Artificial Intelligence 317 (2023) 103859

Contents lists available at ScienceDirect

Artificial Intelligence

journal homepage: www.elsevier.com/locate/artint

Natural language watermarking via paraphraser-based lexical substitution

Jipeng Qiang [a], Shiyu Zhu [a], Yun Li [a,*], Yi Zhu [a], Yunhao Yuan [a], Xindong Wu [b,c]

[a] School of Information Engineering, Yangzhou University, Yangzhou, 225127, Jiangsu, China
[b] Key Laboratory of Knowledge Engineering with Big Data (Hefei University of Technology), Ministry of Education, Hefei, 230009, Anhui, China
[c] College of Computer Science and Information Technology, Hefei University of Technology, Hefei, 230009, Anhui, China

ARTICLE INFO

Article history:
Received 28 July 2022
Received in revised form 9 January 2023
Accepted 11 January 2023
Available online 16 January 2023

Keywords:
Natural language watermarking
Lexical substitution
Paraphrase generation

ABSTRACT

Although powerful pretrained language models generate high-quality output text, they bring new concerns about the potential misuse of such models for malicious purposes. Natural language watermarking (NLW) is a technique that is desgined to help tracing the provenance of texts for againsting possible attacks, where the watermark signals are embedded into cover texts using synonym substitutions. The up-to-date BERT-based NLW methods have made remarkable progress on performance improvement of watermarking through generating substitutes for a masked target word. Yet, the BERT-based NLWs focus on the context of texts rather than the meaning of target words, which might make the capacity of watermark embeddings being lower.

To address the limitations, this study proposes a novel NLW method by incorporating a paraphraser-based lexical substitution method. Under the promise of paraphrase preservation, the proposed NLW method utilizes the knowledge of paraphrase modeling to generate the substitute candidates to replace the words in original sentences capable of carrying the watermark signal in local contexts. We empirically show that our NLW method not only has a better meaning-preserved, but improves the payload more than 2 times compared with the BERT-based NLW method. Besides, compared with previous state-of-the-art method Compared with other state-of-the-art baselines, the experimental results show that the proposed LS method improves the Precision@1 score from 51.7% to 58.3% and from 50.5% to 62.6% on LS07 and CoInCo benchmarks, respectively.

1. Introduction

By taking the advantages of powerful pretrained language models like T5 [31] and GPT-3 [5], natural language generation has made remarkable progress in generating fluent and realistic text, which raises a concern about the misusing of the such large-scale models for potential malicious purposes, such as the propagation of the fake news and misinformation produced by machines. The such disinformation or misinformation is able to be detected by tracing text provenance to claim the ownership of text contents or identify the malicious users who spread such information.

Watermarking is one of the existing prominent techniques on tracing text provenance. It is implemented by covertly embedding watermark signals into an object (image, audio, text) that is helpful to track the ownership of the object. Due

* Corresponding author.
E-mail addresses: jpqiang@yzu.edu.cn (J. Qiang), liyun@yzu.edu.cn (Y. Li).

https://doi.org/10.1016/j.artint.2023.103859
0004-3702/

to texts are inherently discrete, it is a challenge to embed watermarks with imperceptible perturbations into text [1,41]. Previous text watermarking methods try to slightly change the image features like text format [4,32] and fonts [40,27] to insert watermarks. But they are vulnerable to cross-media transmissions like OCR. Due to the semantic information of texts is leveraged, the natural language watermarking (NLW) method has inherent robustness in OCR-style transmissions [37].

The early work of NLW utilizes linguistic databases (e.g., WordNet) [37,8] to substitute words with their synonyms. These methods do not consider the context of target words while generating substitute candidates, which inevitably fail to preserve the sentence's meaning. An adversarial watermarking transformer method based on the encoder-decoder framework is developed to learn how to replace the unobtrusive words (e.g., articles, prepositions, conjunctions) in the input sequence with other inconspicuous words [1]. Although the method can minimal the changes on semantics and maintain the correctness of input texts, the such replacements still impair the logical and semantic coherence of texts.

Recently, a BERT-based NLW method [41] utilizes a BERT-based lexical substitute method [29] to generate substitution candidates. Compared with linguistic databases, BERT entirely or partially masks the target word of original text to generate corresponding substitute candidates. Nevertheless, BERT trained on raw data can not pertinently learn the operation of lexical substitution. There are many words that only fit for the context without considering the meaning of the target word are generated as the substitutes leading to a low watermark embedding capacity. The experimental results show that its payload is lower than 0.1, where the payload is the average amount of information that one single word can hold.

To address those limitations mentioned above, we propose a novel lexical substitution (LS) method based on a paraphraser, and utilize it to design our watermarking method. Considering the generated paraphrases from a paraphraser contain variations in word choice and preserve the sentence's meaning, our idea is whether we can extract the substitute candidates from the paraphrases of original sentence. Since beam decoding concerns the lexical variations of the whole sentence instead of the target word, there are no sufficient appropriate substitutions can be discovered for the target word if we directly extract the substitute candidates from the paraphrases using the beam decoding. To generate more appropriate substitutions, we propose a simple decoding strategy that focuses on lexical variations of the target word during the decoding process. We initially force the decoder to begin with the prefix words of the target word for generating the probability distribution of the target word's position. Then, the words that have highest probabilities are permanently fixed when decoding the rest of the words. We will choose some words of the target word's position in the paraphrases as substitute candidates. In this way, the substitutes are not only semantically consistent with the target word and fit in the context, but also can preserve the sentence's meaning.

Considering the proposed LS method is a context-aware method, it is a challenge to decide which words in a sentence should be to embedded with watermark signals when designing the NLW method. Specifically, the proposed NLW method consists of an embedding stage and an extraction stage. We first generate watermarked sentence by embedding watermarks using our LS method, and extract watermark signals from the watermarked sentence. Therefore, it is necessary to guarantee that the substitutes obtained from the original sentence and watermarked sentence are identical for successfully completing watermark embedding and extraction. Therefore, we design a exchangeability test to search and substitute the words capable of carrying watermark signals, where the exchangeability test needs the embedding and extraction sides to generate identical candidate sets.

The contributions of our paper are as follows:

(1) We propose a novel LS method based on a neural paraphraser. With the help of the paraphraser, we further propose a simple decoding strategy that focuses on lexical variation of the target word during decoding. The experiments conducted on two popular lexical substitution datasets (LS07 and CoInCo) show that ParaLS significantly outperforms the state-of-the-art BERT-based LS methods.

