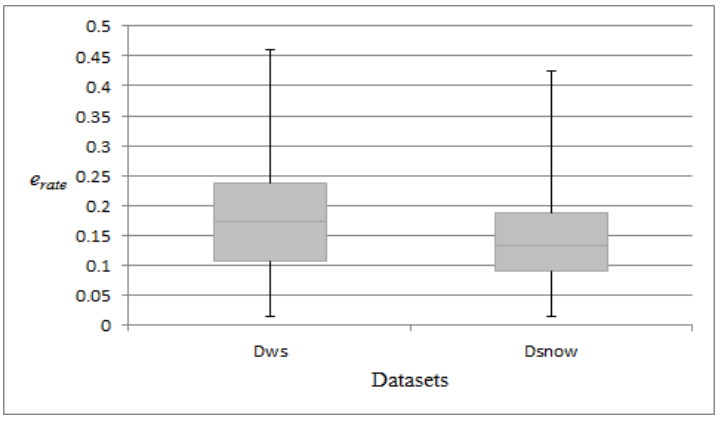


FIGURE 5.3: Box plots for e_{rate}

From the above analysis it is clear that the ICS detection approaches of Sui and Luo and Huang et al. have disadvantages. The proposed DIC algorithm presented in this chapter seeks to address these disadvantages.

5.3 Proposed DIC Detection Method

This section presents the proposed DIC detection method. The idea was to use the frequency distributions of different lengths of sequences of a white space characters as a mechanism for identifying ICS. To the best knowledge of the author the proposed DIC ICS detection approach is unlike any other ICS detection approach presented in the literature. The DIC approach is fully described in this section. However, prior to

TABLE 5.4: First ten white space character segment lengths in selected webpages before and after embedding

Webpage	Before embedding	After embedding (wbStego4open)	After embedding (SNOW)
www.bbc.co.uk	1·1·1·1·1·1·1·1·1·1	1·1·1·1·1·1·1·8·1·1	2·6·5·3·2·3·1·1·1·1
www.dhl.com	1·1·1·1·1·1·1·1·1·1	1·3·1·1·1·1·1·5·1·1	1·1·1·1·1·1·1·7·6·7
www.ieee.org	1·1·1·1·1·1·1·1·1·1	1·1·1·1·1·1·5·1·1·1	1·1·1·1·7·6·1·1·1·1
www.google.com	1·1·1·1·1·1·1·1·1·1	2·1·1·1·1·1·7·1·1·1	6·1·3·3·5·1·1·1·1·1
www.sdwik.net	1·1·1·1·1·1·1·1·1·1	1·1·1·2·1·1·1·1·1·6	1·1·1·1·7·6·7·6·1·1
dailyroutines.typepad.com	1·1·1·1·1·1·1·1·1·1	2·1·2·1·1·1·1·5·1·1	1·1·1·1·6·1·1·1·1·3
www.khanacademy.com	1·1·1·1·1·1·1·8·1·1	4·1·1·1·1·1·4·1·1·1	7·6·7·6·5·3·1·1·1·1
www.sony.com	1·1·1·1·1·1·1·4·4·4	1·3·1·1·1·1·1·5·1·1	7·6·7·6·5·3·1·1·1·1

commencing the description, it should be noted that the term *segment* is used throughout this section to describe contiguous sequences either of white space characters or non-white space characters. Table 5.4 shows the length of the first ten white space character segments before and after embedding for a selection of eight webpages from the Non-Monitoring dataset in Appendix A. The · (dot) symbol used in the table indicates the presence of character segments other than white space segments. From the table it can clearly be seen, as expected, that the length of at least some of the white space segments increases in the presence of ICS. Although some webpages already have space character segments of length more than 1 before any embedding has taken place (the Khan Academy and Sony webpages in the table). However, the table clearly indicates that the idea of using the frequency distribution of different white space segments is worth exploring.

The frequency distribution of a white space character segment of length l in a webpage W, $f_{cscs_l}(W)$, can be calculated using Equation 5.4 where l is the segment length. Note that $\sum_{l=1}^{k} f_{cscs_l}(W) = 1.0$ (where k is the maximum size of the segments featured in W).

$$f_{cscs_l}(W) = \frac{number\ of\ segments\ of\ length\ l\ in\ W}{total\ number\ of\ segments\ in\ W} \tag{5.4}$$

Table 5.5 shows the f_{cscs_5} values, before and after embedding, with respect to some selected webpages from the Non-Monitoring dataset in Appendix A. From the table it can clearly be seen, as expected, that the frequency distributions of the white space segments will increase in presence of ICS. This same phenomena was observed with respect to the remaining webpages (not shown in Table 5.5).

Given the above, the basic idea of using the frequency distribution of white space segments, within webpages, of different length, with and without embedding, as an indicator of ICS seems like a good one. More specifically the idea is to build a binary prediction model (a classifier) generated using a n-dimensional feature space where n is the number of different potential white space segment lengths of interest that might be included in a webpage. The value for each dimension would then be the frequency

TABLE 5.5: f_{cscs_5} for some sample webpages before and after embedding

Webpage	f_{cscs_5} without embedding	f_{cscs_5} after embedding (wbStego4open)	f_{cscs_5} after embedding (SNOW)
www.cnn.com	0.00000	0.03590	0.03990
www.wikipedia.com	0.00000	0.06870	0.15660
www.bbc.co.uk	0.02250	0.05580	0.06120
www.bbc.co.uk/weather	0.00630	0.03180	0.02560
www.adobe.com	0.00080	0.03360	0.02160
www.cisco.com	0.02310	0.05010	0.04320
www.stackoverflow.com	0.00066	0.02999	0.00880
www.sony.com	0.00170	0.03340	0.01760
www.ipod.com	0.00200	0.04330	0.02870
www.nbc.com	0.02300	0.03500	0.02660
www.amazon.com	0.00110	0.03500	0.01520
www.expedia.com	0.00600	0.03660	0.02400

distribution of the indicated white space segment in the given webpage. In this manner a webpage would be defined in terms of a feature vector $V = \{v_1, v_2, \ldots, v_n\}$.

The process for generating a set of feature vectors, given a set of webpages $D = \{w_1, w_2, \ldots, w_m\}$, with respect to the proposes DIC approach, is given by the pseudo code presented in Algorithm 6. The input to the algorithm (line 1) is a set of webpages D and the n the maximum length of white space segment to be considered. Using the algorithm, for each webpage w_i in D, a feature vector V_i of length n is created where the elements are the frequency distribution of white space sequences of lengths 1 to n. The process was used to generate sets of feature vectors for evaluation purposes. Note that for the experimental the Non-Monitoring dataset in Appendix A maximum size of a space character segment was found to be 30, hence $n = 30$.

Note also that for the training data each feature vector has a class label associated with it; class 1 represented a stego webpage while class 2 represented a normal webpage.

Algorithm 6: Feature vector generation

1: Input: A set of webpages $D = \{w_1, w_2, \ldots, w_m\}$, and the maximum size of the white space segments to be considered n
2: Output: A set of feature vectors $\Phi = \{V_1, V_2, \ldots V_m\}$
3: $\Phi = \emptyset$
4: **for all** $w_i \in D$ **do**
5: s = number of white space character segments of length >= 1 in w_i
6: **for** $j = 1$ **to** n **do**
7: t = number of segments in w_i of length j
8: $V_i[j] = \frac{t}{s}$
9: **end for**
10: $\Phi = \Phi \cup V_i$
11: **end for**

5.4 Evaluation

In this section the evaluation of the proposed DIC ICS detection approach is discussed. Two sets of experiments were conducted; using the feature vector represented datasets generated as described above. The first set of experiments was conducted when hidden messages were seeded using wbStego4open and the second set when hidden messages were seeded using SNOW. The objectives of the evaluation were as follows:

1. To determine if the proposed detection approach can be effectively applied to HTML webpage files to detect ICS.
2. To determine the effect of applying a feature selection strategy to the feature vector representation prior to classifier generation.
3. To determine the effect of the nature of embedded message (English language or random) and the message length on classifier performance.
4. To determine if the obtained results were statistically significant or not.
5. To provide a comparison between the proposed detection approach and the approaches to ICS detection of Sui and Lou [68] and Huang et al.[43], as identified in Section 5.2 above.

The first two of the above objectives are considered in Subsection 5.4.1, while the third and fourth in Sub-section 5.4.2 and the fifth in Sub-section 5.4.4.

5.4.1 Effectiveness of the proposed detection approach

The effectiveness of the proposed ICS detection technique was measured in terms of accuracy (Acc.) and AUC. The experiments were conducted using the Non-Monitoring dataset in Appendix A after embedding a message comprised of 72 natural English language characters. The embedding process was conducted using both wbStego4open and SNOW.

Ten Cross Validation TCV was used whereby the dataset was first stratified and then divided into tenths and ten classifiers generated using different nine tenths and tested on the remaining tenth. For the feature selection the "CfsubsetEval" attribute evaluation algorithm, and best first search, as provided in the WEKA machine learning workbench, was used. The idea behind feature selection, as noted in Chapter 2, is to select a subset of the dimensions (recall that each dimension represents an attribute) in the feature space, that are good discriminators of class. The search method used defines how we search the feature space to identify the best attributes. Note that, unlike other feature selection mechanisms, CfsubsetEval does not select the k best dimensions (attributes) but the best performing dimensions according to some threshold. There are many different techniques that can be used for both.

TABLE 5.6: Effectiveness of proposed ICS detection technique using D_{ws} without feature selection

TCV	MLP		SVM		NB	
	Acc.%	AUC	Acc.%	AUC	Acc.%	AUC
1	93.33	1.00	100.00	1.00	1.00	1.00
2	100.00	1.00	93.33	0.94	93.33	0.88
3	86.67	0.96	93.33	0.94	73.33	0.65
4	100.00	1.00	100.00	1.00	100.00	1.00
5	93.33	1.00	93.33	0.94	86.67	0.96
6	86.67	0.80	86.67	0.87	86.67	0.88
7	93.33	0.96	100.00	1.00	86.67	0.86
8	100.00	1.00	100.00	1.00	93.33	1.00
9	86.67	0.86	93.33	0.93	73.33	0.88
10	80.00	0.82	93.33	0.94	86.67	0.95
Average	92.00	0.94	**95.33**	**0.95**	88.00	0.90
SD	6.89	0.08	4.50	0.04	9.32	0.11

TABLE 5.7: Effectiveness of proposed ICS detection technique using D_{SNOW} without feature selection

TCV	MLP		SVM		NB	
	Acc.%	AUC	Acc.%	AUC	Acc.%	AUC
1	86.67	0.89	86.67	0.88	80.00	0.84
2	100.00	1.00	100.00	1.00	66.67	0.78
3	80.00	0.88	86.67	0.87	60.00	0.46
4	80.00	0.96	93.33	0.93	93.33	1.00
5	100.00	1.00	93.33	0.93	86.67	0.84
6	93.33	1.00	80.00	0.79	73.33	0.70
7	93.33	0.98	80.00	0.81	66.67	0.63
8	86.67	0.98	80.00	0.81	80.00	0.79
9	80.00	0.84	86.67	0.86	66.67	0.68
10	86.67	0.88	73.33	0.73	60.00	0.68
Average	**88.67**	**0.94**	86.00	0.86	73.33	0.74
SD	7.73	0.06	7.98	0.08	11.33	0.15

The results are presented in Tables 5.6 to 5.9 (best results highlighted in bold font). Tables 5.6 and 5.7 show the results obtained without feature section, whilst Tables 5.8 and 5.9 the results obtained with feature selection. A summary is presented in Table 5.10. From the summary table it can be seen that better results were obtained when using feature selection than without feature selection. With respect to the D_{ws} dataset SVM produced consistently the best accuracy, whilst best AUC was produced using SVM and NB with feature selection. In the case of the D_{SNOW} dataset best results were obtained consistently using MLP.

With respect to the feature selection it is interesting to note that the selected values of l, the length of white space segments with respect to D_{ws}, were $\{1, 2, 3, 4, 5, 6, 7, 8, 21\}$, the value of 21 seems odd and might simply be an "outlier". In the case of D_{SNOW} the selected values for l were $\{1, 2, 5, 7\}$.

TABLE 5.8: Effectiveness of proposed ICS detection technique using D_{ws} with feature selection

TCV	MLP		SVM		NB	
	Acc.%	AUC	Acc.%	AUC	Acc.%	AUC
1	93.33	1.00	100.00	1.00	100.00	1.00
2	100.00	1.00	100.00	1.00	93.33	0.88
3	93.33	0.96	93.33	0.94	93.33	0.96
4	100.00	1.00	100.00	1.00	100.00	1.00
5	93.33	0.98	93.33	0.94	100.00	1.00
6	86.67	0.98	93.33	0.94	93.33	1.00
7	93.33	1.00	100.00	1.00	93.33	1.00
8	100.00	1.00	100.00	1.00	100.00	1.00
9	93.33	0.86	93.33	0.93	93.33	1.00
10	86.67	1.00	93.33	0.94	86.67	0.98
Average	94.00	**0.98**	**96.67**	0.97	95.33	**0.98**
SD	4.92	0.04	3.51	0.03	4.5	0.04

TABLE 5.9: Effectiveness of proposed ICS detection technique using D_{SNOW} with feature selection

TCV	MLP		SVM		NB	
	Acc.%	AUC	Acc.%	AUC	Acc.%	AUC
1	93.33	0.89	86.67	0.87	73.33	0.89
2	100.00	1.00	86.67	0.86	73.33	1.00
3	86.67	0.93	80.00	0.79	69.33	0.77
4	100.00	1.00	93.33	0.93	93.33	1.00
5	100.00	1.00	100.00	1.00	100.00	1.00
6	100.00	1.00	100.00	1.00	80.00	1.00
7	86.67	0.96	73.33	0.75	73.33	1.00
8	100.00	1.00	80.00	0.81	73.33	1.00
9	93.33	0.86	86.67	0.87	86.67	0.86
10	93.33	0.88	93.33	0.94	66.67	0.88
Average	**95.33**	**0.95**	88.00	0.88	78.89	0.94
SD	5.49	0.06	8.78	0.08	12.19	0.08

TABLE 5.10: Summary of results presented in Tables 5.6 to 5.9 (best results highlighted in bold font)

Data Set	Feature Selection	MLP		SVM		NB	
		Acc.%	AUC	Acc.%	AUC	Acc.%	AUC
D_{ws}	No	92.00	0.94	95.33	0.95	88.00	0.90
	Yes	94.00	**0.98**	**96.67**	0.97	95.33	**0.98**
D_{SNOW}	No	88.67	0.94	86.00	0.86	73.33	0.74
	Yes	**95.33**	**0.95**	88.00	0.88	78.89	0.94

5.4.2 The effect of Different Message Type and Length Embedding

To investigate the effect of the nature of the hidden message, and its length, on the ICS detection two sets of experiments were conducted. The first used English language embedded messages while the second random messages (a mixture of uppercase, lowercase letters and numbers). For both sets of experiments:

- Messages of different lengths were embedded from between 20 to 100 characters,

incrementing of steps of 20 characters. The length of a hidden message this time is absolute not relative to the MEC because as mentioned in Appendix A that ICS tools provide high embedding capacity. A relative increment of a hidden message length (for example 30%, 40%. ..., 100%) with respect to the MEC would result in very long messages that can be easily detected.

