Research Article

Generative Text Secret Sharing with Topic-Controlled Shadows

Long Yu (1,2), Yuliang Lu (1,2), Xuehu Yan (1,2), Yue Jiang (1,2) and Jiayu Wang (1,2)

1National University of Defense Technology, Hefei 230037, China
2Anhui Province Key Laboratory of Cyberspace Security Situation Awareness and Evaluation, Hefei 230037, China

Correspondence should be addressed to Yuliang Lu; publicluyl@126.com and Xuehu Yan; publictiger@126.com

Received 16 June 2022; Revised 5 September 2022; Accepted 12 October 2022; Published 22 November 2022

1.Introduction

Protecting sensitive information from malicious interference is essential when transmitted over public channels. Shannon [1] summarizes three basic information security systems, encryption system, privacy system, and concealment system. By creating a secret key, the encryption system protects the confidentiality of the message itself. The purpose of privacy systems is to prevent unauthorized users from accessing confidential messages. To protect the existence of confidential messages, the concealment system transmits them through open channels using different types of carriers. The secret sharing (SS) of \((k, n)\) threshold [2, 3] satisfies both encryption and privacy requirements. It encrypts a secret message as \(n\) shares (shadows), which are distributed to different participants. Any \(k\) shadows can recover the secret message, less than \(k\) shadows cannot obtain anything. A wide range of applications can be performed with it, including access control, password transmission, distributed storage systems, blockchain security, and cloud computing security [4–7].

Secret image sharing (SIS) has been extensively studied [7–11] since digital images are critical media types. As social networks develop, more and more digital media is being distributed through them. Before being transmitted through social networks, digital media is compressed to reduce storage costs and increase transmission speeds. Digital images are often contaminated by multiple noises due to the defects of transmission and storage media [12]. Secret sharing schemes rely on mathematical models and precise calculations, and slight changes may result in completely different recovery results. Secret sharing schemes often generate noise-like shadow images [8, 11] because they use random during the sharing process, which can easily attract the attention of attackers during transmission. When the communication behavior is discovered, the person using ciphertext communication stands out and thus becomes a priority target for monitoring and analysis. All these realities pose significant challenges to SIS.

Since ancient times, the text has served as the primary means of communication for humans. Generally, the channel does not compress or interfere with text by noise, so
the text are robust when transmitted in a public channel. These indicate that the text may be a more suitable form of data to be transmitted as the shadow in public channels than images. Since the value range of an image’s pixels is \([0, 255]\), SIS can directly establish a mapping relationship from the shared value to the pixel value, thus combining the shared values into an image form. In contrast, text, as a sequence of words with semantic relevance and syntactic rules, cannot form a direct correspondence between shared values and words. Yang et al. [13] proposed a generative text steganography scheme in which they encode candidate words in the text generation process of language model (LM). Later they choose the corresponding words for output according to the secret bits to be embedded, thus completing the mapping of binary bits to word space. With the help of this mapping method, this article proposes to encode the candidate words with the perfect binary tree in the generation process of shadow text, then determine the output words according to the shared values. Since the generation process is under the constraint of the language model, each shadow text is a fluent and natural utterance, which makes this scheme also satisfy the characteristic of the concealment system, that is, imperceptibility.

Social networks’ complex and open characteristics provide an excellent camouflage environment for transmitting shadow texts. The language characteristics of each social account are different due to the interest fields, professional directions, etc. The concealment and security of the transmission of shadow texts through social networks can be further enhanced if the semantics of shadow texts can be effectively controlled combined with the social accounts’ characteristics. Controllable text generation (CTG) can control text characteristics, such as emotion and style, while ensuring the content [14–16]. CTG involves modeling \(P(x|α)\), \(α\) is the target attribute and \(x\) is the sample to generate. A Plug and Play Language Model (PPLM) for controllable language generation was proposed by Dathathri et al. [17], which is a combination of a pretrained generative model \(P(x)\) and attribute models. We propose using this CTG method to control the topic of each shadow text through the bag of words (BoW) associated with different topic words, thus making the shadow texts more suitable for social network scenarios.

In this article, we propose a generative text secret sharing with topic-controlled shadows (GTSS) scheme, which shares a secret message as \(n\) shadow texts, each of which can have a different topic, and any \(k\) shadow texts can obtain the secret message. This article’s motivations and contributions are summarized as follows:

(i) In response to the problems faced by the secret image sharing scheme, such as the transmission process of shadow images easily suffering from compression and noise pollution and the shadow images being susceptible to suspicion, this article proposes to use texts as shadows, which are more suitable for robust and covert transmission in the social network environment.

(ii) To address the problem that the shared values cannot be directly linked to the word space, this article proposes to encode the candidate words with the perfect binary tree in the text generation process to establish the mapping from the shared value space to the word space.

(iii) Due to the differences in characteristics of social network users’ speeches, we propose to control the generated shadow texts’ topics with BoW so that they can be more concealable in social network scenarios.

(iv) Most importantly, we propose two goal programming models, which deeply integrate secret sharing, encoding, and controllable text generation techniques and can enhance the topic relevance and text fluency, respectively. Compared with existing generative text steganography schemes, GTSS has certain advantages in both generated text quality and detection resistance.

2. Preliminaries and Related Work

Preliminaries and related work regarding the proposed scheme are presented here. First, we introduce the definition of SS and the SS scheme based on the matrix theory used by GTSS. Then, we introduce the method of mapping binary bits to word space. Finally, we introduce the transformer principle and the transformer-based controllable text generation method.

2.1. Secret Sharing Based on Matrix Theory. Secret sharing can be defined as follows [18].

Share: a randomized algorithm that outputs a sequence \((s_1, \ldots, s_n)\) of shares based on the input message \(m \in M\).

Reconstruct: a deterministic algorithm that outputs a message based on a collection of \(k\) or more shares.

\(M\) is the space of message, \(k\) is threshold. The users can be numbered as \([1, 2, \ldots, n]\) and the user \(i\) holds share \(s_i\). Denote \(U = \{1, \ldots, n\}\) as a subset of users. The set of shares belonging to users \(U\) is \([s_i|s_i \in U]\). If \(|U| \geq k\), then \(U\) is authorized; otherwise it is unauthorized. Secret sharing aims to allow authorized sets of users/shares to recover the secret, while unauthorized sets cannot.