(2) We propose a novel strategy for ranking candidates. To judge whether the candidates are reasonable, we rank them based on the change of the sentence's meaning after embedding them into original sentences. We adopt a text generation evaluation metric BARTScore [42] to compute the relationship between original sentences and corresponding updated sentences. We found that BARTScore is very suitable for ranking candidates, and it outperforms the previous state-of-the-art methods when the same substitution candidate lists are produced from three popular LS datasets.

(3) We design a novel NLW method based on the proposed LS method via an exchangeability test without relying on any annotated data. Compared with the newest BERT-based NLW method, our method not only has the better performance on meaning preservation, but also improves the payload by more than two times.

2. Related work

Lexical Substitution. Traditional lexical substitution (LS) methods utilize linguistic databases (e.g., WordNet) [9,43] or word embedding models [22,20] to extract synonyms or high-similar words for a target word, and then sort them based on their appropriateness in context. These methods do not consider the context of the target word while generating substitute candidates, thereby inevitably producing massive spurious candidates that may confuse the subsequential ranking phase.

Recent LS methods based on pretrained language models have attracted much attention [44,16,24,28,30], in which the pretrained BERT is the most widely used one. Zhou et al. [44] apply dropout to the embeddings of target words for partially obscuring a word, and obtain a probability distribution over the BERT output vocabulary. Michalopoulos et al. [24] propose a new mix-up embedding strategy by incorporating the knowledge of WordNet into the prediction process of BERT.

Although LS methods based on pretrained language models have made remarkable progress, the shortage of large-scale annotated data blocks the potential benefits that they could bring to this field. To fill this gap up, Lacerra et al. [17,16] tried to train pretrained language models from two aspects. They [17] first merge the development set of two LS datasets (CoInCo and TWSI), and split it into training and development set for fine-tuning an encoder-decoder framework. They propose a novel approach to automatically create large-scale LS dataset, and fine-tune the BERT [16]. By leveraging the latent representations of texts, they retrieve the new contexts that are similar to the context of target words, and then uses the contexts to find meaningful substitutes. Even so, the experiments evidenced from the two works only slightly outperform unsupervised BERT-based LS method [44]. The such situation is caused by that the size of development set limits the applicability of BERT, and the created LS dataset contains a lot of noise.

Overall, pretrained language modeling-based LS methods consider contextual information of target words when generating substitute candidates, but do not concern the impact of applying substitutes on sentence meaning. In contrast to the above methods, we try to utilize the knowledge of a pretrained paraphraser to generate substitute candidates.

Lexical Substitution using Paraphrases. A few studies [25,14] find substitute candidates for complex words from a large-scale paraphrase rule database, e.g., PPDB [7] or its variations [26,25]. A paraphrase rule database consists of large-scale lexical paraphrase rules (e.g., "berries → strawberries") that are extracted from large-scale paraphrase sentence pairs, such as ParaNMT [39] or ParaBank [11]. These methods never consider contexts as linguistic resource-based LS methods. In this paper, we generate substitute candidates of target words using the pretrained paraphrase modeling instead of paraphrase rule databases or paraphrase databases.

Language Watermarking. Watermarking for multimedia documents is applicable to many applications such as identifying and protecting authorship.

Natural language watermarking (NLW) embeds watermarks by changing the syntactic and semantic nature of cover texts without affecting the original meaning of the texts [36,37]. NLW embraces two stages including watermark embedding and decoding, where the embedding stage encodes the watermark information to generate the watermarked text and the decoding stage recovers the watermark from the watermarked text. One early work is the synonym substitution method [37,8] that employs WordNet to search synonyms of words for encoding messages. However, they neglect the context while generating the substitute words of target words, which will inevitably produce a large number of spurious substitutes, thereby significantly harming the original meaning of the text.

Recently, a transformer-based method trained on a specific dataset learns to embed watermarks into unobtrusive words (e.g., articles, prepositions, and conjunctions) [1]. However, the such replacements still impair the logical and semantic coherence of text, since these selected words often convey specific grammatical or semantic information by forming phrases with their adjacent words. A BERT-based NLW method [41] is proposed which uses pretrained language modeling BERT to generate the candidates. Compared with WordNet, BERT generates the candidates directly based on the context of the target word. However, BERT trained on raw data has not pertinently learned the operation of lexical substitution. Therefore, the words that only fit for the context but without considering the meaning of the target word are generated as the candidates.

Decoding Methods. Paraphrase generation can be regarded as a monolingual machine translation task that transforms expressions of an input sentence while retaining its meaning [39]. Recent neural paraphrasers primarily rely on the encoder-decoder framework, achieving inspiring performance gains over the traditional approaches. Beam search decoding is the most common method for inference, which decodes the top-K sequences in a greedy left-to-right fashion. When K is set to 1, beam search decoding is changed into greedy search decoding. In recent years, beam search decoding has multiple variants to deal with various task-specific and diversity/fluency trade-off of outputs, such as noise beam decoding [6], iterative beam decoding [15], clustered beam decoding [35] and diverse beam decoding [38]. Differ to the above decoding methods, our decoding method is a simple strategy focusing on improving the diversity of target words' variation.

3. Paraphraser-based lexical substitution

Formally, given a target word x_i occurring in a context sentence $\mathbf{x} = \{x_1, x_2, ..., x_i, ..., x_n\}$, a lexical substitution method can generate possible substitutions for x_i according to its context $\mathbf{x}$. We outline a three-staged pipeline for LS task, which takes advantage of the knowledge of the paraphrase dataset. Fig. 1 exhibits the details of the pipeline. We first utilize a paraphrase dataset to train a neural paraphraser (Stage 1, Section 3.1). Then, we extract substitute candidates for the target word x_i by a special decoding method after feeding sentence $\mathbf{x}$ into the paraphraser (Stage 2, Section 3.2). Finally, we rank substitute candidates to choose the most appropriate substitution that will not modify the meaning of $\mathbf{x}$ (Stage 3, Section 3.3).