- The wbStego4open and SNOW ICS tools were used to embed messages.
- For each message length ten selected (English language and random) messages were embedded.
- Webpages were represented using a thirty attributes feature vector that containing the white space character segment frequency distributions. These feature vectors were the inputs to the classifiers (with class labels for training).
- Two performance metrics were recorded: (i) Accuracy (Acc) and (ii) Area Under Curve (AUC).
- Three classification models were used: (i) MLP (ii) SVM and (iii) NB.

As a consequence of the above twenty datasets were generated in total using wbStego4open ICS tool grouped into four categories: (i) webstegoNatural1, (ii) webstegoNatural2, (iii) webstegoRandom1 and (iv) webstegoRandom2. Categories webstegoNatural1 and webstegoNatural2 comprised English language message embedding datasets generated using wbStego4open with and without feature selection respectively; whilst categories webstegoRandom1 and webstegoRandom2 comprised random message embedding datasets generated using wbStego4open with and without feature selection respectively. Each category comprised five datasets, each including messages of a different length. The same four categories are generated this time using SNOW ICS tool.
Figures 5.4 and 5.5 present the results obtained using AUC metric. The results showed that the nature of the embedded messages (natural English language or random) has no differential effect on the performance of the proposed DIC ICS detection approach. However, the length of the embedded message did influence the performance of the classifiers where wbStego4open ICS has been used for messages hiding. This influence is clearer with respect to the recorded performance of the MLP and NB classifiers than in the case of the SVM classifier. From the figures it can also be noted that when feature selection is applied this improved the performance of the MLP and NB classifiers than in the case of the SVM classifier. With respect where SNOW ICS has been used for messages hiding it can be noted that from Figure 5.5 the recorded performance of the MLP is better than in the case of the SVM and NB classifiers. The reason behind this is that MLP classifier can support different data types and learn the important features from any data structure.

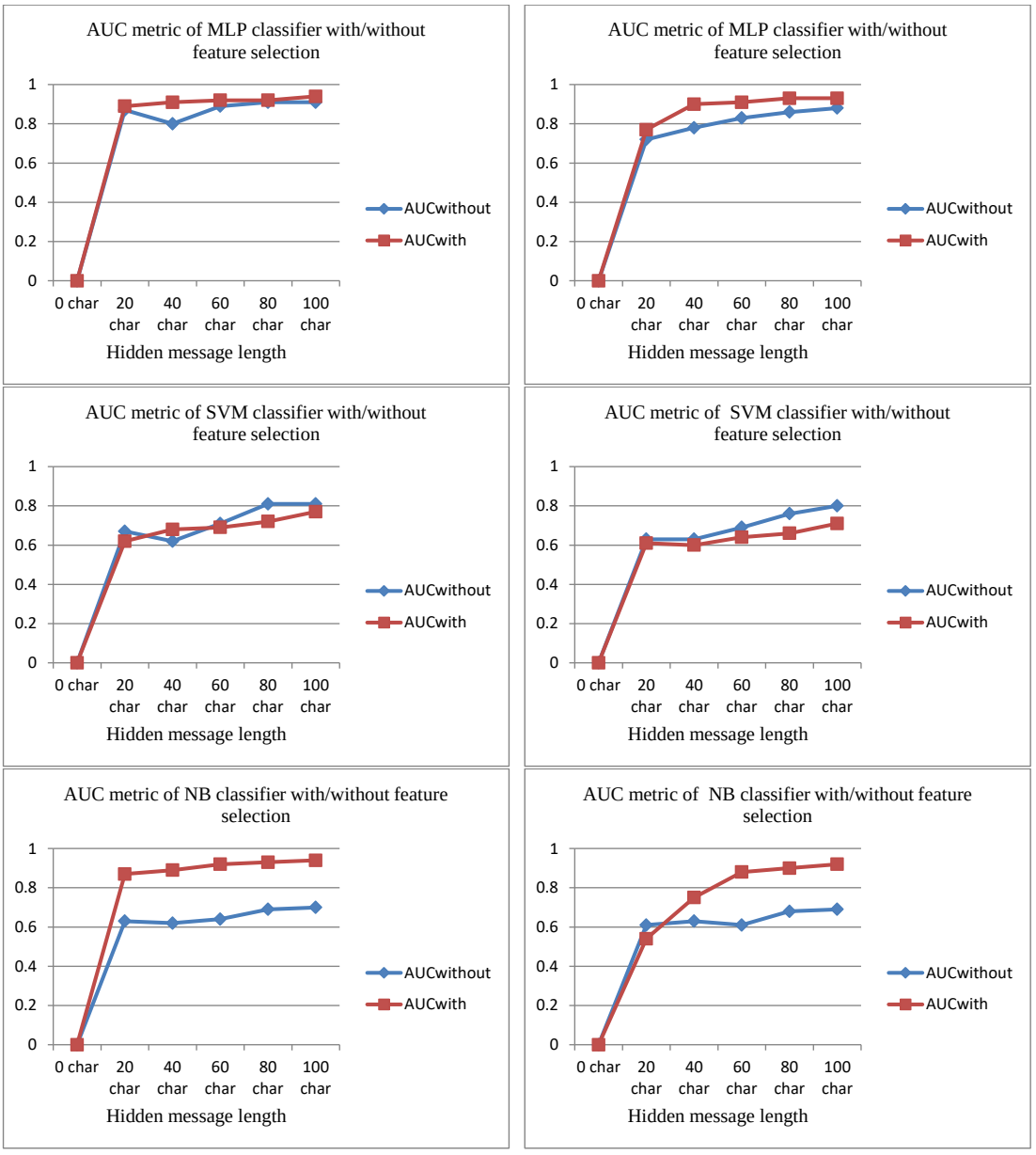


FIGURE 5.4: Operation of proposed DIC ICS detection approach using wbStego4open. Left: English language messages embedding. Right: Random text embedding.

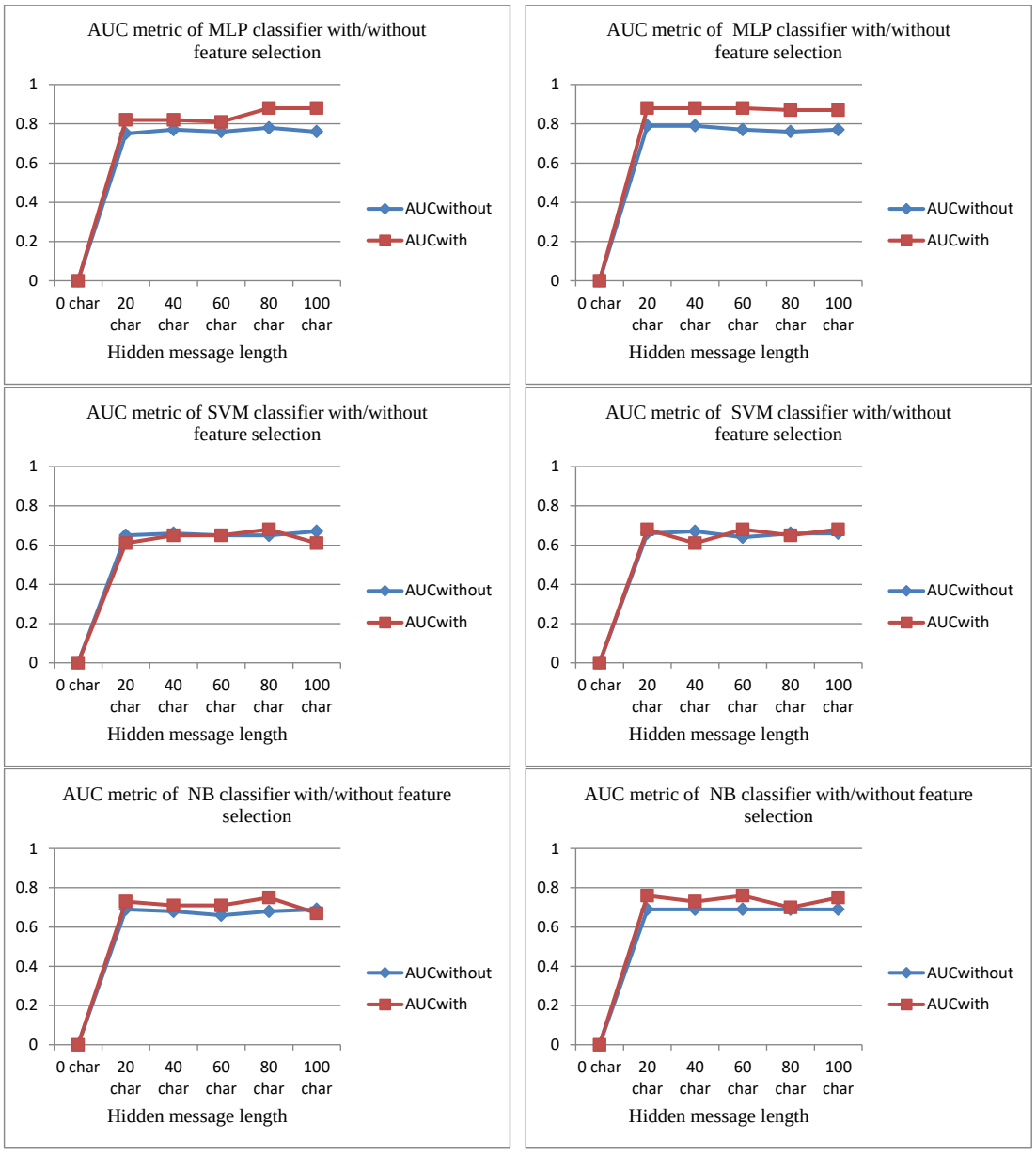


FIGURE 5.5: Operation of proposed DIC ICS detection approach using SNOW. Left: English language messages embedding. Right: Random text embedding.

5.4.3 Statistical Evaluation

To determine whether the obtained results presented in Figures 5.4 to 5.5, using wbStego4open and SNOW, were indeed statistically significant and not simply a matter of chance the Friedman Test was conducted (in similar manner to that described in Chapter 4). The AUC values obtained with respect to the above reported experiments were used for this purpose. The outcomes are presented in Tables 5.11 and 5.12, respectively.

TABLE 5.11: Friedman test results using AUC values for wbStego4open message embedding

Main Datasets	Sub-Data Sets	Feature Selection	Embedded Messages	MLP	SVM	NB	Chi-Sequare	p-value
				AUC				
webstegoNatural1	D_{ws20}	No	Natural English Text	0.87	0.67	0.63	9.579	0.008
	D_{ws40}			0.80	0.62	0.62		
	D_{ws60}			0.89	0.71	0.64		
	D_{ws80}			0.91	0.81	0.69		
	D_{ws100}			0.91	0.81	0.70		
Average Rank				0.87	0.72	0.65		
webstegoNatural2	D_{ws20}	Yes	Natural English Text	0.89	0.62	0.87	8.444	0.015
	D_{ws40}			0.91	0.68	0.89		
	D_{ws60}			0.92	0.69	0.92		
	D_{ws80}			0.92	0.72	0.93		
	D_{ws100}			0.94	0.77	0.94		
Average Rank				0.92	0.69	0.91		
webstegoRandom1	D_{ws20}	No	Random Text	0.72	0.63	0.61	9.579	0.008
	D_{ws40}			0.78	0.63	0.63		
	D_{ws60}			0.83	0.69	0.61		
	D_{ws80}			0.86	0.76	0.68		
	D_{ws100}			0.88	0.80	0.69		
Average Rank				0.81	0.70	0.64		
webstegoRandom2	D_{ws20}	Yes	Random Text	0.77	0.61	0.54	8.400	0.015
	D_{ws40}			0.90	0.60	0.75		
	D_{ws60}			0.91	0.64	0.88		
	D_{ws80}			0.93	0.66	0.90		
	D_{ws100}			0.93	0.71	0.92		
Average Rank				0.88	0.64	0.79		

In Tables 5.11 and 5.12 Column 1 lists the the dataset category as mentioned in previous section, whilst Column 2 lists the 5 datasets (each featuring a different embedded message length) included in each category. Column 3 indicates if feature selection was been applied or not. The nature of the embedded message is shown in Column 4. The *AUC* values associated with the MLP, SVM and NB classification models are given next. The values in Columns 8 and 9 give the Chi-Square Friedman test statistics and the associated p-value.

