Linguistic Steganography Using Automatically Generated Paraphrases

Ching-Yun Chang
University of Cambridge
Computer Laboratory
Ching-Yun.Chang@cl.cam.ac.uk

Stephen Clark
University of Cambridge
Computer Laboratory
Stephen.Clark@cl.cam.ac.uk

Abstract

This paper describes a method for checking the acceptability of paraphrases in context. We use the Google n-gram data and a CCG parser to certify the paraphrasing grammaticality and fluency. We collect a corpus of human judgements to evaluate our system. The ultimate goal of our work is to integrate text paraphrasing into a Linguistic Steganography system, by using paraphrases to hide information in a cover text. We propose automatically generated paraphrases as a new and useful source of transformations for Linguistic Steganography, and show that our method for checking paraphrases is effective at maintaining a high level of imperceptibility, which is crucial for effective steganography.

1 Introduction

Steganography is concerned with hiding information in some cover medium, by manipulating properties of the medium in such a way that the hidden information is not easily detectable by an observer (Fridrich, 2009). The covert communication is such that the very act of communication is to be kept secret from outside observers. A related area is Watermarking, in which modifications are made to a cover medium in order to identify it, for example for the purposes of copyright. Here the changes may be known to an observer, and the task is to make the changes in such a way that the watermark cannot easily be removed.

There is a large literature on image steganography and watermarking, in which images are modified to encode a hidden message or watermark. Image stegosystems exploit the redundancy in an image representation together with limitations of the human visual system. For example, a standard image stegosystem uses the least-significant-bit (LSB) substitution technique. Since the difference between 11111111 and 11111110 in the value for red/green/blue intensity is likely to be undetectable by the human eye, the LSB can be used to hide information other than colour, without being perceptable by a human observer.[1]

A key question for any steganography system is the choice of cover medium. Given the ubiquitous nature of natural languages and electronic text, text is an obvious medium to consider. However, the literature on Linguistic Steganography, in which linguistic properties of a text are modified to hide information, is small compared with other media (Bergmair, 2007). The likely reason is that it is easier to make changes to images and other non-linguistic media which are undetectable by an observer. Language has the property that even small local changes to a text, e.g. replacing a word by a word with similar meaning, may result in text which is anomalous at the document level, or anomalous with respect to the state of the world. Hence finding linguistic transformations which can be applied reliably and often is a challenging problem for Linguistic Steganography.

In this paper we focus on steganography rather than watermarking, since we are interested in the requirement that any changes to a text be imperceptible to an observer. Figure 1 shows the Linguistic Steganography framework. First, some secret message, represented as a sequence of bits, is hidden in a

[1]The observer may also be a computer program, designed to detect statistical anomalies in the image representation which may indicate the presence of hidden information.

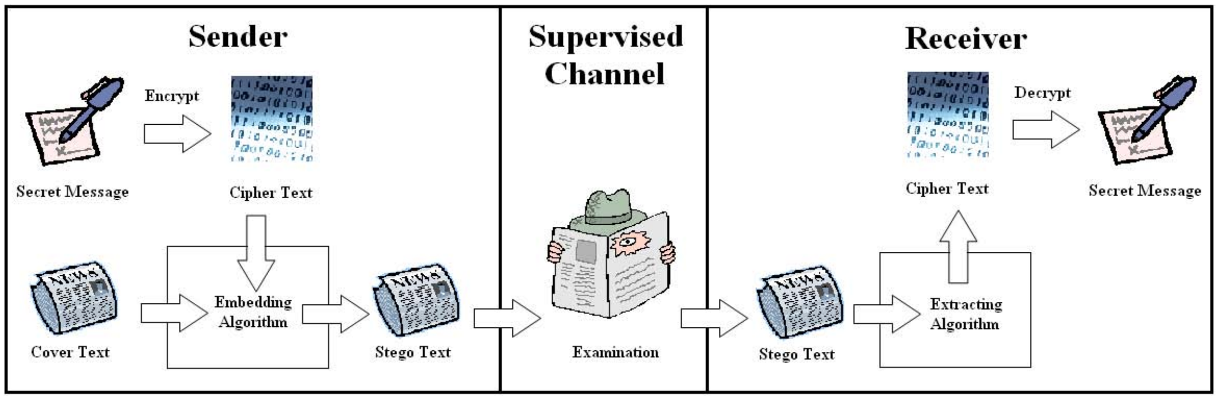


Figure 1: The Linguistic Steganography framework

cover text using the embedding algorithm, resulting in the *stego text*.[2] Next, the stego text passes the human observer, who is happy for innocuous messages to pass between the sender and receiver, but will examine the text for any suspicious looking content. Once the stego text reaches the receiver, the hidden message is recovered using the extracting algorithm.