Yu et al. [19] proposed an SIS scheme modulo 256, which is not limited to the restriction that the modulo (denoted by \(p\) of the traditional SS scheme must be a prime number, and making the shared value space correspond perfectly to the range of grayscale image pixel values. They first construct an \(n \times k\) sharing matrix \(K\), which satisfies the determinant of any \(k \times k\) submatrix is odd. After that, the secret pixel value to be shared is put into the first element of vector \(α = (α_0, α_1, \ldots, α_{k−1})^T\) and the rest elements of \(α\) take value randomly from \([0, p−1]\). Then, \(n\) shared values \(s = (s_1, s_2, \ldots, s_n)^T\) are obtained by matrix multiplication as shown in the following equations:
Each shared value $s_i$ corresponds to one row vector $K[i]$, and recovery can be performed when $k$ combinations of $(s_i, K[i])$ are obtained. During the recovery process, the recovery matrix $K'$ is constructed by $[K[i]|i \in U]$, which is a submatrix of $K$. The vector $a$ can be recovered by (3), and the first element is the secret pixel value.

$$a = K'^{-1} \cdot s', \quad (3)$$

where $K'^{-1}$ is the inverse matrix of the recovery matrix $K'$, which is obtained by dividing the adjoint matrix $K'^* \cdot a$. The vector $s'$ consists of $k$ shared values.

In this article, we choose to encode the candidate pool using a perfect binary tree with tree height $h$. The integer form of the code word ranges in $[0, 2^h - 1]$. If we take $p = 2^h$ then the shared value space corresponds perfectly to the code word space. The size of the candidate pool (CPS) is $2^h$. So GTSS generalizes the SS scheme mentioned above and chooses the modulo to be $2^h$, so that the range of shared values will not exceed CPS, which ensures the feasibility of information embedding and the correctness of extraction. And this scheme chooses to share $l$ ($l \leq k$) secret values at a time, which means that the first $l$ elements of $a$ are secret values, and the rest $k - l$ elements are chosen from $[0, p - 1]$. Sharing $l$ secret values at a time can significantly improve the efficiency of the scheme.

### 2.2. Mapping Method of Binary Bits to Word Space

The natural language processing field typically considers text to be a sequence of words organized according to their semantic associations and syntactic properties. The chain rule of probability [20, 21] can be used to describe the joint probability distribution of word sequences, which is expressed as follows:

$$P(X) = P(x_1, x_2, \ldots, x_N),$$
$$= P(x_1)P(x_2|x_1) \cdots P(x_N|x_1, x_2, \cdots, x_{N-1}),$$
$$= \prod_{i=1}^{N} P(x_i|x_1, x_2, \cdots, x_{i-1}). \quad (4)$$

$P(X)$ is the generation probability of the word sequence $x_1, x_2, \ldots, x_N$, and $P(x_N|x_1, x_2, \cdots, x_{N-1})$ represents the conditional probability of generating word $x_N$ when $x_1, x_2, \cdots, x_{N-1}$ are given. Conditional probability measures the degree of fitness between $x_N$ and the previous text. Generally, the generated text is more reasonable if the conditional probability is higher. In general, multiple candidate words $x_N$ are available for a given string of $x_1, x_2, \cdots, x_{N-1}$, which makes the generated text conform to grammatical and semantic rules.

To achieve the mapping of secret bits to words, Yang et al. [13] proposed to use a fixed-length coding (FLC) based on the perfect binary tree and a variable-length coding (VLC) based on the Huffman tree. The FLC scheme is simpler to implement, and more time-efficient [22]. Figure 1 illustrates FLC schematically, in which we choose a candidate pool size of 4 and a perfect binary tree height of 2 for illustration. At the $t$-th time step, the FLC scheme first inputs the prefix text into LM to get the candidate words and their probability distribution at $t + 1$-th time step, and then intercepts the candidate pool of fixed size in descending probability order, and encodes the words using a perfect binary tree. After that, the codewords of word1, word2, word3, and word4 are 00, 01, 10, and 11, respectively. Therefore, the corresponding candidate word (word3) can be chosen based on the secret bits (01) to be embedded. The output words are added to the prefix for embedding the secret bits in the next time step.

Using the method discussed above, a certain number of secret bits can be carried by each word in the generated text. We first encrypt the secret value into shared values by SS, then complete the mapping from the shared values to words in shadow texts using perfect binary tree encoding.

### 2.3. Transformer-Based Controllable Text Generation

On the basis of traditional text generation, controllable text generation adds control over some key information, styles, attributes, etc., making generated text meet some of our expectations.

To create a conditional generative model, Dathathri et al. used a transformer [23] to model natural language distribution and proposed PPLM [17] to sample from $P(x|a) \propto P(a|x)P(x)$. Equation (5) summarizes the recurrent interpretation of transformers [24].

$$o_{t+1}, H_{t+1} = LM(x_t, H_t). \quad (5)$$

History matrix $H_t$ consists of key-value pairs from time-steps 0 to $t$ in the past. The $x_{t+1}$ is then sampled according to $x_{t+1} \sim P_{t+1} = \text{Softmax}(T o_{t+1})$, where $T$ is a linear transformation that maps the logit vector $o_{t+1}$ to a vector of vocabulary size.

In the next time step, $H_t$ can be adjusted to increase the more relevant words’ probability in the candidate pool. Based on the conditioned attribute model $P(a|\omega)$, history $H_t$ can be shifted toward higher log-likelihood (LL) of attribute $a$ and the distribution of unmodified language model $P(x)$. With $\Delta H_t$ being the update to $H_t$, $(H_t + \Delta H_t)$ will increase the likelihood that the generated text possesses the target attribute. The initial value of $\Delta H_t$ is zero and the attribute model $P(a|x)$ is rewritten as $P(a|H_t + \Delta H_t)$ by PPLM. Then gradient updates are made to $\Delta H_t$ by going as follows:

$$\Delta H_{t+1} = \frac{\nabla_{\Delta H_t} \log P(a|H_t + \Delta H_t)}{\nabla_{\Delta H_t} \log P(a|H_t + \Delta H_t)} \bigg|_{\nabla_{\Delta H_t} \log P(a|H_t + \Delta H_t)} \bigg|_{\nabla_{\Delta H_t} \log P(a|H_t + \Delta H_t)}.$$  

(6)
where $\beta$ indicates the update step size and $\gamma$ represents the scaling coefficient for the normalization term. This updating step can repeat $m$ times. The value of $m$ is usually between 3 and 10. Afterward, the updated logits $\tilde{\delta}_{t+1}$ are obtained by a forward pass through $\tilde{\delta}_{t+1}, H_{t+1} = \text{LM}(x_t, \tilde{H}_t)$, where $\tilde{H}_t = H_t + \Delta H_t$. Using the modified $\tilde{\delta}_{t+1}$ at time step $t + 1$, a new probability distribution $P_{t+1}$ can be generated.