From Table 5.11 it can be seen that in the case of wbStego4open all the obtained p-values are less than the significance level of $\alpha = 0.05$, therefore the null-hypothesis $H0$ can be rejected. It can also be noted from Table 5.11 that regardless of whether feature selection was applied or not, the best performance was obtained using MLP classification. Since as mentioned that $H0$ can be rejected accordingly the Nemenyi test was conducted using the same parameters (K, N and q) as in the previous chapter. The Critical Difference (CD) was again = 1.4 (as in the previous chapter). Figure 5.6 presents a visualization of the Nemenyi test for results in Table 5.11 for both cases with/without feature selection. The figure lists again each classification model along with its CD

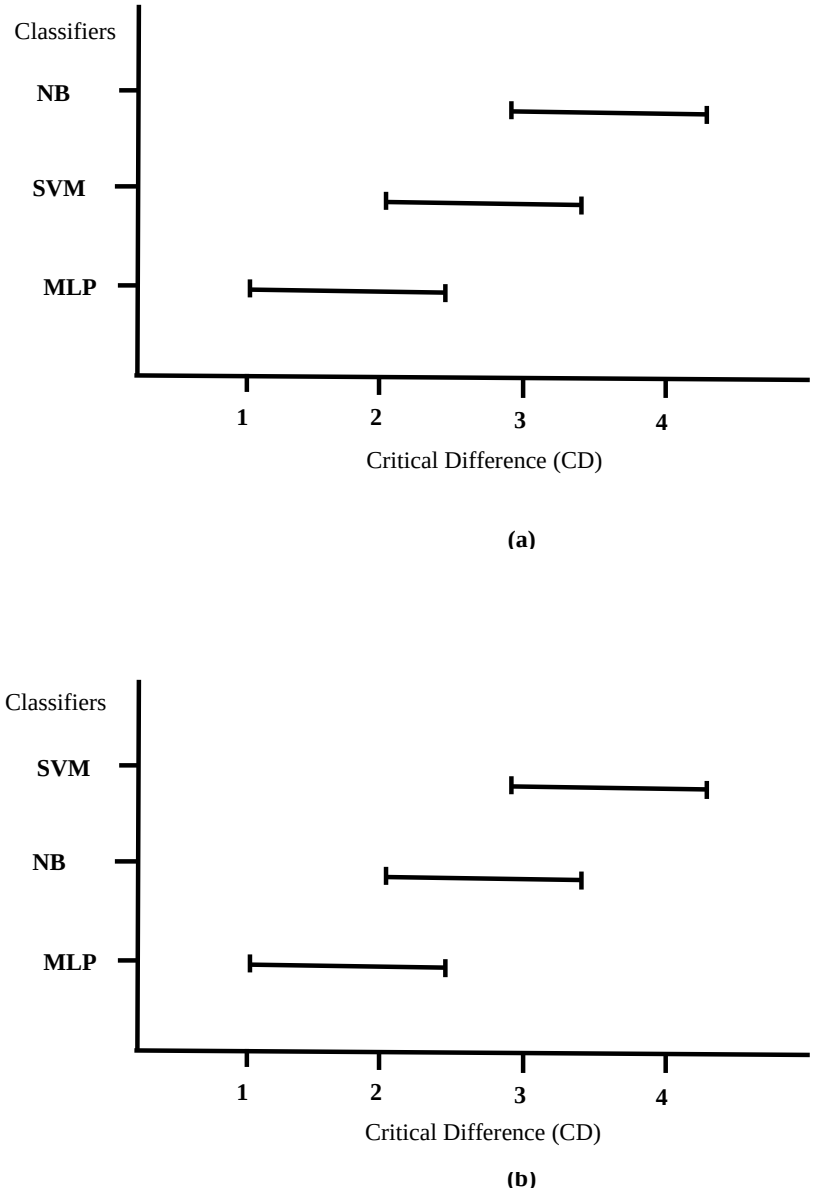


FIGURE 5.6: Visualization of Nemenyi test for results given in Table 5.11 (a): no feature selection is applied (b): feature selection is applied

value. When no feature selection was applied MLP performance was significant only with respect to NB (the CD tails do not overlap). When feature selection was applied MLP classifier performance was significant only with respect to SVM classification.

With respect to the SNOW ICS message embedding tool Table 5.12 presents the Friedman test results. From the table it can be seen that $p-value < \alpha = 0.05$, therefore the null-hypothesis ($H0$) can be rejected and the Nemenyi test was applied. The same CD value of 1.4 was used. A visualization of Nemenyi test for results in Table 5.12 is presented in Figure 5.7. From the figure it can be observed that the best performance

with/without applying feature selection was obtained using MLP classifier and this was only significant for SVM classifier (the CD tails do not overlap).

TABLE 5.12: Friedman test results using AUC values for SNOW of natural and random message embedding with and without feature selection

Main Datasets	Sub-Data Sets	Feature Selection	Embedded Messages	MLP	SVM	NB	Chi-Square	p-value
				AUC				
snowNatural1	D_{Snow20}	No	Natural English Text	0.75	0.65	0.69	10.000	0.007
	D_{Snow40}			0.77	0.66	0.68		
	D_{Snow60}			0.76	0.65	0.66		
	D_{Snow80}			0.78	0.65	0.68		
	$D_{Snow100}$			0.76	0.67	0.69		
Average Rank				0.76	0.65	0.68		
snowNatural2	D_{Snow20}	Yes	Natural English Text	0.82	0.61	0.73	10.000	0.007
	D_{Snow40}			0.82	0.65	0.71		
	D_{Snow60}			0.81	0.65	0.71		
	D_{Snow80}			0.88	0.68	0.75		
	$D_{Snow100}$			0.88	0.61	0.67		
Average Rank				0.84	0.64	0.71		
snowRandom1	D_{Snow20}	No	Random Text	0.79	0.66	0.69	10.000	0.007
	D_{Snow40}			0.79	0.67	0.69		
	D_{Snow60}			0.77	0.64	0.69		
	D_{Snow80}			0.76	0.66	0.69		
	$D_{Snow100}$			0.77	0.66	0.69		
Average Rank				0.77	0.65	0.69		
snowRandom2	D_{Snow20}	Yes	Random Text	0.88	0.68	0.76	10.000	0.007
	D_{Snow40}			0.88	0.61	0.73		
	D_{Snow60}			0.88	0.68	0.76		
	D_{Snow80}			0.87	0.65	0.70		
	$D_{Snow100}$			0.87	0.68	0.75		
Average Rank				0.86	0.66	0.74		

5.4.4 Comparison with other detection approaches

For the fifth and final objective, comparison with existing techniques, the comparison was conducted with respect to the performance of the approaches of Sui and Luo [68] and Huang et al.[43] (both previously described in Section 5.2). So that a fair comparison could be arrived at the evaluation was again conducted using TCV, although it should be noted that the approaches proposed by Sui and Lou and Huang et al. are both statistical in nature and do not require any training. The intuition here was that the evaluation would provide an unfair advantage to the proposed DIC system if it were trained on the entire dataset and then tested on the same dataset (it might also result in "overfitting"). For comparison both SVM and MLP classification with feature selection were used with respect to the proposed method, as these had been shown to produce the best performance. For both detection approaches the reported threshold values were adopted. The evaluation metrics used were again accuracy and AUC.

The results are presented in Tables 5.13 and 5.14. Note that the accuracy values for the proposed approach have been reproduced from Tables 5.8 and 5.9 respectively. From the tables it can clearly be seen that the proposed approach outperformed the previously proposed approaches by a significant margin. The proposed approach was good at identifying ICS webpages while at the same time using both approaches Sui and Luo and Huang et al. was poor. Given that the ICS and non-ICS classes were equally

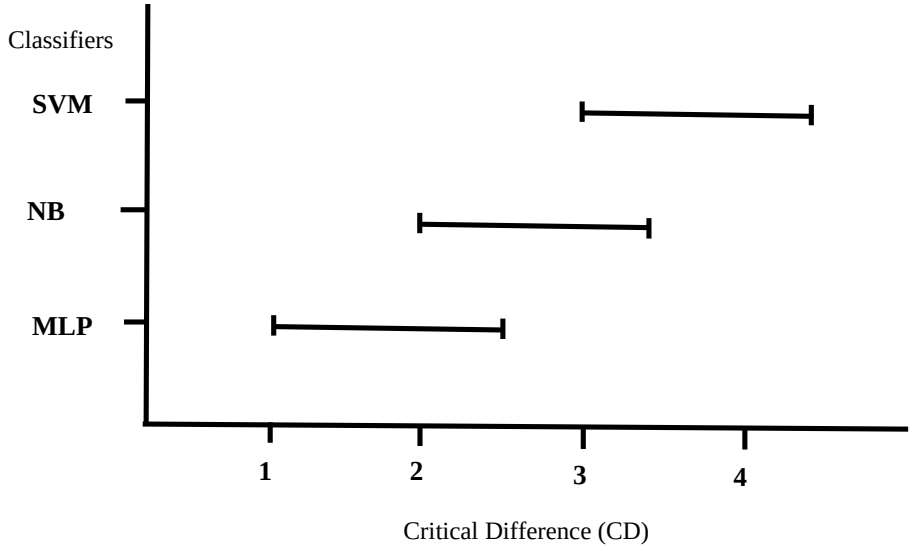


FIGURE 5.7: Visualization of Nemenyi test for results given in Table 5.12

distributed within the dataset the accuracies obtained using Sui and Luo, and Huang, with message embedding using wbStego4open was 54.69% and 50.01% respectively, in other words little better than a guess. In the case of message embedding using SNOW accuracies of 51.00% and 46.00% were obtained, the last one worse than random guessing. IN both cases of detection Sui and Luo, and Huang the average AUC was 0.2.

TABLE 5.13: Comparison of proposed DIC approach to ICS compared with Sui and Luo, and Huang, using wbStego4open message embedding (best results highlighted on bold font)

TCV	DIC Approach		Sui and Luo		Huang et al.	
	Acc.%	AUC	Acc.%	AUC	Acc.%	AUC
1	100.00	1.00	53.00	0.54	53.33	0.56
2	100.00	1.00	60.00	0.58	60.00	0.63
3	93.33	0.94	53.33	0.52	33.33	0.36
4	100.00	1.00	60.00	0.57	53.33	0.56
5	93.33	0.94	40.00	0.38	47.00	0.50
6	93.00	0.94	47.00	0.47	73.00	0.75
7	100.00	1.00	47.00	0.47	33.33	0.38
8	100.00	1.00	60.00	0.59	53.33	0.56
9	93.33	0.93	53.33	0.53	47.00	0.50
10	100.00	0.94	73.33	0.73	47.00	0.50
Average	**96.67**	**0.97**	54.69	0.54	50.01	0.53
SD	3.51	0.03	8.79	0.08	11.16	0.11

TABLE 5.14: Comparison of proposed DIC approach to ICS detection with Sui and Luo, and Huang et al., using SNOW message embedding (best results highlighted on bold font)

TCV	DIC Approach		Sui and Luo		Huang et al.	
	Acc.%	AUC	Acc.%	AUC	Acc.%	AUC
1	93.33	0.89	46.67	0.46	46.67	0.49
2	100.00	1.00	53.33	0.51	60.00	0.63
3	86.67	0.93	53.33	0.52	33.33	0.36
4	100.00	1.00	60.00	0.57	53.33	0.56
5	100.00	1.00	40.00	0.38	40.00	0.34
6	100.00	1.00	40.00	0.40	73.33	0.75
7	86.67	0.96	46.67	0.45	26.67	0.29
8	100.00	1.00	60.00	0.59	53.33	0.56
9	93.33	0.86	46.67	0.46	40.00	0.43
10	93.33	0.88	66.67	0.66	46.67	0.50
Average	**95.33**	**0.95**	51.00	0.50	47.00	0.49
SD	5.49	0.06	0.08	0.08	0.13	0.13

5.5 Summary

In this chapter the proposed DIC approach for ICS detection has been presented. The fundamental idea of the approach was founded on the observation that the length of white space segments in webpages increases in the presence of ICS, and that this information can be captured in a feature vector format and subsequently used to build an ICS prediction (classification) model provided available pre-labeled training data. To evaluate the proposed mechanism the Non-Monitoring dataset in Appendix A is used. This was split into two, half seeded with an embedded messages and half not. In this manner two 150 webpage test datasets were created, one using the wbStego4open ICS tool and one using the SNOW ICS tool. The proposed approach was tested using three well known classifier generation models and with and without feature selection. A best average accuracy of 96.7% was obtained indicating the effectiveness of the proposed approach compared with the previously proposed approaches of Sui and Luo [68], and Huang et al. [43]. In the next chapter the proposed TV approach is presented which was designed to address Tag Letters Case Switching Steganography (TLCSS) detection.

Chapter 6

Tag Letters Case Switching Detection Using Tag Variance

6.1 Introduction

In this chapter the Tag Variance (TV) mechanism for detecting Tag Letters Case Switching Steganography (TLCSS) is presented. Recall that the fundamental idea of TLCSS (see Chapter 2) is to switch the case of tag letters from upper to lower case and vice versa so as to hide messages. Recall also that the switching has no effect on the way that a webpage is rendered by a browser; so an ideal steganography method.

The proposed TV approach operates by comparing the tag letters case variance between the current tag letter cases and the situations where some of the tag letters are switched to either to upper case or to lower case. The idea being to simulate TLCSS embedding. If as result the "smoothness" of the tag decreases TLCSS is considered to have existed. The proposed approach is an alternative to the Tag Offset approach to TLCSS detection proposed by Huang et al. [44] as also described in Chapter 2. The mechanism of Huang uses an aggregate distance measure, the Tag Offset. However, using this approach the identification of short messages becomes challenging because their presence is not readily highlighted using the Tag Offset measure. Part of the motivation underpinning the proposed TV approach presented in this chapter was thus to address the detectection short messages embedded using TLCSS. The motivation was also to try to identify a TLCSS detection mechanism that produced better overall results than the Tag Offset mechanism.

The remainder of this chapter is organized as follows: Section 6.2 presents the proposed TV approach to detecting TLCSS. Calculation of the Maximum Embedding Capacity (MEC) when using TLCSS is presented in Section 6.3 before considering the evaluation of the proposed approach in Section 6.4. The evaluation was conducted by comparing the operation of the proposed TV approach with the Tag Offset approach using the TLCSS embedding mechanisms of Sui and Luo [87] and Shen Y. [96] (discussed previously in Chapter 2). Recall that Sui and Luo utilized all tag letters for embedding

a message while Shen Y used only the first letter. The chapter is concluded with a brief summary presented in Section 6.5.

6.2 Identification of TLCSS using Tag Variance

This section presents the proposed TV approach to HTML TLCSS detection. As noted in the introduction to this chapter the TV approach operates using a measure of the change in distribution (spread) of upper case and lower case letters in a tag, a measure referred to as the Tag Variance. The process of calculating Tag Variance, in contrast to the Tag Offset measure proposed by Huang, is presented in Sub-section 6.2.1.

The calculated Tag Variances are then used to calculate a set of statistics, referred to as the Regular Singular (RS) statistics. The same mechanism was used by Huang et al. (who in turn took the idea from work on image steganography presented in [35]). The mechanism was briefly described in Chapter 2; however, because of its significance with respect to the work presented in this chapter, the process for generating the RS statistics is described in more detail here.

The RS statistics are a set of four measures, $\{|R_{M+}|, |R_{M-}|, |S_{M+}|, |S_{M-}|\}$, where R_{M+}, R_{M-}, S_{M+} and S_{M-} are sets of tags $\{T_1, T_2, \dots\}$. The sets of tags are generated using what is referred to as *positive flipping* and *negative flipping*. The difference between the proposed Tag Variance and the Tag Offset measures is given in Sub-section 6.2.1. The concept of flipping is discussed in further detail in Sub-section 6.2.2. The generation of the final RS statistics is then discussed in Subsection 6.2.3. To summarize the work presented in this section the entire TV process is enumerated in Sub-section 6.2.4

Before continuing with the remainder of the section it should be noted that throughout the section we defined the letters in a tag using a list $T = x_1, x_2, \dots, x_n$ where x_i is the ASCII code for the letter at position i in the tag and n is the total number of letters in the given tag.