3.1. Paraphraser

Recent neural paraphrasers primarily rely on the encoder-decoder learning framework on a large-scale paraphrase dataset, achieving inspiring performance gains over traditional methods [23,12]. For example, a paraphrase dataset ParaBank2 [11] consists of 19,370,798 sentence pairs. The paraphrase rule database, e.g., PPDB2.0 [7] extracted from ParaBank2 contains hundreds of millions of lexical paraphrase rules (e.g., "good → excellent"). If we directly find substitute candidates from the paraphrase rule database, the context of target word will not be considered. In this paper, we generate substitute

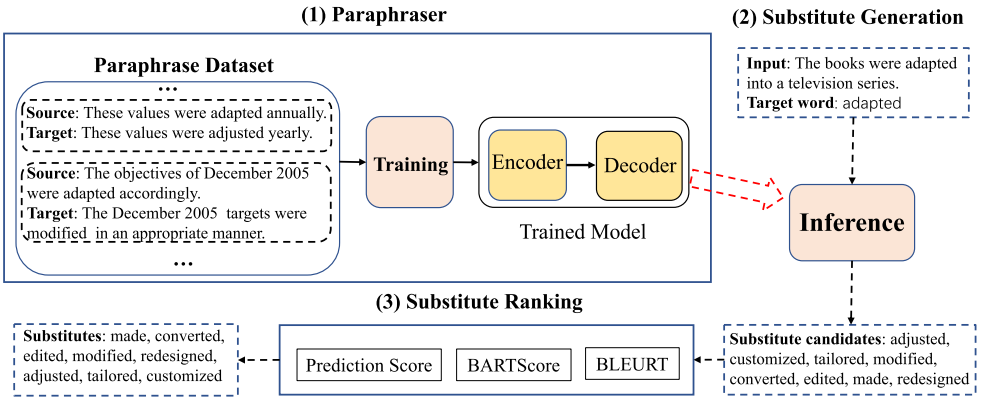


Fig. 1. The framework of paraphraser-based lexical substitution.

candidates of target words using a pretrained paraphrase modeling, rather than a paraphrase rule database or a paraphrase dataset.

Therefore, we train an auto-regressive encoder-decoder paraphraser based on the Transformer model on ParaBank2. If we want to generate multiple paraphrases of sentence $\mathbf{x}$, the beam search decoding is widely used by the auto-regressive method, which maintains a beam of K possible generations, updating them incrementally by ranking their extensions via the model likelihood. Given an input sentence $\mathbf{x}$ and the former generated words $\{y_1, ..., y_{i-1}\}$, the probability of generating next token y_i is computed as,

$$p(y_i|\mathbf{y}_{<i}, \mathbf{x}) = softmax(g(y_{i-1}, t_i, s_i)) \tag{1}$$

where g is non-linear function, t_i a decoding state for time step i, and s_i is a distinct source representation for time i, calculated as a weighted sum of the source annotations by the attention model.

Since beam search decoding devotes to find the most probable hypothesis for the whole sentence during decoding, it is difficult to extract multiple substitute candidates for the target word from the generated paraphrases. Therefore, a novel decoding method need to be developed that only focuses on lexical variations of target words during the decoding process.

3.2. Substitute generation

Since beam search decoding is not fitful for LS tasks, we propose a novel decoding strategy for the paraphraser that is exclusively designed to leverage lexical variations of target word.

Given a sentence $\mathbf{x}$ and a target word x_i, we force the decoder beginning with the prefix words $\mathbf{x}_{<i}$ of the target word, and then decode the next token y_i to predict the probability distribution of the vocabulary $p(y_i|\mathbf{x}_{<i}, \mathbf{x})$. We choose the top K tokens $Y = \{y_1, ..., y_k, ..., y_K\}$ with the highest probability in the distribution as the decoding results.

$$Y = \underset{Y \subseteq \mathcal{V}, |Y|=K}{\arg\max} \; p(Y|\mathbf{x}_{<i}, \mathbf{x}) \tag{2}$$

The decoding phase is crucial to generate substitute candidates since we forcibly generate K different tokens Y with the highest probability. In this case, these generated tokens not only are semantically consistent with the target word and fit to the context, but also preserve the sentence's meaning. For generating the rest of the tokens, we adopt greedy search decoding to pick the token that has maximum probability for each previous token, until reaching the end symbol "EOS" of one sentence.

As shown in Fig. 2(b), we force the decoder to begin with the prefix words of the target word for generating the probability distribution of the target word's position. For example, Fig. 2(b) shows the first 5 paraphrases using our decoding method. Obviously, we can easily access to substitute candidates of target word "adapted" from the paraphrases using our decoding method.

Additionally, to relieve the influence of suffix words $\mathbf{x}_{>i}$ when generating Y, the source representation s_i is calculated by summing the weights of source $\mathbf{x}_{\leq i}$, rather than these of the whole sentence $\mathbf{x}$.

$$s_i = \sum_{j=1}^{i} \alpha_{i,j} \cdot h_j \tag{3}$$

where h_j is the hidden state of x_j from the encoder, $\alpha_{i,j}$ is computed by an attention model that scores how y_i and h_j are matched with each other.

When obtaining K paraphrases of B, we select these words in position i for each paraphrase as substitute candidates, excluding the morphological derivations of the target word.

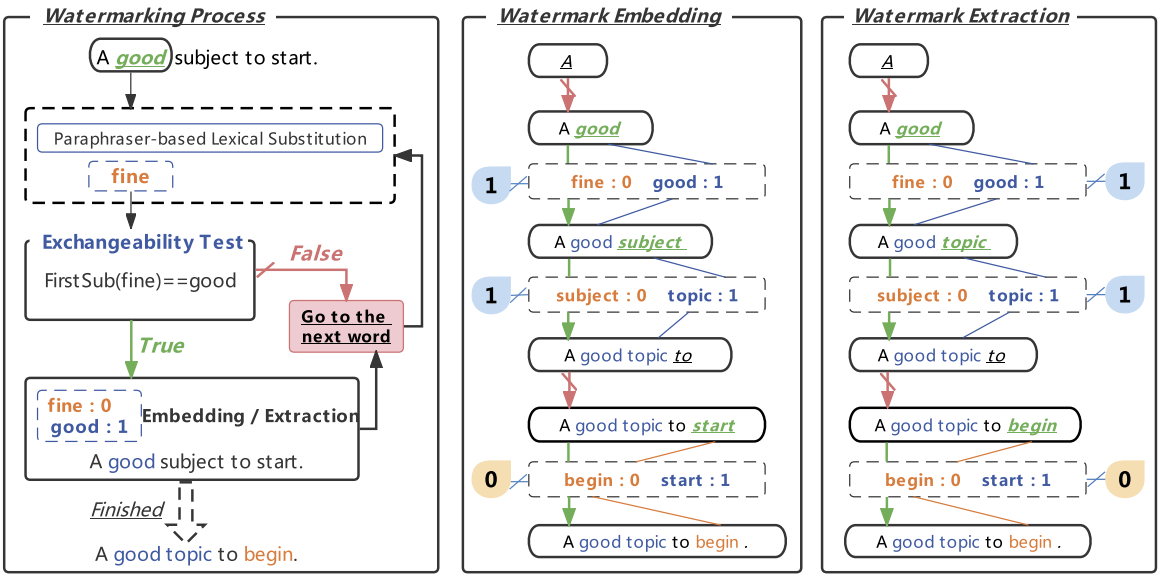


Fig. 2. The watermarking process with one example "A good subject to start." with watermark signals "110".