GTSS chooses BoW as the attribute model to modify $H_t$. After that, the modified probability distribution $P_{t+1}$ is arranged in descending order. The first $2^k$ words are encoded by a perfect binary tree. Then select the corresponding shadow words for output according to the shared values, thus completing the generation of shadow texts that satisfy specific topics. The specific method and detailed algorithm are described in the next Section.

3. GTSS Methodology

In this section, we first define text secret sharing and introduce the basic idea of GTSS, followed by a detailed description of the sharing algorithm and the recovery algorithm, after which we analyze the applicability of GTSS.

3.1. The Basic Idea. Table 1 illustrates the main notations used in this article.

This scheme uses BoW models corresponding to specific topics as attribute models. A BoW is a set of keywords $\{\text{word}_1, \ldots, \text{word}_m\}$ that specify a topic. Equation (7) can be used to represent $\log P(a|x)$.

$$
\log P(a|x) = \log \left( \sum_{i} P_{t+1}[\text{word}_i] \right),
$$

where $P_{t+1}$ represents the conditional probability distribution for the language model output at time $t + 1$. By (6), we can calculate $\Delta H_t$ and modify $H_t$ to obtain the conditional probability distribution $P_{t+1}$.

We define text secret sharing: a secret message is shared as $n$ shadow texts; each shadow text is a natural fluent sentence and can have a specific topic. The original secret message can be recovered by any $k$ shadow texts, while less than $k$ shadow texts cannot complete the recovery.

To achieve this function, we propose using SS based on matrix theory to share the secret message, controlling the topic of shadow text by BoW, and completing the mapping of shared values to word space using perfect binary tree coding. GTSS includes two parts sharing algorithm and recovery algorithm. The sharing algorithm includes three modules: secret sharing module, mapping module, and goal programming model. And the recovery algorithm includes reconstruct module and inverse mapping module. We explain them separately below.

3.2. The Sharing Algorithm. A schematic diagram of the sharing phase is shown in Figure 2, where we consider a scheme with $(2, 2)$ threshold, $h = 2$, $l = 1$, and $p = 2^h$. We assume that the secret message is a piece of secret text, and this scheme can share any binary bits. The secret text is encoded into a binary bitstream, then sliced into several units per $h$ bit, as well as converted into secret integer values. An $n \times k$ sharing matrix $K$ is generated before sharing, which satisfies that any $k \times k$ submatrix’s determinant is odd, after which the secret values are continuously put into the first $l$ positions in $a = (a_0, a_1, \ldots, a_{k-1})^T$, and the rest $k - l$ elements’ value range is $[0, p - 1]$. The secret sharing module performs matrix multiplication to obtain the shared values $s = (s_1, s_2, \ldots, s_n)^T$. The mapping module continuously generates text using the language model. By using BoW corresponding to a specific topic, the mapping module modifies the probability distribution to increase the probability of the more topic-compatible words in the candidate pool. Then the mapping module uses a perfect binary tree to encode the candidate words, and the corresponding words are chosen based on the shared values. The goal programming model (GPM) guides all the above processes.

For different applications, we propose a goal programming model GPM-topic to optimize topic relevance and another goal programming model GPM-ppl to improve text quality. Equation (8) expresses GPM-topic.
The conditional probability distribution $P(w_i|\text{prefix}_i)$ represents the likelihood that the next word $w_i$ will be generated given the prior word prefix of the $i$-th shadow text. The probability distribution $P$ is modified by BoW to increase the relevance to topic. The constraints in GPM-topic are the operations of the secret sharing module and the mapping module. Using the perfect binary tree, the mapping module $M(\cdot)$ completes the mapping of the shared value $s_i$ to the word space generated by LM. For one set of secret values, it would not be unique for the combination of shared values because the rest $k-l$ elements of $a$ take values in $[0, p-1]$. By continuously adjusting the last $k-l$ elements of $a$, we can produce different shadow word combinations. To generate more appropriate shadow texts, GPM-topic utilizes this point to find the combination of shadow words with the largest product of probabilities, which is most relevant to the topic.

The mapping module modifies the original probability distribution $P_{t+1}$ with BoW to get $\tilde{P}_{t+1}$ which has a higher likelihood of fitting the topic. The language model is trained on thousands of natural texts to fit the natural language probability distribution. Therefore, the modification of the
method for improving shadow text quality. As shown in (10), we propose GPM-ppl as a
probability distribution affects the fluency of the generated
text, which is the price of topic control.

(9) shows the perplexity (ppl) as a metric for evaluating
generated text [25–27],
\[
ppl = P(x_1, x_2, \ldots, x_N)^{-1/N}, \\
= \prod_{i=1}^{N} \frac{1}{P(x_i|x_1, x_2, \ldots, x_{i-1})}.
\] (9)
We can see that increasing the conditional probability
decreases the perplexity and improves the quality of the
word sequence. As shown in (10), we propose GPM-ppl as a
method for improving shadow text quality.
\[
\max_{x_{i+1}} \prod_{j=1}^{n} P(w_j|\text{prefix}_j), \\
\begin{cases}
Ka = s, \\
a = (a_0, a_1, \ldots, a_{k-1})^T, \\
s = (s_1, s_2, \ldots, s_n)^T, \\
w_i = M(s_i).
\end{cases}
\] (10)
The conditional probability distribution \( P(w_j|\text{prefix}_j) \)
represents the likelihood that the next word \( w_i \) will be
generated given the prior word prefix, of the \( i \)-th shadow
text, and it is the original probability distribution obtained
by LM. The rest is the same as GPM-topic. To reduce the ppl
and improve the text quality, we have to find the combi-
nation of shadow words with the largest product of original
probabilities. In this way, each shadow word matches the
original distribution with its prior words more closely,
resulting in a reduced ppl of shadow texts. Meanwhile, this
decreases the ability to select words related to the topic and
ultimately reduces shadow text’s relevance. Therefore, GPM
selection should be based on real-world application
requirements.

Algorithm 1 shows the details of the sharing method, by
which we can generate \( n \) natural and topic-controlled
shadow texts based on the input secret bitstream. The
generated shadow texts can then be sent through open
channels carrying confidential messages with high robust-
ness and concealment.