6.2.1 Tag Variance versus Tag Offset

This section presents further detail concerning the calculation of Tag Variance. Before providing any further detail it should be noted that both Tag Variance and Tag Offset are designed to be measures describing the homogeneity or smoothness (the term dispersion is also sometimes used) of a set of tag letters T; the mix of uppercase and lower case letters represented in T. In the case of Huang Tag Offset was calculated as follows:

$$Tag\ Offset(T) = \sum_{i=1}^{n-1} |x_{i+1} - x_i| \tag{6.1}$$

Thus a low Tag Offset value will indicate a smooth (homogeneous) tag, while a high offset will indicate the reverse. Note that the Tag Offset measure is a simple distance measure, which does not take into consideration the "spread" from the average ambient value.

The proposed Tag Variance measure takes this into consideration and is calculated as follows:

$$Tag\ Variance(T) = \frac{\sum_{i=1}^{n} (x_i - \mu)^2}{n} \tag{6.2}$$

Where: μ is the mean of the values in $T = x_1, x_2, \ldots, x_n$. Thus tag variance takes into consideration the spread of uppercase and lowercase letters with respect to the average. As in the case of Tag Offset a low Tag Variance value will indicate a smooth tag and a high value a non-smooth tag.

6.2.2 Flipping Functions and Masks

Both the proposed TV approach and Huang's approach use the concept of flipping functions. Two types flipping are considered: (i) positive flipping using a function $f_{+1}(T_i)$ and (ii) negative flipping using a function $f_{-1}(T_i)$. In the case of positive flipping selected lower case letters in a tag T_i are switched to upper case letters; in the case of negative flipping selected upper case letters in a tag T_i are switched to lower case letters. A further "flipping" function used is the $f_0(T_i)$ function which does not do any letter case changing (flipping without flipping!). To decide which letters to flip the concept of a mask is used. A mask in this context is a vector, of length equal to the input tag length n, with each element corresponding to a letter in the input tag T_i.

The values for the elements in a mask are set according to which flipping function is to be applied. In case of $f_{+1}(T_i)$ a positive mask $M+$ is generated with the elements set to 0 where there is to be no change, and 1 where a lower case letter is to be switched to an upper case letter. In the case of $f_{-1}(T_i)$ a negative mask $M-$ is generated with the elements set to 0 if there is to be no change, and -1 where an upper case letter is to be switched to a lower case letter. Note that for TLCSS detection the locations of 1 and -1 values in both the $M+$ and $M-$ masks are the same. In the case of the f_0 function the mask represents that current case distribution in the tag. The output from a flipping function is an *altered tag*, T_i', that will be the input to the Tag Variance calculation (as described above).

6.2.3 Calculation of RS statistics

The Tag Variance of the tags in a webpage to be tested in their current form and once they have been altered are used to determine the RS statistics. The RS statistics are calculated by generating sets of tags using both positive and negative flipping, and predefined masks $M+$ and $M-$, as described above. In the case of positive flipping three sets of tags are produced $\{R_{M+}, S_{M+}, U_{M+}\}$ (Regular, Singular and Unchanged), generated as follows:

1. Regular tags $R_{M+} = \{T_1, T_2, \ldots\}$, the set of tags where the tag variance after flipping is greater than the tag variance before flipping.

$$R_{M+} = \{T(x_1, x_2...., x_n) | TagVariance(T'_M(x_1, x_2, ..., x_n)) > TagVariance(T(x_1, x_2, ..., x_n))\}.$$

2. Singular tags $S_{M+} = \{T_1, T_2, \dots\}$, the set of tags where the tag variance after flipping is less than the tag variance before flipping.

$$S_{M+} = \{T(x_1, x_2, ..., x_n) | TagVariance(T'_M(x_1, x_2, ..., x_n)) < TagVariance(T(x_1, x_2, ..., x_n))\}.$$

3. Unchanged $U_{M+} = \{T_1, T_2, \dots\}$, the set of tags where the tag variance before and after flipping are equal.

$$U_{M+} = \{T(x_1, x_2, ..., x_n) | TagVariance(T'_M(x_1, x_2, ..., x_n)) = TagVariance(T(x_1, x_2, ..., x_n))\}.$$

In the case of negative flipping we have the sets of tags $\{R_{M-}, S_{M-}, U_{M-}\}$ defined in a similar manner as above.
Once the sets have been determined we calculate the R and S statistics. These are the lengths of the sets R_{M+}, R_{M-}, S_{M+} and S_{M-} (the sets U_{M+} and U_{M-} are generated for completeness); thus $|R_{M+}|$, $|R_{M-}|$, $|S_{M+}|$ and $|S_{M-}|$. Note that $\frac{|R_{M+}|+|S_{M+}|+|U_{M+}|}{m} = 1$ (where m is the total number of tags). Similarly $\frac{|R_{M-}|+|S_{M-}|+|U_{M-}|}{m} = 1$.
The operation of the RS statistic can be illustrated by considering the following. Firstly considering examples where TLCSS is not present:

- If a webpage features all lowercase tag letters then $|R_{M+}| > 0$ and —$S_{M+}| = |S_{M-}| = |R_{M-}| = 0$
- If a webpage features all uppercase tag letters then $|R_{M-}| > 0$ and $|S_{M-}| = |S_{M+}| = |R_{M+}| = 0$.

However, when TLCSS does exist:

- If a webpage features all lowercase tag letters then $|R_{M+}| > |S_{M+}|$, $|S_{M+}| > 0$ and $|S_{M-}| > |R_{M-}|$, $|S_{M-}| > 0$, —$R_{M-}| > 0$.
- If a webpage features all uppercase tag letters then $|R_{M-}| > |S_{M-}|$, $|S_{M-}| > 0$ and $|S_{M+}| > |R_{M+}|$, $|S_{M+}| > 0$, $|R_{M+}| > 0$.
- When the length of a hidden message is 100% of the TLCSS Maximum Embedding Capacity (MEC) of a webpage (see Section 6.3 below) $|S_{M+}| > |R_{M+}|$, $|S_{M-}| > |R_{M-}|$ and $|S_{M+}| > 0$, $|S_{M-}| > 0$ and $|S_{M+}| \approx |S_{M-}|$, $|R_{M+}| \approx |R_{M-}|$, $|R_{M+}| > 0$, $|R_{M-}| > 0$.

6.2.4 The Tag Variance Algorithm

In this section the proposed TV algorithm for TLCSS detection is presented. The algorithm is similar to that presented in Huang et al. [44] with the distinction that Tag Variance is used instead of Tag Offset. The process is as follows:

1. Collect all tags in a webpage in set Q_1 and compute the Tag Variance using Equation 6.2.
2. Apply positive flipping to all tags in Q_1 using $f_{+1}(T)$ and $f_0(T)$ with a mask $M+$ so that altered tags are obtained and saved in the set Q_2.
3. Compute Tag Variance of the altered tags in Q_2
4. Determine R_{M+} and S_{M+}.
5. Apply negative flipping to all tags in Q_1 using $f_{-1}(T)$ and $f_0(T)$ with mask $M-$ so that altered tags are again obtained and saved them in Q_3 set.
6. Compute Tag Variance of the altered tags in Q_3
7. Determine R_{M-} and S_{M-}.
8. Examine the relationship between R_{M+}, S_{M+}, R_{M-}, S_{M-} statistics to decide whether a webpage is normal or stego.

6.3 Maximum Embedding Capacity Using TLCSS

Using TLCSS the maximum size of a message that could be held in a webpage depends on the TLCSS algorithm used. Two TLCSS algorithms were considered with respect to the work presented in this chapter: (i) Sui and Luo [87] and (ii) Shen Y. [96]. In Sui and Luo all tag letters were used for the purpose of message hiding while in Shen Y. only the first letter of each tag was used. Thus, in the case of Sui and Luo the MEC, in bytes, is calculated as follows:

$$MEC_{Sui\ and\ Luo} = \frac{\sum_{i=1}^{m} |T_i|}{8} \tag{6.3}$$

Where $|T_i|$ is the length (number of letters) in tag T_i and m is the number of tags in the given webpage. Division by 8 so the result is given in bytes. With respect to Shen Y. the MEC is then calculated as follows:

$$MEC_{Shen\ Y.} = \frac{m}{8} \tag{6.4}$$

6.4 Evaluation

In this section the results obtained from the evaluation of the proposed TV TLCSS detection approach is presented. Four sets of experiments were conducted; the objectives of which were as follows:

1. To analyze the behavior of the RS statistics given hidden messages of different lengths.
2. To determine the effect of applying different masks M on the TLCSS detection.
3. To provide a comparison between the proposed TV approach and Tag Offset approach of Huang et al. with respect to both natural English language and random message embedding.
4. To determine whether the results obtained with respect to the experiments conducted to address the previous objective are statistically significant or not.

These objectives are considered in turn in Sub-sections 6.4.1, 6.4.2, 6.4.3 and 6.4.4 respectively. Note that for all the evaluation objectives the Non-Monitoring dataset in Appendix Awas used. Two metrics were used for the evaluation:

- The True Positive rate (TP) , the percentage of correctly detected stego webpages with respect to the total number of webpages considered.
- The False Negatives (FN) rate, the percentage of stego webpages identified as non-stego webpages with respect to the total number of webpages considered.

It should be noted that the higher the TP rate the lower the FN and vice versa.

6.4.1 Behavior of The RS Statistics Given Hidden Message of Different Length

With respect to the first of the above evaluation objectives, to analyze the behavior of the RS Statistics given hidden message of different length, the experimentation comprised two individual experiments as follows:

- A set of experiments using a single webpage from the TV dataset, the Wikipedia landing page, repeatedly seeded with English language hidden messages of increasing size, from 10% to 100% of the MEC, incrementing in steps of 10%.
- A set of experiments using thirty webpages from the TV dataset each seeded with an English language hidden message of length equivalent to 100% of the MEC for each webpage.

In both cases Sui and Luo's TLCSS embedding method was used. The proposed TV approach was then applied and the obtained RS statistics recorded. The masks used were: $M+ = [1, 1, 1]$ for positive flipping and $M- = [-1, -1, -1]$ for negative flipping. The results are presented in Figures 6.1 and 6.2.

From Figure 6.1 it can be seen how the RS statistics increase/decrease in a more or less linear manner as the relative size of the hidden message increases. Both S_{M+} and S_{M-} increase significantly, while in the case of R_{M-} the increase is less significant. The R_{M-} statistic decreases as the size of the message decreases. These results confirm the expected outcomes identified in Sub-section 6.2.3 .

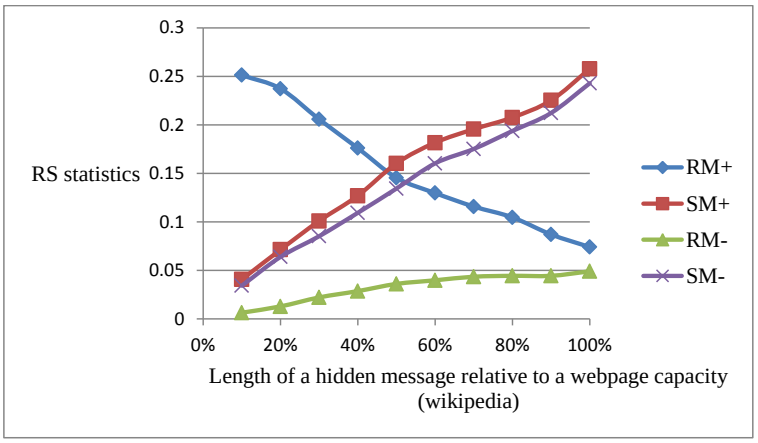


FIGURE 6.1: Effect of incremental increases in the hidden message size on the RS Statistics (R_{M+}, S_{M+}, R_{M-} and S_{M-})

In Figure 6.2 the X-axis; lists the ID number for the selected webpages, while the Y-axis gives the RS values (R_{M+}, S_{M+}, R_{M-} and S_{M-}). From the figure the following can be observed: (i) S_{M+} and S_{M-} closely approximate each other, (ii) R_{M+} and R_{M-} also approximate each other (but less closely) and S_{M+} and S_{M-} are greater that R_{M+} and R_{M-}. These results again confirm the expected outcomes identified in Sub-section 6.2.3.

6.4.2 The Effect of Masking

Recall from Section 6.2 that the RS statistics are calculated as a consequence of applying flipping functions to the letters featured in tags and that the nature of this flipping is dictated by a mask. Recall also that we apply both positive and negative flipping (upper case to lower case, and vice versa) using positive and negative masks ($M+$ and $M-$) respectively. Thus the effectiveness of the proposed TLCSS detection approach was

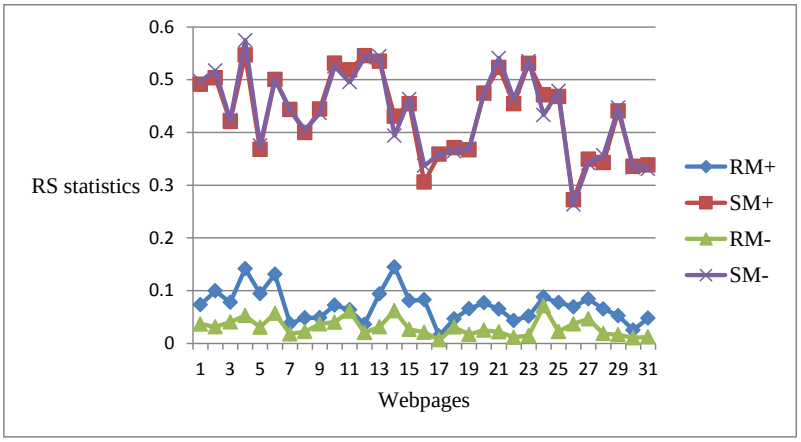


FIGURE 6.2: RS statistics generated using the TV approach when hidden messages (of length equivalent to 100% of MEC) are embedded in selected webpages from the TV dataset

dependent on the nature of the masks. A second set of experiments was thus conducted to evaluate the usage of different masks on TLCSS detection.