There is a fundamental tradeoff in all steganography systems, and one that is especially apparent in the Linguistic Steganography framework: the tradeoff between *imperceptibility* and *payload*. Payload is the number of bits that can be encoded per unit of cover medium, for example per sentence in the linguistic case. The tradeoff arises because any attempt to hide additional information in the cover text, through the application of more linguistic transformations, is likely to increase the chances of raising the suspicions of the observer, by introducing anomalies into the text.

The key elements of a Linguistic Steganography system are the linguistic transformation and the embedding method. In this paper we focus on the linguistic transformation. Section 5 describes a possible embedding method for our framework, and for readers unfamiliar with linguistic steganography shows how linguistic transformations can be used to embed hidden bits in text.

Section 2 describes some of the previous transformations used in Linguistic Steganography. Note that we are concerned with transformations which are linguistic in nature, rather than dealing with superficial properties of the text, e.g. the amount of white space between words (Por et al., 2008). Our proposed method is based on the automatically acquired paraphrase dictionary described in Callison-Burch (2008), in which the application of paraphrases from the dictionary encodes secret bits. One advantage of the dictionary is that it has wide coverage, being automatically extracted; however, a disadvantage is that it contains many paraphrases which are either inappropriate, or only appropriate in certain contexts. Since we require any changes to be imperceptible to a human observer, it is crucial to our system that any uses of paraphrasing are grammatical and retain the meaning of the original cover text.

In order to test the grammaticality and meaning preserving nature of a paraphrase, we employ a simple technique based on checking whether the contexts containing the paraphrase are in the Google n-gram corpus. This technique is based on the simple hypothesis that, if the paraphrase in context has been used many times before on the web, then it is an appropriate use. We test our n-gram-based system against some human judgements of the grammaticality of paraphrases in context. We find that using larger contexts leads to a high precision system (100% when using 5-grams), but at the cost of a reduced recall. This precision-recall tradeoff reflects the inherent tradeoff between imperceptibility and payload in a Linguistic Steganography system. We also experiment with a CCG parser (Clark and Curran, 2007), requiring that the contexts surrounding the original phrase and paraphrase are assigned

[2]The message may have been encrypted initially also, as in the figure, but this is not important in this paper; the key point is that the hidden message is a sequence of bits.

the same CCG lexical categories by the parser. This method increases the precision of the Google n-gram check with a slight loss in recall.

A contribution of this paper is to advertise the Linguistic Steganography problem to the ACL community. The requirement that any linguistic transformation maintain the grammaticality and meaning of the cover text makes the problem a strong test for existing NLP technology.

2 Previous Work

2.1 Synonym Substitution

The simplest and most straightforward subliminal modification of text is to substitute selected words with their synonyms. The first lexical substitution method was proposed by Chapman and Davida (1997). Later works, such as Atallah et al. (2001a), Bolshakov (2004), Taskiran et al. (2006) and Topkara et al. (2006b), further made use of part-of-speech taggers and electronic dictionaries, such as WordNet and VerbNet, to increase the robustness of the method. Taskiran et al. (2006) attempt to use context by prioritizing the alternatives using an n-gram language model; that is, rather than randomly choose an option from the synonym set, the system relies on the language model to select the synonym. Topkara et al. (2005) and Topkara et al. (2006b) report an average embedding capacity of 0.67 bits per sentence for the synonym substitution method.

2.2 Syntactic Transformations

The second and the most widely used manipulations for linguistic steganography are syntactic transformations. This method is based on the fact that a sentence can be transformed into more than one semantically equivalent syntactic structure, using transformations such as passivization, topicalization and clefting. The first syntactic transformation method is presented by Atallah et al. (2001a). Later, Atallah et al. (2001b) embedded information in the tree structure of the text by adjusting the structural properties of intermediate representations of sentences. In other words, instead of performing lexical substitution directly to the text, the secret message is embedded into syntactic parse trees of the sentences. Liu et al. (2005), Meral et al. (2007), Murphy (2001), Murphy and Vogel (2007) and Topkara et al. (2006a) all belong to the syntactic transformation category. After embedding the secret message, modified deep structure forms are converted into the surface structure format via language generation tools. Atallah et al. (2001b) and Topkara et al. (2006a) attained the embedding capacity of 0.5 bits per sentence with the syntactic transformation method.

