Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2023.025082

check for updates

ARTICLE

Topic Controlled Steganography via Graph-to-Text Generation

Bowen Sun[1], Yamin Li[1,2,3,*], Jun Zhang[1], Honghong Xu[1], Xiaoqiang Ma[4] and Ping Xia[2,3,5]

[1]School of Computer Science and Information Engineering, Hubei University, Wuhan, 430062, China

[2]Hubei Key Laboratory of Intelligent Vision Based Monitoring for Hydroelectric Engineering, China Three Gorges University, Yichang, 443002, China

[3]Yichang Key Laboratory of Intelligent Vision Based Monitoring for Hydroelectric Engineering, China Three Gorges University, Yichang, 443002, China

[4]Department of CSIS, Douglas College, New Westminster, BC, V3L 5B2, Canada

[5]College of Computer and Information Technology, China Three Gorges University, Yichang, 443002, China

*Corresponding Author: Yamin Li. Email: yamin.li@hubu.edu.cn

Received: 21 June 2022 Accepted: 06 September 2022

ABSTRACT

Generation-based linguistic steganography is a popular research area of information hiding. The text generative steganographic method based on conditional probability coding is the direction that researchers have recently paid attention to. However, in the course of our experiment, we found that the secret information hiding in the text tends to destroy the statistical distribution characteristics of the original text, which indicates that this method has the problem of the obvious reduction of text quality when the embedding rate increases, and that the topic of generated texts is uncontrollable, so there is still room for improvement in concealment. In this paper, we propose a topic-controlled steganography method which is guided by graph-to-text generation. The proposed model can automatically generate steganographic texts carrying secret messages from knowledge graphs, and the topic of the generated texts is controllable. We also provide a graph path coding method with corresponding detailed algorithms for graph-to-text generation. Different from traditional linguistic steganography methods, we encode the secret information during graph path coding rather than using conditional probability. We test our method in different aspects and compare it with other text generative steganographic methods. The experimental results show that the model proposed in this paper can effectively improve the quality of the generated text and significantly improve the concealment of steganographic text.

KEYWORDS

Information hiding; linguistic steganography; knowledge graph; topic controlled; text generation

1 Introduction

With the development of information technology, human society has entered the era of big data. While the new round of technological revolution has brought convenience to our work and life, many security issues have become increasingly apparent. In recent years, there have been frequent threats to existing information network structure security, data security, and information content security,

especially in banking, transportation, commerce, medical care, communications, electricity, etc., which are highly dependent on informatization.

At present, traditional information security is mainly realized by encryption technology and its application system. Encryption Techniques encode secret information into an incomprehensible form, which can ensure content security to a certain extent. However, with the development of data mining and machine learning technologies in the era of big data, encrypted data, as a kind of abnormal data, tends to make it easier to expose the existence of secret information, which makes it become a key goal of network data analysis. Information Hiding, also known as Steganography, is another key technology in the field of information security. It has a long history and is widely used in applications such as military intelligence secret communications, user privacy protection, and digital media copyright protection [1–4]. Compared with traditional encryption technology, steganography embeds the secret message in the public information carrier to complete the communication, so as not to attract the attention of the monitors, reduce the possibility of being attacked and detected, and effectively ensure the security of information.

In today's era of big data, various digital media such as images, audios, and texts have become important ways for people to transmit information. They are also ideal carriers for technical researchers to hide information [5–7]. Many researchers have studied steganography in image, audio and other fields and published lots of related steganography [8,9] and Steganalysis [10] models. Text is the most widely used information carrier in our daily life, and the steganography method that uses text as an information hiding carrier has attracted the attention of many researchers. However, compared with images and audio, text has less redundant information, so it is quite challenging to hide the information inside. At present, there have been a large number of in-depth studies on the text-based steganography, which can be divided into text retrieval [11,12], text modification [13–15] and text generation [16–18] three types of methods. The text retrieval steganography method expresses different meanings by selecting different text characters. The steganography method of text modification is realized by making minor modifications to the text, such as adjusting the word spacing and replacing synonyms. However, the secret information embedding rate of these methods is very limited, which is not practical in actual application scenarios. Therefore, researchers began to try to use automatic text generation models to realize the reversible embedding of secret information. Since it can generate natural language text close to human communication and is not restricted by the text format, the generation-based linguistic steganography method has become a popular research direction in the field of information hiding.

The problem to be solved by information hiding technology can be summarized by a classic model of "The Prisoners' Problem" [19]: Alice and Bob are separated in prison, and they need to complete the transmission of some secret information without being discovered by the guard Eve. Therefore, Alice and Bob need to hide the secret information in some kind of carrier. In this paper whose task is to generate steganographic text guided by the knowledge graph, Alice uses the knowledge graph to hide the secret information in the steganographic text embedded with the secret information. The mathematical description of this steganography task is: given a specific semantic subgraph $g(g \in G)$ in the knowledge graph space G and a secret message to be hidden $m(m \in M)$ in the secret message space M, the task goal is to generate steganographic text $s(s \in S)$, S is the steganographic text space, and ensure that: a) steganographic text is a smooth natural language paragraph; b) expresses specific semantics; c) hidden secret messages can be extracted correctly. Therefore, the steganographic task can be expressed as

$$Emb : M \times K \times G \rightarrow S, f(m, k_a, g) = s \tag{1}$$

$$Ext : S \times K \rightarrow M, f(s, k_b) = m \tag{2}$$

Among them, *Emb* is the process of embedding secret information, *Ext* is the process of extracting secret information, k_a and k_b are two keys in the key space K, f and g are the corresponding mapping functions.

The current text generation-based steganography method mainly relies on the Statistical Language Model in Natural Language Processing (NLP) technology, and it contains two parts: a) the text generation part that based on statistical language models; b) the encoding part based on conditional probability distribution. The text generation part uses a well-designed model to learn a language statistical distribution model from a large number of natural texts. Then during the process of text generation, the conditional probability distribution of each word will be coded according to the secret information. As many steganalysis methods keep developing, the concealment of the steganographic text needs to be improved, and the text generative steganography method based on the current method framework faces severe challenges.

First, it is important to design a model that can generate texts with high quality. Since this technical framework needs to encode the conditional probability of the generated text to embed information, as the secret information embedding rate increases, the quality of the generated text will decrease significantly, and even generate meaningless or grammatically wrong sentences. It means this method has the inherent defect that it cannot simultaneously increase the secret information hiding capacity and improve the concealment of the algorithm. Second, it is also necessary to improve the relevance and coherence of steganographic text in content, so that the text content can have a certain theme and complete semantics, thereby improving the concealment of the generation-based steganography method. Therefore, the generation-based steganography method should not only consider the similarity of the probability distribution between text characters to make the generated text more natural and smoother, but also ensure that the text content has a certain theme, complete semantics, and consistent emotions, which are closer to human communication.

