



BUCKINGHAMSHIRE NEW UNIVERSITY

EST. 1891

Downloaded from: <https://bucks.repository.guildhe.ac.uk/>

This document is protected by copyright. It is published with permission and all rights are reserved.

Usage of any items from Buckinghamshire New University's institutional repository must follow the usage guidelines.

Any item and its associated metadata held in the institutional repository is subject to

Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0)

Please note that you must also do the following;

- the authors, title and full bibliographic details of the item are cited clearly when any part of the work is referred to verbally or in the written form
- a hyperlink/URL to the original Insight record of that item is included in any citations of the work
- the content is not changed in any way
- all files required for usage of the item are kept together with the main item file.

You may not

- sell any part of an item
- refer to any part of an item without citation
- amend any item or contextualise it in a way that will impugn the creator's reputation
- remove or alter the copyright statement on an item.

If you need further guidance contact the Research Enterprise and Development Unit
ResearchUnit@bnu.ac.uk

Detecting the Manipulation of Text Structure in Text Steganography using Machine Learning

Benjamin Aziz^a, Aysha Bukhelli^b

^a*School of Creative and Digital Industries
Buckinghamshire New University*

High Wycombe HP11 2JZ, United Kingdom

^b*Office of the Prime Minister, Bahrain*
benjamin.aziz@bucks.ac.uk

Keywords: Information Hiding, Lexical Steganography, Machine Learning, Text Steganography.

Abstract: We evaluate in this paper the security of a recent method proposed in literature for the embedding of hidden content in textual documents using paragraph size manipulation. Our steganalysis is based on machine learning, and the classification method we use for the analysis of a document utilises text attributes, such as words per paragraph, paragraph proportion based on sentences and other English document features. The embedding model showed to be resilient against the analysis techniques, where the highest plotted accuracy was 0.601, which is considered poor. The analysis methods were able to detect around half of the embedded corpus, which is equivalent to random guess. We concluded that it is difficult to detect an embedding model that manipulates paragraphs of novel texts, as the structure of these texts depend fully on the writer's style of writing. Thus by shifting the sentences up and down paragraphs without changing the order of the sentences and affecting the context of the text, it yields a reasonably secure method of embedding.


1 INTRODUCTION


The ancient technique of information hiding known as steganography has enjoyed much research in recent years due to the rising popularity of social media platforms and the abundant availability of online literature and other text as cover media for steganography. *Text steganography*, which refers to all techniques and methods used for hiding secret messages in textual documents (Agarwal, 2013; Lockwood and Curran, 2017; Taleby Ahvanooy et al., 2019; Kumar and Singh, 2020; Majeed et al., 2021), is the most difficult form of steganography due to the unavailability of redundant bits, compared with other file types, such as image, video and audio files. The structure of the text is identical to what is observed by the human eye, whereas the structure of other files, such as images and videos, is different from what has been observed. Thus, it is relatively easy to hide information in multimedia documents since no changes may be observed compared to pure textual media. In contrast, if slight changes are made to a text document, they can be easily detected by the human eye.

In contrast to text steganography, text steganalysis is the estimation process and science of identifying whether a given text document (file) has some hidden content and, if possible, extracting and recovering this hidden content. In practice, text steganalysis is a complicated task, because of the wide variety of digital text characteristics, the extensive variation of embedding approaches and usually, the low embedding distortion. In some cases, text steganalysis is possible due to the fact that data embedding modifies the statistics of the textual document. Hence, the existence of embedded content will render the original text different from its modified version.

According to (Taleby Ahvanooy et al., 2019), there are generally three methods for attacking text documents with hidden content: Visual attacks that involve a human in comparing two documents visually, structural attacks that involve modifying the structure of the suspected document hence destroying its embedded content, and finally, statistical attacks where the attacker uses statistical methods to estimate the probability that a document has some hidden content.

In this paper, we apply machine learning algorithms to analyse a new and interesting text embedding method recently proposed in (Aziz et al., 2022), to test the resilience of this method. Unlike other text embed-

^a  <https://orcid.org/0000-0001-5089-2025>

^b  <https://orcid.org/0000-0001-7578-977X>

ding approaches, the method proposed in (Aziz et al., 2022) relies on the manipulation of paragraph sizes as a way of embedding 0s and 1s in the document. We were not able to find any existing attack methods that can specifically tackle this type of document manipulation. There are no current data in literature regarding statistics of paragraph breaks, nor data regarding the features to create such statistics. Thus, we created an algorithm to extract features of English language text documents, for example, words per paragraph and sentences per paragraph. Using such features, we integrated them with machine learning tools, in order to gain understanding of the resilience of the embedding method proposed in (Aziz et al., 2022). In effect, our analysis implements a model of the *passive warden*, who is unable to do nothing but spy on the communication channel between the communicating prisoners (Simmons, 1984). It has the limited capability to only detect the existence of a secret message, without giving any information about the secret message itself.