2.3 Semantic Transformations

The semantic transformation method is the most sophisticated approach for linguistic steganography, and perhaps impractical given the current state-of-the-art for NLP technology. It requires some sophisticated tools and knowledge to model natural language semantics. Atallah et al. (2002) used semantic transformations and embed information in text-meaning representation (TMR) trees of the text by either pruning, grafting or substituting the tree structure with information available from ontological semantic resources. Vybornova and Macq (2007) aimed to embed information by exploiting the linguistic phenomenon of presupposition, with the idea that some presuppositional information can be removed without changing the meaning of a sentence.

3 Data Resources

3.1 Paraphrase Dictionary

The cover text used for our experiments consists of newspaper sentences from Section 00 of the Penn Treebank (Marcus et al., 1993). Hence we require possible paraphrases for phrases that occur in Section 00. The paraphrase dictionary that we use was generated for us by Chris Callison-Burch, using the technique described in Callison-Burch (2008), which exploits a parallel corpus and methods developed for statistical machine translation.

Table 1 gives summary statistics of the paraphrase dictionary and its coverage on Section 00 of the Penn Treebank. The length of the extracted n-gram phrases ranges from unigrams to five-grams. The coverage figure gives the percentage of sentences which have at least one phrase in the dictionary. The coverage is important for us because it determines the payload capacity of the embedding method described in Section 5.

Table 2 lists some examples 5-gram phrases and paraphrases from the dictionary. The format of the

N-gram	Number of phrases	Coverage on section 00 (%)
Unigrams	5,856	99
Bigrams	13,473	96
Trigrams	6,574	65
Four-grams	1,604	40
Five-grams	295	10

Table 1: Statistics for the paraphrase dictionary

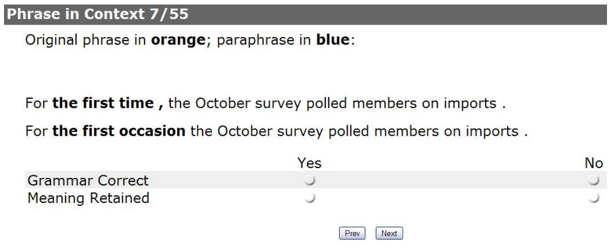
Original phrase	Paraphrases
the end of this year	later this year the end of the year year end
a number of people	some of my colleagues differences the European peoples party the PPE group

Table 2: Example phrases and paraphrases from the dictionary

dictionary is a mapping from phrases to sets of possible paraphrases. Each paraphrase also has a probability, based on a statistical machine translation model, but we do not use that feature here. The examples show that, while some of the paraphrases are of a high quality, some are not. For example, *differences* is unlikely to be a suitable paraphrase for *a number of people* in any context. Moreover, there are some ⟨phrase, paraphrase⟩ pairs which are only suitable in particular contexts. For example, *year end* is an unsuitable paraphrase for *the end of this year* in the sentence *The chart compares the gold price at the end of last year with the end of this year*. Barzilay and McKeown (2001) also note that the applicability of paraphrases is strongly influenced by context. Section 4 describes our method for determining if a paraphrase is suitable in a given context.

3.2 Google N-gram Data

The Google n-gram data was collected by Google Research for statistical language modelling, and has been used for many tasks such as lexical disambiguation (Bergsma et al., 2009), and contains English n-grams and their observed frequency counts, for counts of at least 40. The striking feature of the n-gram corpus is the large number of n-grams and the size of the counts, since the counts were extracted from over 1 trillion word tokens of English text on publicly accessible Web pages collected in January 2006. For example, the 5-gram phrase *the part that you were* has a count of 103. The compressed data is around 24 GB on disk.


Phrase in Context 7/55

Original phrase in **orange**; paraphrase in **blue**:

For **the first time ,** the October survey polled members on imports .

For **the first occasion** the October survey polled members on imports .