3.3. The Recovery Algorithm. The secret message can be
recovered when \( k \) or more shadow texts are obtained.
Figure 3 shows the schematic diagram of the recovery phase.
Based on the same text generation process as in the sharing
phase, the inverse mapping module calculates the condi-
tional probability distribution for the next time step. A
perfect binary tree is used to encode the candidate pool.
In order to obtain the shared values, we need not use a sampling
strategy similar to the sharing stage to select the shadow
words, but to find the corresponding codewords through the
determined shadow words. After that, the reconstruct
module is used to put the obtained shared values into the
vector \( s' \) and multiply it with the inverse matrix of the

\[
\begin{align*}
\text{Input:} & \quad \text{Secret bitstream } B = \{0, 0, 1, \ldots, 1, 0\}; (k,n) \text{ threshold; the number of secret units to be shared at one time } k; \text{ perfect binary tree’s height } h \text{ (then } \text{CPS} = p = 2^h); \text{ the topics of each shadow text } \text{topic}_1, \ldots, \text{topic}_n; \text{ bag of words BoW}_1, \text{BoW}_2, \ldots, \text{BoW}_n \text{ related to topic}; \text{ initial words for each shadow text } \text{prefix}_1, \text{prefix}_2, \ldots, \text{prefix}_n. \\
\text{Output:} & \quad \text{n shadow texts shadow}_1, \text{shadow}_2, \ldots, \text{shadow}_n. \\
(1) & \quad \text{Slice } B \text{ per } h \text{ bits, transform each unit into integer form to get the secret values Secrets; } \\
(2) & \quad \text{Construct the sharing matrix } K, \text{ which satisfies any } k \times k \text{ submatrix’s determinant is odd; } \\
(3) & \quad \text{for each prefix, } \text{do} \\
(4) & \quad \text{Input prefix, into LM to get } H^t_i \text{ of shadow}_i; \\
(5) & \quad \text{shadow}_i ← \text{prefix}_i; \\
(6) & \quad i ← 0; \\
(7) & \quad \text{while } i ≤ \text{len (Secrets)} \text{ do } \\
(8) & \quad \text{if not } \max_{x_{i+1}} \prod_{j=1}^{n} P(w_j|\text{prefix}_j) \text{ then } \\
(9) & \quad \text{Create a vector } \alpha \text{ with the first } l \text{ values being } l \text{ secret values, and the remaining } k - l \text{ values coming from } [0, p - 1]; \\
(10) & \quad s ← Ka; \\
(11) & \quad \text{for each } s_j \text{ in } s \text{ do } \\
(12) & \quad \text{Based on BoW}_j, \Delta H^t_i \text{ can be obtained through (6); } \\
(13) & \quad H_{i+1}^t ← H_{i+1}^t + \Delta H^t_i; \\
(14) & \quad \text{Input the last word of shadow}_i \text{ and } H^t_i \text{ into LM, then } \tilde{H}_{i+1}^t, H_{i+1}^t ← \text{LM } (x_i, H^t_i); \\
(15) & \quad \tilde{P}_{i+1}^t ← \text{Softmax}(\tilde{H}_{i+1}^t); \\
(16) & \quad \text{Arrange the } \tilde{P}_{i+1}^t \text{ in descending order, encode the words using the perfect binary tree, and determine the word } w_i \text{ for output by the shared value } s_j; \\
(17) & \quad \text{else } \\
(18) & \quad \text{Add } w_i \text{ to shadow}_i; \\
(19) & \quad i ← i + 1; \\
(20) & \quad \text{return shadow}_1, \text{shadow}_2, \ldots, \text{shadow}_n.
\end{align*}
\]
reconstruction matrix to get vector $a$, whose first $l$ elements are filled by secret values. The recovery process is shown in detail in Algorithm 2. The $k$ shadow texts obtained are assumed to be the first $k$ of the $n$ shadow texts for convenience of representation.

In contrast to images or videos, texts are not compressed or distorted during transmission. Texts, therefore, have excellent robustness, making shadow texts very suitable for transmission in many scenarios. For example, shadow texts can be transmitted by instant messaging software such as Telegram and Skype, or by posting them on social media platforms such as Twitter and Facebook. Then, the receiver can obtain the shadow texts by browsing and copying through the above platforms, using the recovery algorithm to get the secret message.

The topic of each shadow text does not need to be transmitted but can be obtained from the shadow text itself. The topic can be identified artificially by the semantics of the shadow text, or it can be determined by which BoW has the highest number of words in the text.

3.4. Theoretical Analysis. First, we analyze the algorithm’s input and output text length.

Since GTSS uses GPT-2 [28] as the generation model, the most extended sequence length it can process at one time is 1024, so the upper limit of the length of the most extended input and output text supported is 1024. Therefore, the most extended length of both input prefix text and output shadow text is 1024.

The design idea of GTSS is that $l$ consecutive secret units are shared into $n$ shadow values, each corresponding to a word in $n$ shadow texts. When all the secret units are shared, the generation of shadow text is ended. Assuming that the length of the secret bitstream $B$ to be shared is $|B|$, then $B$ is divided into $(|B|/h)$ secret units after being split, and GTSS shares $l$ secret units at a time, so a total of $(|B|/h \cdot l)$ sharing process will be performed, and the shadow text will generate $(|B|/h \cdot l)$ words later. The actual length of the final generated shadow text is the length of the prefix text plus the number of subsequent generated words, that is, $|\text{prefix}| + (|B|/h \cdot l)$.

Then we discuss the computational complexity of the algorithm. Since this article proposes a secret sharing scheme, the analysis focuses on the secret sharing module and reconstruction module for the complexity calculation.

Each sharing process is multiplied by an $n \times k$ matrix and a $k \times 1$ matrix, as shown in (2), and a total of $(|B|/h \cdot l)$ sharing processes will be performed, so the time complexity of the secret sharing module in GTSS is $O(k \cdot n \cdot 1 \cdot (|B|/h \cdot l))$. In GTSS, $n = k$ or $n = k + 1$, so the time complexity is $O(k^2 \cdot |B|/h \cdot l)$.

