Improving Confidentiality and Integrity of Syntactic Steganography by using Error Control Techniques

Ei Nyein Chan Wai, May Aye Khine
University of Computer Studies, Yangon
einyeinchanwai@gmail.com

Abstract

Among many types of security techniques, steganography is the one that used to build private communication over the public channel. This paper proposes linguistic steganography system by utilizing lossless compression methods, error control techniques and syntax transformation of English language based embedding. The secret message is first compressed with Huffman or Shannon-Fano compression methods to achieve higher capacity. To maintain integrity and confidentiality of the secret message, Hamming code error correction and SHA-1 based Keyed-hash Message Authentication Code (HMAC) may be applied after compression according to the required security level. Because of using transformable syntax forms to carry hidden information, the imperceptibility cannot be damaged by producing meaning preserving sentences. The proposed system is evaluated by using Reuter corpus as a testing environment and Machine Translation (MT) evaluation toolkit of NIST as a similarity measuring tool. It randomly chooses 1804 sentences from about 190 files of three publication days to compare with an existing stego system. The result shows that the proposed system can enhance integrity and confidentiality of secret information by using error control techniques.

1. Introduction

Information security is the protection of information and information systems against unauthorized access or modification of information, whether in storage, processing, or transit, and against denial of service to authorized users. It is classified as the provision of the following three services:

1. Confidentiality : concealment of data from unauthorized parties
2. Integrity : assurance that data is genuine
3. Availability : the system still functions efficiently after security provisions are in place

As there are more demands to improve techniques for information security, many techniques like cryptography, steganography, and digital watermarking have contributed much. Steganographic techniques have been the most successful in supporting hiding of critical information in ways that prevent the detection of hidden messages. The stego process generally involves placing a hidden message within some transport medium, called the carrier. The secret message is embedded within the carrier to form the stego medium. The use of a stego key may be employed for encryption of the hidden message and/or for randomization within the stego scheme. There are three different aspects in steganography systems:

1. Capacity : the amount of information that can be hidden in the cover medium
2. Security : an eavesdropper's inability to detect hidden information
3. Robustness : the amount of modification the stego medium can withstand before an adversary can destroy the hidden information

Today, steganography can be applied in text documents and web pages. In general, text steganography methods can be classified into formatted and linguistics methods. Formatted methods include word shifting, line shifting, and other techniques by changing the physical formatting of cover text. In these methods, the locations of text lines and words in the cover text are shifted horizontally and/or vertically to hide information. Apart from this, word substitution and syntax transformation methods are used to conceal the intended secret information by means of linguistics approach.

In this paper, a linguistic steganography system is constructed by using the compressing algorithms, syntax extraction by the statistical Stanford parser, and a syntactic method that is based on the syntax bank. The confidentiality of secret information can be achieved by applying error correction code (ECC) to compressed secret message.

The rest of the paper is organized as follows. In section 2, a brief overview of existing linguistic steganography methods will be presented. Section 3 will explain the syntax of English language and Section 4 describes briefly about two error control techniques that are used in the proposed system. Section 5 presents our proposed method and evaluation of the proposed system. Finally, the conclusion and further extension will be placed in section 6.

2. Linguistic Steganography

Linguistic Steganography is concerned with making changes that the changes do not result in ungrammatical or unnatural text. Most of the linguistic steganography methods use either lexical (semantic) or syntactic transformations or combination of both. The synonym substitution is the popular lexical steganography method. It substitutes the original word with one of the word that belongs to the same synonym set of the original word. The syntactic methods transform the grammatical style of the original sentences.

2.1. Lexical Steganography

In [1], the original message was hidden through use of a cover text which was shared between sender and receiver. The algorithm replaced all the nouns, adjectives, verbs and adverbs of cover text by their respective synonyms from a word dictionary. All synonyms were put in a frequency table according to their frequencies obtained from WordNet and Huffman coding was done to obtain codes for all synonyms. The input text to be hidden was compressed using Huffman Compression Algorithm and a generated string of bits was consumed in selection of synonyms.