In order to further improve the concealment of the text generative steganography method, and in response to the above problems and challenges, we attempt to break through the current generation-based steganography method framework. So that we can overcome the inherent defects of the text generative steganographic methods based on conditional probability coding. The main contributions of this paper can be summarized as the following three points: a) a topic controlled steganography method is proposed, which can automatically generate steganographic texts that carry secret information; b) we use knowledge graph to guide the text generation, combining with our topic matching method, we make the topic of the generated texts controllable. c) we get better quality of generated texts by encoding the secret information during graph path coding rather than using conditional probability, and the method of graph path coding is provided with detailed algorithms.

2 Related Works

The text generative steganography method has high concealment and security, which has attracted wide attention from researchers. It has long been the focus of information security and research hotspots in the field of information hiding. In early days, researchers tried some information hiding methods based on text generation, but the generated sentences did not have complete semantics and contained many grammatical errors [20]. Subsequently, the researchers tried to introduce syntactic

rules to constrain the generated text. Chapman et al. [21] designed a kind of password privacy protection software based on syntactic structure, which can convert the ciphertext into natural text, and then extract the original ciphertext from the natural text. However, the steganographic sentences generated by this method are relatively simple and have a single structure, which can be easily detected and recognized by the monitors, and the security of the algorithm in the application cannot be guaranteed.

The traditional generation-based steganography framework is mainly composed of two parts: a text generation model based on statistical language models and a coding method based on conditional probability distribution. Based on this framework, some researchers use Markov chains to calculate the number of co-occurrences of each phrase and obtain the transition probability, and then use transition probability to encode words, so as to hide secret information in the process of text generation [22–24]. The Markov chain model can be used to generate natural text that conforms to the statistical language model to a certain extent, and even poetry [25] with a fixed format. However, due to the limitations of the Markov chain model itself, unmeaning or grammatical sentences are often generated, and the quality of text generation is limited. Therefore, in practical applications, it can be easily detected and recognized by text steganalysis algorithms.

In recent years, researchers have combined text-generated steganography with statistical language models in natural language processing, and conducted a series of innovative explorations. Taking the advantages of Recurrent Neural Network (RNN) to extract sequence signal features, first learn a statistical language model from a corpus containing a large number of natural language texts. Then, at time t when the text is generated, the RNN can calculate the conditional probability distribution p of the t-th word based on the t−1 word.

Many researchers have adopted the RNN and put forward lots of valuable steganography methods. Fang et al. [16] proposed a text generative steganography system based on Long Short-Term Memory (LSTM) neural network at ACL 2017 in the NLP field. The system uses the LSTM network to learn statistical language models from natural text, and in the process of text generation, selects different words from the precoding dictionary according to the secret information to realize the hiding of the secret information. Compared with the method based on Markov Chains, this system has larger information hiding capacity and higher information embedding rate. Yang et al. [17] also used a multi-layer recurrent neural network with LSTM units to perform steganographic text generation, which is the "RNN-Stega" method. By learning and training from a large corpus, they can obtain a better statistical language model, and estimate the conditional probability distribution of each word, encode the conditional probability of each word in the generated text through a fixed-length Perfect Binary Tree and a variable-length Huffman Coding, then output the corresponding word according to the secret information bit stream to realize the secret information hiding. The experimental results showed that the method had reached the highest level at the time in terms of the hidden efficiency, concealment, and hidden capacity. Then on the basis of the "RNN-Stega" method, Ziegler et al. [18] used Arithmetic Coding to encode strings of known probabilities, which is more effective than Huffman coding and has less damage to the probability distribution of the language. In addition, the system uses one of the best pre-trained language models in the experiment, the GPT-2 model [26], which can generate natural text that is more in line with the statistical language model. And then, based on arithmetic coding, Shen et al. [27] proposed a text-generating steganography algorithm "SAAC" that uses self-adjusting arithmetic coding. The method encodes the conditional probability to further reduce the Kullback-Leibler Divergence of steganographic text, thereby improving the concealment of the algorithm in language statistics. Also, Yang et al. [28] used Variational Auto-Encoder (VAE) to learn the overall

statistical distribution characteristics of texts from the dataset, which further improves the ability of anti-detection of the generated steganographic texts.

After summarizing the relevant results that the former researchers have led to, we find that these algorithms based on the existing text generation steganography model framework which use the conditional probability coding still cannot avoid the inherent shortcoming: unable to increase secret information hiding capacity and improve algorithm concealment at the same time. In the latest study of Yang et al. [29], they revealed that due to the uncontrollable semantics of the generated text, even if the steganographic text is sufficiently concealed in terms of statistical distribution characteristics, there are still some risks. Especially in practical application scenarios, the content and topic of the generated text should also conform to the specific context, especially the long text paragraph containing multiple sentences. The text content must maintain a certain degree of relevance and coherence with the specific topic. However, in the current generation-based steganography method, it is still a challenge to generate controllable text content.

To control the semantics of generated text, Li et al. [30] proposed a Topic-Aware neural linguistic steganography method, which links the generated texts with specific topics by introducing the knowledge graph. As a result, their method performed better on text quality and anti-detection ability when compared with traditional framework. Furthermore, there are still many researchers [31,32] who have made great progress on steganography methods and improved the security performance of linguistic steganography. However, they still preferred to use traditional way rather than generating steganographic texts from the graph structure, so there is still room for improvement in the quality and concealment of the text.

In order to ensure the versatility of the steganography algorithm, we assume that the secret information to be embedded is arbitrary, that is, the content of the secret information is not restricted. Therefore, it will be quite a challenge to not only embed the secret information in the text generation process, but also realize the content control of the generated text. In this paper, to make the text content more coherent, so as to improve the concealment of the secret information, we use keyword matching in the knowledge graph and the graph path encoding, then complete the steganography of secret information by generating sentences corresponding to the KG triples.

3 Method

In this section, we will introduce the methods and model we use in this paper. The overview of our method is shown in Fig. 1. We use "The Prisoners' Problem" model to show the whole transmission process of secret information under monitoring with a part of an example from the experiment, including encoding, generating and decoding.

3.1 Topic Matching

In order to control the semantics of the generated text, we use the knowledge graph to guide the semantic expression of the steganographic text. Comparing with the traditional methods, we can decide the topic of generated texts, so that when the length of the generated text increases, the topic of generated texts will not be jumping, which improves the concealment of the steganographic text. We construct a topic list, add a large number of different topic keywords contained in the knowledge graph to the topic list, then randomly select a topic keyword from it each time, use the method of string matching [33] to find the matching subgraph with specific semantics, and use this keyword as the starting node of the subgraph. In this way, we can ensure the topic consistency of the generated text.