The rest of the paper is organised as follows. In Section 2, we briefly discuss related state-of-the art literature. In Section 3, we give an overview of the text steganography model introduced in (Aziz et al., 2022) as a background to our analysis. In Section 4, we discuss our analysis methodology, including giving a description of the dataset used in our analysis. In Section 5, we discuss the outcomes of our analysis, which used some of the most popular machine learning classification algorithms. Finally, in Section 6, we conclude the paper giving directions for future work.

2 RELATED WORK

We review here a few works in literature we consider of direct relevance to our proposed analysis in this paper. We focus on works that have adopted a machine learning approach in attacking textual steganography embedding methods.

We start with Taskiran et al. (Taskiran et al., 2006), who partitioned 40,000 sentences in their data set into two sets: a 30,000 sentence set, on which they trained out 8 language models by alternating the values for the parameters listed, and a 10,000 sentence set, on which they performed their classification experiments. For each of the 8 language models they extracted a feature vector for each sentence in their training set using features such as number of words, out of vocabulary words, zero probability words and minimum n-gram context length matching the model. They inserted a 781-bit long plain text message into the text consisting of 10,000 sentences using the Tyrannosaurus Lex system. This resulted in 1169 stegano-

graphically modified sentences, and 8831 unmodified sentences. They then extracted feature vectors from each of these 10,000 sentences using the features described beforehand. A Support Vector Machine (SVM) (Cortes and Vapnik, 1995) classifier was trained on the features of randomly selected 500 unmodified sentences and 500 steganographically modified sentences. They used the *libsvm* library to train the SVM classifier. Using the trained SVM classifier, they classified the remaining 669 steganographically modified and 8331 unmodified sentences. The accuracy on steganographically modified sentences was found to be as high as 84.9% and that for unmodified sentences to be 38.6%. The classification of each sentence was performed independently. In practice whole paragraphs would be steganographically modified, so they believed that classification on multiple consecutive sentences picked from a text will give much better results. This is easily achieved by using the output of the SVM classifier to train a second SVM classifier for the whole text, rather than on a sentence by sentence basis.

Another notable work is that of Zhi-Li et al. (Zhi-Li et al., 2008), who were able to detect three different linguistic steganography embedding methods: NICETEXT (Chapman and Davida, 1997), TEXTO (Maher, 1995) and the Markov-Chain-Based method of (Wu et al., 2019). The total accuracy of discovering embedded content segments and normal text segments were found to be high reaching 87.39% 95.51%, 98.50%, 99.15% and 99.57%, respectively, for the segment sizes 5kB, 10kB, 20kB, 30kB and 40kB of text.

Zhao (Zhao et al., 2009) proposed a steganalysis technique using the Support Vector Machine (SVM)-based hidden information detection algorithm. The SVM Classifier was built by training and testing the machine using normal text and a small sample of embedded text using a certain steganalysis technique. The better generalisation ability of the classifier is used to classify the unknown text to either clean or embedded text. The model has great generalisation capabilities and the SVM classifier also has an excellent classification effect (Zhao et al., 2009).

Wen et al. (Wen et al., 2019) proposed a novel universal text steganalysis model based on the Convolutional Neural Network (CNN) framework (LeCun et al., 1998), which is able to capture complex dependencies and learn feature representations, automatically from texts. Unlike CNN methods used in image steganalysis, the word embedding layer is utilised to extract the semantic and syntax features of words, and the sentence features are learnt through a convolutional layer with rectangular kernels of different sizes.

The authors presented a decision strategy to improve the performance of long texts.

Yang et al. (Yang et al., 2019) proposed a technique using recurrent neural networks (Rumelhart et al., 1986) to extract feature distribution differences of conditional probability distribution of each word in the automatically generated steganographic texts that are distorted after being embedded with hidden information. Yang then classifies these features into cover texts and embedded texts. Experimental results showed that the model achieved high detection accuracy of between 70% and 90% (Yang et al., 2019).

Wu et al. (Wu et al., 2021) presented a linguistic steganalysis method using graph neural networks (Zhou et al., 2020). In the proposed method, texts are translated as directed graphs with the associated information, where nodes denote words and edges show associations between those words. By training a graph network for feature extraction, each node can then collect contextual information to update self-expression. As a result, solving effectively the problem of poor representation of polysemous words.

