

Linguistic Steganography Detection Based on Perplexity

Peng Meng, Liusheng Huang, Zhili Chen, Wei Yang, Dong Li
(National High Performance Computing Center at Hefei, Department
of Computer Science and Technology, University of Science and
Technology of China, Hefei, 230027, P. R. China)
(E-mail:mengpeng@mail.ustc.edu.cn lshuang@ustc.edu.cn)

Abstract

Linguistic steganography is a technique for hiding information into text carriers. Most of the previous work on linguistic steganography was focused on steganography and there were few researches on steganalysis. Research on attacking methods against linguistic steganography plays an important role in Information Security (IS) area. In this paper, a linguistic steganography detection algorithm based on perplexity is presented. An experiment to detect text segments generated by linguistic steganography system NICETEXT is carried out. The result of our experiment is pretty promising. The experimental accuracy of our method on classification of generated text and normal text exceeds 99% when text size is larger than 400 bytes. Even for sentences, the experimental accuracy exceeds 85%.

Keywords: *Steganography, Language Model, Perplexity, Text Processing, steganalysis*

1. Introduction

Steganography is the art of hiding secret information in carriers. The carriers can be videos, images and texts [1]. Because texts have less redundant, unnecessary and unnoticed data space than videos or images, research on text steganography is very short. With the development of computer networks, much information exchanged and distribution on Internet is based on text. Text with hiding information will draw less attention. This attracts us to focus on analysis of text steganography. Linguistic steganography which generates natural like text or modifies cover text to hide information in text is a branch of text steganography [1].

Steganography is very useful in Information Security area. For example, we can use it disguising ciphertext as normal text, so to thwart the censorship of ciphertext. The NICETEXT system [2] is a method for hiding ciphertext into text that looks like natural language. At the same time the system retains the ability to recover the original data from the generated text. The NICETEXT system changes ciphertext to natural-like text (in this paper, we call it stego-text) according to two components: a dictionary table and

a style template. The dictionary table is a large list of (type,word) pairs where the type may be based on the part-of-speech of word or its synonym set. Such tables may be generated using a part-of-speech tagger. The dictionary is used to randomly generate sequences of words and the style template, which is generated by PCFG or an example text, improves the quality of the generated text by selecting natural sequences of parts-of-speech while controlling capitalization, punctuation, and white space generation. we give an example of NICETEXT system generated text [2]:

...No, don't love me before another minute or two, if you please, John! how little he had looked for such a close to such a wind! Within these words he carried it off, and carried herself off too: nearly stopping at the door ...

steganalysis is to analyze stego-text in order to detect or extract secret messages [1]. A successful steganography algorithm requires that the steganographically hidden message is not detectable by the adversary. This problem conclude two sides: one is the attacker should not be able to find the message, the other is he should not even know whether it exists. Steganalysis is generally considered to be successful when the existence of a message is detected [3].

Our research examines drawbacks of the NICETEXT system, aiming to accurately detect the application of the approach in given text segments. We have found an efficient and accurate detection algorithm which is based on perplexity.

2. Related Work

A few detecting algorithms against linguistic steganography have been proposed. A very famous linguistic steganography detection method was presented in paper [4], which used the statistical characteristics of correlations between the general service words gathered in a dictionary to classify the given text segments into stego-text segments and normal text segments. The total accuracy of discovering stego-text segments and normal text segments was found to be 94.55% when text segments size was 20KB. The paper [5] brought forward a detection method against NICETEXT system which took advantage of distribution of words , the accuracy on discovering stego-text segments and normal

text segments was found to be 87.39% when text segments size was 5KB. Another effective linguistic steganography detection method was proposed by the paper [6], the method used an information entropy-like statistical variable of words together with its variance as two features to classify detected text segments, the Support Vector Machine(SVM) was used as classifier. The detection accuracy could exceed 90% when text size was larger than 4KB. All the three methods fell short of accuracy when text size is less than 5KB.

Our research examines drawbacks of the NICETEXT system, aiming to accurately classify stego-text and normal text in small size. We have used perplexity to distinguish between stego-text segments and normal text segments. The results of our experiment are pretty promising. The experimental accuracy of our method on classification of stego-text and normal texts exceeds 99% when text size is 400 bytes. Even for sentences, the experimental accuracy exceeds 85%. The text size that can effectively classify is one-tenth of the last three algorithms.

3. Preliminaries and Notations

We introduce some preliminaries and definitions in this section.

3.1. N-gram Language Models

A n-gram Language Model (LM) is a statistical model that estimates the prior probabilities of n-gram word strings [7]. The task of predicting the next word can be stated as attempting to estimate the probability of function P:

$$P(w_n|w_1, \dots, w_{n-1}) \quad (1)$$

In this a stochastic problem, we use the history to predict the next word. In practice we often assumption that current word only based on the n-1 words preceding it. If we construct a model that assume all words only based on the n-1 words preceding it, then we have an (n -1)th order Markov model or an n-gram word model [7]. In our experiments, We have used 3-gram model.