3.3. Substitute ranking

When the substitute candidates are generated, existing BERT-based LS methods [44,42,17] obtain a contextualized representation of substitutes by replacing a target word with each of its substitutes, and rank the substitutes by computing the cosine similarity between the vector of the target word and that of each substitute. The similarity between the target word and each substitute does not provide sufficient information about whether the substitute will modify the sentence's meaning. In this section, we rank the substitutes by calculating the semantic textual similarity between the original sentence $\mathbf{x}$ and the updated sentence after the substitute is embedded into the original sentence.

In this work, we formulate the evaluation for updated sentences as a text generation evaluation task. Assume that an updated sentence is denoted as $\mathbf{x}'$ after replacing the target word x_i in $\mathbf{x}$ with a substitute. In order to accurately calculate the similarity score between $\mathbf{x}$ and $\mathbf{x}'$, we choose two newest metrics (BARTScore and BLEURT) to evaluate the generated text in different perspectives (e.g., informativeness, fluency, and coherence). We found that BARTScore [42] and BLEURT [33] have the highest correlation on ranking substitutes by human judgments, because the pretrained modeling used by BARTScore and BLEURT are fine-tuned for learning the operation of lexical substitution.

BARTScore uses the pretrained encoder-decoder model BART [18] as the backbone to compute the generation probability of $\mathbf{y}$ conditioned on $\mathbf{x}$ as follows,

$$\text{BARTScore}(\mathbf{y}) = \sum_{i=1}^{n} \log p(y_i|\mathbf{y}_{<i}, \mathbf{x}) \tag{4}$$

where $\mathbf{y}$ is $\mathbf{x}'$ in our method.

In our method, we do not use the default parameter of BART to compute BARTScore, but fine-tune BART on ParaBank2 dataset, which will make the pretrained domain being closer to our task.

BLEURT is a BERT-based text generation metric, which describes the extent to which a generated sentence is similar to the original one. BLEURT is a fully learned metric that is pretrained on large amounts of synthetic data before fine-tuning it on human ratings, where the creation of synthetic data uses random perturbations of Wikipedia sentences augmented with a diverse set of lexical and semantic-level supervision signals. Based on its training processing, we see that BLEURT is fitful to rank the substitutes.

We have also included the prediction scores of the substitute candidates in Equation (2), as they consider both the context and the meaning preservation of original sentences for guiding the substitutes' generation. Finally, ParaLS uses a linear combination of the above-mentioned scores to calculate the final score for each substitute candidate.

4. Natural language watermarking method

NLW method consists of an embedding stage and an extraction stage. Given one sentence $\mathbf{x}$, we generate a watermarked sentence $\mathbf{x}'$ after embedding the watermark signals using our proposed LS method. In the watermark extraction stage, we need to extract the watermark signals from $\mathbf{x}'$. Considering our LS method depends on the context information for

generating the substitute candidates, the candidates generated for $\mathbf{x}$ are probably different from those candidates for $\mathbf{x}'$, thereby resulting in extraction failure.

To overcome this challenge, we further introduce an exchangeability test, which guarantees that both the embedding and extraction stages can generate identical substitute candidates. Therefore, we design a novel NLW method based on the exchangeability test. The whole process of an example is illustrated in Fig. 2.

4.1. Exchangeability test

Suppose one word y_i is the first substitution of a target word x_i in $\mathbf{x} = \{x_1, x_2, ..., x_i\}$ via our proposed LS method. We only choose the first substitute y_i to replace x_i for generating the watermarked sentence, since the first substitute can better maintain the original meaning of the original sentence. The process of obtaining y_i via our proposed LS method is formulated as,

$$y_i = \text{FirstSub}_{\mathbf{x}_{<i}}(x_i) \tag{5}$$

The exchangeability test is a mechanism: x_i and y_i can be exchanged with each other under the same prefix words $\mathbf{x}_{<i}$. After replacing x_i of $\mathbf{x}$ with y_i, we say the word x_i in sentence $\mathbf{x}$ passes the exchangeability test if it meets the following condition: the first substitute $\text{FirstSub}_{\mathbf{x}_{<i}}(y_i)$ of target word y_i equals to x_i, namely x_i equals to $\text{FirstSub}_{\mathbf{x}_{<i}}(\text{FirstSub}_{\mathbf{x}_{<i}}(x_i))$.

The exchangeability test claims that we can re-generate the target word after two lexical substitution operations under the same context. It implicitly indicates that the substitutions in the watermarked sentence can excellently preserve the meaning of the target word under the corresponding context.

4.2. Watermarking process

Given a text, we need to split it into sentences using the NLP tools in NLTK.[1] For each sentence, we try to embed and extract the watermark signals via the exchangeability test. Our NLW method consists of the embedding stage (Algorithm 1) and the extraction stage (Algorithm 2).

Given the i-th word $x_i (2 \leq i \leq n)$ in sentence $\mathbf{x} = \{x_1, x_2, ..., x_n\}$, we incrementally search and substitute the words that can carry watermark signals in the local context. In the embedding stage, given the sentence $\mathbf{x}_{\leq i}$ and the target word x_i, we get the first substitution y_i by Eq. (5) (step 3). Likewise, we obtain the first substitution z_i after replacing x_i with y_i (step 4). Then, we will justify whether the target word x_i meets the exchangeability test by comparing x_i with z_i (step 5).

If x_i is not equal to z_i, the next word will be traveled. If x_i equals z_i, we sort both y_i and x_i that follow the alphabet in ascending order as $C = \{c_0, c_1\}$ (line 8). We replace x_i with the word in C for embedding a one-bit watermark signal from the watermark binary sequence $\mathbf{m}$ based on the following rule:

$$x_i = \begin{cases} c_0, & if\ signal = 0, \\ c_1, & if\ signal = 1. \end{cases} \tag{6}$$

In the extraction stage (Algorithm 2), we only need the watermarked sentence $\mathbf{x}'$ to extract the watermark signals. The steps in the extraction stage are exactly similar to the steps in the embedding stage. Firstly, we generate the first substitute for the current word x_i, and judge whether it meets the exchangeability test (steps 2-4). The candidates $C = \{c_0, c_1\}$ we generated in the stage are identical to those in the embedding stage due to the exchangeability test. Next, we decide which signal is embedded in $\mathbf{x}'$ by comparing x_i with the candidate of C (steps 8-14). Then, we append the obtained signal into the watermark binary sequence $\mathbf{m}$ and update the current sentence $\mathbf{x}'$. Finally, the embedding watermark signals will be successfully acquired as soon as the sentence is completely updated.