For the experiments half of the webpages in the TV dataset were seeded with different English Language messages, again using Sui and Luo's TLCSS algorithm. The length of the seeded messages varied between 2% and 10% of MEC, incrementing in steps of 2%. The reason for using relatively short hidden messages was that the approach of Huang et al. had founded these to be particularly challenging. A range of different masks were used:

- For positive flipping:
  $M+ = \{[1,0,1,1],[1,0,1,0],[1,1,1],[0,1,1],[1,1,0],[1,0,0],[0,0,1]$ *and* $[0,1,0]\}$.
- For negative flipping:
  $M- = \{[-1,0,-1,-1],[-1,0,-1,0],[-1,-1,-1],[0,-1,-1],[-1,-1,0],[-1,0,0],$
  $[0,0,-1]$ *and* $[0,-1,0]\}$.

In each case the TV algorithm was applied and the TP and FN rates were recorded. Figure 6.3 shows the results obtained when using two of the above masks; $M+ = [1,0,1,1]$, $M- = [-1,0,-1,-1]$ and $M+ = [1,0,1,0]$, $M- = [-1,0,-1,0]$. The graph on the top considers the TP rate, and that on the bottom the FN rate. In the figure the X-axis represents length of the hidden messages, while the Y-axis represents either the True Positives (TP) or False Negatives rate (FN). From the figure it can be seen that the (TP) rate of 96% was attained with respect to hidden messages of a length of 2% of the MEC when using masks $M+ = [1,0,1,1]$ and $M- = [-1,0,-1,-1]$. While

when using masks $M+ = [1, 0, 1, 0]$ and $M- = [-1, 0, -1, 0]$, many stego webpages were missed. This is consequently reflected on the recorded FN rate values. Using masks $M+ = [1, 0, 1, 1]$ and $M- = [-1, 0, -1, -1]$ resulted that the FN rate was much lower than when using masks $M+ = [1, 0, 1, 0]$ and $M- = [-1, 0, -1, 0]$. From this set of experiments it was concluded that the most appropriate masks to use were $M+$=[1,0,1,1] and $M-$=[-1,0,-1,-1].

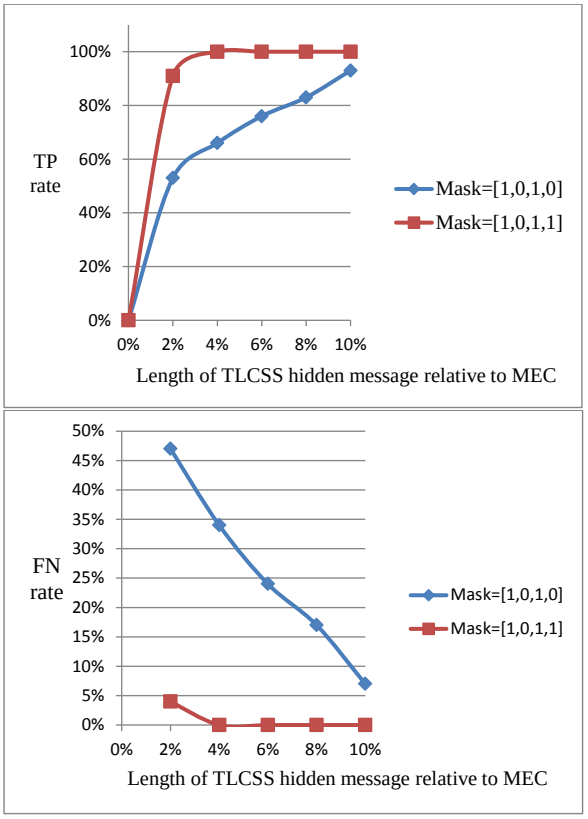


FIGURE 6.3: Illustration of the use of different masks for TLCSS detection using the proposed TV approach; top: TP rate, bottom: FN rate

6.4.3 Comparisons with Other Detection Approaches

For the comparison with other TLCSS detection approach the operation of the proposed TV approach was compared to the Tag Offset approach of Huang et al. as presented in [44]. The TLCSS embedding mechanisms of both Sui and Luo [87] and Shen Y. [96] were used together with the Non-Monitoring dataset in Appendix A also used for earlier experiments. The dataset was half seeded with hidden messages using one or other of the selected TLCSS mechanisms. The comparisons were conducted firstly using natural English language message embedding, and secondly using random message embedding. Recall that when using Sui and Luo all the tag letters are used, while when using

Shen Y. only the first letter of each tag are used. The MEC calculation for each is therefore different as discussed in Section 6.3. For the experiments the masks were used $M+$=[1,0,1,1] and $M-$=[-1,0,-1,-1] because earlier experiments, reported in the foregoing subsection, has indicated that these masks worked well. In the case of Sui and Luo the message lengths ranged from 2% to 6%, increasing in steps of 2%, with respect to the relevant MEC. In the case of Shen Y. the lengths ranged from 4% to 14% incrementing in steps of 2%.

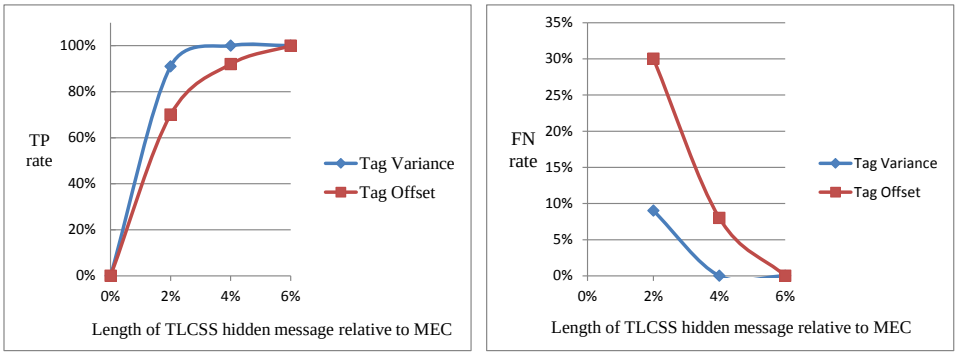
This distinction was because experiments conducted earlier had indicated that proposed Tag Variance could detect Sui and Luo TLCSS embedding, even when very short hidden messages were used. In the case of detecting Shen Y. TLCSS embedding, experiments conducted had indicated that the proposed Tag Variance operated best with longer hidden messages (although still relatively short).

The results obtained are given in Figures 6.4 and 6.5. Figure 6.4 shows the results obtained using Sui and Luo embedding, (a) natural English language messages and (b) random messages. The X-axis in each case represents the hidden message length relative to the MEC, while the Y-axis represents either: TP rate (left); or FN rate (right). From the figure it can be noted that the performance of the TV proposed approach is better than the performance of Huangs Tag Offset approach. In some cases the proposed TV approach identified all of the stego webpages, a detection rate of 100% whereas the Tag Offset approach did not. As a consequence the FN rate (missing stego webpages) in the case of using proposed TV approach was much less than in the case of Huangs Tag Offset approach. From the figure it can also be noted that the nature of the message embedding made little difference with respect to TLCSS detection.

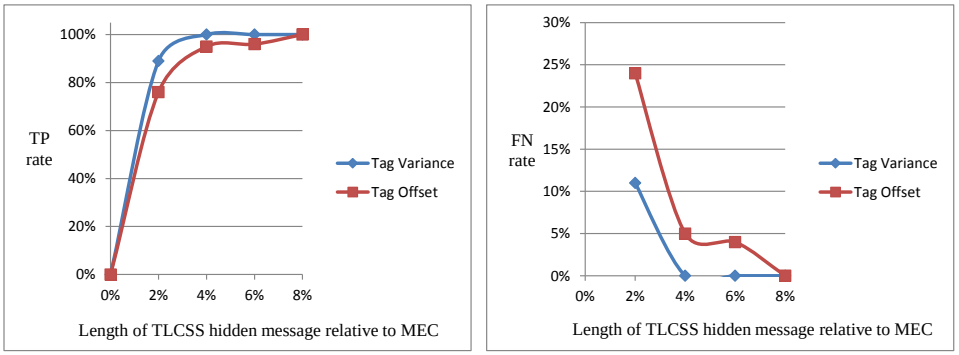
Figure 6.5 in turn shows the results obtained using Shen Y. From the figure it can be seen that both approaches achieved a similar performance. An approximate detection rate of 90% was achieved by both detection approaches with respect to hidden messages of length 14% of the webpage MEC. webpages featuring shorter messages were undetected, hence the high FN rate. Again the nature of message embedding made little difference with respect to TLCSS detection.

6.4.4 Statistical Evaluation

This sub-section reports on the results of the statistical significance testing conducted with respect to the results reported on in the previous sub-section. The aim of the significance testing was to answer the question whether the performance of the Tag Variance approach compared to the performance of the Tag Offset is statistically significant or simply a matter of chance. The test used was the Wilcoxon signed-rank test [23] because only two True Positives (TP) rate samples, that of Tag Variance and Tag Offset, were collected. The Wilcoxon signed-rank test is a non-parametric test used to compare two related samples, matched samples or repeated measurements on a single sample. It is an alternative to the paired t-test and is typically used to test the differences in the mean of paired observations. These differences are ranked to produce two rank totals



(a) Natural English language message embedding



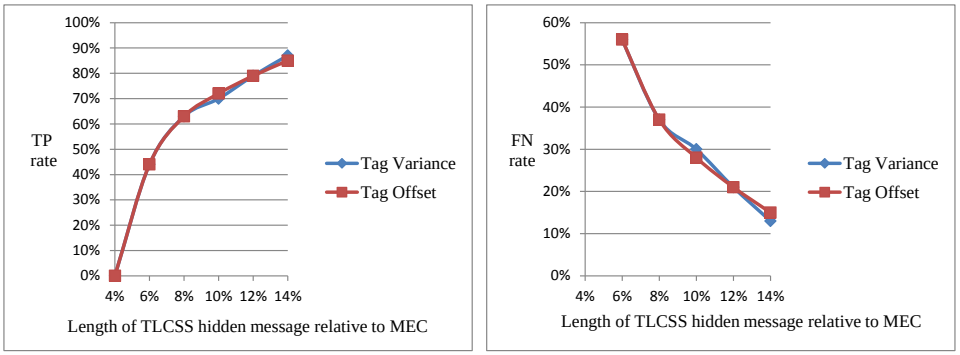
(b) Random message embedding

FIGURE 6.4: Comparison between the proposed Tag Variance approach and the Tag Offset approach using Sui and Luo TLCSS embedding

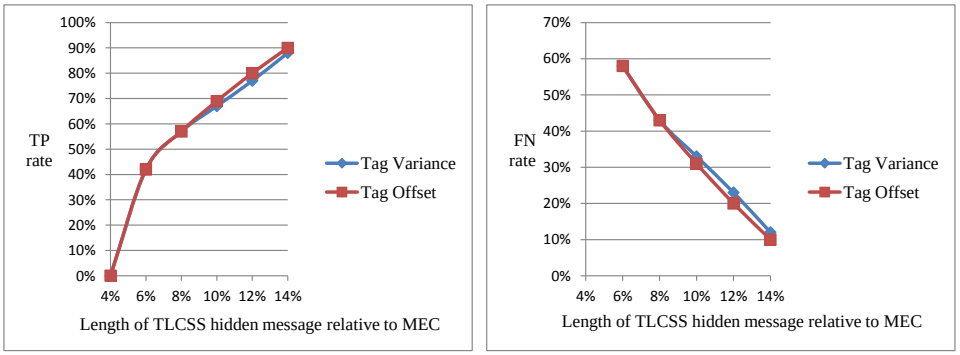
(positive and negative) that are used later to find test statistics Appendix B. Under the null hypothesis, $H0$, these two totals are the same. Since the results, presented above, indicated that the nature of the embedded message has no effect on the True Positives rate (TP) only the results of obtained using natural English language messages embedding from Figures 6.4 and 6.5 were used. Figures 6.6 and 6.7 present the Wilcoxon signed-rank test results obtained using Sui and Luo and Shen Y embedding respectively. In Figure 6.6 it can be seen that the test statistics were $z = -1.342$ and $p = 0.18$, and in Figure 6.7 that the test statistics were $z = 0$ and $p = 1.000$. Since p for both tests is greater than 0.05 (the significant level) it can be concluded that there is no statistical difference in operation between the proposed TV approach and the Tag Offset approach; in other words the null-hypothesis $H0$ is accepted.

6.5 Summary

In this chapter the proposed Tag Variance (TV) approach to detecting TLCSS has been presented. The idea is founded on the Tag Offset approach proposed in Huang et al.

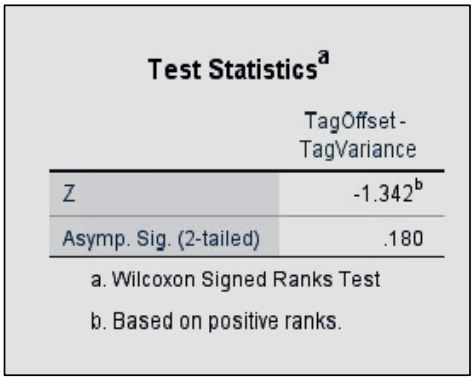


(a) Natural English messages embedding



(b) Random messages embedding

FIGURE 6.5: Comparison between the proposed Tag Variance approach and the Tag Offset approach using Shen Y. TLCSS embedding



Test Statistics[a]

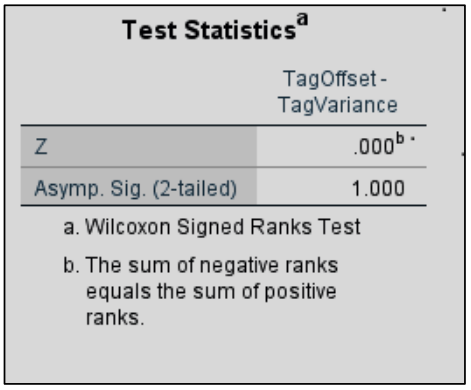
	TagOffset - TagVariance
Z	-1.342[b]
Asymp. Sig. (2-tailed)	.180

a. Wilcoxon Signed Ranks Test

b. Based on positive ranks.

FIGURE 6.6: Results of Wilcoxon test comparing statistical significance of the TV and Tag Offset approaches using Sui and Luo TLCSS embedding

[44]. The reported evaluation demonstrated that the proposed Tag Variance mechanism, when coupled with appropriate positive and negative masks, outperformed the TLCSS detection approach of Huang et al., especially with respect to detection of Sui and Luo



Test Statistics[a]

	TagOffset - TagVariance
Z	.000[b]
Asymp. Sig. (2-tailed)	1.000

a. Wilcoxon Signed Ranks Test

b. The sum of negative ranks equals the sum of positive ranks.