Finally, Xiang et al. (Xiang et al., 2020) proposed a linguistic steganalysis method based on two-level cascaded convolutional neural networks to improve the ability to detect embedded texts, which use synonym substitutions. The first-level network, sentence-level, consists of one convolutional layer with multiple convolutional kernels in different window sizes, one pooling layer to deal with variable sentence lengths, and one fully connected layer with dropout as well as a soft-max output, such that two final steganographic features are obtained for each sentence. The unmodified and modified sentences, along with their words, are represented in the form of pre-trained dense word embedding, which serve as the input of the network. Sentence-level networks provide the representation of a sentence, and thus are able to be utilised to predict whether a sentence is unmodified or modified by synonym substitutions. In the second level, a text-level network exploits the predicted representations of sentences obtained from the sentence-level CNN to determine whether the detected text is embedded or clean text. The method achieved an average accuracy of 82.245%. Although, the accuracy of the method was high, the authors had to use more than 2 million samples per set, making it a total of above 4 million texts, which requires high computational processing power.

One can conclude from the above works (and other text steganalysis methods) that the *structure of a paragraph* is not considered to be an important feature that steganalysis users would be looking for or considering when analysing documents for possible hidden content, making it one of the least common and hence, de-

tectable document manipulation methods. None of the above studies mentioned the concept of *erroneously structured paragraphs*, in the sense that some paragraphs may be too short or too long. As a result, we consider the target property (paragraph size manipulation) of the study proposed in this paper to be a pioneering effort in the field of text steganalysis.

3 BACKGROUND

Recently, in (Aziz et al., 2022), the authors defined a new method for embedding secret messages in textual documents based on the changing of the sizes of paragraphs and the comparison of the sizes of subsequent paragraphs. We give a brief summary of this method here for the purpose of introduction. For more details, we refer the reader to the original method proposed in (Aziz et al., 2022).

The method assumes that a text document (or file) contains a finite number of paragraphs. Each of these paragraphs are defined as a list of multiple sentences separated from other list of sentences by a newline character. And that each sentence in the paragraph contains a number of characters. Thus the size of a paragraph is its *total number of characters combined*. The comparison of each paragraph pair size will represent a bit, 1 or 0. The manipulation method of the paragraph size is by shifting either a sentence from the top of a paragraph to the bottom of the previous paragraph, or by shifting a sentence from the bottom of a paragraph to the top of its subsequent paragraph. Thus leaving the order of sentences untouched. This method is context free, however the sentence shifting is static, to try and preserve the array of sentences without compromising the context. The extraction process is simple; it runs consecutively through the number of paragraphs by comparing the size of the 1st to that of the 2nd and then 2nd to 3rd and so on. This produces $n - 1$ number of bits, where n is the number of paragraphs in a text file. The comparison assigns each time a 0 or a 1 depending on whether one paragraph is larger or smaller than its successor.

As a simple example, let us consider the excerpt in Figure 1 taken from Charles Dickens's "Oliver Twist", where we have assumed that if the size of paragraph i is larger than the size of paragraph $i + 1$, then this represents a 1 bit, otherwise, it represents a 0 bit. Hence, the excerpt in Figure 1 represents the message [1, 0]. However, if we were to alter the excerpt to that of Figure 2, then this would represent a new (secret) message representing [0, 1].

P1: "Bow to the Board," said Bumble. Oliver brushed away two or three tears that were lingering in his eyes, and seeing no board but the table, fortunately bowed to that.

P2: "What's your name, boy?" said the gentleman in the high chair.

P3: Oliver was frightened at the sight of so many gentlemen, which made him tremble; and the beadle gave him another tap behind, which made him cry; and these two causes made him answer in a very low and hesitating voice; whereupon a gentleman in a white waistcoat said he was a fool. Which was a capital way of raising his spirits, and putting him quite at ease.

Fig. 1: A three-paragraph excerpt from Charles Dickens's "Oliver Twist" (taken from (Aziz et al., 2022)).

P1: "Bow to the Board," said Bumble. Oliver brushed away two or three tears that were lingering in his eyes, and seeing no board but the table, fortunately bowed to that.

P2: "What's your name, boy?" said the gentleman in the high chair.

Oliver was frightened at the sight of so many gentlemen, which made him tremble; and the beadle gave him another tap behind, which made him cry; and these two causes made him answer in a very low and hesitating voice; whereupon a gentleman in a white waistcoat said he was a fool.

P3: Which was a capital way of raising his spirits, and putting him quite at ease.