	Yes	No
Grammar Correct	○	○
Meaning Retained	○	○

Figure 2: The web-based annotation system

3.3 Paraphrase Judgement Corpus

The focus of the paper is to develop an automatic system for checking the grammaticality and fluency of paraphrases in context. In order to evaluate the system, we collected some human judgements, based on 70 sentences from Section 00 of the Penn Treebank. For each sentence, we took every phrase in the sentence which is in the dictionary, and for each paraphrase of that phrase, replaced the phrase with the paraphrase to create an instance. This procedure resulted in 500 cases of paraphrases in context.

Each case was then evaluated by a human judge, using a web-based annotation system that we developed. The judges were asked to judge each case on two dimensions: a) whether the paraphrase is *grammatical* in context; and b) whether the paraphrase *retains the meaning* of the original phrase given the context. Figure 2 gives a screen shot of the annotation system.

50 of the 500 cases were judged by two judges, in order to obtain some indication of whether the grammaticality and meaning retention judgements are viable; the rest were judged by one annotator. (The 500 instances were randomly distributed among 10 native speakers, each being given 55 instances to judge.) For the meaning retention check, only 34 out of the 50 cases received the same judgement. One reason for the low agreement may be that, for 11 of the 16 disagreement cases, we were asking annota-

tors to judge the meaning retention of paraphrases which had been judged to be ungrammatical in context, which may not be a meaningful task. For the grammatical check, 42 out of the 50 cases received the same judgement, a much higher level of agreement.

Since the meaning retention judgements were unreliable, we used only the grammatical judgements to evaluate our system. Hence we are interested in evaluating whether our n-gram and parser-based systems can determine if a paraphrase is *grammatical* in context. Meaning retention is important for the imperceptibility requirement, but grammaticality is even more so, since ungrammatical sentences will be easy for an observer to spot. However, we recognise that only testing for grammaticality does not fully test the imperceptibility properties of the system, only part of it.

For the 8 cases which received different judgements on grammaticality, the second author of this paper made the definitive judgement, which resulted in a test set of 308 paraphrases judged as grammatical in context, and 192 paraphrases judged as ungrammatical in context.

4 Proposed Method and Experiments

4.1 Google N-gram Method

The main idea for testing the use of paraphrases is to check if the various contextual n-grams appear in the Google n-gram data, or were already in the original sentence (before paraphrasing). Let us first define some notation to be used in describing the method. The leftmost and rightmost <*m*> words in the phrase/paraphrase are represented as <*m*>*INLeft* and <*m*>*IN**Right*, respectively. Words at the left and right side of the substituted phrase are defined as <*c*>***OUT****Left* and <*c*>***OUT****Right*, where <*c*> is an integer which indicates the number of words represented. Also, we define a context window pair $W^{<c>}_{<n>} = (W_L{}^{<c>}_{<n>}, W_R{}^{<c>}_{<n>})$, where $W_L{}^{<c>}_{<n>}$ is composed by <*c*>***OUT****Left* concatenated with <*n-c*>*IN**Left*, and $W_R{}^{<c>}_{<n>}$ is composed by <*n-c*>*IN**Right* concatenated with <*c*>***OUT****Right*. Figure 3 gives an example of the context window pairs W^1_3 and W^2_3 in the sentence *Soviets said that it is* ***too early to*** *say whether that will happen* where the phrase *too early to* is being considered in context.

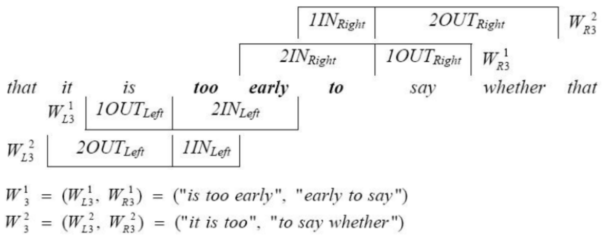

1IN_Right
2OUT_Right
W_R3^2
2IN_Right
1OUT_Right
W_R3^1
that it is too early to say whether that
W_L3^1
1OUT_Left
2IN_Left
W_L3^2
2OUT_Left
1IN_Left
W_3^1 = (W_L3^1, W_R3^1) = ("is too early", "early to say")
W_3^2 = (W_L3^2, W_R3^2) = ("it is too", "to say whether")