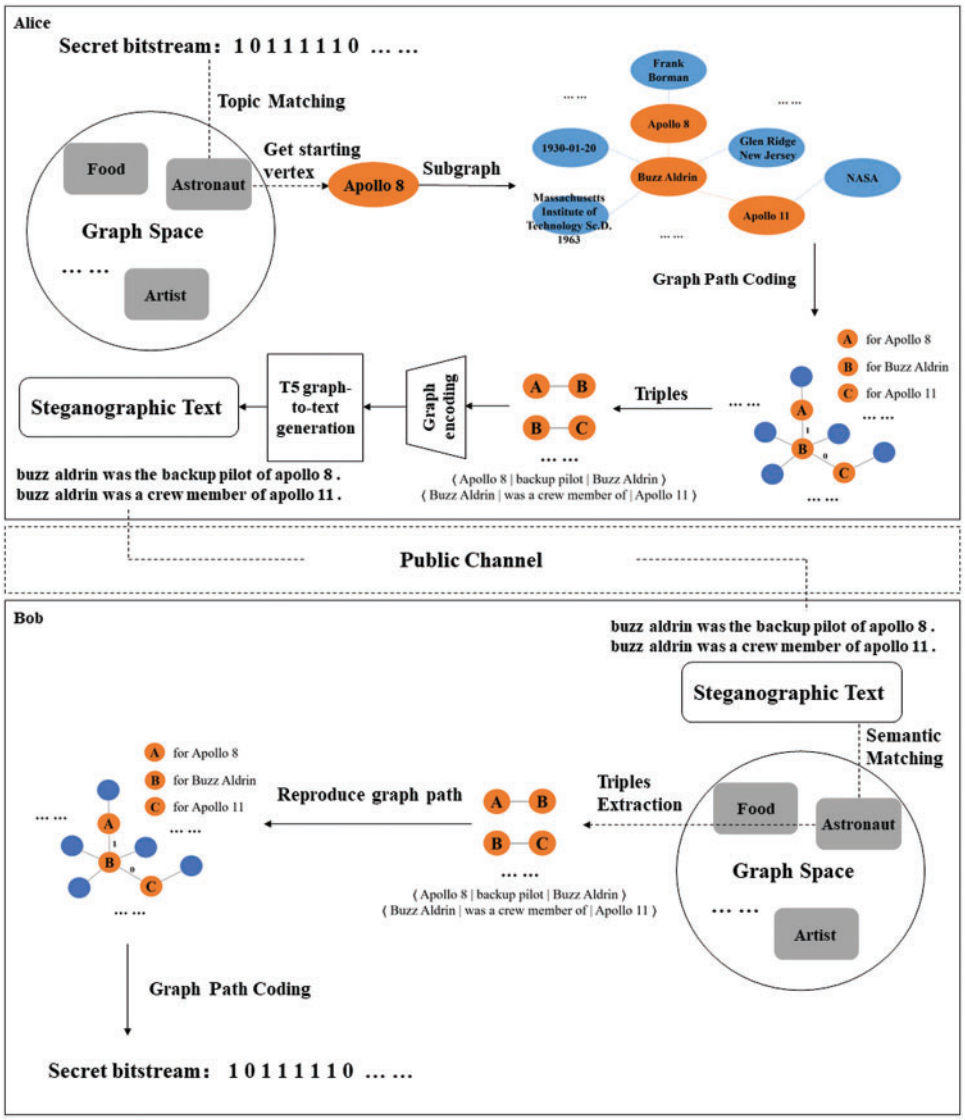

Figure 1: The overview of proposed method

3.2 Graph Path Coding

First, we define a graph structure to represent the knowledge graph, which is a directed graph with entities as vertices and relations as edges, and there is no self-loop and Parallel-edges. A graph with N vertices is defined as:

$$\begin{cases} G = (V \in \mathbf{V}, E \in \mathbf{E} \subseteq (\mathbf{V} \times \mathbf{V})) \\ \mathbf{V} = \cup_{i=1}^{N} \{V_i\}, \mathbf{E} = \cup \{E_{i,j}\}\big|_{i,j \in [1,N]} \end{cases} \tag{3}$$

Among them, $\mathbf{V}$ and $\mathbf{E}$ respectively represent the vertex set and edge set of the graph, that is, the entity set and relation set of the knowledge graph, $\mathbf{E}_{i,j}$ represents the edge from vertex V_i to vertex V_j. Therefore, the edge set $\mathbf{E}_{\text{out}}^{i}$ with vertex V_i as the starting vertex is:

$$\begin{cases} \forall i,j \in [1,N], E_{i,j} : V_i \to V_j \\ \mathbf{E}_{\text{out}}^{i} = \cup \left\{ E_{i,j} \right\}\big|_{j\in[1,N]} \end{cases} \tag{4}$$

Among them, $\left|\mathbf{E}_{\text{out}}^{i}\right|$ represents the number of elements in the edge set, that is, the number of edges starting with vertex V_i.

For any entity vertex V_i in the graph structure, there are $\left|\mathbf{E}_{\text{out}}^{i}\right|$ edges with it as the starting node, and each edge connects to another vertex. In the knowledge graph, such a triple composed of two vertices and one edge can express certain semantic information, so that multiple connected triples can express the trend of related semantic information.

When encoding the graph structure of the knowledge graph, we convert the path encoding of the subgraph to the encoding of the edge set of the starting vertex. However, the size of the edge set of various starting vertices are quite different, that is, the value of $\left|\mathbf{E}_{\text{out}}^{i}\right|$ is uncertain and not necessarily a multiple of 2, it is not recommended to simply use binary fixed-length coding to encode its edge set.

Therefore, in this paper, we intend to use Huffman coding to encode the edge set of each vertex, and its weight is the frequency of appearance of the adjacent entity vertex in the corpus. The process of path coding and secret information hiding on a subgraph based on Huffman coding is shown in Fig. 2, and the corresponding algorithm is shown in Algorithm 1, which contains the detailed logic and measures of our coding method.

Algorithm 1: Information Hiding

Input: subgraph g, starting vertex V_0,
secret binary bitstream: $B = \{100110\ldots\}$
Output: chained subgraph $g_c = \{\}$

Add V_0 to the chained subgraph g_c;
while not the end of B **do**
  Get the edge set $\mathbf{E}_{out}^{0}$ of the starting vertex V_0;
  Construct a Huffman tree according to the weights of each edge;
  if weights of each edge are the same **then**
    The vertex of the entity word in the front is regarded as the left child node;
  else
    The vertex with a larger weight is the left child vertex;
  end if
  The code of the left child vertex is 1, and the code of the right child vertex is 0, get the Huffman code of each edge, that is, the code of each path;
  Select the corresponding path according to the secret binary bitstream B, get the next vertex V_1;
  Remove the edge $E_{i,j}$ from the subgraph g that connects the previous vertex V_0;
  Add V_1 to the chained subgraph g_c;
  Take vertex V_1 as the new starting node.
end while
return chained subgraph g_c