In the part of the shadow text that removes the prefix, one word corresponds to one shared value. So $k$ shared values can be extracted from the words at the corresponding positions in the $k$ shadow texts, and then one unit of the secret message can be recovered. The number of words corresponding to shared values in a shadow text is $(|B|/h \cdot l)$, and GTSS uses matrix multiplication in the recovery phase, a $k \times k$ matrix is multiplied by a $k \times 1$ matrix, as shown in (3), so the time complexity of the reconstruction module is $O(k \cdot k \cdot 1 \cdot (|B|/h \cdot l)) = O(k^2 \cdot |B|/h \cdot l)$.

3.5. Application Analysis. The GTSS scheme proposed in this article has two main application scenarios, multi-channel transmission and access control for the secret message. The application scenario diagrams are shown in Figure 4, where $ST_i$ represents the $i$-th shadow text.

Considering that social networks are public channels, and each social platform has staff monitoring it, a suspicious account might be deleted or banned. Whenever the transmission encounters the above scenario, the secret message will be lost. Therefore, we can use GTSS to share the secret message as multiple shadow texts with different topics and transmit them through different social accounts or even social platforms. Thanks to the loss tolerance feature of secret sharing, even if some of the shadow texts are lost due to abnormal reasons, the secret message can be reconstructed when the receiver gets $k$ shadow texts.
In the traditional secret image sharing scheme, $n$ participants hold $n$ shadow images, where $k$ or more participants use the shadow images in hand to recover the secret message. Since an image is a digital media, a storage device is needed to keep it. In contrast, the shadow of GTSS is in the form of text, whose carrier can be a simple paper or even the participants' memory, so it is not easily limited by the storage device and is easy to remember and manage.

4. Experiments

This section evaluates the proposed GTSS scheme regarding shadow text quality, topic relevance, and anti-detection capability, conducts an ablation study to verify each module’s effectiveness, and compresses it with text steganography schemes regarding embedding rate and perplexity.

4.1. Experimental Setup

4.1.1. Dataset. The "Microsoft Coco" [29] dataset for object recognition, captioning, and segmentation, is used to evaluate GTSS’ performance. Our corpus is the portion of the dataset used for image captions, which contains 591,753 sentences.

4.1.2. Evaluation Indicators. Perplexity, as shown in (9), is used to evaluate the fluency of shadow text. The smaller ppl indicates that the closer the statistical distribution of the
generated shadow text and the natural text, the higher the text quality. Since we use BoW as the attribute model to control the topic by adjusting the conditional probability distribution at each time step, we evaluate topic relevance (TR) to topic $c_i$ using the percentage of words belonging to BoW, in the shadow text, as shown in the following equation:

$$TR_i = \frac{N_{\text{Bow},i}}{N} \times 100\%,$$

(iii) The shadow texts of BP method optimized by GPM-ppl possess the smallest complexity and highest quality. In comparison with both BT and BTT, the GPM-ppl optimized BTP method has a lower perplexity.

(iv) The shadow texts of BTT method optimized by GPM-topic possess the highest topic relevance.

4.4. Comparative Experiment. Although we design a text secret sharing scheme in this article, the shared value is mapped to word space in generating shadow text, which inevitably affects the normal text generation process so that it will have a certain impact on the concealment of shadow text. In this section, to examine the concealment of shadow text, we conduct experiments on GTSS and two classical text steganography schemes Bins [25] and FLC [13] in terms of embedding rate (ER) and perplexity (ppl).

The embedding rate is the average number of effective secret bits carried by each word in the text. The Bins scheme divides the word space into blocks and then encodes the blocks. In the text generation process, the corresponding block is determined according to the secret bitstream, and the appropriate word is selected for output, thereby completing the embedding of the secret bits. Therefore, the ER of the Bins scheme is related to the size of block block.size. The larger the block.size, the more secret bits each word can carry, and the larger the ER, and ER = block.size. The FLC scheme performs perfect binary tree coding on the candidate pool and then output the codeword according to the secret bits, so ER = tree_height. GTSS will share l units of the secret message (a total of $l \times h$ bits) into n shared values at a time, and the length of each unit of the secret message is h bits, and then map the shared values to the word space through the perfect binary tree. Each word in the shadow text corresponds to a shared value, so ER = $h \cdot (l/n)$ in GTSS.

To reflect the effect of parameters $k, n, l$, and $h$ in GTSS, we conducted three sets of experiments on the two variants of GTSS, GTSS-BTP, and GTSS-BTT, respectively. The parameters were selected as $k = 2, n = 2, l = 1, h = 2, 4, 6, 8, 10$; $k = 4, n = 4, l = 3, h = 1, 2, \ldots, 7$; and $k = 6, n = 6, l = 5, h = 1, 2, \ldots, 6$. For the Bins scheme, we choose block.size = 1, 2, \ldots, 5. For the FLC scheme, we choose tree_height = 1, 2, \ldots, 5. The experimental results are shown in Tables 9–11. Since the embedding rate of each scheme cannot be accurately matched, we draw two line graphs as shown in Figure 5 for a more intuitive display.

It can be concluded from the tables and figures that for the same scheme, with the increase in embedding rate, perplexity tends to increase, and the quality of the text continues to decline. For GTSS and the FLC scheme, with the increase of $h$ and tree_height, the candidate word space will become larger, and the possibility of selecting words with small conditional probability becomes larger, so the overall perplexity will increase. For the Bins scheme, as the block.size increases, the number of words in each block will decrease, resulting in no words matching the previous text being selected, so the text quality will decrease.

For the two variants of GTSS, GTSS-BTP and GTSS-BTT, under the same embedding rate, the higher the

4.2. Some Examples. The hyperparameters of GTSS include $(k, n, l, \text{topic}, h, \text{prefix})$. Below we show some examples of shadow texts when these parameters are denoted as different values (Tables 2–5). The secret text to share is “Secret message.” The target topics of the shadow texts are colored and bracketed (e.g., [science]). The words of BoW are highlighted brightly (e.g., evolution). A softer highlight is used for words related to the topic but not in BoW (e.g., brain). The prefix of every sentence is underlined (e.g., It has been shown).

To further demonstrate the scalability of the shadow text generated by GTSS in terms of topic control, we add a sample of the $(2, 2)$ threshold, where the topics of the first shadow text are restricted to “space” and “military,” and the topics of the second shadow text are restricted to “technology” and “science.” As shown in Table 6, the shadow texts generated by GTSS can satisfy multiple topics at the same time.

4.3. Ablation Study. An ablation study was conducted on five variants: B: the baseline with no topic control method, and $a_1 \sim a_{n-1}$ are random selected; BP: the variant with no topic control method under the constraint of GPM-ppl; BT: the variant with topic control method, and $a_1 \sim a_{n-1}$ are random selected; BTP: the GTSS scheme with topic control method under the constraint of GPM-ppl; and BTT: the GTSS scheme with topic control method under the constraint of GPM-topic.