Fig. 2: The same three-paragraph excerpt from Charles Dickens's "Oliver Twist" but with a different paragraph layout (taken from (Aziz et al., 2022)).

4 METHODOLOGY

In this section, we outline the methodology used in our analysis. The methodology consists of the following steps: Identifying the features of an English language document, defining the cover text selection method, populating the dataset used in the analysis and finally, identifying the machine learning algorithms to be used in the analysis. We expand on these steps in the following sections.

4.1 Features of English Documents

The general understanding of English text is that a document contains characters which create words used to formulate sentences. These sentences are then structured into paragraphs, and paragraphs are grouped together with newline character breaks to create a document. This document also has grammatical and linguistic features, such as verbs, nouns and adjectives. In English text, the letter 'e' has the highest frequency of occurrence (Ridley et al., 1999). Thus its distribution in a text holds some significance in analysing clean and embedded pieces of text.

We have identified the following features, which will be used by the machine learning algorithms to differentiate between clean and embedded documents:

1. Total number of sentences in the file
2. Total number of words in the file
3. Total number of characters in the file
4. Total number of special word or letter in the file

5. Average number of sentences per paragraph
6. Average number of words per paragraph
7. Average number of characters per paragraph
8. Distribution of sentences in a file
9. Distribution of words in a file
10. Distribution of characters in a file
11. Distribution of special word or letter in a file
12. Average number of verbs, nouns, and adjectives per paragraph
13. Number of paragraphs starting with a noun, verb, and adjective
14. Number of paragraphs ending with a noun, verb, and adjective

These 14 features have a frequency of distribution in a document and probability of occurrence, which the machine learning algorithms rely on.

4.2 Cover Text Selection

The cover texts we used were downloaded from Project Gutenberg eBook Repository (Hart, 1971). The selection process was created to ensure the following characteristics were met in all cover the files:

- Texts must be of UTF-8 encoding and *.txt* format.
- All titles and chapter headings are removed manually, including any commentary.
- Larger files were preferred to smaller files as they contained bigger paragraphs. Thus allowing for easier manipulation.

- Preferred genre is fiction and non-fiction novels.
- Poems are excluded as they do not meet the criteria of punctuation marks and newline characters, which separate paragraphs, and are not therefore easily manipulated.

All cover texts downloaded had to be slimmed down to three separate sizes: 5 paragraph documents, 10 paragraph documents and 15 paragraph documents. Additional paragraphs from each category were removed. Moreover, selecting cover texts by multiple authors ensured good diversity of writing styles and techniques, and consequently, good coverage in terms of the features of the textual document.

4.3 Dataset

After categorising the cover texts into the above three size sets, the files are then equally separated into two types: One type used as our controlled element of the clean text (i.e. cover text), and the other type containing files embedded with secret messages using the method of (Aziz et al., 2022). The secret messages were random sequences of 0s and 1s, of lengths $n - 1$, hence 4, 9 and 14 bits long.

Table 1 shows the total number of files, for each dataset, divided roughly equally between clean and embedded cases.

| Number of Paragraphs | Number of Clean Files | Number of Embedded Files |
|----------------------|-----------------------|--------------------------|
| 5 | 1089 | 1089 |
| 10 | 788 | 787 |
| 15 | 754 | 753 |

Table 1: Sizes of datasets for the three cases.

We created a notebook in Google Colab Notebook, using SKlearn environment, where we uploaded our clean and embedded datasets for analysis, and programmed a machine learning simulation to test and train the nine machine learning algorithms to figure out which algorithm was best in detecting the embedding method. The standard split of 20%-80% was used for training and testing. The algorithms used the features identified in Section 4.1 to attempt to classify a document as ‘clean’ or ‘embedded’.

4.4 Machine Learning Algorithms

We chose the following machine learning classification algorithms to help us analyse the features extracted from both clean and embedded datasets.

- The k Nearest Neighbour (k -NN) (Mucherino et al., 2009)
- Linear Support Vector Machine (LSVM) (Yang et al., 2015)
- Radial Basis Function Support Vector Machine (RBF SVM) (Cao et al., 2008)
- Decision Tree (DT) (Swain and Hauska, 1977)
- Random Forest (RF) (Parmar et al., 2019)
- Neural Network (NN), (Rumelhart et al., 1986), using the multi-layer perceptron classifier (thought to have been originally discovered by (Rosenblatt, 1958))
- Adaptive Boosting (AdaBoost) (Schapire, 2013)
- Naïve Bayes (NB) (Bayes, 1763)
- Quadratic Discriminant Analysis (QDA) (Tharwat, 2016)

Typical performance indicators were used, such as accuracy, precision, recall and the F-measure (Chinchor, 1992), calculated as follows:

$$Accuracy = \frac{\# \text{ of correct predictions } (TP+TN)}{\# \text{ of predictions } (TP+TN+FP+FN)}$$

$$Precision = \frac{TP}{TP+FP}$$

$$Recall = \frac{TP}{TP+FN}$$

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Such that True Positive (TP) is a positive instance classified correctly as positive, True Negative (TN) is a negative instance classified correctly as negative, False Positive (FP) is a negative instance classified wrongly as positive and False Negative (FN) is positive instance classified wrongly as negative.