Two improvements by means of the WebIT Google n-gram corpus and vertex color coding can be seen in [2] to address the problem that arises from words with more than one sense. This attempt used WordNet to provide sets of synonyms (synsets). In addition, it only took single word substitution into consideration in order to avoid the confusion of finding information-carrying words during the decoding phase. They proposed a novel coding method based on vertex coloring by which each synonym was assigned a unique codeword.

2.2. Syntactic Steganography

In [5], the authors developed a morphosyntax-based natural language watermarking scheme in which a text is first transformed into a syntactic tree diagram where the hierarchies and the functional dependencies were made explicit. The watermarking software then executes binary changed under control of Wordnet and Dictionary to avoid semantic drops. The security of the watermarked text was enhanced in two ways: (i) the pseudo-random order of tool selection, and (ii) the insertion of a ‘‘pass” tool creating void watermarks.

The research presented at [13] explored the method of text watermarking for Korean by using Korean syntactic dependency parser for syntactic analysis. First, they constructed a syntactic dependency tree of input text. Next, target syntactic constituents were chosen to move and watermark bits were embedded. If the watermark bit did not coincide with the movement bit of the target constituent, the proposed system moved the syntactic constituent in the syntactic tree. Finally, from the modified syntactic tree, a marked text was obtained.

The work in [7] described a method for hiding secret information underneath a Modern Greek cover text by applying shallow syntactic transformations to it. The transformations were extracted automatically by making use of limited external resources, rendering the process easily portable to other free-phrase-order languages. No use of Grammars, syntactic parsers, paraphrase lexica, parallel corpora, semantic lexica and thesauri of any kind was made.

2.3. Combining Lexical and Syntactic Steganography

M.Topkara proposed Enigmark [11, 12], that used orthogonal features of sentences separately for selection and embedding. The way of embedding was done by modifying the embedding features until they “speak the desired message bits”. The selection features were determined by Equmark [11, 18], where a sentence that had a word from the selected subset of the vocabulary was an information-carrier, and the embedding features were based on sentence-level linguistic features which can be “number of prepositions in a sentence”, “a sentence being passive or active”, “distance of certain functional words”, or “the verb classes of the verbs in a sentence”.

3. Syntax of English Language

The syntax of a language is the set of rules that language uses to combine words to create sentences. In English, the parts of speech of words combine into phrases: noun phrase, verb phrase, propositional phrase, adjectival phrase, and adverbial phrase. A clause is a set of words that includes at least a verb and probably a subject noun. A sentence is actually a clause. But a sentence can have more than one clause. There may be a main clause (or independent clause) and one or more subordinate clauses. Just about all sentences in the English language fall into ten patterns determined by the presence and functions of nouns, verbs, adjectives, and adverbs [16].

The nature of the sentences can be changed without changing the meaning of the sentences [4]. The most possible transformation of English is active-passive transformation. This can be used for all sentences and clauses that contain subject, verb, and object. In addition, there is also possible to interchange the clauses back and front. Apart from this, there may be many other ways to transform the sentence retaining its meaning such as topicalization, adverb displacement, and so on.

4. Error Control Techniques

Error control techniqes are used to construct reliable communication over unreliable media with errors caused by channel noise during transmission. In steganography, error controls are applied to improve the robustness, confidentiality and integrity of secret information before embedding secret message's bits into the cover media.

4.1. Hamming Code

Hamming codes can detect up to two simultaneous bit errors, and correct single-bit errors. All bit positions that are powers of two are parity bits. All other bit positions are data bits. Each data bit is included in a unique set of 2 or more parity bits, as determined by the binary form of its bit position. The sum of the positions of the erroneous parity bits identifies the erroneous bit. If only one parity bit indicates an error, the parity bit itself is in error [6]. It needs to trick to correct burst errors. To send k codewords of each length n, these codewords are required to arrange in matrix, each row is a codeword. This matrix has width n and height k. It would normally transmit this row-by-row. The trick is to transmit column-by-column [10].