We can see that our method watermarks an entire text rather than just certain parts of it. There are the following several reasons. One reason is to ensure that the watermark is difficult to remove. If the watermarked text is only present in a few parts of the document, it might be easier for someone to remove the watermark by simply deleting those parts. On the other hand, if the entire text is watermarked, it becomes much more difficult to remove the watermark without also significantly altering the content of the document.

Another reason to watermark the entire text is to make it easier to detect the watermark. If the watermarked text is only present in certain parts of the document, it might be more difficult to detect the watermark, especially if the document is large or the watermarked text is spread out over many pages. By watermarking the entire text, it becomes much easier to detect the watermark using automated methods.

Overall, the decision to watermark an entire text or just certain parts of it depends on the specific goals of the watermarking process. If the goal is to protect the copyright of a document or to establish ownership, it may be necessary to watermark the entire text to make it more difficult to remove the watermark and to make it easier to detect. On the other hand, if the goal is to preserve the original text and reduce computational efficiency, it may be sufficient to only watermark certain parts of the text.

[1] https://www.nltk.org/.

Algorithm 1 The embedding stage of our NLW method.

Input: source sentence: $\mathbf{x} = x_1, ..., x_n$, the watermark binary bit sequence: $\mathbf{m}$
Output: watermarked sentence $\mathbf{x}'$

$\mathbf{x}' \leftarrow \mathbf{x}$
for $i \leftarrow 2$ to n **do**
  $y_i \leftarrow \text{FirstSub}_{\mathbf{x}'_{<i}}(x_i)$
  $z_i \leftarrow \text{FirstSub}_{\mathbf{x}'_{<i}}(y_i)$
  if $z_i \neq x_i$ **then**
    continue
  else
    $\{c_0, c_1\}$=$Sort_{alphabet}(x_i, y_i)$
    Fetch one bit *signal* from m.
    Replace x_i in $\mathbf{x}'$ with word c_{signal} of $\{c_0, c_1\}$
  end if
end for
Return $\mathbf{x}'$

Algorithm 2 The extraction stage of our NLW method.

Input: watermarked sentence: $\mathbf{x}' = x_1, ..., x_i, ..., x_n$
Output: the watermark binary bit sequence: $\mathbf{m}$

for $i \leftarrow 2$ to n **do**
  $y_i \leftarrow \text{FirstSub}_{\mathbf{x}'_{<i}}(x_i)$
  $z_i \leftarrow \text{FirstSub}_{\mathbf{x}'_{<i}}(y_i)$
  if $z_i \neq x_i$ **then**
    continue
  else
    $\{c_0, c_1\}$=$Sort_{alphabet}(x_i, y_i)$
    if $x_i = c_0$ **then**
      Append one bit 0 into $\mathbf{m}$.
      Replace x_i in $\mathbf{x}'$ with c_0
    else
      Append one bit 1 into $\mathbf{m}$.
      Replace x_i in $\mathbf{x}'$ with c_1
    end if
  end if
end for
Return $\mathbf{m}$

5. Experimental results on lexical substitution

In this section, we verify the performance of our proposed lexical substitution method. We conduct a group of experiments to answer the following questions:

Q1. The effectiveness of the decoding strategy of our LS method: Does the decoding strategy fit for LS task?

Q2. The effectiveness of our proposing substitute ranking: Does the substitute ranking step of our proposed method outperform the state-of-the-art LS competitors?

5.1. Experiment settings

(1) Implementation Details. To implement an English paraphraser, we train a Transformer model in FairSeq with an 8-layer encoder and decoder, 1024 dimensional embeddings, 16 encoder-decoder attention heads, and 0.1 dropout. The initial learning rate is set to $lr = 3 \times 10^{-4}$. We adopt the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$. We choose an English paraphrase dataset ParaBank2 [11] to train the paraphraser. In our experiments, we duplicate all the samples by exchanging source sentences and target sentences. We use the BLEURT large model[2] for the calculation of BLEURT score. BARTScore is fine-tuned on ParaBank2 that can be downloaded through the link.[3] We use the LS07 dev set for tuning the hyperparameters in our model. The weights of the prediction score (PREDICT), BARTScore, and BLEURT are 0.02, 1, and 1, respectively. The number of outputted paraphrases K is set to 50.

(2) Datasets. For LS tasks, two widely used datasets, LS07 [19] and CoInCo [13], are chosen for the performance evaluation of LS methods. Each instance in LS dataset is composed of a target word, its context, and corresponding substitutes. Each of the substitutes has an associated frequency that indicates the number of annotates. LS07 consists of 300 development examples and 1710 test instances for 201 polysemous words. For each target word, 10 sentences are provided. The annotators' task deployed on Amazon Mechanical Turk gives up to 3 possible substitutes. Concept In Context (ConInCo) is a

[2] https://huggingface.co/Elron/bleurt-large-512.
[3] https://github.com/neulab/BARTScore.

dataset containing 2474 sentences that cover 3874 distinct targets with different part-of-speech tags, which is the current largest LS benchmark. It consists of 15K target instances with a given 35% development and 65% test.

(3) Comparison Methods. We compare our unsupervised method ParaLS with the following baselines. The two recent BERT-based methods BERT-Drop [44] and LexSubCon [24]) are chosen. One method based on pretrained XLNet is chosen, denoted as XLNet [2]. We choose one newest supervised method GeneSis and other previous LS methods including two embedding-based methods (Embedding [22] and Addocs [20]), two supervised methods (MachineLearning [34] and TransferLearning [10]), and two knowledge-based methods (KU [43] and UNT [9]).

(4) Metrics. Following previous LS methods [44,24], we use the official metrics *best*, *best-mode*, *oot*, *oot-mode* in SemEval 2007 task as well as Precision@1(P@1) as our evaluation metrics. In detailed, *best*, *best-mode* and P@1 evaluate the quality of the best predictions, yet *oot* (out-of-ten) and *oot-mode* benefit verifying the coverage of the gold substitute candidate list by the 10-top predictions.