FIGURE 6.7: Results of Wilcoxon test comparing statistical significance of the TV and Tag Offset approaches using Shen Y. TLCSS embedding

TLCSS embedding of short hidden messages (2% of a webpage's MEC). Even though the reported results showed that TV performed better than the Tag Offset approach, the results were not found to be statistical significance using the Wilcoxon signed-rank test. The next chapter concludes this thesis with a summary of the contributions and main findings, and some suggestions for future research directions.

Chapter 7

Conclusions and Future Work

7.1 Introduction

This chapter provides a summary of the work presented in this thesis, a review of the main findings in the context of the research question and research issues identified in Chapter 1 and some suggestions for fruitful the future work directions.

7.2 Summary

In this thesis a set of HTML steganography detection methods have been proposed directed at three different kinds of HTML steganography methods: (i) Attribute Permutation Steganography (APS), (ii) Invisible Character Steganography (ICS) and (iii) Tag Letters Case Switching Steganography (TLCSS). In total four algorithms were proposed: (i) Statistical Detection (SD) (ii) Attribute Position Changes Count (APCC) (iii) Detect Invisible Character (DIC) and (iv) Tag Variance (TV). The first two directed at APS, and the third and fourth directed at ICS and TLCSS, respectively. A distinction was also made between steganography monitoring in relation to dynamic context and steganography detection in relation to static context.

The thesis commenced, in Chapter 2 with a literature review of previous work relevant to the work presented in the thesis. The following four chapters covered the proposed HTML steganography detection approaches, each directed at a specific approach (as listed above). These chapters were all structured in a similar manner starting with a description of the proposed approach and ending with a review of the evaluation conducted with respect to each approach. All the proposed approaches were tested using both natural English language and random embedded messages.

In more detail Chapter 3 presented the SD approach designed to detect APS in the dynamic monitoring context. The fundamental idea was to use the standard deviation of the change in tag attribute position as a measure for detecting the presence (or otherwise) of APS. The proposed detection was tested with respect to the APS methods of: Deogol [80], Huang et al.[45] and Shen et al.[81]. A mechanism whereby the SD threshold σ value for a particular WWW page could be learnt was also given.

Chapter 4 considered the proposed APCC approach to detecting APS in the non-monitoring context. In this proposed approach the concept of the attribute position changes count was used to detect the presence of steganography. Unlike in the case of the proposed SD approach the steganography detection (steganalysis) problem was formulated as a supervised learning problem. For the presented evaluation three classification models were used: (i) Multi-Layer Perceptron (MLP) Neural Network, (ii) Support Vectors Machine (SVM) and (iii) Naive Bayes(NB). Classifiers performance was evaluated in terms of: (i) Accuracy and (ii) Area Under the Receiver Operator Characteristic Curve (AUC). The evaluation included an investigation of whether the results were statistically significant or not by applying the Freidman Test. As in the case of the SD approach the APCC approach was tested using the APS embedding methods of Deogol, Huang et al. and Shen et al. The operation of the APCC approach was compared with the approaches of L.Polak and Z.Kotulski [65] (dynamic context) and W.Jian-feng et al. [95] (static context) and the SD approach from the previous chapter. The evaluation results indicated the followings: (i) the APCC approach was more efficient than the approach of W.Jian-feng et al (ii) the SD approach produced better results than that of L.Polak and Z.Kotulski in the non-monitoring context.

Chapter 5 considered the DIC approach directed at detecting ICS in the non-monitoring context. The fundamental idea was to use the frequency distribution of continuous sequences (segments) of a white space character to distinguish normal webpages from stego webpages. An idea founded on the observation that the frequency of space character segments will change when inserting additional white space characters to embed messages. Two freely available tools, SNOW [60] and wbStego4open [91], were used for the purpose of embedding hidden messages using ICS. Again a supervised learning approach was adopted, and for the evaluation the same classification models as considered previously with respect to the work presented in Chapter 4 were considered. The operation of the proposed DIC approach was compared with the ICS detection approaches of Sui and Luo [68] and Huang et al. [43]. The results demonstrated a significantly better detection rate with respect to the DIC approach over the other two considered approaches. The statistical significance of the results was also reported on.

Chapter 6 considered the proposed TV approach directed at the detection of TLCSS, again, in the non-monitoring context. The idea was to use a measure of tag letters case "smoothness" called Tag Variance, an improved version of the Tag offset mechanism of Huang et al. [44]. For the evaluation the TLCSS embedding mechanisms of Sui and Luo [87] and (ii) Shen Y. [96] were considered. The evaluation outcomes indicated that the TV mechanism performed better than the TLCSS detection approach of Huang et al., especially with respect to short hidden messages. The evaluation included an investigation of whether the results were statistically significant or not by applying the Wilcoxon-signed rank test.

7.3 Main Findings and Contributions

This section presents the main findings resulting from the research presented in this thesis. Recall that the original research question from Chapter 1 was *"What are the most appropriate detection approaches, dynamic and static, required to efficiently and effectively detect hidden messages in HTML files?"*. In Chapter 1 a number of subsidiary questions were also postulated. Each of these will be considered in turn, before returning the main overriding research question, as follows:

1. *"What are the features of steganography that can best be adopted to facilitate detection?"*. The answer to this is that is depends on which HTML steganography approach is adopted. Three were considered in this thesis: (i) APS (ii) ICS and (iii) TLCSS. In the case of APS it was found that the standard deviation of the position changes count worked well as did APCC. In the case of ICS it was found that the frequency distributions of space character segments could be usefully employed. Finally in the case if TLCSS the proposed tag variance measure was found to produce good results.

2. *"How can steganography detection effectiveness be measured so that different techniques can be assessed and compared to one another?"*. Two different steganograhy detection contexts were considered: (i) the dynamic (monitoring) context and (ii) the static (non-monitoring) context. In the first case the idea presented was to adopt a mechanism to identify a webpage steganography detection threshold. After that measures such as True Positives (TP) and False Negatives (FN) could be used to assess a steganography detection mechanism. In the second case the main idea presented was to adopt a classification approach. Measures typically used to evaluate supervised learning techniques were adopted. The used measures were: (i) Accuracy and (ii) Area Under the Receiver Operator Characteristic Curve (AUC). In the case of TLCSS detection as in the case of detecting APS in the monitoring context the detection rate True Positives (TP) and False Negatives (FN) rate were used.

3. *"Given a set of webpages how should the content of those webpages be represented so as to facilitate, dynamic or static, steganography detection?"*. The resolution to this question proved straight-forward. The answer, as in the case of the answer to subsidiary Question 1 above, is that this depended on the nature of the HTML steganography under consideration: (i) APS, (ii) ICS or (iii) TLCSS. In the case of APS we only need to consider the attributes that appear in webpages. These were therefore represented using either the standard deviation of their positions as in Chapter 3 or by a feature vector of their positions changes count as in Chapter 4. In case of ICS the entire webpage of interest needed to be considered, and thus was represented by a feature vector of white space segment frequency distributions. In case of TLCSS we only needed to consider the tags that appear in webpages.

4. *"Given a solution to the above, how can the performance of such techniques be best improved so as to maximize effectiveness?*. From the foregoing, at least for static steganography, detection using feature selection maximized the effectiveness. Parameter tuning was also important.

5. *"What is the most appropriate mechanism for webpage monitoring in the context of dynamic steganography detection?"*. Monitoring was considered in detail in Chapter 3. The fundamental idea presented was to repeatedly consider snapshots of a webpage of interest, separated by a time intervals in a period of time. This seemed to work well as indicated in the reported evaluation with respect to learning a webpage steganography detection threshold.

6. *"What is the effect on detection of the nature of messages that are hidden (English language or random)?"*. The reported evaluation results, using both English language and random messages (the later simulating encrypted messages), indicated that whether a hidden messages was a natural English language text or a random one had no effect with respect to the operation of the proposed steganography detection techniques.

7. *"What is the effect on detection of the length of messages that are hidden?"*. The evaluation conducted using messages of different length reported on in the previous four chapters, indicated that the length of a hidden message had a significant influence on the effectiveness of detection.

Returning to the overriding research question in Chapter 1 *"What are the most appropriate detection approaches, dynamic and static, required to efficiently and effectively detect hidden messages in HTML files?"* it can be concluded that for HTML steganography detection in the dynamic context that the proposed SD algorithm was effectively a mechanism whereby the steganography detection threshold for a particular webpage could be learnt. In the case of HTML steganography detection in the static context the work presented in this thesis clearly indicates that the both proposed APCC and DIC approaches, coupled with appropriate classification techniques, could clearly distinguish between stego and clean webpages in a way that was effective. Also proposed TV approach was an alternative enhancement of the approach of Huang et. al to detect very short messages in the (non-monitoring) static context.

It is found that detection of HTML attribute permutation steganography APS mechanism is more attractive than other HTML steganography mechanisms. This is because APS still difficult to discover as it does not neither cause increasing of the carrier object size as in ICS nor it can be suspicious from the source code as in TLCSS.

Finally, to complete the main findings reported on in this section, the contributions of the research presented in this thesis (originally itemized in Chapter 1) are presented again here for completeness:

1. The SD approach to detect APS in the dynamic (monitoring) context.

2. The APCC approach also directed at the detection of APS detection in the static (non-monitoring) context.

3. The DIC approach to detect ICS in the static (non-monitoring) context.

4. The TV approach to detect TLCSS in the static (non-monitoring) context.

7.4 Future work

The research described in this thesis has suggested a number potential areas for future work. This thesis is thus concluded with an itemization of these potential future research directions:

- Alternative HTML steganography methods: A number of established HTML steganography methods were investigated with respect to the work presented in this thesis. It would be interesting to investigate the detection of further alternative steganography methods, such as hiding data using links and scripts.

- Alternative datasets: The proposed approaches in this thesis have been tested using small size datasets. A much more comprehensive evaluation is therefore desirable.

- A universal HTML steganography detection technique: The proposed approaches in this thesis have each been designed to detect a particular steganography HTML method. A universal HTML steganography detection technique is desirable to detect all forms of HTML steganography. This could also then be applicable to HTML steganography mechanisms of the future. How this would operate is unclear at present, but some form of machine learning approach might be appropriate.

- We can also consider the perspective of the steganographer and examine ways, given knowledge of the techniques presented in this thesis, whereby detection can be avoided!

Appendix A

Evaluation Datasets

This appendix presents a description of the datasets used for evaluation purposes with respect to the work presented in this thesis. The techniques presented in this thesis are divided between techniques for static non-monitoring steganography detection, and dynamic monitoring steganography detection.
In total four distinct HTML Steganography detection mechanisms were proposed in this thesis:

- Statistical Detection (SD) was directed at Attribute Permutation Steganography (APS) in the monitoring context.
- Attribute Position Changes Count (APCC) was directed at Attribute Permutation Steganography (APS) in the non-monitoring context.
- Detect Invisible Characters (DIC) was directed at Invisible Characters Steganography (ICS), again in the non-monitoring context.
- Tag Variance (TV) was directed at Tag Letter Case Switching Steganography (TLCSS) also in the non-monitoring context.

Whatever the case each dataset was half seeded with one of HTML steganography mechanisms and constructed using a standard process as shown in Figure A.1 to produce stego-webpages. From the figure it can be seen that a four steps process was adopted. This process commenced by collecting a set of webpages (webpages collection was for the period of 1/7/2015 to 7/7/2015) . In the case of static (non-monitoring) steganography detection one webpage per web site was collected, in the case of dynamic (monitoring) steganography a sequence of webpages was collected with respect to each site. The next step in the process was to generate the messages to be hidden. Two categories of message generation were considered: (i) natural English language messages and (ii) random messages made up of mixtures of uppercase and lowercase letters, symbols and numbers. At the same time, for each category, messages of different length were considered. The prepared messages were then used to seeded the collected webpages

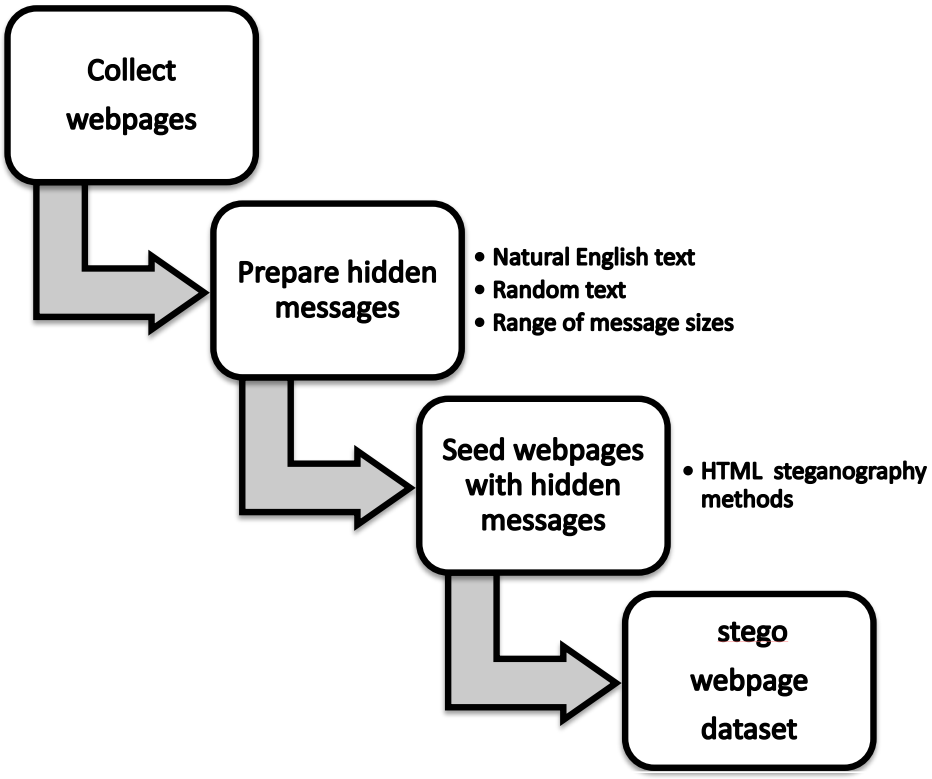


FIGURE A.1: Stego-webpages generation process

using one of the selected HTML steganography methods: (i) APS, (ii) ICS and (iii) TLCSS.