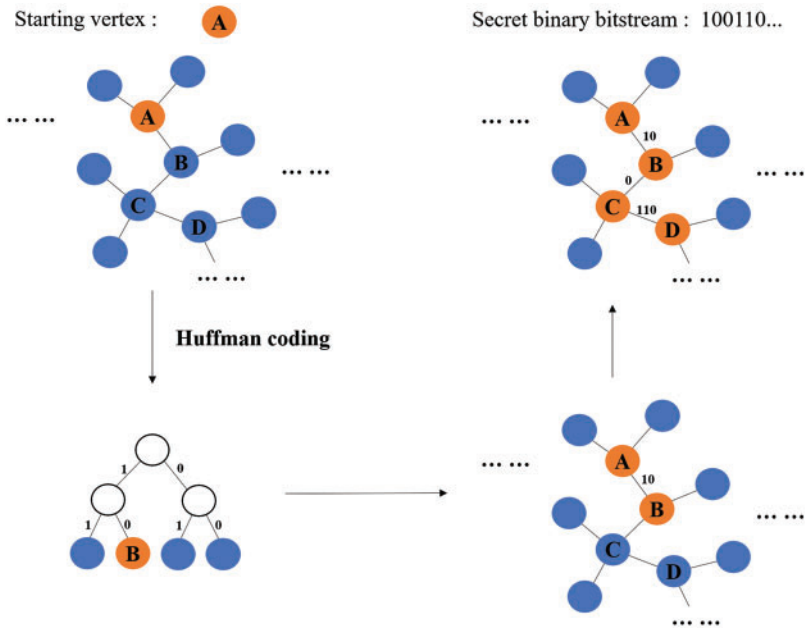

Figure 2: The process of path coding

Through the above method, while completing the secret information embedding, we realize the construction of an ordered directed subgraph chain containing specific semantics from the non-hierarchical knowledge graph. The subgraph chain contains several connected knowledge graph triples, which represent the semantic information on the path.

In the knowledge graph structure, the path coding based on Huffman coding not only guarantees the uniqueness of each path coding, and it will not be the prefix of other path coding, but also, it ensures that any binary bitstream can match the corresponding path. The length of the path or the size of the subgraph depends on the length of the secret information. In this way, the conditional probability distribution of the texts will not get destroyed, and the generated texts will be under the same topic, which can improve the quality and concealment of the steganographic texts.

3.3 Text Generation

In this paper, we use the pre-trained model T5 proposed by Raffel et al. [34]. The T5 model uses a standard Transformer-based encoder-decoder framework and is trained on a large cleaned network text corpus C4. It uses a unified framework to transform various problems in NLP into a text-to-text format.

In order to complete the downstream task of graph-to-text that we need in this paper, we fine-tune the T5 model [35]. First, we preprocess each subgraph obtained in the graph dataset to get RDF-triples, and input them into a sequence. Then we turn the problem into a text-to-text task, and translate the triples into text to train the model. Finally, we add the ordered chained subgraphs obtained in the path encoding as input to the trained model, so that we can get the text $(text1, text2, text3, \ldots)$ generated by each subgraph in an orderly manner to form the steganographic text S. The Fig. 3 shows the whole process of steganographic text generation introduced above.

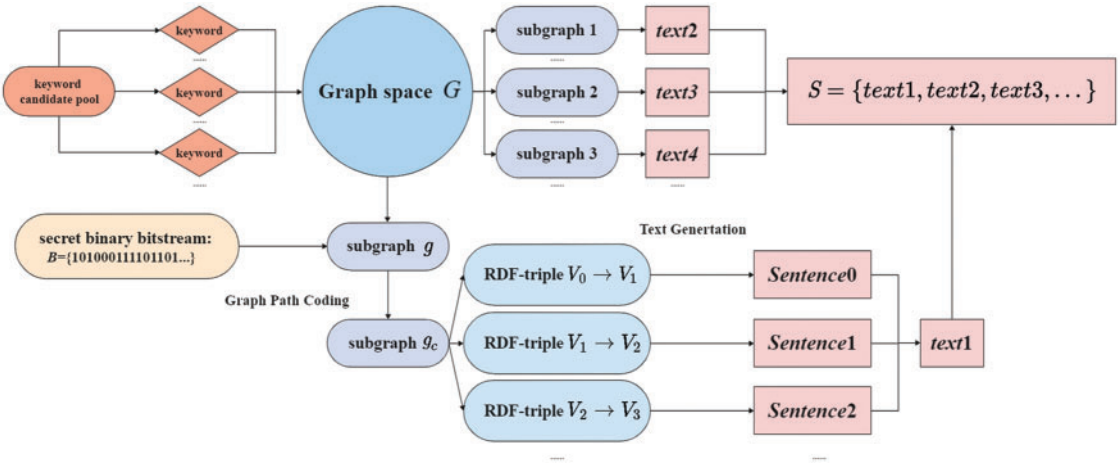

Figure 3: Steganographic text generation

Algorithm 2: Information Extraction

Input: steganographic text $S = \{text1, text2, text3, \ldots\}$,
knowledge graph space G
Output: secret binary bitstream: $B = \{\}$

Add V_0 to the chained subgraph g_c;
while not the end of S **do**
  Read $text1$ from S;
  Get the matching subgraph g according to G;
  Get the corresponding chained subgraph g_c according to $text1$;
  Follow the chained subgraph g_c, get the chain of ordered vertexes $V_0, V_1, V_2, \ldots$;
  while not the end of g_c **do**
    Get the edge set E^0_{out} of the starting vertex V_0;
    Construct a Huffman tree according to the weights of each edge;
    if weights of each edge are the same **then**
      The vertex of the entity word in the front is regarded as the left child node;
    else
      The vertex with a larger weight is the left child vertex;
    end if
    The code of the left child vertex is 1, and the code of the right child vertex is 0, get the Huffman code of each edge, that is, the code of each path;
    Add the code that matches the path $V_0 \rightarrow V_1$ to secret binary bitstream B;
    Remove the edge E_{ij} from the subgraph g that connects the previous vertex V_0;
    Take vertex V_1 as the new starting node.
  end while
  Read the next text $text2$ from S.
end while
return secret binary bitstream B

3.4 Information Extraction

Information extraction and information hiding are a pair of opposite operations. After the receiver Bob receives the steganographic text from the public network transmission channel, the receiver needs to correctly decode the secret information contained in it. In the KG-guided text generation steganography framework proposed in this paper, the embedding of secret information is realized by encoding the node path of the knowledge graph before the text is generated. Therefore, the receiver only needs to reconstruct the node path through graph matching to realize the extraction of secret information.

The detailed algorithm of information extraction is shown in Algorithm 2. In our method, when Bob receives the steganographic text, he first extracts entity keywords from the generated text, identifies the subgraphs in sequence, and then performs subgraph matching and coding in the same knowledge graph shared by both the sender and the receiver to reconstruct the path of the subgraph nodes, so that the code of the subgraph can be extracted. It should be noted that Alice and Bob must have exactly the same knowledge graph dataset and use the same path encoding method. Because in the knowledge graph shared by both parties, the path coding of each subgraph is unique, so it can be guaranteed that Bob can accurately extract the hidden secret information.

4 Experiments and Analysis

In this section, we will introduce the details and environments of our experiments. And we evaluate the method from four aspects: semantic correlation, text quality, topic correlation and anti-detection.