Let T and H be the set of test instances and annotators, respectively. We use A being the set of instances in T where the system provides at least one substitute. For each item $i \in A$, a_i denotes the set of substitutes generated from the system, and h_i indicates the set of substitutes provided by annotator $h \in H$. For each i, we calculate the multiset union (H_i) of all h_i for each annotator, and each unique type res in H_i will have an associated frequency $freq_{res}$ by capturing the times of its appearance in H_i.

For example, a sentence contains the target word *happy*, and the five annotators supply the answers as follows: {glad, merry}, {glad}, {cheerful, glad}, {merry}, {jovial}. Then, H_i would be {glad glad glad merry merry cheerful jovial}. The *res* with associated frequencies would be {*glad* 3, *merry* 2, *cheerful* 1, *jovial* 1}.

As regards the variations of *mode*, let the mode m_i be the most frequent response for instance $i \in T$, if it exists. For the gold substitutes and the system ones, the sets that belong to this mode are TM and AM, respectively.

best and best-mode. The precision and recall of *best* are formulated as,

$$best = 2 * \frac{P_b R_b}{P_b + R_b} \tag{7}$$

$$P_b = \frac{1}{|A|} \sum_{a_i : i \in A} \frac{\sum_{res \in a_i} freq_{res}}{|H_i| * |a_i|} \tag{8}$$

$$R_b = \frac{1}{|T|} \sum_{a_i : i \in T} \frac{\sum_{res \in a_i} freq_{res}}{|H_i| * |a_i|} \tag{9}$$

As regards the *mode* variation, we modify precision and recall as

$$P_b = \frac{1}{|AM|} \sum_{bg_i \in AM} 1 \; if \; bg = m_i \tag{10}$$

$$R_b = \frac{1}{|TM|} \sum_{bg_i \in TM} 1 \; if \; bg = m_i \tag{11}$$

where bg is the best guess in the list of substitutes provided by the system.

oot and oot-mode. The precision and recall of *oot* are formulated as,

$$oot = 2 * \frac{P_o R_o}{P_o + R_o} \tag{12}$$

$$P_o = \frac{1}{|A|} \sum_{a_i : i \in A} \frac{\sum_{res \in a_i} freq_{res}}{|H_i|} \tag{13}$$

$$R_o = \frac{1}{|T|} \sum_{a_i : i \in T} \frac{\sum_{res \in a_i} freq_{res}}{|H_i|} \tag{14}$$

While in the *mode* variation precision and recall are slightly modified as

$$P_o = \frac{1}{|AM|} \sum_{a_i : i \in AM} 1 \; if \; any \; guess \in a_i = m_i \tag{15}$$

$$R_o = \frac{1}{|TM|} \sum_{a_i : i \in TM} 1 \; if \; any \; guess \in a_i = m_i \tag{16}$$

Table 1
Results on LS07 and CoInCo datasets. "±" means the standard deviation of five runs. † means these methods utilize annotated LS data for training. Best values are bolded. 'w/o ranking' means it only chooses the top substitutes from the step of substitute generation. The baselines are KU [43], UNT [9], Embedding [22], Addocs [20]), MachineLearning [34], TransferLearning [10], BERT-Drop [44], XLNet [3], GeneSis [17], and LexSubcon [24].

Dataset	Method	best	best-m	oot	oot-m	P@1
LS07	KU	12.9	20.7	46.2	61.3	-
	UNT	12.8	20.7	49.2	66.3	-
	Embedding	12.7	21.7	36.4	52.0	-
	Addocs	8.1	13.4	27.4	39.1	-
	MachineLearning †	15.9	-	48.8	-	40.8
	TransferLearning †	17.2	-	48.4	-	-
	BERT-Drop (w/o ranking)	12.8	22.1	43.9	59.7	31.7
	XLNet	20.6	33.2	50.9	61.6	49.5
	GeneSis †	21.2	34.1	52.2	66.4	51.2
	LexSubCon(w/o ranking)	16.3	27.6	45.6	62.4	40.8
	LexSubCon †	21.1 ± 0.03	35.5 ± 0.07	51.3 ± 0.05	68.6 ± 0.05	51.7 ± 0.03
	ParaLS(Ours, w/o ranking)	18.3 ± 0.02	31.7 ± 0.02	50.0 ± 0.01	67.4 ± 0.01	45.1 ± 0.02
	ParaLS(Ours)	**24.1** ± 0.02	**42.4** ± 0.02	**58.2** ± 0.01	**76.5** ± 0.01	**58.3** ± 0.02
CoInCo	Embedding	8.1	17.4	26.7	46.2	-
	Addocs	5.6	11.9	20.0	33.8	-
	BERT-Drop (w/o ranking)	11.8	24.2	36.0	56.8	43.5
	XLNet	14.4	30.1	39.2	60.7	51.5
	LexSubCon(w/o ranking)	11.3	23.8	33.6	54.4	41.3
	LexSubCon †	14.0 ± 0.02	29.7 ± 0.03	38.0 ± 0.03	59.2 ± 0.04	50.5 ± 0.02
	ParaLS(Ours, w/o ranking)	13.0 ± 0.02	27.0 ± 0.02	40.3 ± 0.01	63.8 ± 0.01	47.0 ± 0.02
	ParaLS(Ours)	**18.1** ± 0.02	**40.0** ± 0.02	**49.2** ± 0.01	**75.4** ± 0.01	**62.6** ± 0.02

Table 2
Ablation study of ParaLS. "-w/o" indicates a ParaLS framework without the specific feature. "o." indicates that only one specific ranking feature is used.

Method	LS07					CoInCo				
	best	best-m	oot	oot-m	P@1	best	best-m	oot	oot-m	P@1
ParaLS	**24.1**	**42.4**	**58.2**	**76.5**	**58.3**	**18.1**	**40.0**	**49.2**	**75.4**	**62.6**
-w/o PREDICT	22.7	39.4	57.6	75.9	55.7	17.8	39.5	48.9	75.1	61.6
-w/o BARTScore	22.3	38.1	57.1	74.6	54.7	16.5	35.1	48.0	73.8	58.2
-w/o BLEURE	22.8	40.5	56.8	76.2	54.7	16.7	36.6	47.5	73.8	58.1
o. PREDICT	14.7	25.5	46.5	65.7	36.8	13.0	27.0	40.3	63.8	47.0
o. BARTScore	20.4	34.9	56.2	75.0	51.2	16.0	34.9	47.1	73.3	56.0
o. BLEURE	19.5	32.0	55.6	72.4	48.7	15.2	31.9	47.4	73.2	54.6

5.2. Results and discussion

Performance on Lexical Substitution. The results of our method coupled with the state-of-the-art methods are shown in Table 1. In general, the top substitutes by executing the step of substitutes ranking are chosen to evaluate the performance. To eliminate the influence of substitute ranking, we also provide the results of three methods (BERT-Drop, LexSubCon, and ParaLS) without ranking.