5 RESULTS AND ANALYSIS

We present here the results of our machine learning-based analysis for each of the three cases, 5, 10 and 15 paragraph documents, embedded with secret messages using the method by (Aziz et al., 2022). Table 2 shows the accuracy levels for the various classification algorithms used versus each of the document sizes.

The results revealed that the QDA algorithm was most accurate in predicting documents with embedded content for the cases of 5 and 10 paragraph documents (at 0.553 and 0.584, respectively), whereas LSVM was best in predicting the 15 paragraph documents (at 0.555). The accuracy values for all these

| Algorithm | 5 Paragraph | 10 Paragraph | 15 Paragraph |
|--------------|--------------|--------------|--------------|
| <i>k</i> -NN | 0.485 | 0.512 | 0.537 |
| LSVM | 0.546 | 0.525 | 0.555 |
| RBFSVM | 0.520 | 0.473 | 0.462 |
| DT | 0.527 | 0.536 | 0.553 |
| RF | 0.529 | 0.561 | 0.529 |
| NN | 0.532 | 0.526 | 0.484 |
| AdaBoost | 0.525 | 0.555 | 0.524 |
| NB | 0.529 | 0.536 | 0.537 |
| QDA | 0.553 | 0.584 | 0.527 |

Table 2: Accuracy rates for the different algorithms.

| Algorithm | 5 Paragraph | 10 Paragraph | 15 Paragraph |
|--------------|--------------|--------------|--------------|
| <i>k</i> -NN | 0.484 | 0.514 | 0.536 |
| LSVM | 0.542 | 0.532 | 0.555 |
| RBFSVM | 0.340 | 0.322 | 0.316 |
| DT | 0.521 | 0.540 | 0.545 |
| RF | 0.541 | 0.541 | 0.528 |
| NN | 0.514 | 0.547 | 0.474 |
| AdaBoost | 0.555 | 0.555 | 0.524 |
| NB | 0.522 | 0.526 | 0.490 |
| QDA | 0.549 | 0.580 | 0.518 |

Table 5: F_1 rates for the different algorithms.

cases are low representing medium performance only. The values of the remaining performance indicators are shown in Tables 3, 4 and 5.

| Algorithm | 5 Paragraph | 10 Paragraph | 15 Paragraph |
|--------------|--------------|--------------|--------------|
| <i>k</i> -NN | 0.484 | 0.514 | 0.536 |
| LSVM | 0.545 | 0.533 | 0.562 |
| RBFSVM | 0.260 | 0.237 | 0.231 |
| DT | 0.528 | 0.540 | 0.548 |
| RF | 0.545 | 0.541 | 0.535 |
| NN | 0.516 | 0.567 | 0.551 |
| AdaBoost | 0.555 | 0.555 | 0.525 |
| NB | 0.526 | 0.532 | 0.524 |
| QDA | 0.553 | 0.583 | 0.520 |

Table 3: Precision rates for the different algorithms.

| Algorithm | 5 Paragraph | 10 Paragraph | 15 Paragraph |
|--------------|-------------|--------------|--------------|
| <i>k</i> -NN | 0.0008 | 0.0009 | 0.0047 |
| LSVM | 1.0000 | 1.0000 | 1.0000 |
| RBFSVM | 0.1094 | 0.0975 | 0.1306 |
| DT | 0.0021 | 0.0036 | 0.0044 |
| RF | 0.0028 | 0.0051 | 0.0095 |
| NN | 0.1729 | 0.34845 | 0.2496 |
| AdaBoost | 0.0414 | 0.06384 | 0.0951 |
| NB | 0.0004 | 0.0006 | 0.0011 |
| QDA | 0.0008 | 0.00129 | 0.0021 |

Table 6: Training times for the different classification algorithms (seconds).