Figure 3: An example of the context window pair

```
INPUT: $S, P, P', n, maxC$
OUTPUT: the acceptability of paraphrase $P'$
checked by $(n, maxC)$

FOR each context size $C$ from 1 to $maxC$
    GET a context window pair $W^C_n$
    IF $O(W^C_n)$ is zero THEN
        OUTPUT paraphrase $P'$ fails
END FOR
OUTPUT paraphrase $P'$ passes
```

Figure 4: Procedure for checking acceptability

We define a *google-count function G()*. This function takes a context window pair $W^{<c>}_{<n>}$ as input and outputs a frequency count pair of $W^{<c>}_{<n>}$ recorded in the Google n-gram data. If a context window cannot be found in the Google n-gram data, the frequency count of that window is zero. Also, we define a binary *occurrence function O()*. It is used to determine whether a context window pair can be passed as acceptable. The input of this function is $W^{<c>}_{<n>}$. The function outputs one if either both $W_L{}^{<c>}_{<n>}$ and $W_R{}^{<c>}_{<n>}$ already occurred in the original sentence (before paraphrasing) or if the frequency counts output by *G(*$W^{<c>}_{<n>}$*)* are both greater than zero.

The two major components in our method are the paraphrase dictionary and the Google n-gram data. Once a phrase P in the cover sentence S is matched with that in the paraphrase dictionary, we test the use of its paraphrase P' by the following method. This method takes into account maximum C contextual words at both sides of the target phrase, and uses Google n-gram data as a check, where n = 2, 3, 4 or 5, and $maxC$ = 1 to $n - 1$. Each pair of (n, $maxC$) provides a separate check, by considering both left and right contexts for these values.

Figure 4 describes the procedure for checking the acceptability of paraphrasing phrase P with P' in a given sentence S, given the n-gram size and the maximum considered context size $maxC$. For example, we want to check the acceptability of the paraphrase in context shown in Figure 3 by using google tri-gram data (n = 3) and taking maximum context size equal to two into consideration ($maxC$ = 2). The procedure starts from taking context size C equal to one into account, namely checking the occurrence of W_3^1. If the paraphrase P' passes the current test, in the next iteration it will be tested by taking one more context word into account, namely W_3^2. However, If the paraphrase P' fails the current (n, C) check the checking procedure will terminate and report that the paraphrase fails. In contrast, if the paraphrase passes all the (n, C) checks where C = 1 to $maxC$, the procedure determines the paraphrase as acceptable. What is happening is that an n-gram window is effectively being shifted across the paraphrase boundary to include different amounts of context and paraphrase.

4.2 Syntactic Filter

In order to improve the grammaticality checking, we use a parser as an addition to the basic Google n-gram method. We use the Clark and Curran (2007) CCG parser to analyse the sentence before and after paraphrasing. Combinatory Categorial Grammar (CCG) is a lexicalised grammar formalism, in which CCG lexical categories — typically expressing subcategorisation information — are assigned to each word in a sentence. The grammatical check works by checking if the words in the sentence outside of the phrase and paraphrase receive the same lexical categories before and after paraphrasing. If there is any change in lexical category assignment to these words then the paraphrase is judged ungrammatical. Hence the grammar check is at the word, rather than derivation, level; however, CCG lexical categories contain a large amount of syntactic information which this method is able to exploit.

4.3 Results

The test corpus described in Section 3.3 was split into development and test data: 100 instances for development and 400 for testing. The development data was used for preliminary experiments. For the test data, 246 of the examples (61.5%) had been judged as grammatical, and 154 (38.5%) had been judged as ungrammatical by the annotators.

	Acc%	P%	R%	F%
baseline	61.5	61.5	100.0	76.2
parser	68.3	67.4	93.9	78.4

Table 3: Grammar check using CCG parser

The performance of the system is evaluated using accuracy, precision, recall and balanced F-measure. Accuracy is the percentage of correct judgements over all grammatical and ungrammatical paraphrases. Precision is the percentage of paraphrases judged grammatical by the system which are judged grammatical by the human judges, and recall is the percentage of paraphrases judged grammatical by human judges which are also judged grammatical by the system. Precision and recall are relevant in our setting because high precision implies high imperceptibility, since grammatical phrases in context are less likely to be viewed as suspicious by the observer; whereas high recall maximises the payload (given the dictionary), since high recall implies that phrases are being paraphrased where possible (and hence embedding as much information as possible).