As we can see from the results, our method ranking-excluded ParaLS outperforms the previous BERT-based LS methods (BERT-Drop and LexSubCon) across all metrics evaluated on the LS07 and CoInCo datasets. Since BERT is trained on raw data, it does not pertinently learn the operation of lexical substitution. Although ParaLS does not rely on any LS dataset, the paraphrasing modeling used by ParaLS focuses on variations in word choice. It is verifiable that our decoding strategy based on the paraphrasing modeling can be a valuable tool for providing accurate substitution candidates.

After ranking the substitutes, our method outperforms the previous unsupervised and supervised methods. When both ParaLS and LexSubCon incorporate the step of substitute ranking, ParaLS further improve the performance than LexSubCon. Although GeneSis fine-tunes one encoder-decoder modeling using the development sets of LS datasets, the size of the development set limits the applicability of large pretrained models, thereby hampering the potential benefits of supervised approaches applied to the task.

Ablation Study. To verify the effect of each feature on the performance of ParaLS, we conduct an ablation study. The results are shown in Table 2. As shown in the table, ParaLS achieves the best performance when it accesses to the information from all the features (PREDICT, BARTScore, and BLEURT). By testing the performance of individual features, we observe that BARTScore achieves the best performance, because we fine-tune BARTScore on ParaBank2 that will make the pretrained domain being similar to a LS task. The PREDICT feature achieves the worst performance, which means it does not provide sufficient information about whether the substitute words will modify the sentence's meaning.

Table 3
Comparison of GAP scores (%) in the substitute ranking task. † means these methods are supervised. The newly added baselines: XLNet+embs [3] and Context2Vec [21].

Method	LS07	CoInCo
ParaLS(Ours)	**65.1**	**60.7**
-w/o Prediction	64.5	59.2
-w/o BARTScore	62.0	58.1
-w/o BLEURE	63.9	58.4
LexSubCon†	60.6	58.0
BERT-Drop	58.6	55.2
XLNet+embs	57.3	54.8
Context2Vec	56.0	47.9
TransferLearning†	51.9	-
SupervisedLearning†	55.0	-
Embedding	55.1	50.2
Addcos	52.9	48.3

Table 4
The first five generated paraphrases of two instances in CoInCo using ParaLS.

Instance 1	With warmest regards, Your Past.
Labels	wish, respect, hope, concern, address
Para1	With warmest **greetings**, Your Past.
Para2	With warmest **enthusiasm**, Your Past.
Para3	With warmest **respect**, Your Past.
Para4	With warmest **salute**, Your Past.
Para5	With warmest **regard** for your past.
Instance 2	this is an **unusual** situation.
Labels	strage, weird, exceptional, different, ...
Para1	this is an **unorthodox** situation.
Para2	this is an **atypical** situation.
Para3	this is an **uncommon** situation.
Para4	this is an **aberrant** situation.
Para5	this is an **unusual** situation.

Substitute Ranking. We also evaluate ParaLS in the substitute ranking task for both the LS07 and CoInCo datasets. In this sub-task of LS task, we assume that the substitute candidates are provided, each method aims to create the most appropriate ranking of the candidates. We use GAP score[4] to evaluate the performance of the following prior works [44,24] on the subtask, where the GAP score is a variant of MAP (Mean Average Precision).

As shown in Table 3, our method ParaLS outperforms the previous state-of-the-art methods when they are all provided with the same substitute candidates. Although LexSubCon is a supervised method, ParaLS still yields better results. Additionally, our method is simpler than LexSubCon, since LexSubCon utilizes the combination of the four features from contextual embedding models and external lexical knowledge bases for ranking the substitutes. It verifies that text generation evaluation metrics are more fitful to substitute ranking than all the previous methods. By taking account of the performance from ParaLS by removing one feature of ParaLS, we see that all the features have a positive effectiveness on the performance of ParaLS.

Case Study. We sample two instances of CoInCo to analyze the generated paraphrases using our decoding method. Table 4 shows the top five paraphrases using ParaLS. We found that the three substitutes of the first instance "greetings, enthusiasm, salute" are not annotated in CoInCo. Even if the labels of instance 2 in CoInCo consist of 10 words, these words "unorthodox atypical, uncommon, aberrant" are not annotated. Since the labels are annotated by humans, it is impossible to provide all the suitable substitutes for each word. We affirm that the real performance of ParaLS is better than the results calculated by the metrics. Additionally, ParaLS can help to improve the coverage of the substitutes in existing LS datasets.

We give an instance to show the drawback of ParaLS. In Table 5, the output paraphrases contain multiple suitable substitutes. The words (held, stretched, and extended) are pruned by ParaLS, since ParaLS only chooses the words owning the next word same with the target word. We observe that the paraphrases include multiple phrases for the target word, which means ParaLS has great potential on producing "phrase" substitutes.

[4] https://tinyurl.com/gap-measure.

Table 5
A sample: the substitute candidates are unable to be extracted using ParaLS.

Instance 3	She reached for another dial on the house controls.
Labels	extend an arm, catch up, come up, extend
Para1	She reached for the next dial ···
Para2	She held out her hand to the next dial ···
Para3	She stretched out her hand to the next ···
Para4	She extended her hand to the next ···
Para5	She reaching for the next dial ···
ParaLS	-

6. Experimental results on natural language watermarking

In this section, we verify the performance of our proposed NLW method. We design experiments to answer the following questions:

Q1. The payload capacity of our NLW method: Does the payload capacity of our NLW method outperform the state-of-the-art NLW competitors?

Q2. The effectiveness of preserving the semantics of original sentences: Does our NLW method better preserve the semantics of original sentences?

6.1. Experiment settings

(1) Implementation Details. Based on the LS method proposed above, we implement our NLW method to embed/extract the watermark signals into/from the target text one by one. Generally, the data related to the target document is selected as the source of the watermark signal. For example, the novel "Pride and Prejudice" was written by a British novelist 'Jane Austen' in 1796 and published in 1813. We can use the information "Jane Austen published 'Pride and Prejudice' in 1813" as watermark information for protecting its copyright. We transform the watermark information into binary watermark signals through ASCII code. The binary watermark signals are the watermark binary bit sequence **m** in Algorithm 1. In the embedding stage, we embed **m** into the text. In the extraction stage, we extract **m** for identifying the copyright.