4.2. SHA-1 based HMAC

HMAC is a specific construction for calculating a message authentication code (MAC) involving a cryptographic hash function in combination with a secret key. It may be used to simultaneously verify both the data integrity and the authenticity of a message. Any cryptographic hash function, such as SHA-1, may be used in the calculation of an HMAC. The cryptographic strength of the HMAC depends upon the cryptographic strength of the underlying hash function, the size of its hash output length in bits, and on the size and quality of the cryptographic key. SHA-1 operates on 512-bit blocks and produces 160 bits hash value [8]. As the estimated collision resistance strength of any approved cryptographic hash function is half the length of its hash value, it is believed to have collision resistance strength of 80 bits. Again, the estimated preimage resistance strength is 160 bits [15].

5. Proposed Approach

In the proposed system, the input cover text must be in English in order to use the syntax transformation capability. It also tries to use some error control techniques to improve confidentiality and integrity as in other multimedia steganography. Figure 5.1 and 5.2 respectively show the sender and receiver of the proposed system.

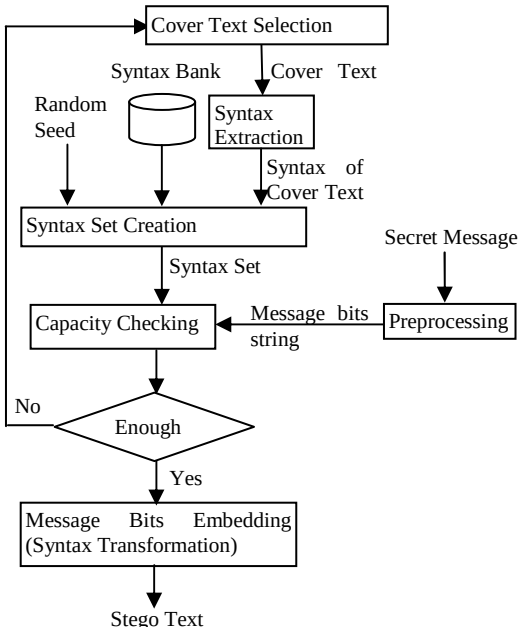

Figure 5.1 Proposed System (Sender Side)

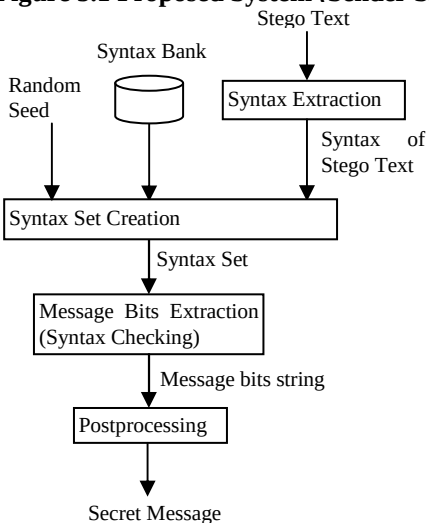

Figure 5.2 Proposed System (Receiver side)

When using error control mechanisms, it tends to increase the number of required message bits by adding some extra redundancy bits to control errors. To cover this fact, the approach firstly compresses the message as possible, and uses this saving to add the error control bits in the embedded message sequence at the preprocessing step. It then utilizes Stanford parser to extract the phrase structure of the input text sentences to get their syntax. Moreover, we propose the syntax bank based linguistic steganography system by doing syntax set creation, capacity checking and syntax transformation steps at the sender's side. At the receiver's side, syntax set creation and syntax checking steps are processed to extract the secret information from the stego text.

5.1. Preprocessing

The system first collects the characters frequencies from a predefined text that is already known by both sender and the receiver. Then it compresses the incoming secret message by one of two popular compression algorithms, Huffman and Shannon-Fano compression methods, to achieve the higher payload. Huffman algorithm is used when the predefined key file has unequal character frequencies and Shannon-Fano algorithm is used when it contains equal frequencies to reduce the overhead of compression. Finally, the message bits sequence is terminated by the compressed code of the termination character "%" because it is not possible to delete the extra part of the cover text.

Error control codes are added to the compressed secret message bits before embedding into the cover text. The system allows user to choose from four options about the robustness level – "none (only compressed)", "with Hamming", "with SHA-1 based HMAC", "with both Hamming and SHA-1 based HMAC". Depending on the user's choice, the compressed message string is appended by Hamming code and/or SHA-1 based HMAC as needed. In the case with HMAC, a predefined key is used.