3.2. Maximum Likelihood Estimation

One approach to estimate n-gram probabilities from training text is to count the number of n-grams occurring in the text and then define the probabilities as the maximum likelihood estimate [7].

$$P(w_1 \dots w_n) = \frac{C(w_1 \dots w_n)}{N} \quad (2)$$

$$P(w_n|w_1 \dots w_{n-1}) = \frac{C(w_1 \dots w_n)}{C(w_1 \dots w_{n-1})} \quad (3)$$

where N is the number of n-grams in the training text and $C(w_1 \dots w_n)$ is frequency of n-gram $w_1 \dots w_n$ in training text.

The maximum likelihood approach in this case leads to a related problem: observed n-grams will be assigned too much probability and none to the ones that are not observed. There are many methods to solve this problem, in our experiment, we have used Good-Turing estimation [8] to this problem.

3.3. Perplexity

Perplexity is the probability of the test text segment (assigned by the language model), normalized by the number of words N. When we use 3-gram, Perplexity is defined as:

$$P(W) = \sqrt[N]{\prod_{i=1}^N \frac{1}{P(w_i|w_{i-2}, w_{i-1})}} \quad (4)$$

where the initial condition $P(w_1|w_{-1}, w_0)$ is chosen suitably. Minimizing perplexity is the same as maximizing the text's probability.

4. Detection Process

In natural language, normal text has many inherent statistical characteristics that can't be provided by stego-text that generated by NICETEXT system. In this paper, we try to find out the perplexity difference between natural language text and stego-text. Our experiments show that if we create a Language Model from normal text, then use it to calculate the perplexity of normal text and stego-text, the results are very similar, but if we create a Language Model from stego-text, then we use it to calculate the perplexity of normal text and stego-text, the results are very different. So we can use Language Model created by stego-text to distinguish between normal text and stego-text.

In our experiment, experimental data set contains training data set and testing data set. The training data set consists of stego-text and the testing data set consists both stego-text and normal text. The training data set is used to train the Language Model to well distinguish between the "Good" data and "Bad" data. The testing data set is the data we want to detect. We built a corpus from The New York Times and name it Good-Corpus. We built another corpus from text generated by NICETEXT system and name it Bad-Corpus. In the experiment process, The training data set comes from Bad-Corpus and testing data set comes both GoodCorpus and BadCorpus. The experiments have used Stanford Research Institute Language Modeling (SRILM) Toolkit to train LMs [9].

In the algorithm, there are three main processes employed: training, testing and classifying .

First, we check out our training text from Bad-Corpus, the size is about 200M. Parse it into sentences. Because we only want to get relations between word and word, the parsing accuracy is not very important. We use periods mark the end

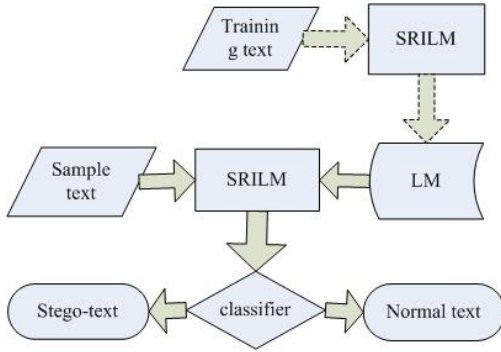


Figure 1. Flow of the detection procedure.

of sentence. We then train the training text using SRILM Toolkit with Good-Turing discounting method and 3-gram model to get our Language Model(LM).

Second, we use SRILM and LM to process a sample text segment to get the perplexity of it. In this step our program just read a text segment of certain size. In this way, we can control the size of the sample text segment.

Finally, we use a simple classifier to classify the sample text segment as generated text segment or normal text segment according the perplexity value. Figure 1 shows the flow of the detection procedure. The real line arrowhead represents the flow of data, while the dashed line arrowhead represents that nothing will be transferred or processed if the LM has already been stored.

5. Results and Discussion

In our experiment, text segments are split into three levels: sentence, 200bytes and 400bytes. We total tested 1351 sentences which have at least 8 words, 1018 200bytes and 509 400bytes text segments. The composing of the sample text set and the experiment result is present in Table 1. figure 2 to figure 3 shows comparison of the perplexity of different text size.

The accuracy is computed as follows:

$$Accuracy = \frac{SUC(G) + SUC(B)}{SAM(G) + SAM(B)} \quad (5)$$

Where SUC represents the number of success text segments, and SAM represents the number of sample text segments. G represents normal text, and B represents text that generated by NICETEXT system.

In addition, our algorithm is time efficient because time complexity is $O(n)$. Note that our classifier is simply designed: we use perplexity 100 to classify between normal text and stego-text. In practice, some better classifiers can be used to get better results. After all, the results of our research appear pretty good.