| Algorithm | 5 Paragraph | 10 Paragraph | 15 Paragraph |
|--------------|--------------|--------------|--------------|
| <i>k</i> -NN | 0.484 | 0.514 | 0.537 |
| LSVM | 0.540 | 0.531 | 0.561 |
| RBFSVM | 0.500 | 0.500 | 0.500 |
| DT | 0.514 | 0.540 | 0.547 |
| RF | 0.537 | 0.541 | 0.535 |
| NN | 0.513 | 0.529 | 0.526 |
| AdaBoost | 0.520 | 0.555 | 0.525 |
| NB | 0.519 | 0.521 | 0.517 |
| QDA | 0.546 | 0.577 | 0.520 |

Table 4: Recall rates for the different algorithms.

| Algorithm | 5 Paragraph | 10 Paragraph | 15 Paragraph |
|--------------|-------------|--------------|--------------|
| <i>k</i> -NN | 0.5373 | 0.5173 | 0.4838 |
| LSVM | 0.4162 | 0.4031 | 0.4269 |
| RBFSVM | 1.0000 | 1.0000 | 1.0000 |
| DT | 0.0080 | 0.0141 | 0.0134 |
| RF | 0.0168 | 0.0330 | 0.0421 |
| NN | 0.0202 | 0.0256 | 0.0280 |
| AdaBoost | 0.0808 | 0.1344 | 0.1509 |
| NB | 0.0079 | 0.0139 | 0.0149 |
| QDA | 0.0126 | 0.0220 | 0.0228 |

Table 7: Testing times for the different classification algorithms (seconds).

Our analysis also calculated each of the classification algorithms' training and testing times, as shown in Tables 6 and 7, respectively. In general, LSVM proved to be the slowest during the training phase, whereas RBFSVM was the slowest during the testing phase.

We also plotted the ROC (Receiver Operating Characteristic) curves for the 5, 10 and 15 paragraph

TP and FP rates, are shown in Figures 3, 4 and 5, respectively, with the Area Under ROC (AUROC) curve values also shown on the plots, for all the classification algorithms.

For all of these cases, we found that the classifiers remained close to the linear line (i.e. when AUROC=0.5), and hence did not outperform by any useful margins the *random guess* case. For some classifiers, e.g. *k*-NN with the 5 paragraph documents and RBF SVM with the 10 paragraph documents, the classifier's performance actually was worse than the random guess line. On the other hand, QDA performed the best for the 5 and 10 paragraph cases, slightly outperforming the random guess case (AUROC=0.568

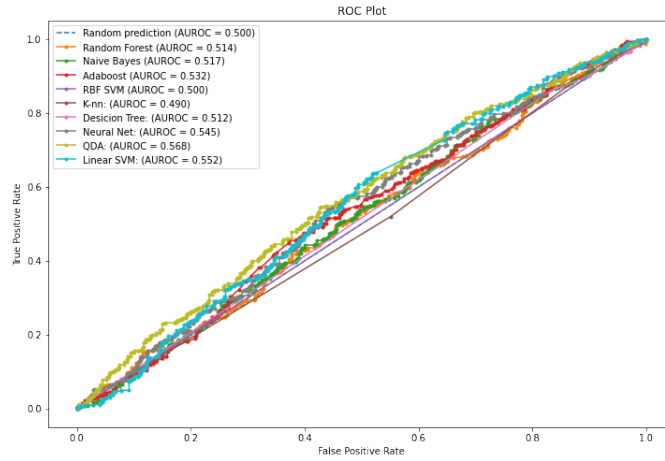


Fig. 3: 5 paragraph AUROC graph.

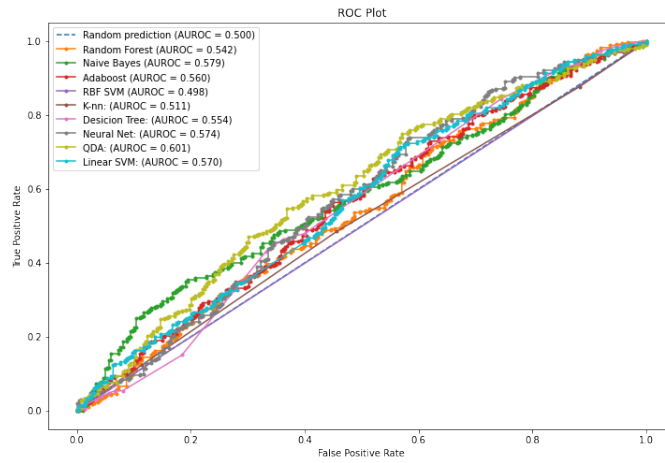


Fig. 4: 10 paragraph AUROC graph.