The message bits that are used to embed in the cover text become longer as it applies error controls to achieve robustness. When it uses Hamming, the length will increase 75% of the original compressed message sequence. With HMAC, as the proposed method applies SHA-1 based HMAC, the length will raise by adding 160 bits to the original sequence.

5.2. Syntax Extraction

This step uses Stanford parser to extract the phrase structure of the input sentence. This parser is a Java implementation of probabilistic natural language parsers, a program that works out the grammatical structure of sentences. For instance, which groups of words go together (as "phrases") and which group of words is the subject or object of a verb. [3]

The syntax extraction step modifies the output of this parser as necessary to get the syntax structure of the sentence. The system firstly produces the "wordsgroup" that contains three attributes- Name (e.g. NP), Type (e.g. NN NN), Words (e.g. animal testing). These wordsgroups are combined to form a phrase to achieve the phrase structure. After that, the phrases are grouped together to create the clause structure as below:

NP	DT NN	the cure
VP	MD RB VB	would not exist

The summary of this clause structure is written as the sequence of phrases' Name attribute except VP, verb phrase, and PP, proposition phrase started with "by" in passive clause. In the case of VP, the summary will use the Type attribute for determining the sense of the clause, active or passive. The summary will add the Word attribute when it finds out that the current proposition phrase is started with "by". This summary of clause can be used to find out the alternative syntax forms in the syntax bank at the syntax set creation step.

These clauses are connected by conjunctions words that are tagged with SBAR or SINV by the parser. Beside these words, "S" is also the symbol of representing the start of a sentence or clause by the parser. With this sense, these words can be used to decide the start and end of the clause within the sentence. In the proposed system, "S", "SINV", "SBAR", and "," are used to define the boundary of the clauses inside of a sentence.

Finally, the clause structures of the overall sentence are grouped together in a vector of phrase structures, a sentence's syntax structure.

5.3. Syntax Transformation based Linguistic Steganography

This step can be divided into two sub-steps: syntax set creation and syntax transformation at the sender side or syntax checking at the receiver side.

The syntax set creation takes the syntax phrase structure of an input sentence produced by the syntax extraction step as input, constructs and provides a syntax set for this sentence as output.

At the sender side, the capacity checking step checks whether or not the selected cover text have enough hidden capacity for the intended compressed secret message. If so, the syntax transformation step decides which syntax alternative to transform according to the assigned binary sequence, and transforms the input cover text sentence into this chosen syntax. If not, the cover text must be re-chosen.

For the receiver side, the syntax checking step uses the syntax phrase structure of the stego text sentence that is produced by the syntax extraction step and the syntax set that is the output of the syntax set creation step to finds out the corresponding binary sequence.

5.3.1. Syntax Set Creation

The proposed method uses syntax bank that consists of a number of the syntax groups and has already shared between the sender and the receiver. This set creation task takes the syntax phrase structure produced by the above extraction step as input. It then uses this syntax to search for its transformable syntax alternatives group in syntax bank. If there is more than one clause in the input sentence, the syntax set forms by the combination of syntax groups of all clauses in the sentence.

A syntax set is a combination of all available syntax alternatives for all clauses of a sentence. Hence, the bigger the syntax set, the more message bits it can be hidden. All members of the set are semi-randomly assigned with a unique binary sequence for each. This syntax set's size of a sentence can be calculated by the following equation.

$$L = \prod_{i=1}^{N} M_i \quad (1)$$

Where

M= the number of syntactic forms for each clause

N = the number of clauses in a sentence

L = the size of syntax set

The number of secret bits which can be hidden in a sentence is $\log_2$ of the size of syntax set of the sentence: $\log_2 L$ of Eq (1).

Moreover, sender and receiver have already shared a key that is used as a seed to produce the same random sequence that is used to assign to the syntactic rules of the set. To generate the random sequence without repeat, a newly generated random number is checked whether it is already exist in the sequence. Only the sender and receiver who shared the seed can generate the random sequence of correct order. Even the intruder obtains the syntax set; it cannot be possible to assign the correct binary numbers sequence because of lack of knowledge about the seed to produce the sequence.