4.1 Dataset and Model Training

During the experiments, we use the pre-trained model T5 as we introduced in Section 3.3. To evaluate the proposed method, we fine-tune the model with WebNLG dataset [36], which contains pairs of knowledge graph and corresponding target text that can describe the graph. Specifically, the WebNLG dataset consists of 21855 data/text pairs with a total of 8372 distinct data input. And the input describes entities belonging to 9 distinct DBpedia categories namely, Astronaut, University, Monument, Building, ComicsCharacter, Food, Airport, SportsTeam and WrittenWork. Since the small subgraph which contains few triples are meaningless for path coding, we use the subgraphs in the text dataset that contains more than five triples as our test dataset, and there are 307 subgraphs that meet the requirement. The data statistics of WebNLG are shown in Table 1.

Table 1: The data statistics of WebNLG

	Train	Dev	Test (original)	Test (our)
Number of subgraphs	18102	872	1862	307
Number of triples	53786	2563	5378	1601
Average triples per subgraph	2.97	2.93	2.88	5.21
Average length of gold text	19.84	20.15	19.52	32.82

To preprocess the dataset, we identify the entities and relationships in the dataset triples as the model's vocabulary. Then during the training, we set the initial learning rate as $3 \cdot 10^{-5}$, and we choose the batch size as 4. GeForce RTX 3070 GPU and CUDA 11.1 are used for training acceleration.

To evaluate our method, we compare our method with three other steganography methods. Firstly, following the Topic-Aware method proposed in [30], we use conditional probability coding in the same Graph2Text model and WebNLG dataset. Secondly, following the RNN-Stega method proposed in Yang et al. [17], we use RNN for the train set of WebNLG dataset, while using conditional probability coding for the generation. The initial learning rates are set as 0.001 and batch size is set as 128, dropout rate is 0.5. GeForce RTX 3070 GPU and CUDA 8.0 are used for training acceleration. Thirdly, following the VAE-Stega method proposed in Yang et al. [28], two different encoders are used, one of them uses a recurrent neural network with LSTM units as the encoder (shown as VAE-Stega (lstm) in the tables of experimental results) and the other uses Bidirectional Encoder Representations from Transformers (BERT) [37] as the encoder (shown as VAE-Stega(bert) in the tables of experimental results). For VAE-Stega(lstm), the initial learning rates are set as 0.001 and batch size is set as 128. And for VAE-Stega(bert), a pre-trained model released by [37] is also used, the initial learning rates are set as 0.001 while batch size is set as 20. Quadro RTX 5000 and CUDA 9.0 are used for training acceleration.

4.2 Embedding Rate

In general, the generated texts of linguistic steganography consist of N steganographic sentences. The embedding rate ER stands for how many bits a word can carry (bits per word) during the generation. For the methods using conditional probability coding like RNN-Stega, Topic-Aware and VAE-Stega, ER can be defined as:

$$ER = \frac{1}{N}\sum_{i=1}^{N}\frac{b_i}{n_i} \tag{5}$$

Among them, N represents the number of the sentences in the generated texts, while n is the number of words each steganographic sentence contains and b is the number of bits that each steganographic sentence can carry.

The embedding rate of our method can be defined as:

$$ER = \frac{1}{N}\sum_{i=1}^{N}\frac{\sum_{j=1}^{t}B_j}{n_i} \tag{6}$$

Among them, N represents the number of the sentences in the generated texts, n is the number of words each steganographic sentence contains, t represents how many triples each steganographic sentence uses for generation and B is the number of bits that each triple can carry. The embedding rate of our method depends on the number of triples that each steganographic sentence contains, which is highly relevant to the dataset we use. Therefore, if the experiments are conducted on larger datasets which contain more relations, the embedding efficiency of the proposed method could be higher.

4.3 Evaluation of Semantic Correlation

Semantic correlation is an important evaluation index in the fields of text generation and machine translation. By calculating the similarity between the generated text of the model and the artificial reference text on the vocabulary level, we can more intuitively understand the reliability of the generated texts.

In this paper, the WebNLG dataset provided pairs of knowledge graph and corresponding target text, so that we can use random bitstreams to generate texts from the subgraphs, then analyze the semantic correlation between the generated texts and the target texts. We use the automatic metrics BLEU [38], ROUGE-L [39] and CIDEr [40] for evaluation. We also test the texts generated from

the RNN-Stega, Topic-Aware and VAE-Stega as comparison, and we set different embedding rate on RNN-Stega, Topic-Aware and VAE-Stega for experiments.

We use all the 307 subgraphs that contains more than five triples in the test dataset for text generation. Since we will not use all the triples in the subgraph, the generated texts are shorter than the target texts. In order to set the experiments under the same condition, we adjust the length of the text generated by RNN-Stega, Topic-Aware and VAE-Stega to the average length of the text generated by our method. The results are shown in Table 2.

Table 2: Evaluation of semantic correlation

Method	Metrics			
	BLEU-1	BLEU-2	Rouge-L	CIDEr
OurMethod	**3.8804**	**2.3030**	**12.8880**	**0.0081**
Topic-Aware(embedding rate = 1)	3.6962	1.6493	11.6101	0.0075
Topic-Aware(embedding rate = 2)	2.3598	0.7996	9.3598	0.0049
Topic-Aware(embedding rate = 3)	1.8712	0.5750	7.4555	0.0041
RNN-Stega(embedding rate = 1)	3.3912	1.4241	11.4362	0.0025
RNN-Stega(embedding rate = 2)	2.9437	1.0440	10.2071	0.0034
RNN-Stega(embedding rate = 3)	2.9916	1.0835	10.1558	0.0023
VAE-Stega(lstm)(embedding rate = 1)	3.4698	1.4724	11.5866	0.0029
VAE-Stega(lstm)(embedding rate = 2)	3.3401	1.4785	11.3752	0.0069
VAE-Stega(lstm)(embedding rate = 3)	3.1381	1.3397	11.0880	0.0029
VAE-Stega(bert)(embedding rate = 1)	3.5337	1.4670	11.7155	0.0037
VAE-Stega(bert)(embedding rate = 2)	3.3084	1.5250	11.5249	0.0033
VAE-Stega(bert)(embedding rate = 3)	3.2304	1.4386	11.1096	0.0031

As we can conclude from the results, the scores of RNN-Stega, Topic-Aware and VAE-Stega decrease when the embedding rate increases, and the scores of our method are obviously higher, which indicates that our method can generate texts that follow the input semantic information to a certain extent.

4.4 Evaluation of Text Quality

The text quality represents the imperceptibility of information hiding, which is one of the most important evaluation of a concealment system. In this paper, we set the embedding rate of RNN-Stega, Topic-Aware and VAE-Stega from 1 to 3, and we use *perplexity* [41] to analyze the text quality. It is a widely used evaluation method in the field of NLP, and it is defined as the average per-word log-probability on the test texts:

$$\begin{aligned}\text{Perplexity} &= 2^{-\frac{1}{n}\log p(S)} \\ &= 2^{-\frac{1}{n}\log p\left(\text{Word},\text{Word}_2,\ldots,\text{Word}_n\right)} \\ &= 2^{-\frac{1}{n}\sum_{j=1}^{n}\log p\left(\text{Word}_j|\text{word}_1,\text{Word}_2,\ldots,\text{Word}_{j-1}\right)}\end{aligned} \tag{7}$$

Among them, $S = \{\text{Word}_1, \text{Word}_2, \ldots, \text{Word}_n\}$ represents the generated text, while $p(S)$ is the probability distribution of the text. During the experiment, we test the texts that generated in Section 4.3. The results are shown in Table 3.