A summary of the datasets generated is presented in Table A.1. The table gives: (i) the size of each dataset in terms of number of webpages, (ii) the HTML steganography mechanism used for the "seeding". Seven mechanisms were used in total: (i) Shen et. al [81], (ii) Deogol [80] (iii) Huang et. al [45] (iv) wbstego4open [91] (v) SNOW [60] (vi) Sui and Luo [87] and (vii) Shen Y. [96]. The first three for APS, the fourth and fifth for ICS and the last two for TLCSS.

TABLE A.1: Summary of generated evaluation datasets

Dataset	Dataset size (wepages no.)	Steganography seeding mechanisms
Monitoring	400	(APS) Shen et. al, Deogol and Huang et. al
Non-Monitoring	150	(APS) Shen et. al, Deogol, Huang et. al , (ICS) wbstego4open, SNOW, (TLCSS) Sui and Luo and Shen Y.

A.1 Monitoring dataset

The Monitoring dataset was used to evaluate the proposed Statistical Detection (SD) monitoring approach presented in this thesis (Chapter 3). In this approach the idea was to monitor the presence of APS in HTML files using an attribute position standard deviation mechanism. This dataset, was generated for monitoring purposes; in other words, for each website considered, a sequence of samples of the website landing page was collected at intervals of time τ over a period of time T. For the evaluation the compiled dataset consisted of 10 well-known landing webpages:

- www.bbc.co.uk.
- www.nytimes.com (New York Times).
- www.wikipedia.org.
- www.stackoverflow.com.
- www.sony.com.
- www.liverpool.ac.uk (The University of Liverpool).
- www.ieee.org.
- www.webmd.com.
- www.microsoft.com.
- www.amazon.com.

These were all landing pages that were updated frequently. For each landing page 40 snapshots were collected at an interval of $\tau = 2\ hours$ over a period of seven consecutive days. The total final number of downloaded webpages was therefore 400 (10×40).

The adopted steganography seeding mechanism was as follows: a message was embedded in thirty out of the 40 snapshots for each webpage. The Standard Deviation (St.D) values of the attributes position change were calculated before and after embedding so that the detection threshold could be learned . The last 10 webpage were used for steganography detection (half seeded and half unseeded). Hidden messages were seeded using the Shen et. al [81], Deogol [80] and Huang et. al [45] mechanisms. Various lengths for the hidden messages were considered from 10% to 100%, incrementing in steps of 10%, of Maximum webpage Embedding Capacity (MEC).

A.2 Non-Monitoring dataset

This dataset was used to evaluate three HTML steganography detection approaches in non-monitoring context. These approaches were:

- The Attribute Position Changes Count (APCC) approach to detecting APS, more details in Chapter 4.
- The Detect Invisible Characters approach (DIC) to detecting ICS, more details in Chapter 5.
- The Tag Variance (TV) approach to detecting TLCSS , more details in Chapter 6.

This dataset also included the 10 landing pages used to construct the Monitoring dataset. The webpages included in the dataset can be loosely categorized according to " the topic", five topics were identified: (i) education, (ii) entertainment, (iii) business, (iv) health and (v) news. Figure A.2 shows the distribution of topics across the dataset. From the figure it can be seen that: 61 webpages were concerned with the education (such as universities and schools), 32 with entertainment (such as sport, games and music), 21 with business, 18 with health (such as nutrition, fitness and medical centers) and 18 with news sites.

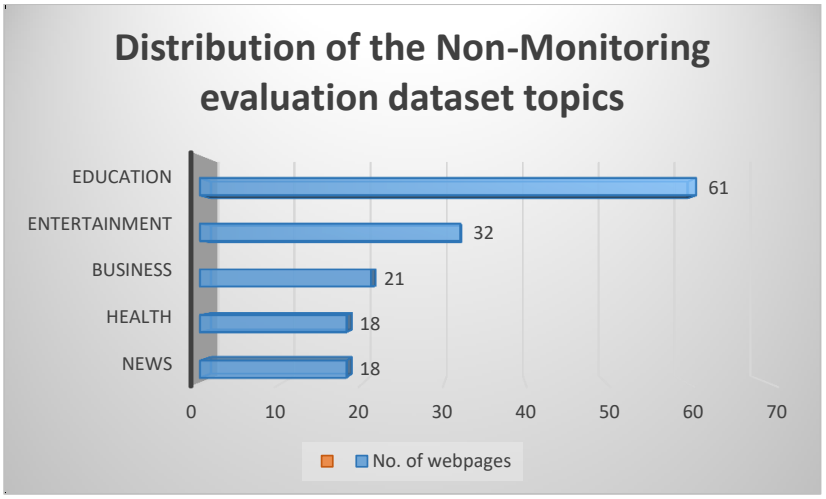


FIGURE A.2: Distribution of webpages topics for the Non-Monitoring dataset

Subsections A.2.1 to A.2.3 give more details on how the Non-Monitoring dataset was used to evaluate HTML steganography proposed approaches in non-monitoring context in this thesis: APCC, DIC and TV in Chapters 4, 5 and 6 respectively.

A.2.1 Non-Monitoring Dataset usage for APCC Evaluation

To evaluate the APCC approach in Chapter 4 the Non-Monitoring dataset was prepared as follows: the webpages were half seeded with hidden messages using the Shen et. al [81],

Deogol [80] and Huang et. al [45] mechanisms. Various lengths for the hidden messages were considered from 10% to 100%, incrementing in steps of 10%, of Maximum webpage Embedding Capacity (MEC).

A.2.2 Non-Monitoring Dataset usage for DIC Evaluation

To evaluate DIC approach in 5, the Non-Monitoring dataset was used as follows: the webpages were half seeded with hidden messages using ICS tools presented in Chapter 2, namely: (i) wbstego4open [91] and (ii) SNOW [60]. Messages of different length were again embedded, although in this case message length was defined in terms of a number of characters instead of a percentage of the MEC. The reason for this was that the ICS tools used provided a high embedded capacity so a relative increment of a hidden message length (for example 30%, 40%. ..., 100%) with respect to the MEC would result in very long messages that can be easily detected. Thus messages of lengths of between 20 and 100 characters, incrementing in steps of 20 characters, were used.

A.2.3 Non-Monitoring Dataset usage for TV Evaluation

To evaluate TV approach described in Chapter 6. The Non-Monitoring dataset was used as follows: the webpages were half seeded with hidden messages using TLCSS methods, namely: (i) Sui and Luo [87] and (ii) Shen Y. [96] (both described previously in Chapter 2). Again messages of different length were considered, defined in terms of a percentage of the MEC, however in this case the lengths considered were from 2% to 10% of the MEC incrementing in steps of 2%. The reason for this incrementing was to investigate the TV approach effectiveness to detect very short messages.

Appendix B

Wilcoxon-signed rank test

This appendix presents the steps for computing the Wilcoxon-signed rank test statistics W that is used in this thesis.
The Wilcoxon-signed rank test is used to test for the median differences. The test statistics W can be obtained using the following steps:

1. State the null hypothesis - in this case it is that the median difference, M, is equal to zero.

2. Calculate each paired difference, $d_i = x_i$-y_i, where x_i , y_i are the pairs of observations.

3. Rank the d_is, ignoring the signs (i.e. assign rank 1 to the smallest absolute value of d_i, rank 2 to the next etc.).

4. Label each rank with its sign, according to the sign of y_i.

5. Calculate W+, the sum of the ranks of the positive d_is, and W-, the sum of the ranks of the negative d_is. (As a check the total, W+ + W-, should be equal to $\frac{n(n-1)}{2}$, where n is the number of pairs of observations in the sample).

6. Choose W = min(W-, W+).

7. Use tables of critical values for the Wilcoxon signed rank sum test to find the probability of observing a value of W or more extreme. Most tables give both one-sided and two-sided p-values. If not, double the one-sided p-value to obtain the two-sided p-value.

Appendix C

Feature Selection Methods

This appendix presents feature selection methods. There are three general methods of feature selection: [59]:

- **Filter methods**: In these methods a rank or score is assigned to each feature and the top k selected. This ranking can be determined using a variety of statistical metrics. This rank is assigned according to the correlation of each feature with the output variable (the class label). Figure C.1 presents the general process adopted by these methods. Filter methods are frequently independent of any machine learning algorithm, they are also fast because there is no training as in the case of Wrapper methods (see below). Examples of these methods include: correlation coefficient, chi square and information gain.

FIGURE C.1: Filter methods [55]

- **Wrapper methods**: In these methods a feature subset is chosen and a particular model is trained using this subset. Based on the results obtained the relevance of the features is verified. Various features combinations are considered when using these methods, not the case with respect to Filter methods. However, searching all combinations requires high computational time. Figure C.2 presents the general process adopted by these methods. Examples of these methods include: forward selection, backward selection [39] and recursive feature elimination [38].

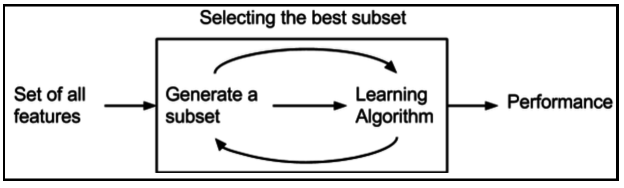


FIGURE C.2: Wrapper methods [55]

- **Embedded methods**: This is where the feature selection is embedded in the data mining and hence it is no longer a preprocessing method. However, there are a number of regression methods that use this approach; examples include: LASSO (Least Absolute Shrinkage and Selection Operator) regression [30] and RIDGE regression [61]. Figure C.3 presents the approach of these methods.

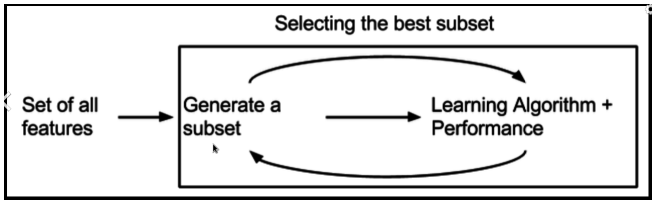


FIGURE C.3: Embedded methods [55]

Bibliography

[1] *Notes on boxplots*, http://web.pdx.edu/ stipakb/download/PA551/boxplot.html.

[2] Charu C. Aggarwal, *Data classification algorithms and applications*, Boca Raton, FL, USA:CRC Press, 2014.

[3] Megha Aggarwal, *Performance analysis of different feature selection methods in intrusion detection*, International Journal of Scientific and Technology Research **vol. 2** (2013), no. 6, pp. 225–231.

[4] Adnan M. Alattar and Osama M. Alattar, *Watermarking electronic text documents containing justified paragraphs and irregular line spacing*, Proc. of SPIE 5306, Security, Steganography, and Watermarking of Multimedia Contents VI **vol. 5306** (2004), pp. 685–695.

[5] Ross J. Anderson, *Stretching the limits of steganography*, 1st International Workshop, Springer-Verlag **vol. 1174** (1996), pp. 39–48.

[6] Ross J. Anderson, Roger Needham, and Adi Shami, *The steganographic file system*, "In Proc. of International Workshop on Information Hiding IWIH" (1998), pp. 73–82.

[7] Ross J. Anderson and Fabien A. P. Petitcolas, *On the limits of steganography*, IEEE Journal on Selected areas in Communications **vol. 16** (1998), no. 4, pp. 474–481.

[8] I. Avcibas, N. Memon, and B. Sankur, *Steganalysis using image quality metrics*, IEEE Trans. Image Processing **vol. 12** (2002), no. 2, pp. 221–229.

[9] T.Oladipupo Ayodele, *Types of machine learning algorithms*, http://www.http://citeseerx.ist.psu.edu/viewdoc/download, 2010.

[10] W. Bender, D. Gruhl, N. Morimoto, and A. Lu, *Techniques for data hiding*, IBM Systems Journal **vol. 35** (1996), no. 3, pp. 313–336.

[11] Krista Bennett, *Linguistic steganography: Survey, analysis and robustness: Concerns for hiding information in text*, Purdue University, CERIAS Tech. Report (2004).

[12] Chidansh Amitkumar Bhatt and Mohan S. Kankanhalli, *Multimedia data mining: state of the art and challenges*, Multimedia Tools and Applications **vol. 5** (2010), no. 1, pp. :35–76.

[13] Dan Boneh and Victor Shoup, *A graduate course in applied cryptography*, https://crypto.stanford.edu/ dabo/courses/OnlineCrypto, Sep. 2017.

[14] Hong Cao and Alex C. Kot, *Eag: Edge adaptive grid data hiding for binary image authentication*, Proc. . Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC) (2012), pp. 1–6.

[15] R. Chandramouli and K.p. Subbalakshmi, *Current trends in steganalysis: a critical survey*, 8th International Conference on Control, Automation, Robotics and Vision Kunming, China (2004), pp. 964–967.

[16] Ching-Yun Chang and Stephen Clark, *Practical linguistic steganography using contextual synonym substitution and a novel vertex coding method*, **vol. 40** (2012), no. 2, pp. 403–448.

[17] Abbas Cheddad, Joan Condell, Kevin Curran, and Paul Mc Kevitt, *Digital image steganography: Survey and analysis of current methods*, Signal Processing **vol. 90** (2010), no. 3, pp. 727–752.

[18] Z. Chen, Land Miao Huang, W. Yang, and P Meng, *Steganalysis against substitution-based linguistic steganography based on context clusters*, Computer and Electrical Engineering **vol. 37** (2011), no. 6, pp. 1071–1081.

[19] Scott Craver, *On public-key steganography in the presence of an active warden*, In Information Hiding, volume Springer Lecture Notes in Computer Science **vol. 1525** (1998), pp. 355–368.

[20] Kevin Curran and Karen Bailey, *An evaluation of image based steganography methods*, International Journal of Digital Evidence **vol. 30** (2006), no. 1, pp. 55–88.

[21] Solomon D., *Data hiding in text*, Springer, New York (2003).

[22] Gerard E. Dallal, *Why p=0.05?*, http://www.jerrydallal.com/lhsp/p05.htm.

[23] J. Demsar, *Statistical comparisons of classifiers over multiple data sets*, Machine Learning Research **vol. 7** (2006), pp. 1–30.

[24] Franck Dernoncourt, *What does auc stand for and what is it?*, https://stats.stackexchange.com/questions/132777/what-does-auc-stand-for-and-what-is-it, January 2015.

[25] Chintan Dhanani and Krunal Panchal, *Html steganography using relative links and multi web-page embedment*, International Journal of Engineering Development and Research. IJEDR **vol. 2** (2014), no. 2, pp. 1960–1965.

[26] Kwankamon Dittakan, *Population estimation mining from satellite imagery*, Ph.D Dissertation, Department of Computer Science, University of Liverpool (2016).