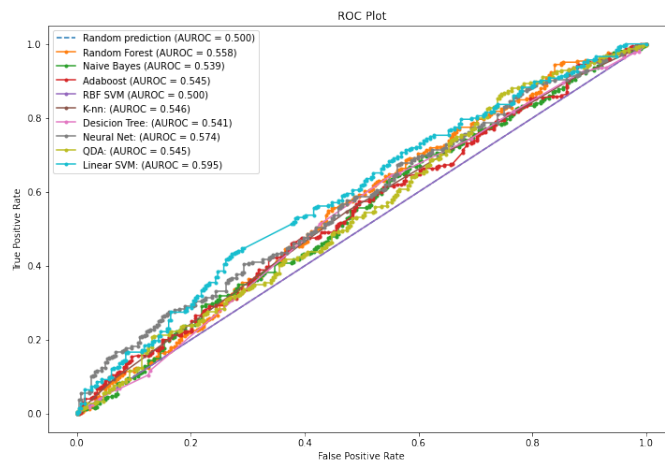


Fig. 5: 15 paragraph AUROC graph.

and AUROC=0.601, respectively), whereas LSVM was best in the case of the 15 paragraph documents (AUROC=0.595).

6 CONCLUSION

We presented in this paper an steganalysis attack based on nine popular machine learning algorithms of the newly proposed text embedding method of Aziz et al. (Aziz et al., 2022). We found that the accuracy values of detecting embedded content using our machine learning classifiers was surprisingly low (0.553, 0.584 and 0.555), generally resulting in a ROC curve close to random guess. This initial study indicates that the new method proposed in (Aziz et al., 2022) so far can withstand standard machine learning-based attacks.

We plan to continue applying other attack methods, in particular, other statistical attack methods such as the χ^2 attack (Pearson, 1900; Plackett, 1983), to determine if the method can still withstand security attacks. We also plan to test the embedding capacity of the (Aziz et al., 2022) to determine its efficiency in embedding large sized hidden content, and whether size is a factor in undermining the security of this embedding method.