5.3.2. Capacity Checking

This step checks the hidden capacity of the input text. It first calculates how many bits can be hidden in the input cover text according to the syntax sets of the input cover text sentences. At the text level, the total number of message bits that can be hidden in a text is the summation of the hidden capacity of all sentences in the text.

$$\text{Capacity} = \sum_{i=1}^{k} S_i \quad (2)$$

where k is the number of sentences in the text and S is the hidden capacity, $\text{Log}_2 L$, of each sentence. If the capacity is greater than or equal to the number of secret message bits intended to hide in this cover text, the cover text is ready for actual transformation. If not enough, the sender has to re-choose again for a larger cover text.

5.3.3. Syntax Transformation

This step transforms the input sentence into the desired syntax form. As for a prototype, our system now implemented and tested with only active-passive transformation. This can be done by the following procedure.

- The phrase structure of the sentence produced by the parser is used to define subject (noun phrase that come before verb phrase), verb (verb phrase), object (noun phrase that come after verb phrase), and other complement phrases (such as adverb phrase).
- The main action verb in the verb phrase is then transformed into its past participle form with the help of the verb table. The verb phrase for the passive form of the sentence is constructed by adding the appropriate singular/plural form of helping verb to the past participle form of the main verb.
- The passive sentence is constructed by making direct object into the subject, adding the passive formed verb phrase, and placing the original subject into a propositional phrase beginning with "by".

There are some limitations in interchanging the active sentence into passive form. These are because of the performance of the parser used. For our system, we assume that the parser used, the Stanford parser, is a perfect parser.

5.4. Postprocessing

This step is applied at the receiver side. It first generates the exact compressed code table by using the predefined text and the compression algorithms. It then searches the termination bits in the extracted bits sequence and decompresses by the code table. If needed, the system processes error control code decoding of the extracted bits according to the sender's choice. The output of this step is a secret message in human readable form.

5.5. Experimental Result

The proposed system is evaluated with its three requirements, capacity, imperceptibility, and robustness concerns by using Reuters corpus [17] as testbed and MT Evaluation Tool Kit of NIST [14] as a testing tool.

5.5.1. Evaluation of Capacity

With the proposed system, it can be seen that the number of message bits to embed in the cover text increases when applying error control methods. As a result, this fact causes more sentences that are required to carry these bits. But, the length of error controlled message bits is nearly the same as their ordinary ASCII code length because of compression methods that are applied to secret message before embedding into the cover text. According to the experiment done on the Reuter corpus, the payload ratio is about 1:8 for message bits to clauses required to carry these message bits.

The number of bits after adding one or both of ECC methods always higher than that of ordinary only compressed message. This fact can be seen at Figure 5.3 by means of 10 messages that regularly increased in lengths. As the original message length increases, the number of message bits to embed in the cover text also increases. Aside from this,

application of error control methods, Hamming code or SHA-1 based HMAC, provides less message bits than that of using both two methods.

Figure 5.4 represents the number of sentences required to embed these 10 messages by adding error control methods. This graph can tell that the application of error control codes required larger cover text for embedding secret information while maintaining its security.

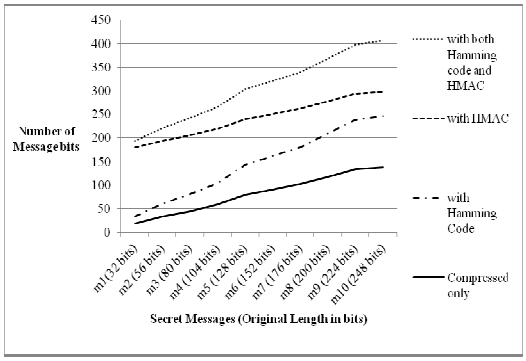

Figure 5.3 Comparison of the number of message bits when applying error control methods