In order to identify the average perplexity and topic relevance of each shadow text, we randomly select sentences from “Microsoft Coco” as the secret texts. Tables 7 and 8 show the experimental results.

Following are the conclusions we can draw from the above experimental results.

(i) Through GTSS’s topic control mechanism, words that match a specific topic are more likely to be selected when creating shadow texts, resulting in shadow texts that satisfy the topic.

(ii) Because of the topic control means, the modified probability distribution does not match the training samples, so the BT method without the optimization of the goal programming model has the poorest text quality. The hyperparameters of GTSS include $(k, n, l, \text{topic}, h, \text{prefix})$. Below we show some examples of shadow texts when these parameters are denoted as different values (Tables 2–5). The secret text to share is “Secret message.” The target topics of the shadow texts are colored and bracketed (e.g., [science]). The words of BoW are highlighted brightly (e.g., evolution). A softer highlight is used for words related to the topic but not in BoW (e.g., brain). The prefix of every sentence is underlined (e.g., It has been shown).

To further demonstrate the scalability of the shadow text generated by GTSS in terms of topic control, we add a sample of the $(2, 2)$ threshold, where the topics of the first shadow text are restricted to “space” and “military,” and the topics of the second shadow text are restricted to “technology” and “science.” As shown in Table 6, the shadow texts generated by GTSS can satisfy multiple topics at the same time.

4.3. Ablation Study. An ablation study was conducted on five variants: B: the baseline with no topic control method, and $a_1 \sim a_{n-1}$ are random selected; BP: the variant with no topic control method under the constraint of GPM-ppl; BT: the variant with topic control method, and $a_1 \sim a_{n-1}$ are random selected; BTP: the GTSS scheme with topic control method under the constraint of GPM-ppl; and BTT: the GTSS scheme with topic control method under the constraint of GPM-topic.

In order to identify the average perplexity and topic relevance of each shadow text, we randomly select sentences from “Microsoft Coco” as the secret texts. Tables 7 and 8 show the experimental results.

Following are the conclusions we can draw from the above experimental results.

(i) Through GTSS’s topic control mechanism, words that match a specific topic are more likely to be selected when creating shadow texts, resulting in shadow texts that satisfy the topic.

(ii) Because of the topic control means, the modified probability distribution does not match the training samples, so the BT method without the optimization of the goal programming model has the poorest text quality.
Table 2: Shadow texts of “secret message” when the threshold is $\left(2, 2\right)$, $l = 1$, and $h = 3$.

| shadow$_1$ | More importantly, we have been working on this for a long period of time, but the last few months have been a battle between the government and the army. We have been fighting the army and |
| shadow$_2$ | It has been shown time and again that the brain is not only involved in memory processes but also in learning and memory consolidation, which is a key process for human brain structure evolution (Bertsch et al) |
| shadow$_3$ | Views on abortion rights in general are generally favorable, and support for laws requiring a court challenge to state laws is low in a national law-law environment, according to the law firm representing a legal |

Table 3: Shadow texts of “secret message” when the threshold is $\left(2, 3\right)$, $l = 1$, and $h = 3$.

| shadow$_1$ | More importantly, this will help us to develop the tools for fighting terrorism in the Middle East. |
| shadow$_2$ | It has been shown experimentally that the amount (in mmol) of calcium in the serum of young healthy men is about 10 to 25 per kilobase per day higher with the presence or absence of calcium |
| shadow$_3$ | Views on abortion rights in general are generally favorable, and support for laws requiring a court challenge to state laws is low in a national law-law environment, according to the law firm representing a legal |

Table 4: Shadow texts of “secret message” when the threshold is $\left(3, 3\right)$, $l = 2$, and $h = 3$.

| shadow$_1$ | The connection between the two worlds was made by the great sea-giant planet of Saturn in the |
| shadow$_2$ | To review, there are two major ways to determine the tax treatment of a business: by statute of |
| shadow$_3$ | In brief, it is believed that the ancient Roman Empire was ruled through a monarchy. The Roman Emperor |

Table 5: Shadow texts of “secret message” when the threshold is $\left(3, 4\right)$, $l = 2$, and $h = 3$.

| shadow$_1$ | The connection between the United Nation Earth Day 2018 and the upcoming International Earth Observed satellite was made by |
| shadow$_2$ | To review, the U.S. Constitution gives citizens the right to bear arms. The U. |
| shadow$_3$ | In brief This paper reviews how a novel strategy to address social inequality has been applied at a global scale |
| shadow$_4$ | The key aspect that separates the best Android applications from the best iOS or Windows based app, is the developer |

Table 6: Shadow texts of “secret message” when the threshold is $\left(2, 2\right)$, $l = 1$, and $h = 2$.

| shadow$_1$ | The connection that I have made in writing and writing on this blog and the comments and suggestions I have gotten on Twitter are very similar to one another (I think that is a fact of nature that you can learn by observing others and comparing the comments you get with the ones you write) I will try my hardest on this website for research articles |
| shadow$_2$ | To review and analyze the evidence of an alleged conspiracy among the CIA and its satellite programs in a way to create and promote global hegemony is not only unethical but potentially deadly as satellite warfare technology and capabilities could cause the loss of life on space space stations as satellites are launched at high velocity into high alt space space debris orbits. This space-space |
threshold \((k, n)\), the better the text quality. This is because \(ER = h \cdot (l/n)\) in GTSS, we choose \(l = n - 1\) in the experiment, and \(\lim_{n \to \infty} h \cdot (l/n) = \lim_{n \to \infty} h \cdot (n - 1/n) = h\), so the higher the threshold, the closer the embedding rate is to \(h\). When \(h\) is equal, that is, when the size of candidate word space is equal, the high-threshold GTSS scheme has a higher embedding rate, so the text generated by the high-threshold scheme is of higher quality under the same ER.

The quality of the text generated by GTSS-BTP is better than that of the Bins scheme under various thresholds. Under the \((4, 4)\) and \((6, 6)\) thresholds, when \(ER\) is relatively large, the text quality of GTSS-BTP is better than that of the FLC scheme. At the \((2, 2)\) threshold of GTSS-BTT, the text quality drops sharply when the \(ER\) is greater than 4. The text quality of GTSS-BTT still outperforms the Bins scheme at the high threshold and is not much different from the FLC scheme.