Bibliography

- [Agarwal, 2013]Agarwal, M. (2013). Text steganographic approaches: A comparison. *International Journal of Network Security and its Applications*, 5:91–106.
- [Aziz et al., 2022]Aziz, B., Bukhelli, A., Khusainov, R., and Mohasseb, A. (2022). A novel method for embedding and extracting secret messages in textual documents based on paragraph resizing. In *19th International Conference on Security and Cryptography*. SciTePress.
- [Bayes, 1763]Bayes, T. (1763). Lii. an essay towards solving a problem in the doctrine of chances. by the late rev. mr. bayes, frs communicated by mr. price, in a letter to john canton, amfr s. *Philosophical transactions of the Royal Society of London*, 53:370–418.
- [Cao et al., 2008]Cao, H., Naito, T., and Ninomiya, Y. (2008). Approximate rbf kernel svm and its applications in pedestrian classification. In *The 1st International Workshop on Machine Learning for Vision-based Motion Analysis-MLVMA'08*.
- [Chapman and Davida, 1997]Chapman, M. and Davida, G. (1997). Hiding the hidden: A software system for concealing ciphertext as innocuous text. In *International Conference on Information and Communications Security*, pages 335–345. Springer.
- [Chinchor, 1992]Chinchor, N. (1992). Muc-4 evaluation metrics. In *Proceedings of the 4th Conference on Message Understanding, MUC4 '92*, pages 22–29, Stroudsburg, PA, USA. Association for Computational Linguistics.
- [Cortes and Vapnik, 1995]Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20:273–297.
- [Hart, 1971]Hart, M. (1971). Free ebooks - Project Gutenberg.
- [Kumar and Singh, 2020]Kumar, R. and Singh, H. (2020). Recent trends in text steganography with experimental study. In *Handbook of Computer Networks and Cyber Security*, pages 849–872. Springer.
- [LeCun et al., 1998]LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.
- [Lockwood and Curran, 2017]Lockwood, R. and Curran, K. (2017). Text based steganography. *International Journal of Information Privacy, Security and Integrity*, 3(2):134–153.
- [Maher, 1995]Maher, K. (1995). Texto. URL: <ftp://ftp.funet.fi/pub/crypt/steganography/texto.tar.gz>.
- [Majeed et al., 2021]Majeed, M. A., Sulaiman, R., Shukur, Z., and Hasan, M. K. (2021). A review on text steganography techniques. *Mathematics*, 9(21).
- [Mucherino et al., 2009]Mucherino, A., Papajorgji, P. J., and Pardalos, P. M. (2009). K-nearest neighbor classification. In *Data mining in agriculture*, pages 83–106. Springer.
- [Parmar et al., 2019]Parmar, A., Katariya, R., and Patel, V. (2019). A Review on Random Forest: An Ensemble Classifier. In Hemanth, J., Fernando, X., Lafata, P., and Baig, Z., editors, *International Conference on Intelligent Data Communication Technologies and Internet of Things (ICICI) 2018*, pages 758–763, Cham. Springer International Publishing.
- [Pearson, 1900]Pearson, K. (1900). X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 50(302):157–175.
- [Plackett, 1983]Plackett, R. L. (1983). Karl Pearson and the Chi-Squared Test. *International Statistical Review / Revue Internationale de Statistique*, 51(1):59–72.
- [Ridley et al., 1999]Ridley, D. R., Dominguez, P. S., and Walker, C. B. (1999). English letter frequencies in transcribed speech versus written samples. *Perceptual and Motor Skills*, 88(3 part 2):1181–1188.
- [Rosenblatt, 1958]Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.
- [Rumelhart et al., 1986]Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, 323(6088):533–536.
- [Schapire, 2013]Schapire, R. E. (2013). Explaining adaboost. In *Empirical inference*, pages 37–52. Springer.
- [Simmons, 1984]Simmons, G. J. (1984). The prisoners' problem and the subliminal channel. In *Advances in Cryptology*, pages 51–67. Springer.
- [Swain and Hauska, 1977]Swain, P. H. and Hauska, H. (1977). The decision tree classifier: Design and potential. *IEEE Transactions on Geoscience Electronics*, 15(3):142–147.
- [Taleby Ahvanooy et al., 2019]Taleby Ahvanooy, M., Li, Q., Hou, J., Rajput, A. R., and Chen, Y. (2019). Modern text hiding, text steganalysis, and applications: A comparative analysis. *Entropy*, 21(4).
- [Taskiran et al., 2006]Taskiran, C., Topkara, U., Topkara, M., and Delp, E. (2006). Attacks on lexical natural language steganography systems. *Proceedings of SPIE - The International Society for Optical Engineering*.
- [Tharwat, 2016]Tharwat, A. (2016). Linear vs. quadratic discriminant analysis classifier: a tutorial. *International Journal of Applied Pattern Recognition*, 3(2):145–180.
- [Wen et al., 2019]Wen, J., Zhou, X., Zhong, P., and Xue, Y. (2019). Convolutional neural network based text steganalysis. *IEEE Signal Processing Letters*, 26(3):460–464.
- [Wu et al., 2021]Wu, H., Yi, B., Ding, F., Feng, G., and Zhang, X. (2021). Linguistic steganalysis with graph neural

- networks. *IEEE Signal Processing Letters*, 28:558–562.
- [Wu et al., 2019]Wu, N., Shang, P., Fan, J., Yang, Z., Ma, W., and Liu, Z. (2019). Coverless text steganography based on maximum variable bit embedding rules. In *Journal of Physics: Conference Series*, volume 1237:2, page 022078. IOP Publishing.
- [Xiang et al., 2020]Xiang, L., Guo, G., Yu, J., Sheng, V. S., and Yang, P. (2020). A convolutional neural network-based linguistic steganalysis for synonym substitution steganography. *Mathematical Biosciences and Engineering*, 17(2):1041–1058.
- [Yang et al., 2015]Yang, Y., Li, J., and Yang, Y. (2015). The research of the fast SVM classifier method. In *2015 12th international computer conference on wavelet active media technology and information processing (IC-CWAMTIP)*, pages 121–124. IEEE.
- [Yang et al., 2019]Yang, Z., Wang, K., Li, J., Huang, Y., and Zhang, Y.-J. (2019). Ts-rnn: Text steganalysis based on recurrent neural networks. *IEEE Signal Processing Letters*, 26(12):1743–1747.
- [Zhao et al., 2009]Zhao, X., Huang, L., Li, L., Yang, W., Chen, Z., and Yu, Z. (2009). Steganalysis on character substitution using support vector machine. In *2009 Second International Workshop on Knowledge Discovery and Data Mining*, pages 84–88.
- [Zhi-Li et al., 2008]Zhi-Li, C., Liu-Sheng, H., Zhen-Shan, Y., Ling-Jun, L., and Wei, Y. (2008). A statistical algorithm for linguistic steganography detection based on distribution of words. In *ARES 2008 - 3rd International Conference on Availability, Security, and Reliability, Proceedings*, pages 558–563.
- [Zhou et al., 2020]Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. (2020). Graph neural networks: A review of methods and applications. *AI open*, 1:57–81.