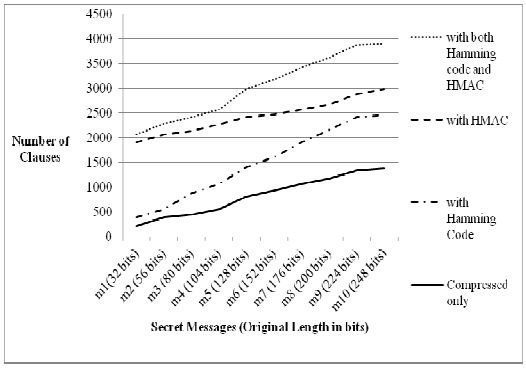

Figure 5.4 Comparison of the number of sentences to carry message bits when adding error control methods

5.5.2. Evaluation of Imperceptibility

Imperceptibility of the proposed system can be measured by BLEU and NIST scores. These measures can be shown by ordinary compressed form of secret message "test" and its error controlled forms. NIST comparison of these forms is described at Figure 5.5 while BLEU scores can be presented at Figure 5.6. From these figures, we can summarize that their BLEU scores are always greater than or equal to 0.88. For NIST, all forms of sentences provide higher scores as their N-gram length increases. Thus, we can say that the usage of error control codes cannot harm the imperceptibility of the proposed system considerably.

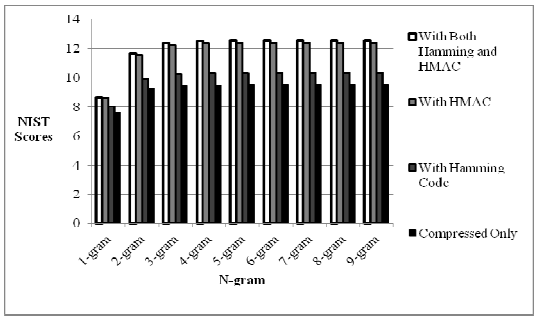

Figure 5.5 NIST Comparison of Ordinary Compressed Message and its error controlled forms

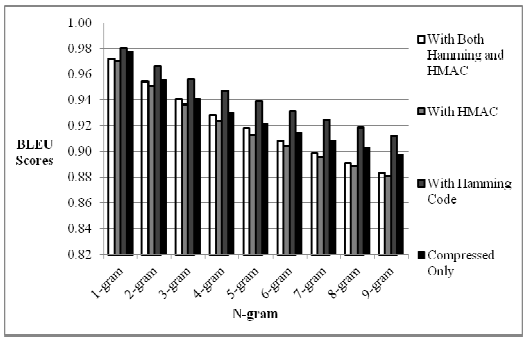

Figure 5.6 BLEU Comparison of Ordinary Compressed Message and its error controlled form

5.5.3. Evaluation of Robustness

There are three types of attacks that can be seen in linguistic steganography: changing, insertion, and deletion. Here, while changing attack can change the value of message bits, two other attacks can cause burst errors propagated to the end of the message bits string. By means of security, these attacks can threaten confidentiality and integrity of secret message.

To overcome these attacks, the proposed system applies two error control codes. Hamming code is used for controlling the changing attacks while SHA-1 based HMAC is applied to face with the burst error propagation caused by insertion and deletion attacks.

Hamming code detects up to 2 errors and corrects 1 error per each 7 bits codeword. In [10], they suggest to transmit codeword in the column order instead of ordinary row order. When using this trick, they say that without any other errors, up to k consecutive errors can be corrected when k codewords are used. In the proposed approach, the message bits are divided into 4 bits blocks to use Hamming code of 4 data bits of 7 bits codeword. According to [9], the proposed system can correct burst error of up to 25% of embedded message bits.

In the case of HMAC, the security relies on the resistance of underlying hash algorithm, SHA-1 in the proposed system. According to its output hash size, SHA-1 can resist against less than 80 bits for

collision resistance and 160 bits for preimage resistance. This fact says that the proposed system can detect up to 160 errors in the message bits string.

6. Conclusion and Further Extension

This work developed a prototype system to express the usability of error control techniques in syntactic steganography. To overcome the capacity requirement of adding error control codes for improving robustness, the system first compress the secret message. According to the experiments, it can be shown that ECC applied compressed message length is nearly the same as its ordinary ASCII coded length. To maintain secrecy, the system uses the key-controlled random assignment for syntax forms in the syntax set. The same random sequence cannot be generated by the intruders without having the key that is the basic of random number assigned to syntax forms. Based on the clause level structure of sentence to determine the transformability of the sentence, the proposed method can carry more secret bits than the existing methods. Moreover, the perceptibility of the sentence will not be higher after embedding because some sentences can carry message bits without transformation. This is because the ordinary syntax of these sentences already represents the required bits.