An accuracy baseline is obtained by always returning the majority class, in this case always judging the paraphrase grammatical, which gives an accuracy of 61.5%. Table 3 gives the performance when only the CCG parser is used for checking grammaticality. As far as steganography is concerned, the precision is low, since over 30% of the paraphrases used are ungrammatical, which is likely to raise the suspicions of the observer.

Table 4 gives the results for the Google n-gram method, for various n-gram and context sizes. As the n-gram size increases — meaning that a larger part of the context is used — the accuracy falls below that of the baseline. However, from a steganography aspect, accuracy is not useful, since the trade-off between precision and recall is more relevant. As expected, with larger n-grams checking the left and right contexts, the precision increases, reaching 100% for the 5-grams. Hence, as far as grammaticality judgements are concerned, the imperceptibility requirement is completely satisified. However, the large drop in recall means that the imperceptibil-

N-gram	Context Size	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)
2-gram	1	62.0	62.1	98.0	76.0
3-gram	1	62.5	65.1	84.2	73.4
	2	67.3	72.9	74.4	73.6
4-gram	1	58.5	71.3	54.5	61.8
	2	53.2	84.7	29.3	43.5
	3	51.8	89.6	24.4	38.3
5-gram	1	54.8	85.0	32.1	46.6
	2	43.5	95.5	8.5	15.7
	3	41.0	100.0	4.1	7.8
	4	41.0	100.0	4.1	7.8

Table 4: Performance of google n-gram method

N-gram	Context Size	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)
2-gram	1	68.0	67.7	91.9	78.0
3-gram	1	67.3	70.9	79.3	74.9
	2	69.5	77.7	70.7	74.0
4-gram	1	59.5	75.6	50.4	60.5
	2	53.8	88.6	28.5	43.1
	3	52.0	92.2	24.0	38.1
5-gram	1	53.8	86.8	29.3	43.8
	2	43.3	95.2	8.1	15.0
	3	41.0	100.0	4.1	7.8
	4	41.0	100.0	4.1	7.8

Table 5: Performance of google n-gram method with the CCG parser filter

ity is achieved at the cost of a reduced payload, since many of the grammatical paraphrases that could be used to embed information are being discarded.

Table 5 shows the results for the Google n-gram method followed by the parser check; that is, if the Google n-gram method judges the paraphrase to be grammatical, then it is passed to the CCG parser for an additional check. Adding the parser generally increases the precision with a slight loss in recall. Which settings are best to use in practice would depend on how the steganography user wished to trade off imperceptibility for payload.

5 Possible embedding method

In this section, we propose a linguistic hiding method which can be integrated with an automatic paraphrasing system. It needs a large paraphrase dictionary to determine modifiable phrases and provide available paraphrases. The embedding capacity of the proposed linguistic stegosystem relies on the number of paraphrasable sentences in the cover text. If every sentence in the cover text is paraphrasable, the system can have the maximum embedding capacity equal to 1 bit per sentence which is comparable to other linguistic steganography methods using syntactic transformations and synonym substitution.

5.1 Data Embedding Procedure

First the sentences in a cover text T are identified using a sentence segmentation algorithm, giving N sentences s_1, s_2,..., s_N. The paraphrasability of each sentence is then checked using our automatic method. If a sentence contains at least one paraphrasable phrase, we call the sentence a *paraphrasable sentence* or a *non-paraphrasable sentence* otherwise. Let *D* be the maximum number of sentence boundaries between two subsequent paraphrasable sentences in *T*. Thus, for every *D* sentences within a cover text *T*, there will be at least one paraphrasable sentence. Let every unit of *D* sentences serve as one embedding unit in which a single secret bit can be embedded. If we want to embed 0 in an embedding unit, we transform all the paraphrasable sentences in this embedding unit to non-paraphrasable sentences (assuming certain properties of the dictionary; see end of this section for discussion). If we want to embed 1, we leave the embedding unit without any modifications.