Table 3: Evaluation of text quality

Method	Metrics
	Perplexity
OurMethod	**109.14**
Topic-Aware(embedding rate = 1)	167.99
Topic-Aware(embedding rate = 2)	212.16
Topic-Aware(embedding rate = 3)	763.39
RNN-Stega(embedding rate = 1)	221.72
RNN-Stega(embedding rate = 2)	409.06
RNN-Stega(embedding rate = 3)	327.77
VAE-Stega(lstm)(embedding rate = 1)	161.01
VAE-Stega(lstm)(embedding rate = 2)	262.74
VAE-Stega(lstm)(embedding rate = 3)	318.87
VAE-Stega(bert)(embedding rate = 1)	130.52
VAE-Stega(bert)(embedding rate = 2)	386.65
VAE-Stega(bert)(embedding rate = 3)	629.77

From the results, we can clearly see that the perplexity of RNN-Stega, Topic-Aware and VAE-Stega gets higher as the embedding rate increases, and the perplexity of our method is much lower than that of RNN-Stega, Topic-Aware and VAE-Stega, which means that the texts generated by our method are closer to the real semantic expression, that is, our method performed better in information imperceptibility.

4.5 Evaluation of Topic Correlation

In this part, we use two evaluating indicators to analyze topic correlation of our method. First, we use Topic-Coherence, which is an important measurement for evaluating topic models. We adopt LDA topic model [42] to train the target texts and evaluate Topic-Coherence score of our generated texts. The results are shown in Table 4, where the "tp" in the table means the number of topic words that used in the evaluation. From the results we can see that our method get better scores with different topic numbers, which means our method performs better in controlling the topic in steganography texts.

Second, we use an IE (Information Extraction) model proposed in [43]. After training the IE model using the WebNLG dataset, we use the model to extract triples from our generated texts and calculate the Precision, Recall and F1 score of the extraction. Since the conditional probability coding changes the statistical distribution characteristics of texts, which makes it hard for the IE model to extract the triples from the texts generated by Topic-Aware and VAE-Stega, we only present the results on other methods. The results are shown in Table 5 with the corresponding figure shown in Fig. 4, where the correct, Predict, Gold in the table stand for the number of triples extracted correctly, the number of triples predicted from the model, the number of corresponding triples matched in the gold texts, the results of IE system are also shown here as a baseline. An example of information extraction on our dataset is also proposed in Table 6. From the results we can see that the steganography texts generated from our method present the entities and relationship in the text more precisely, which means the topic of generated texts closer to the gold texts.

Table 4: Evaluation of topic-coherence

Method	Coherence (tp5)	Coherence (tp10)	Coherence (tp20)
OurMethod	**0.395**	**0.612**	**0.769**
Topic-Aware	0.382	0.608	0.744
RNN-Stega	0.369	0.464	0.572
VAE-Stega(lstm)	0.350	0.553	0.691
VAE-Stega(bert)	0.308	0.511	0.689

Table 5: Evaluation of topic correlation

Method	Correct	Predict	Gold	Precision	Recall	F1
IE	705	954	988	0.739	0.714	0.726
OurMethod	**157**	**261**	**236**	**0.602**	**0.665**	**0.632**
RNN-Stega	15	304	38	0.049	0.395	0.088

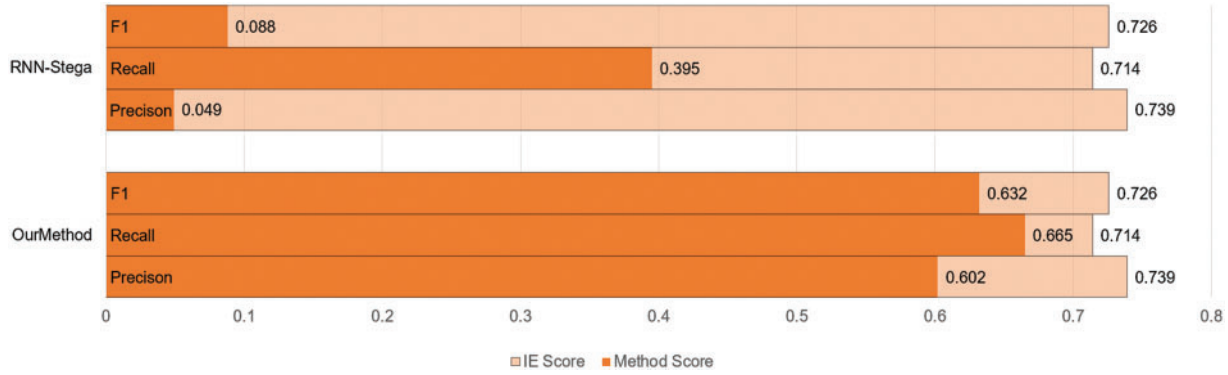

Figure 4: Evaluation of topic correlation

Table 6: Information extraction results of topic correlation

Generated texts	Predict triples	Correct triples
1634:the bavarian crisis is followed by ring of fire ii.	⟨1634:the bavarian crisis | successor | ring of fire ii⟩, ⟨**1634: the bavarian crisis** | **followedBy** | **ring of fire ii**⟩	⟨1634 : the bavarian crisis | followedBy | ring of fire ii⟩
The baku turkish martyrs' memorial is located in azerbaijan. artur rasizade is the leader of azerbaijan.	⟨**Azerbaijan** | **leader** | **Artur Rasizade**⟩, ⟨Azerbaijan | leaderName | Artur Rasizade⟩, ⟨**Baku Turkish Martyrs' Memorial** | **location** | **Azerbaijan**⟩	⟨Azerbaijan | leader | Artur Rasizade⟩, ⟨Baku Turkish Martyrs' Memorial | location | Azerbaijan⟩
Buzz aldrin served as a fighter pilot.	⟨**Buzz Aldrin** | **occupation** | **Fighter pilot**⟩, ⟨Buzz Aldrin | 1st_runway_Surface Type | Fighter pilot⟩	⟨Buzz Aldrin | occupation | Fighter pilot⟩
Buzz aldrin was a crew member of apollo 11. apollo 11 was operated by nasa.	⟨**Apollo 11** | **operator** | **NASA**⟩, ⟨**Buzz Aldrin** | **was a crew member of** | **Apollo 11**⟩	⟨Apollo 11 | operator | NASA⟩, ⟨Buzz Aldrin | was a crew member of | Apollo 11⟩

4.6 Evaluation of Anti-Detection

With the continuous development of steganography, various steganalysis methods are also developing. During the experiment, we also use three different steganalysis methods: FCN [44], CNN [45], RBiLSTMC [46], so that we can analyze the anti-detection ability of our method. The results are shown in Table 7. Since the length of texts that generated from our method is uncontrollable, the advantage of the anti-detection ability of our method is not huge, but we can still see from the results that the steganalysis method can detect the other two methods more easily, which means that our method can actually avoid detection to a certain degree.