The future works can be done by applying more secure random number generation methods. A higher degree of confidence can be achieved by using MAC as a basis of a pseudo random number generator (PRNG). Moreover, when linguistic structure is used as an embedding unit, the method depends on the development of natural language processing (NLP) tools. There is no perfect NLP tool now. This can cause imperfect implementation of steganography application. The more powerful NLP tools are used, the more prefect steganography application can result. Therefore, it is looking forward to use perfect NLP tools.

References

[1] A.M.Nanhe, M.P.Kunjir, S.V.Sakdeo, "Improved Synonym Approach to Linguistic Steganography", http://dsl.serc.iisc.ernet.in/~mayuresh/ImprovedSynonymApproachToLinguisticSteganography.pdf, 2008 (accessed at 15.3.2011).

[2] C.Y.Chang, S.Clark, "Practical Linguistic Steganography using Contextual Synonym Substitutionand Vertex Colour Coding", in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-10), pp.1194-1203, Cambridge, MA, October 2010.

[3] D. D. Lewis, Y. Yang, T. Rose, F. Li,"RCV1: A New Benchmark Collection for Text Categorization Research", Journal of Machine Learning Research, 5:361-397, 2004.

[4] Error detection and correction, Encyclopedia of Microcomputers: Volume 6, p. 343.

[5] H.M.Meral, B.Sankur, A.S.Özsoy, T.Güngör, and E.Sevinc, "Natural language watermarking via morphosyntactic alterations", Journal of Computer Speech and Language Vol.23 Iss.1, pages 107–125, 2009.

[6] Introduction to linear codes, January 2010, http://www.cs.cmu.edu/~venkatg/teaching/codingtheory/notes/notes1.pdf, Accessed at 25.7.2012

[7] K. L. Kermanidis, "Capacity-rich Knowledge-poor Linguistic Steganography", Journal of Information Hiding and Multimedia Signal Processing, Volume 2, Number 3, July 2011.

[8] M. Bellare, R. Canetti, H. Krawczyk, "Keying Hash Functions for Message Authentication", 1996.

[9] M. Grosvald, C. O. Orgun, "Free from the Cover Text: A Human-generated Natural Language Approach to Text-based Steganography", Journal of Information Hiding and Multimedia Signal Processing, Volume 2, Number 2, April 2011.

[10] M. Humphrys, "Hamming Code (1 bit error correction)",http://computing.dcu.ie/~humphrys/Notes/Networks/data.hamming.html

[11] M. K. Topkara, "New Designs for Improving The Efficiency and Resilence of Natural Language Watermarking", Ph.D thesis, Purdue University, West Lafayette, Indiana, August 2007.

[12] M.Topkara, U.Topkara, and M.J.Atallah, "Words Are Not Enough: Sentence Level Natural Language Watermarking", MCPS'06, Santa Barbara, California, USA, October 2006.

[13] M.Y.Kim, "Text Watermarking by Syntactic Analysis", 12th WSEAS International Conference on Computers, Heraklion, Greece, July 2008.

[14] N. I. of Standards and Technology, "Machine translations benchmark tests provided by national institute of standards and technology," http://www.nist.gov/speech/tests/mt/resources/scoring.htm, Accessed at 21.11.2012.

[15] Q.Dang, "Recommendation for Applications Using Approved Hash Algorithms", NIST Special Publication 800-107, February 2009.

[16] Sentence patterns, http://www.towson.edu/ows/SentPatt.htm, Accessed at 28.11.2012

[17] Text Categorization Corpora, http://disi.unitn.it/moschitti/corpora.htm, Accessed at 21.11.2012.

[18] U. Topkara, M. Topkara, M. J. Atallah, "The Hiding Virtues of Ambiguity: Quantifiably Resilient Watermarking of Natural Language Text through Synonym Substitutions", MM&Sec'06, Geneva, Switzerland, September 26–27, 2006.