Figure 5 demonstrates the embedding of the secret bitstring 101 in a cover text containing nine sentences t_1, t_2,..., t_9 defined by a sentence segmentation algorithm. First, t_1, t_3, t_4, t_7 and t_9 are determined as paraphrasable sentences and thus *D*, the

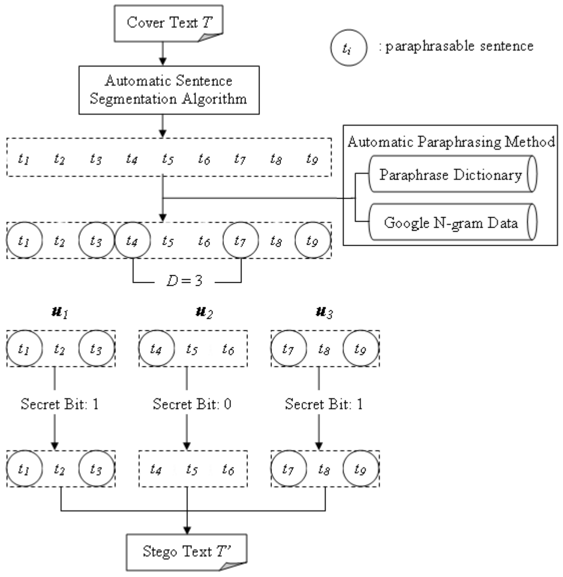

Cover Text T
t_i : paraphrasable sentence
Automatic Sentence Segmentation Algorithm
Automatic Paraphrasing Method
Paraphrase Dictionary
Google N-gram Data
t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9
$D=3$
u_1
u_2
u_3
Secret Bit: 1
Secret Bit: 0
Secret Bit: 1
Stego Text T'

Figure 5: Embedding secret bits in a cover text using sentence segmentation method

size of an embedding unit, is 3. Next, we segment the cover text into three embedding units u_1, u_2 and u_3, each of which contains three sentences. Since we want to embed secret bits 101 in u_1, u_2 and u_3 respectively, the embedding unit u_2 should contain no paraphrasable sentence. That is, the paraphrasable phrase in t_4 should be replaced by its paraphrase. Finally, the stego text is output and sent along with the private key *D* to the other party. A private key is known only to the parties that exchange messages.

In order for this method to work, we require certain properties of the paraphrase dictionary. For example, it is crucial that, once a phrase has been paraphrased, it does not produce another phrase that can be paraphrased. This can be achieved by simply requiring that any paraphrase 'on the RHS' of the dictionary does not also appear as a phrase on the LHS. In fact, this is not so unnatural for the Callison-Burch dictionary, which consists of phrases mapped to sets of paraphrases, many of which only appear on one side.

5.2 Data Extracting Procedure

For extracting the secret data, first, the stego text T' undergoes sentence segmentation, and *N* defined sentences s'_1, s'_2,..., s'_N are obtained. According to the private key *D*, every *D* sentences are treated as an information unit, and in each unit we check the occurrence of paraphrasable sentences making use of our paraphrasing method. If an information unit contains at least one paraphrasable sentence, this information unit implies the embedding of 1. In contrast, if none of the sentences in the information unit are paraphrasable, it implies the embedding of 0. Hence, in order to recover the hidden message, the receiver requires the sentence segmentation algorithm, the paraphrase dictionary, the automatic program determining grammaticality of paraphrases in context, and the secret key D. The extraction process essentially reverses the embedding method.

6 Conclusions

The contributions of this paper are to develop an automatic system for checking the grammaticality and fluency of paraphrases in context, and the proposal of using paraphrases as a suitable transformation for Linguistic Steganography. An advantage of our proposed method is that it is somewhat language and domain independent, requiring only a paraphrase dictionary and a Google n-gram corpus, both of which are likely to be available for a range of languages in the future.

There are various practical issues in the application of Linguistic Steganography systems that we have chosen to ignore. For example, we have not discussed the choice of cover text. If a newspaper article were chosen as the cover text, then any changes could be easily found in practice by comparing the stego text with the original article, which is likely to be readily available. Another interesting question that we have not addressed is whether some languages are better suited to Linguistic Steganography than others, or whether some languages are better suited to particular linguistic transformations than others. Finally, we have only evaluated our grammatical checker and not the steganography system itself (other than giving an indication of the likely payload). How best to evaluate the imperceptibility of such a system we leave to future work.

Acknowledgements

We would like to thank Chris Callison-Burch for providing the paraphrase dictionary, Katja Markert, Stephen Pulman, Laura Rimell, and the anonymous reviewers for useful comments. Ching-Yun Chang was funded by an Oxford University Clarendon scholarship.