GTSS has an advantage over existing text steganography schemes in terms of the ppl of the generated text at higher thresholds under the same embedding rate.
4.5 Evaluation of Anti-Detection. In addition, we tested the anti-detection capability of GTSS. The text steganalysis algorithm TS-CSW [30] is used to classify the generated steganographic sentences, which uses convolutional sliding windows with multiple sizes to extract relevant features. These subtle variations in the distribution of relevant features can be used for text steganalysis. We conducted experiments for Bins, FLC, and several variants of GTSS, respectively. We chose to compare the anti-detection capability of each scheme at approximately the same embedding rate (ER ≈ 3), so that block_size = 3 for the Bins scheme, tree_height = 3 for the FLC scheme, h = 6 for the (2, 2) threshold GTSS, h = 4 for the (4, 4) threshold GTSS, and h = 4 for the (6, 6) threshold GTSS, at which point Bins, FLC, GTSS-(2, 2), and GTSS-(4, 4) schemes all have the embedding rate of 3 bits/word, and GTSS-(6, 6) has an embedding rate of 3.33 bits/word.

We choose Accuracy, Precision, Recall, and F1-score, standard evaluation metrics for binary classification models, to evaluate the detection resistance of the generated text, as shown in equations (12)–(15).

\[
\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \quad (12)
\]

\[
\text{Precision} = \frac{TP}{TP + FP}, \quad (13)
\]

\[
\text{Recall} = \frac{TP}{TP + FN}, \quad (14)
\]

\[
\text{F1}\_\text{score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \quad (15)
\]

TP is positive sample success prediction, TN is negative sample success prediction, FP is negative sample error prediction, and FN is positive sample error prediction. The closer the accuracy of the scheme is to 0.5, or the smaller the values of Precision, Recall, and F1-score, the more resistant the generated steganographic text or shadow text is to detection. The related results are shown in Table 12.

We can see that the Accuracy of Bins, FLC, and GTSS is close to 50%, indicating that these schemes have some

|                | Accuracy (%) | Precision (%) | Recall (%) | F1_score (%) |
|----------------|--------------|---------------|------------|-------------|
| Bins (ER = 3)  | 52.70        | 53.51         | 73.84      | 56.90       |
| FLC (ER = 3)   | 51.11        | 32.19         | 46.36      | 33.66       |
| GTSS-BTP       |              |               |            |             |
| \(k = 2, n = 2, l = 1, h = 6, ER = 3\) | 51.01        | 30.06         | 33.88      | 25.90       |
| \(k = 4, n = 4, l = 3, h = 4, ER = 3\) | 51.38        | 18.77         | 18.18      | 16.50       |
| \(k = 6, n = 6, l = 5, h = 4, ER = 3.33\) | 51.10        | \textbf{14.26} | 29.66      | 19.26       |
| GTSS-BTT       |              |               |            |             |
| \(k = 2, n = 2, l = 1, h = 6, ER = 3\) | 52.08        | 31.26         | 19.91      | 16.46       |
| \(k = 4, n = 4, l = 3, h = 4, ER = 3\) | 51.84        | 24.29         | 19.00      | 15.66       |
| \(k = 6, n = 6, l = 5, h = 4, ER = 3.33\) | 52.14        | 20.87         | 19.72      | 16.37       |

4.5. Evaluation of Anti-Detection. In addition, we tested the anti-detection capability of GTSS. The text steganalysis algorithm TS-CSW [30] is used to classify the generated steganographic sentences, which uses convolutional sliding windows with multiple sizes to extract relevant features. These subtle variations in the distribution of relevant features can be used for text steganalysis. We conducted experiments for Bins, FLC, and several variants of GTSS, respectively. We chose to compare the anti-detection capability of each scheme at approximately the same embedding rate (ER ≈ 3), so that block_size = 3 for the Bins scheme, tree_height = 3 for the FLC scheme, h = 6 for the (2, 2) threshold GTSS, h = 4 for the (4, 4) threshold GTSS, and h = 4 for the (6, 6) threshold GTSS, at which point Bins, FLC, GTSS-(2, 2), and GTSS-(4, 4) schemes all have the embedding rate of 3 bits/word, and GTSS-(6, 6) has an embedding rate of 3.33 bits/word.

We choose Accuracy, Precision, Recall, and F1-score, standard evaluation metrics for binary classification models, to evaluate the detection resistance of the generated text, as shown in equations (12)–(15).
resistance to detection. However, in the performance of the remaining three metrics Precision, Recall, and F1-score, the variants of GTSS are better than Bins and FLC, which indicates that the shadow text generated by GTSS has better resistance to detection than the Bins and FLC schemes.

5. Conclusions

A text secret sharing scheme is proposed in this article, where the secret message is divided as $n$ topic-controlled and fluent shadow texts, and any $k$ shadow texts can reconstruct the secret message. First, we encrypt the secret message using matrix theory to get shared values. Then we use BoW to modify the conditional probability distribution in order to increase the probability of words meeting the topic. Shadow texts are generated by mapping the shared values into word space using the perfect binary tree. Most importantly, we propose two goal programming models that deeply integrate secret sharing, encoding, and controllable text generation techniques. The two GPM can enhance the fluency and topic relevance of the shadow text, respectively. We discuss two application scenarios of this scheme: multi-channel transmission and access control of the secret message. Our experimental section illustrates the effectiveness of GTSS through examples and ablation studies. Comparative and anti-detection experiments show that the text generated by GTSS has good quality and anti-detection ability. In the meantime, there are still some deficiencies in the proposed scheme, which need to be addressed in the future.

(i) The limitation of the SS scheme used in GTSS itself, which can only satisfy the $(k,k)$ and $(k,k+1)$ thresholds, leads to the limitation of the current GTSS scheme in the values of the threshold parameters. The operations in the finite field $GF(p^k)$ are polynomial operations, which can avoid the restriction that the modulo number of operations must be prime in the prime field $GF(p)$, and using it in GTSS may extend the threshold parameters.

(ii) A modified or deleted word in a shadow text will cause GTSS to fail to find the corresponding word in its candidate pool at a certain point, which will further affect the extraction of shared values, as well as the recovery of the secret message. There is an urgent need to improve shadow text’s ability to withstand word modification or deletion attacks.

Data Availability

The BoW we used is available at https://github.com/yuxiaoxiaoachun/GTSS.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was funded by the National Natural Science Foundation of China (Grant Number: 62271496). This article is an extension of the conference paper [31]. The authors discuss it in terms of text secret sharing and add a lot of theoretical analysis and experiments.