Table 7: Evaluation of anti-detection

Steganalysis method	Method	Accuracy	Precision	Recall	F1-score
FCN [44]	OurMethod	**0.6509**	0.6333	**0.7169**	**0.6725**
	RNN-Stega	0.7075	0.6666	0.8301	0.7394
	Topic-Aware	0.6603	**0.6103**	0.8867	0.7230
	VAE-Stega(lstm)	0.6666	0.6444	0.7435	0.6904
	VAE-Stega(bert)	0.6875	0.6667	0.7500	0.7058
CNN [45]	OurMethod	**0.7238**	0.7174	**0.6735**	**0.7091**
	RNN-Stega	0.8462	0.7500	0.9000	0.8182
	Topic-Aware	0.7480	**0.7000**	0.7636	0.7304
	VAE-Stega(lstm)	0.7949	0.7188	0.7667	0.7419
	VAE-Stega(bert)	0.8125	0.8400	0.6810	0.7368

(Continued)

Table 7 (continued)

Steganalysis method	Method	Accuracy	Precision	Recall	F1-score
RBiLSTMC [46]	OurMethod	**0.7239**	**0.6923**	0.7947	**0.7129**
	RNN-Stega	0.8076	0.8182	0.8372	0.8275
	Topic-Aware	0.7804	0.7857	**0.7457**	0.7652
	VAE-Stega(lstm)	0.8462	0.7812	0.8333	0.8065
	VAE-Stega(bert)	0.8625	0.8387	0.8125	0.8254

4.7 Example of Generation

In this section, we will demonstrate an example of subgraph and corresponding text that generated from our method. To present the process of coding and generation more directly, we artificially combine several subgraphs from the original WebNLG dataset to get a bigger subgraph. During the path coding, 7 triples were used to generate text in this subgraph. The example is shown in Fig. 5, the entities that used for generation were tagged in orange, while the not used ones were tagged in blue. The used triples, bitstream and the corresponding generated text are shown in Table 8.

Figure 5: Example of a knowledge graph

Table 8: Example of generation

Triples	Encoding
⟨Apollo 8 | backup pilot | Buzz Aldrin⟩	1
⟨Buzz Aldrin | was a crew member of | Apollo 11⟩	0
⟨Apollo 11 | backup pilot | William Anders⟩	1
⟨William Anders | nationality | United States⟩	11
⟨United States | largest City | New York City⟩	1
⟨New York City | is Part Of | Manhattan⟩	1
⟨Manhattan| leader Name | Cyrus Vance Jr.⟩	0
Bitstream: 10111110 . . .	
Generated text:	
Buzz aldrin was the backup pilot of apollo 8.	
Buzz aldrin was a crew member of apollo 11.	
William anders was the backup pilot of apollo 11.	
William anders is from the United States.	
The largest city in the United States is new york city.	
Manhattan is part of New York City.	
Cyrus Vance Jr. is the leader of Manhattan.	

5 Conclusion

Steganography is a hot topic with challenge and huge research value. In this paper, we propose a model that can automatically generate steganographic texts from knowledge graphs. In order to hide secret information, we abandon traditional way of using conditional probability and choose to encode during the process of graph-to-text generation. We realize the semantic control of the steganographic texts by introducing knowledge graphs with our topic matching method. We also provide detailed algorithms for our path coding method, and we carry out many comparative experiments to verify the effect of our method. Compared with the previous steganography methods, the experiments confirm the feasibility of our method. The texts generated by our method have a certain improvement in imperceptibility and anti-detection ability. The results show that compared with the previous methods, our method improves the quality of steganographic texts by more than 34%, and improves semantic correlation and anti-detection ability by more than 3% and 6%. In future work, we look forward to building our own dataset which takes the depth of the graph structure as the construction standard and has a larger capacity; in this way, we can better use our method and put it into practical application. Furthermore, we hope to create more effective methods in linguistic steganography. This paper successfully reveals the possibility of combining graph-to-text generation with steganography and we hope that our work can bring help and inspiration to more researchers in this field.

Funding Statement: This work was supported in part by the National Natural Science Foundation of China [62102136], the 2020 Opening Fund for Hubei Key Laboratory of Intelligent Vision Based Monitoring for Hydroelectric Engineering [2020SDSJ06] and the Construction Fund for Hubei Key Laboratory of Intelligent Vision Based Monitoring for Hydroelectric Engineering [2019ZYYD007].

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Mazurczyk, W., Caviglione, L. (2015). Information hiding as a challenge for malware detection. *IEEE Security & Privacy, 13(2),* 89–93. DOI 10.1109/MSP.2015.33.
2. Bash, B. A., Goeckel, D., Towsley, D., Guha, S. (2015). Hiding information in noise: Fundamental limits of covert wireless communication. *IEEE Communications Magazine, 53(12),* 26–31. DOI 10.1109/MCOM.2015.7355562.
3. Hassan, F. S., Gutub, A. (2022). Improving data hiding within colour images using hue component of HSV colour space. *CAAI Transactions on Intelligence Technology, 7(1),* 56–68. DOI 10.1049/cit2.12053.
4. Hassan, F. S., Gutub, A. (2021). Efficient image reversible data hiding technique based on interpolation optimization. *Arabian Journal for Science and Engineering, 46(9),* 8441–8456. DOI 10.1007/s13369-021-05529-3.
5. Yong, F. H., Tang, S., Jian, Y. (2011). Steganography in inactive frames of voip streams encoded by source codec. *IEEE Transactions on Information Forensics & Security, 6(2),* 296–306. DOI 10.1109/TIFS.2011.2108649.
6. Peng, F., Long, M., Zhang, X. (2018). Robust coverless image steganography based on DCT and LDA topic classification. *IEEE Transactions on Multimedia, 20(12),* 3223–3238. DOI 10.1109/TMM.2018.2838334.
7. Yang, Z., Peng, X., Huang, Y. (2017). A sudoku matrix-based method of pitch period steganography in low-rate speech coding. *International Conference on Security and Privacy in Communication Systems*, pp. 752–762. Cham, Springer.
8. Gutub, A., Al-Shaarani, F. (2020). Efficient implementation of multi-image secret hiding based on LSB and DWT steganography comparisons. *Arabian Journal for Science and Engineering, 45(4),* 2631–2644. DOI 10.1007/s13369-020-04413-w.
9. Al-Shaarani, F., Gutub, A. (2022). Increasing participants using counting-based secret sharing via involving matrices and practical steganography. *Arabian Journal for Science and Engineering, 47(2),* 2455–2477. DOI 10.1007/s13369-021-06165-7.
10. Yang, W., Li, M., Zhou, B., Liu, Y., Liu, K. et al. (2021). Steganalysis of low embedding rate CNV-QIM in speech. *Computer Modeling in Engineering & Sciences, 128(2),* 623–637. DOI 10.32604/cmes.2021.015629.
11. Zhang, J., Shen, J., Wang, L., Lin, H. (2016). Coverless text information hiding method based on the word rank map. *International Conference on Cloud Computing and Security*, pp. 145–155. Cham, Springer.
12. Zhang, J., Xie, Y., Wang, L., Lin, H. (2017). Coverless text information hiding method using the frequent words distance. *International Conference on Cloud Computing and Security*, pp. 121–132. Cham, Springer.
13. Chotikakamthorn, N. (1998). Electronic document data hiding technique using inter-character space. *1998 IEEE Asia-Pacific Conference on Circuits and Systems. Microelectronics and Integrating Systems. Proceedings (Cat. No. 98EX242)*, pp. 419–422. Chiang Mai, Thailand: IEEE.
14. Xiang, L., Wang, X., Yang, C., Liu, P. (2017). A novel linguistic steganography based on synonym run-length encoding. *IEICE Transactions on Information and Systems, 100(2),* 313–322. DOI 10.1587/transinf.2016EDP7358.
15. Xiang, L., Sun, X., Luo, G., Xia, B. (2014). Linguistic steganalysis using the features derived from synonym frequency. *Multimedia Tools and Applications, 71(3),* 1893–1911. DOI 10.1007/s11042-012-1313-8.
16. Fang, T., Jaggi, M., Argyraki, K. (2017). Generating steganographic text with lstms. arXiv Preprint arXiv:1705.10742.
17. Yang, Z. L., Guo, X. Q., Chen, Z. M., Huang, Y. F., Zhang, Y. J. (2018). RNN-Stega: Linguistic steganography based on recurrent neural networks. *IEEE Transactions on Information Forensics and Security, 14(5),* 1280–1295. DOI 10.1109/TIFS.2018.2871746.