[27] Bret Dunbar, *A detailed look at steganographic techniques and their use in an open-systems environment*, Sans Institute * Whitepaper:http://www.attackprevention.com/article/2586, 2002.

[28] Atallah M.J. et al, *Natural language watermarking and tamperproofing*, Proc. of 5th International Workshop of Information Hiding. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg **vol. 2578.** (2002), pp. 196–212.

[29] Dalson Britto Figueiredo Filho, Ranulfo Paranhos, Enivaldo C. da Rocha, Mariana Batista, Jos Alexandre da Silva Jr, Manoel L. Wanderley D. Santos, and Jacira Guiro Marino, *When is statistical significance not significant?*, Brazilian Political Science Review **vol. 7** (2013), no. 1, pp. 31–55.

[30] Valeria Fonti, *Feature selection using lasso*, Technical report, VU Amsterdam (2017).

[31] Matteo Fortini, *steganography and digital watermarking: a global view*, http://lia.deis.unibo.it/Courses/RetiDiCalcolatori/Progetti00/fortini/project.pdf., 1998.

[32] Eibe Frank, Mark A. Hall, and Ian H. Witten, *The weka workbench. online appendix for data mining: Practical machine learning tools and techniques, morgan kaufmann, fourth edition*, (2016).

[33] Jessica Fridrich, *Feature-based steganalysis for jpeg images and its implications for future design of steganographic schemes*, Proc. . of the 6th Information Hiding Workshop, Springer **vol. 3200** (2004), pp. 67–81.

[34] Jessica Fridrich, Miroslav Goljan, , and Rui Du, *Reliable detection of lsb steganography in color and grayscale images*, Proc. of ACM workshop on Multimedia and security: new challenges (2001), pp. 27–30.

[35] Jessica Fridrich and Miroslav Goljan, *Practical steganalysis of digital images state of the art*, Proc. of SPIE Photonics Imaging, Security and Watermarking of Multimedia Contents **vol. 4675** (2002), pp. 1–13.

[36] Jessica Fridrich, Miroslav Goljan, and Dorin Hogea, *Attacking the outguess*, Proc. of ACM Workshop on Multimedia and Security, ACM Press (2002), pp. 3–6.

[37] Jessica Fridrich, Miroslav Goljanb, and Rui Dub, *Invertible authentication watermark for jpeg images*, Proc. SPIE Security Watermarking Multimedia Contents **vol. 4314** (2001), pp. 197–208.

[38] Joanna Goscik and Tomasz Lukaszuk, *Application of the recursive feature elimination and the relaxed linear sseparability feature selection algorithms to gene expression data analysis*, Advances in Computer Science Research **vol. 10** (2013), pp. 39–52.

[39] Isabelle Guyon and Andre Elisseeff, *An introduction to variable and feature selection*, The Journal of Machine Learning Research **3** (2003), pp. 1157–1182.

[40] Vladimir Hajduk and Dusan Livecky, *Covert selection steganography*, 58th International Somposium ELMAR, Zadar, Croatia (2016), pp. 205–208.

[41] Jiawei Han, Micheline Kamber, and Jian Pei, *Data mining concepts and techniques*, Morgan Kaufmann, San Fransisco, (2012).

[42] Pedram Hayati, Vidyasagar Potdar, and Elizabeth Chang, *A survey of steganographic and steganalytic tools for the digital forensic investigator*, in Workshop of Information Hiding and Digital Watermarking held in conjunction with IFIPTM, Canada, (2007).

[43] H.Huang, X.Sun, Z.Li, and G.Sun, *Detection of hidden information in webpage*, Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD) (2007).

[44] H. Huang, J.Tan, X. Sun, and L. Liu, *Detection of hidden information in webpage based on higher-order statistics*, 7th International Workshop, IWDW 2008 LNCS 5450 Springer-Verlag Berlin Heidelberg (2009), pp.293–302.

[45] Huajun Huang, Shaohong Zhong, and Xingming Sun, *An algorithm of webpage information hiding based on attributes permutation*, 4th International Conference on Intelligent Information Hiding and Multimedia Signal Processing (2008), pp. 257–260.

[46] Shingo Inoue, Kyoko Makino, Ichiro Murase, Osamu Takizawa, Tsutomu Matsumoto, and Hiroshi Nakagawa Shingo, *A proposal on information hiding methods using xml*, 2002.

[47] Pan J.-S.a, Sung M.-T.a, Huang H.-C.b, and Liao B.-Y.a, *Robust vq-based digital watermarking for the memoryless binary symmetric channel*, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **vol. 87** (2004), no. 7, pp. 1839–1841.

[48] Joyce Jackson, *Data mining: A conceptual overview*, Communications of the Association for Information Systems **vol. 8** (2002), pp. 267–296.

[49] Neil F. Johnson, *Steganography software*, http://www.jjtc.com/Steganography/tools.html.

[50] Neil F. Johnson and Sushil Jajodia, *Steganalysis: The investigation of hidden information*, IEEE Information Technology Conference, Syracuse, New York, USA (1998).

[51] Neil F. Johnson and Stefan Katzenbeisser, *A survey of steganographic techniques*, In: Katzenbeisser, S., Petitcolas, F.A.P. (Eds.), Information Hiding Techniques for Steganography and Digital Watermarking, Artech House, Inc., Norwood (2000).

[52] James C. Judge, *Steganography: Past, present, future*, "SANS Institution" (2001).

[53] J.Zollner, H.Federrath, H.Klimant, A.Pfitzmann, R.Piotraschke, A.Westfeld, G.Wicke, and G.Wolf, *Modeling the security of steganographic systems*, Proc. 2nd Workshop on Information Hiding, Portland, LNCS 1525, Springer-Verlag (1998), pp. 345–355.

[54] Stefan Katzenbeisser and Fabien A. P. Petitcolas, *Information hiding techniques for steganography and digital watermarking*, (2000).

[55] Saurav Kaushik, *Introduction to feature selection methods with an example (or how to select the right variables?)*, https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/, 2016.

[56] Tom Kellen, *Hiding in plain view: Could steganography be a terrorist tool?*, http://www.sans.org/reading room/whitepapers/stenganography/hiding plain view steganography terrorist tool, 2001.

[57] Jack Kelley, *Terrorist instructions hidden online*, http://usatoday30.usatoday.com/tech/news/2001-02-05-binladen-side.htm, 2001.

[58] Gary C. Kessler, *An overview of steganography for the computer forensics examiner*, Forensic Science Communications **vol. 6** (2004), no. 3.

[59] Vipin Kumar and Sonajharia Minz, *Feature selection: A literature review*, Smart Computing Review **vol. 4** (2014), no. 3.

[60] Mattew Kwan, *The snow home page: http://www.darkside.com.au/snow/*, (2006).

[61] Thomas Navin Lal, Olivier Chapelle, Jason Weston, and Andre Elisseeff, *Embedded methods*, (2006), pp. 137–165.

[62] Y.K. Lee and L.H. Chen, *High capacity image steganographic model*, IEEE Proc. on Vision, Image and Signal Processing **vol. 147** (2000), no. 3, pp. 288–294.

[63] Bin Li, Junhui He, Jiwu Huang, and Yun Qing Shi, *A survey on image steganography and steganalysis*, Information Hiding and Multimedia Signal Processing **vol. 2** (2011), pp. 142–172.

[64] C.Y. Lin and Chin-Chen Chang, *Hiding data in vq-compressed images using dissimilar pairs*, Journal of Computers **vol. 17** (2006), no. 2, pp. 3–10.

[65] L.Polak and Z.Kotulski, *Sending hidden data through www pages detection and prevention*, Engng.Trans. **vol.58** (2010), pp. 75–89.

[66] Haiping Lu, Alex C. Kot, and Jiancheng Zeng, *Secure data hiding in binary document images for authentication*, Proc. of the 2003 International Symposium on Circuits and Systems **vol. 3** (2003), pp. 806–809.

[67] Jozef Lubacz, Wojciech Mazurczyk, and Krzysztof Szczypiorski, *Principles and overview of network steganography*, IEEE Communications Magazine **vol. 52** (2014), pp. 225–229.

[68] Hui Luo and Xin-Guang Sui, *A steganalysis method based on the distribution of space characters*, in Proc. of Communications, Circuits and Systems International conference Guilin Guangzi, China (2006), pp. 54–56.

[69] L.Y.Por, T.F.Ang, and B.Delina, *Whitesteg: A new scheme in information hiding using text steganography*, WSEAS Transactions on Computers **vol. 7** (2008), pp. 735–745.

[70] M. Shirali-Shahreza M. Hassan Shirali-Shahreza, *A new approach to persian/arabic text steganography*, Proceedinds of 5th Intnernational Conference of Computer and Information Science, Washington (2006), pp. 310–315.

[71] I.Venkata Sai Manoj, *Cryptography and steganography*, International Journal of Computer Applications **vol. 1** (2010), no. 12, pp. 63–68.

[72] Andrew D. McDonald and Markus G. Kuhn, *Stegfs: A steganographic file system for linux*, In Information Hiding (1999), pp. 462–477.

[73] D. McKellar, *Space mimic : http://www.spammimic.com/encodespace.shtml*, (2000).

[74] Jarno Mielikainen, *Lsb matching revisited*, IEEE Signal Processing Letters **vol. 13** (2006), no. 5, pp. 285 – 287.

[75] Cisco Press, *Internetworking basics*, https://www.cisco.com/cpress/cc/td/cpress/fund/ith/, (1998).

[76] Niels Provos, *Defending against statistical steganalysis*, Proc. of the 10th USENIX Security Symposium (2001), pp. 323–325.

[77] Niels Provos and Peter Honeyman, *Detecting steganographic content on the internet*, Proc. of NDSS02: Network and Distributed System Security Symposium, Internet Society **vol. 1768** (2002), pp. 1–13.

[78] R.Baeza-Yates and G.Navarro, *Modeling text databases*, "Recent Advences in Applied Probablity" (2006), pp. 1–25.

[79] Iman Sedeeq, Frans Coenen, and Alexei Lisitsa, *A statistical approach to the detection of html attribute permutation steganography*, In Proceedings of 2nd International Conference on Information Security Systems and Privacy (ICISSP 2016), SCITEPRESS-Science and Technology Publications (2016), pp. 522–527.

[80] S.Forrest, *Introduction to deogol*, http://www.wandership.ca/projects/deogol, 2006.

[81] Dongsheng Shen and Hong Zhao, *A novel scheme of webpage information hiding based on attributes*, IEEE International Conference on Information Theory and Information Security (ICITIS) (2010), pp. 1147–1150.

[82] Gustavus J. Simmons, *The prisoners' problem and the subliminal channel*, Proc. IEEE Workshop Communications Security CRYPTO'83 (1998), 51–67.

[83] Lyu Siwei and H. Farid, *Detecting hidden message using higher-order statistics and support vector machines*, Proc. . of the 5th Information Hiding Workshop, Springer **vol. 2578** (2002), pp. 131–142.

[84] Softpedia, *Steganography tools*, http://www.softpedia.com/hubs/Steganography-Tools/.

[85] Sourceforge, *Steganography studio*, http://stegstudio.sourceforge.net/.

[86] Caitlin Stier, *Russian spy ring hide secret messages on the web*, http://www.newscientist.com/article/-dn19126-russian-spy-ring-hid-secret-messages-on-the-web.html., 2010.

[87] Xin-Guang Sui and Hui Luo, *A new steganography method based on hypertext*, Radio Science Conference (2004), pp. 181–184.

[88] K. Sullivan, U. Madhow, S. Chandrasekaran, and B. S. Manjunath, *Steganalysis for markov cover data with applications to images*, IEEE Trans. Information Forensics and Security **vol. 1** (2006), no. 2, pp. 275–287.

[89] Krzysztof Szczypiorski, *Hiccups - steganography in tcp/ip networks. state of the art and a proposal of a new system*, Proc. 10th Int'l. Multi- Conf. Advanced Computer Systems (2003), pp. 31–40.

[90] Akadej Udomchaiporn, *Volumetric data classification: A study directed at 3-d imagery*, Ph.D Dissertation, Department of Computer Science, University of Liverpool (2016).

[91] wbStego4open, *http://www.wbstego.wbailer.com/*, (2004).

[92] Steffen Wendzel, Wojciech Mazurczyk, Luca Caviglione, and Michael Meier, *Hidden and uncontrolled - on the emergence of network steganographic threats*, in ISSE Securing Electronic Business Processes., Wiesbaden, Germany:Springer (2014), pp. 123–133.

[93] A. Westfeld and A. Pfitzmann, *Attacks on steganographic systems-breaking the steganographic utilities ezstego, jsteg, steganos, and s-tools-and some lessons learned*, Proc. of the 3rd Information Hiding Workshop, Springer **vol. 1768** (1999), pp. 61–76.

[94] Andrew Westfeld, *F5-a steganographic algorithm: high capacity despite better steganalysis*, Proc. of of the 4th Information Hiding Workshop, Springer **vol. 2137** (2001), pp. 289–302.

[95] W.Jian-feng, H.Liu-sheng, T. Miao-miao, C.Zhi-li, and M.Hai-bo, *Detection of html steganography based on statistics and svm*, Journal of Chinese Computer Systems **vol. 35** (2014), no. 6, pp. 1221–1225.

[96] Shen Y., *A scheme of information hiding based on html document*, Journal of Wuhan University **vol. 50** (2004), pp. 217–220.

[97] Huijuan Yang, Xudong Jiang, and Alex C. Kot, *Image watermarking using dual-tree complex wavelet by coefficients swapping and group of coefficients quantization*, EEE International Conference on Multimedia and Expo (2010), pp. 1673 – 1678.

[98] Yi-TaWua and FrankY. Shihb, *Digital watermarking based on chaotic map and reference register*, Pattern Recognition (2007), no. 40, pp. 3753–3763.

[99] Longjiang Yu, Xiamu Niu, and Shenghe Sun, *Print-and-scan model and the watermarking countermeasure*, Image and Vision Computing, Elseiver **vol. 23** (2005), pp. 807–814.

[100] Elzbieta Zelinska, Wojciech Mazurczyk, and Krzysztof Szczypiorski, *Development trends in steganography*, Communications of the ACM **vol. 57** (2014), no. 3, pp. 86–95.

[101] X Zhao, L. Huang, L. Li, W. Yang, Z. Chen, and Z. Yu, *Steganalysis on character substitution using support vector machine*, Proc. of the Second International Workshop on Knowledge Discovery and Data Mining (2009), no. 5, pp. 84–88.