References

[1] C. E. Shannon, “Communication theory of secrecy systems,” Bell System Technical Journal, vol. 28, no. 4, pp. 656–715, 1949.
[2] A. Shamir, “How to share a secret,” Communications of the ACM, vol. 22, no. 11, pp. 612–613, 1979.
[3] G. Robert Blakley, “Safeguarding cryptographic keys,” in Proceedings of the National Computer Conference, pp. 313–317, IEEE Computer Society, New York, USA, June 1979.
[4] R. Ciegis, V. Starikovičius, N. Tumanova, and M. Ragulskis, “Application of distributed parallel computing for dynamic visual cryptography,” The Journal of Supercomputing, vol. 72, no. 11, pp. 4204–4220, 2016.
[5] P. Palevicius and M. Ragulskis, “Image communication scheme based on dynamic visual cryptography and computer generated holography,” Optics Communications, vol. 335, pp. 161–167, 2015.
[6] X. Yan, Y. Lu, L. Liu, S. Wan, W. Ding, and H. Liu, “Exploiting the homomorphic property of visual cryptography,” International Journal of Digital Crime and Forensics, vol. 9, no. 2, pp. 45–56, 2017.
[7] X. Yan, Y. Lu, L. Liu, and X. Song, “Reversible image secret sharing,” IEEE Transactions on Information Forensics and Security, vol. 15, pp. 1–3858, 2020.
[8] C.-C. Lin and J.-C. Lin, “Secret image sharing,” Computers & Graphics, vol. 26, no. 5, pp. 765–770, 2002.
[9] C.-C. Lin and W.-H. Tsai, “Secret image sharing with steganography and authentication,” Journal of Systems and Software, vol. 73, no. 3, pp. 405–414, 2004.
[10] Y. Ke, M. Zhang, X. Zhang, J. Liu, T. Su, and X. Yang, “A reversible data hiding scheme in encrypted domain for secret image sharing based on Chinese remainder theorem,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 4, pp. 2469–2481, 2022.
[11] C.-N. Yang, X. Wu, M.-J. Chung, and X. Zhang, “Ambtc-based secret image sharing by simple modular arithmetic,” Journal of Visual Communication and Image Representation, vol. 84, Article ID 103482, 2022.
[12] X. Zhang, F. Peng, and M. Long, “Robust coverless image steganography based on dct and lida topic classification,” IEEE Transactions on Multimedia, vol. 20, no. 12, pp. 3223–3238, 2018.
[13] Z.-L. Yang, X.-Q. Guo, Z.-M. Chen, Y. F. Huang, and Y. J. Zhang, “Rnn-stega: linguistic steganography based on recurrent neural networks,” IEEE Transactions on Information Forensics and Security, vol. 14, no. 5, pp. 1280–1295, 2019.
[14] Z. Hu, Z. Yang, X. Liang, R. Salakhutdinov, and E. P. Xing, “Toward controlled generation of text,” in International Conference on Machine LearningPMLR, Sydney, NSW, Australia, 2017.
[15] A. Chan, Y.-S. Ong, B. Pung, A. Zhang, and J. Fu, “Cocoon: A self-supervised approach for controlled text generation,” in Proceedings of the International Conference on Learning Representations, Addis Ababa, Ethiopia, 2020.
[16] R. Zellers, A. Holtzman, H. Rashkin et al., “Defending against neural fake news,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[17] S. Dathathri, A. Madotto, J. Lan et al., “Plug and play language models: a simple approach to controlled text generation,” in *Proceedings of the International Conference on Learning Representations*, Orleans, LA, USA, 2019.

[18] M. Rosulek, “Secret Sharing,” *The Joy of Cryptography OE*, Oregon State University, no. 1st, Corvallis, OR, USA, 2017.

[19] L. Yu, L. Liu, Z. Xia, X. Yan, and Y. Lu, “Lossless and efficient secret image sharing based on matrix theory modulo 256,” *Mathematics*, vol. 8, no. 6, p. 1018, 2020.

[20] C. Manning and H. Schutze, *Foundations of Statistical Natural Language Processing*, MIT press, Cambridge, MA, USA, 1999.

[21] Y. Bengio, R. Ducharme, and P. Vincent, “A neural probabilistic language model,” *Advances in Neural Information Processing Systems*, vol. 13, 2000.

[22] L. Yu, Y. Lu, X. Yan, and Y. Yu, “Mts-stega: linguistic steganography based on multi-time-step,” *Entropy*, vol. 24, no. 5, p. 585, 2022.

[23] A. Vaswani, N. Shazeer, N. Parmar et al., “Attention is all you need,” *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[24] T. Wolf, L. Debut, V. Sanh et al., “Transformers: state-of-the-art natural language processing,” in *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pp. 38–45, Demos, 2020.

[25] F. Tina, J. Martin, and K. Argyraki, “Generating steganographic text with lstms,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics-Student Research Workshop*, pp. 100–106, Graz, Austria, 2017.

[26] T. Mikolov, M. Karafiát, L. Burget, C. Jan, and S. Khudanpur, “Recurrent neural network based language model,” in *Proceedings of the 11th Annual Conference of the International Speech Communication Association*, vol. 2, pp. 1045–1048, Makuhari, Chiba, Japan, 2010.

[27] D. Jurafsky, *Speech & Language Processing*. Pearson Education India, Noida, UP, India, 2000.

[28] A. Radford, J. Wu, R. Child, D. Luan, A. Dario, and S. Ilya, “Language models are unsupervised multitask learners,” *OpenAI blog*, vol. 1, no. 8, 2019.

[29] T.-Y. Lin, M. Maire, S. Belongie et al., “Microsoft coco: common objects in context,” in *European conference on computer vision* Springer, New York, NY, USA, 2014.

[30] Z. Yang, Y. Huang, and Yu-J. Zhang, “Ts-csw: text steganalysis and hidden capacity estimation based on convolutional sliding windows,” *Multimedia Tools and Applications*, vol. 79, no. 25-26, Article ID 18293, 2020.

[31] Y. Long, Y. Lu, X. Yan, and X. Wang, “Generative text steganography via multiple social network channels based on transformers,” in *Natural Language Processing and Chinese Computing*, W. Lu, S. Huang, H. Yu, and X. Zhou, Eds., Springer International Publishing, New York, NY, USA, 2022.