18. Ziegler, Z. M., Deng, Y., Rush, A. M. (2019). Neural linguistic steganography. arXiv preprint arXiv:1909.01496.
19. Simmons, G. J. (1984). The prisoners' problem and the subliminal channel. In: *Advances in cryptology*, pp. 51–67. Boston, MA: Springer.
20. Wayner, P. (1992). Mimic functions. *Cryptologia, 16(3),* 193–214. DOI 10.1080/0161-119291866883.
21. Chapman, M., Davida, G. (1997). Hiding the hidden: A software system for concealing ciphertext as innocuous text. *International Conference on Information and Communications Security*, pp. 335–345. Berlin, Heidelberg, Springer.
22. Dai, W., Yu, Y., Dai, Y., Deng, B. (2010). Text steganography system using markov chain source model and DES algorithm. *Journal of Software, 5(7),* 785–792. DOI 10.4304/jsw.5.7.785-792.
23. Moraldo, H. H. (2014). An approach for text steganography based on markov chains. arXiv preprint arXiv:1409.0915.
24. Shniperov, A. N., Nikitina, K. (2016). A text steganography method based on markov chains. *Automatic Control and Computer Sciences, 50(8),* 802–808. DOI 10.3103/S0146411616080174.
25. Luo, Y., Huang, Y., Li, F., Chang, C. (2016). Text steganography based on ci-poetry generation using markov chain model. *KSII Transactions on Internet and Information Systems (TIIS), 10(9),* 4568–4584.
26. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. et al. (2019). Language models are unsupervised multitask learners. *OpenAI Blog, 1(8),* 9.
27. Shen, J., Ji, H., Han, J. (2020). Near-imperceptible neural linguistic steganography via self-adjusting arithmetic coding. arXiv preprint arXiv:2010.00677.
28. Yang, Z. L., Zhang, S. Y., Hu, Y. T., Hu, Z. W., Huang, Y. F. (2020). VAE-Stega: Linguistic steganography based on variational auto-encoder. *IEEE Transactions on Information Forensics and Security, 16,* 880–895. DOI 10.1109/TIFS.10206.
29. Yang, Z., Hu, Y., Huang, Y., Zhang, Y. (2019). Behavioral security in covert communication systems. *International Workshop on Digital Watermarking*, pp. 377–392. Cham, Springer.
30. Li, Y., Zhang, J., Yang, Z., Zhang, R. (2021). Topic-aware neural linguistic steganography based on knowledge graphs. *ACM/IMS Transactions on Data Science, 2(2),* 1–13. DOI 10.1145/3418598.
31. Yang, Z., Xiang, L., Zhang, S., Sun, X., Huang, Y. (2021). Linguistic generative steganography with enhanced cognitive-imperceptibility. *IEEE Signal Processing Letters, 28,* 409–413. DOI 10.1109/LSP.97.
32. Zhang, S., Yang, Z., Yang, J., Huang, Y. (2021). Provably secure generative linguistic steganography. arXiv preprint arXiv:2106.02011.
33. Knuth, D. E., Morris, J. H., Pratt, V. R. (1977). Fast pattern matching in strings. *Siam Journal on Computing, 6(2),* 323–350. DOI 10.1137/0206024.
34. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S. et al. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research, 21(140),* 1–67.
35. Ribeiro, L. F., Schmitt, M., Schütze, H., Gurevych, I. (2020). Investigating pretrained language models for graph-to-text generation. arXiv preprint arXiv:2007.08426.
36. Gardent, C., Shimorina, A., Narayan, S., Perez-Beltrachini, L. (2017). The webnlg challenge: Generating text from rdf data. *Proceedings of the 10th International Conference on Natural Language Generation*, pp. 124–133. Santiago de Compostela, Spain.
37. Devlin, J., Chang, M. W., Lee, K., Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
38. Papineni, K., Roukos, S., Ward, T., Zhu, W. J. (2002). BLEU: A method for automatic evaluation of machine translation. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318. Philadelphia, Pennsylvania, USA.
39. Lin, C. Y. (2004). Rouge: A package for automatic evaluation of summaries. In: *Text summarization branches out*. Barcelona, Spain: Association for Computational Linguistics.

40. Vedantam, R., Lawrence Zitnick, C., Parikh, D. (2015). CIDEr: Consensus-based image description evaluation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4566–4575. Boston, MA, USA.
41. Jurafsky, D. (2000). *Speech & language processing.* India: Pearson Education India.
42. Ramage, D., Hall, D., Nallapati, R., Manning, C. D. (2009). Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pp. 248–256. Singapore.
43. Zheng, H., Wen, R., Chen, X., Yang, Y., Zhang, Y. et al. (2021). PRGC: Potential relation and global correspondence based joint relational triple extraction. arXiv preprint arXiv:2106.09895.
44. Yang, Z., Huang, Y., Zhang, Y. J. (2019). A fast and efficient text steganalysis method. *IEEE Signal Processing Letters, 26(4),* 627–631. DOI 10.1109/LSP.2019.2902095.
45. Wen, J., Zhou, X., Zhong, P., Xue, Y. (2019). Convolutional neural network based text steganalysis. *IEEE Signal Processing Letters, 26(3),* 460–464. DOI 10.1109/LSP.2019.2895286.
46. Niu, Y., Wen, J., Zhong, P., Xue, Y. (2019). A hybrid r-bilstm-c neural network based text steganalysis. *IEEE Signal Processing Letters, 26(12),* 1907–1911. DOI 10.1109/LSP.2019.2953953.