Table 1. Sample text set and detection results

| Size | Sen. | | 200B | | 400B | |
|----------|--------|-----|--------|-----|--------|-----|
| | G | B | G | B | G | B |
| Sample | 846 | 505 | 608 | 410 | 304 | 205 |
| Success | 785 | 376 | 585 | 401 | 302 | 205 |
| Failure | 61 | 129 | 23 | 9 | 2 | 0 |
| Accuracy | 85.93% | | 96.86% | | 99.61% | |

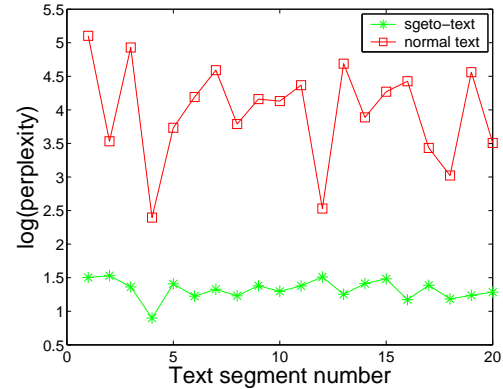


Figure 2. Comparison of the perplexity values of 20 normal text segments and 20 stego-text segments when text size is 400bytes .

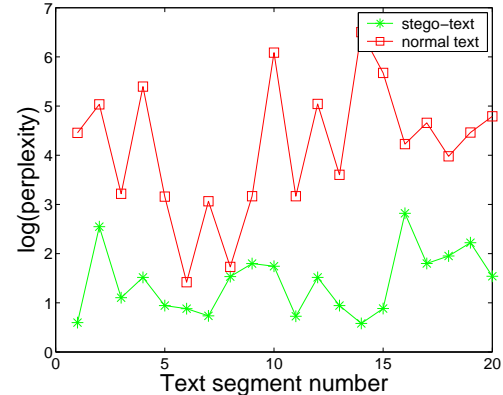


Figure 3. Comparison of the perplexity values of 20 normal sentences and 20 stego-text sentences.

6. Conclusion

In this paper, a linguistic steganography detection algorithm based on perplexity has been presented. We use the perplexity of given text segments to classify them into stego-text segments and normal text segments. The accuracy of our algorithm on discovering stego-text segments and normal text segments is found to be 99.61% when the segment size is not less than 400 bytes. Even for sentences, our algorithm detection accuracy exceeds 85.93%.

Our experiment has tested texts generated by NICETEXT system and normal texts. But it is easy to modify our method

to fit other linguistic steganography algorithms, the only changes is using different training data to built LMs.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (No. 60773032 & 60703071), the Ph.D. Program Foundation of Ministry of Education of China (No. 20060358014), the Natural Science Foundation of Jiangsu Province of China (No. BK2007060), and the Anhui Provincial Natural Science Foundation (No. 070411043).

References

- [1] K. Bennett, "Linguistic Steganography: Survey, Analysis, and Robustness Concerns for Hiding Information in Text," *Purdue University, CERIAS Tech. Report*, vol. 13, p. 2004, 2004.
- [2] M. Chapman and D. Davida, "Hiding the Hidden: A software system for concealing ciphertext as innocuous text," *LECTURE NOTES IN COMPUTER SCIENCE*, pp. 335–345, 1997.
- [3] F. Petitcolas, R. Anderson, and M. Kuhn, "Information hiding-a survey," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1062–1078, 1999.
- [4] Z. Chen, L. Huang, Z. Yu, W. Yang, L. Li, X. Zheng, and X. Zhao, "Linguistic Steganography Detection Using Statistical Characteristics of Correlations between Words," in *Information Hiding: 10th International Workshop, Ih 2008, Sana Barbara, CA, USA, May 19-21, 2008, Revised Selected Papers*, p. 224, Springer, 2008.
- [5] C. Zhi-li, H. Liu-sheng, Y. Zhen-shan, L. Ling-jun, and Y. Wei, "A Statistical Algorithm for Linguistic Steganography Detection Based on Distribution of Words," in *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on*, pp. 558–563, 2008.
- [6] Z. Chen, L. Huang, Z. Yu, X. Zhao, and X. Zhao, "Effective Linguistic Steganography Detection," in *Computer and Information Technology Workshops, 2008. CIT Workshops 2008. IEEE 8th International Conference on*, pp. 224–229, 2008.
- [7] C. Manning and H. Schütze, *Foundations of statistical natural language processing*. MIT Press, 1999.
- [8] S. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," in *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pp. 310–318, Association for Computational Linguistics Morristown, NJ, USA, 1996.
- [9] A. Stolcke, "SRILM-an Extensible Language Modeling Toolkit," in *Seventh International Conference on Spoken Language Processing, ISCA, 2002*.