References

Mikhail J. Atallah, Craig J. McDonough, Victor Raskin, and Sergei Nirenburg. 2001a. Natural language processing for information assurance and security: an overview and implementations. In *Proceedings of the 2000 workshop on New security paradigms*, pages 51–65, Ballycotton, County Cork, Ireland.

Mikhail J. Atallah, Victor Raskin, Michael C. Crogan, Christian Hempelmann, Florian Kerschbaum, Dina Mohamed, and Sanket Naik. 2001b. Natural language watermarking: design, analysis, and a proof-of-concept implementation. In *Proceedings of the 4th International Information Hiding Workshop*, volume 2137, pages 185–199, Pittsburgh, Pennsylvania.

Mikhail J. Atallah, Victor Raskin, Christian F. Hempelmann, Mercan Karahan, Umut Topkara, Katrina E. Triezenberg, and Radu Sion. 2002. Natural language watermarking and tamperproofing. In *Proceedings of the 5th International Information Hiding Workshop*, pages 196–212, Noordwijkerhout, The Netherlands.

Regina Barzilay and Kathleen R. McKeown. 2001. Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th ACL*, pages 50–57, Toulouse.

Richard Bergmair. 2007. A comprehensive bibliography of linguistic steganography. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505.

Shane Bergsma, Dekang Lin, and Randy Goebel. 2009. Web-scale n-gram models for lexical disambiguation. In *Proceedings of the 21st International Joint Conference on Artifical Intelligence*, pages 1507–1512, Pasadena, CA.

Igor A. Bolshakov. 2004. A method of linguistic steganography based on coladdressally-verified synonym. In *Information Hiding: 6th International Workshop*, volume 3200, pages 180–191, Toronto, Canada.

Chris Callison-Burch. 2008. Syntactic constraints on paraphrases extracted from parallel corpora. In *Proceedings of the EMNLP Conference*, pages 196–205, Honolulu, Hawaii.

Mark Chapman and George I. Davida. 1997. Hiding the hidden: A software system for concealing ciphertext as innocuous text. In *Proceedings of the First International Conference on Information and Communication Security*, volume 1334, pages 335–345, Beijing.

Stephen Clark and James R. Curran. 2007. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Comp. Ling.*, 33(4):493–552.

Jessica Fridrich. 2009. *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge University Press, first edition.

Yuling Liu, Xingming Sun, and Yong Wu. 2005. A natural language watermarking based on Chinese syntax. In *Advances in Natural Computation*, volume 3612, pages 958–961, Changsha, China.

Mitchell P. Marcus, Beatrice Santorini, and Mary A. Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19:313–330.

Hasan M. Meral, Emre Sevinc, Ersin Unkar, Bulent Sankur, A. Sumru Ozsoy, and Tunga Gungor. 2007. Syntactic tools for text watermarking. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, San Jose, CA.

Brian Murphy and Carl Vogel. 2007. The syntax of concealment: reliable methods for plain text information hiding. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, San Jose, CA.

Brian Murphy. 2001. Syntactic information hiding in plain text. Masters Thesis. Trinity College Dublin.

Lip Y. Por, Ang T. Fong, and B. Delina. 2008. Whitesteg: a new scheme in information hiding using text steganography. *WSEAS Transactions on Computers*, 7:735–745.

Cuneyt M. Taskiran, Mercan Topkara, and Edward J. Delp. 2006. Attacks on linguistic steganography systems using text analysis. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6072, pages 97–105, San Jose, CA.

Mercan Topkara, Cuneyt M. Taskiran, and Edward J. Delp. 2005. Natural language watermarking. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 5681, pages 441–452, San Jose, CA.

Mercan Topkara, Umut Topkara, and Mikhail J. Atallah. 2006a. Words are not enough: sentence level natural language watermarking. In *Proceedings of the ACM Workshop on Content Protection and Security*, pages 37–46, Santa Barbara, CA.

Umut Topkara, Mercan Topkara, and Mikhail J. Atallah. 2006b. The hiding virtues of ambiguity: quantifiably resilient watermarking of natural language text through synonym substitutions. In *Proceedings of the 8th Workshop on Multimedia and Security*, pages 164–174, Geneva, Switzerland.

M. Olga Vybornova and Benoit Macq. 2007. A method of text watermarking using presuppositions. In *Proceedings of the SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, San Jose, CA.