(2) Datasets. For evaluating NLW methods, we choose the four datasets with different writing styles (Novels, WikiText-2, IMDB, and NgNews) that comply with the study [41]. There are three novels ("Wuthering Heights", "Dracula", and "Pride and Prejudice") come from Project Gutenberg.[5] For the rest of the datasets (WikiText-2, IMDB, and NgNews) provided by HuggingFace,[6] the first 10,000 sentences are only selected.

(3) Comparison Methods. We choose the state-of-the-art NLW method (BERT-NLW) [41] as the baseline for comparison. Both our proposed NLW method (Para-NLW) and BERT-NLW utilize the context to suggest LS candidates. The transformer-based method AWT [1] is not considered as the baseline because it only substitutes the unobtrusive words in a given context.

(4) Metrics. By learning from the existing NLW method [41], we use payload and text recoverability as our evaluation metrics. We will explain these metrics in the following experiments.

6.2. Results and discussion

Payload. The payload is the maximum amount of information (number of bits) that can be embedded in the original text. In general, the payload is measured by the average amount of information carried in one single word. One NLW method owning good payload capacity is critical for its application [8].

As shown in Fig. 3, BERT-NLW has low capacity of watermark embedding. The average payload of our method Para-NLW is higher than that of BERT-NLW by 2-3 times upon six datasets. Since BERT-NLW is less concerned with target words than contexts, it is difficult to generate suitable substitutions for many words leading to the disability of carrying the watermark signals.

To highlight the difference between Para-NLW and BERT-NLW, we also show the average number of words in each sentence that can carry the watermark signals. The results are shown in Fig. 4. The average number of words in three novels ("Wuthering Heights", "Dracula", and "Pride and Prejudice") are 21, 21, and 24, respectively. The average number of words in the three datasets (WikiText-2, IMDB, and NgNews) are 121, 112, and 97, respectively. We can clearly see that Para-NLW allows each sentence carries more watermark signals compared with BERT-NLW. We also find that thousands of sentences in Fig. 4(b) do not contain any watermarked word, because they only contain a few words.

[5] https://www.gutenberg.org/.
[6] https://huggingface.co/datasets.

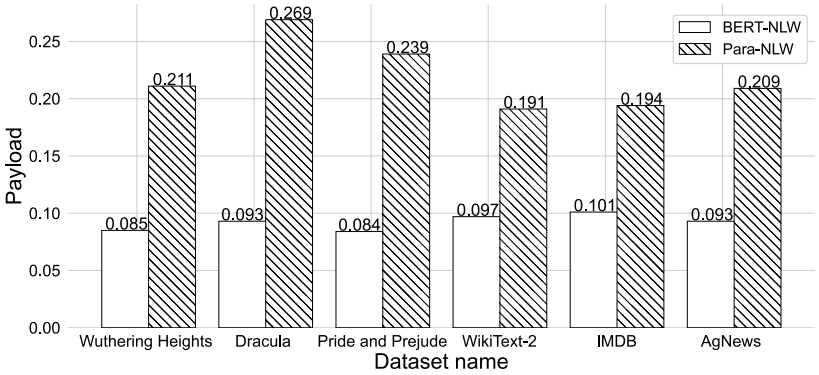
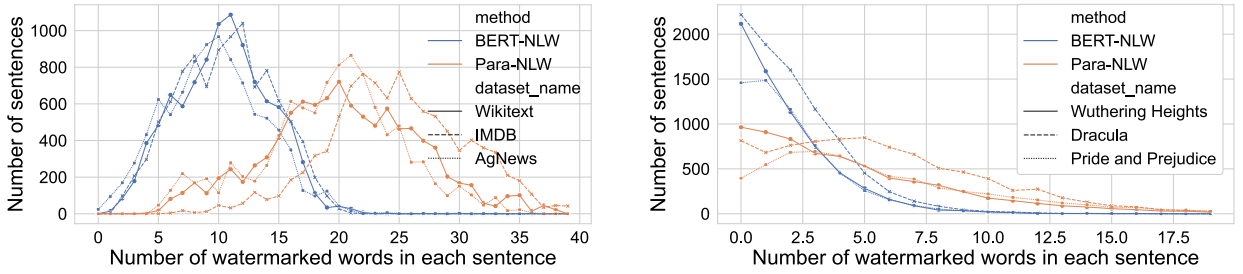


Fig. 3. The payload of our proposed method Para-NLW compared with BERT-NLW.



(a) WikiText-2, IMDB, and NgNews. (b) Three novels.

Fig. 4. The average number of words in each sentence that can carry the watermark signals.

Table 6
The proportion of the substituted words that can be recovered in six datasets.

Method	Wuthering Height	Dracula	Pride and Prejudice	WikiText-2	IMDB	AgNews
BERT-NLW	81.65%	82.38%	81.29%	87.04%	82.95%	86.14%
Para-NLW	87.88%	89.31%	88.77%	89.87%	87.94%	89.91%

Text Recoverability. According to the exchangeability test, we can regenerate the original word for each substitution word. In this experiment, we try to reconstruct the original text from the watermarked text. The higher text recoverability means that the method is more effective in preserving the semantics of original sentences.

Following with the experimental process in the study [41], we mask each candidate in the substitute candidates, and use BERT to predict the probability. Then we choose top probability words as recovered words to replace the corresponding watermarked word for the original sentence reconstruction.

The results are shown in Table 6. We find that approaching 90% of watermarked words in watermarked texts using our method Para-NLW can be successfully recovered. BERT-NLW only recovers about 80% of the watermarked words. The results indicate that Para-NLW can better preserve the semantics of original sentences compared with BERT-NLW.

Case Study. We give seven examples to analyze the watermarked words among AWT [1], BERT-NLW, and our method Para-NLW. The results are shown in Table 7. The watermarked words in AWT are unobtrusive. Some unobtrusive words may change the logic and semantics, e.g., "but → and", "as → an". We see that only a few words are chosen as watermarked words in BERT-NLW. Compared with BERT-NLW, our method Para-NLW can search and substitute more words to carry watermark signals without hurting the original sentence's meaning.

The importance of Exchangeability Test. The exchangeability test aims to guarantee that both the embedding and extraction stages can generate identical substitute candidates. As shown in Table 8, if there is no exchangeability test, we cannot correctly extract the watermark signals since the watermarked words can not be located, e.g., FirstSub(FirstSub(*elements*)) $\neq$ *elements*, or FirstSub(FirstSub(*space*)) $\neq$ *space*. After extracting the watermark signals, we transform the signals into watermark information that can help tracing text provenance.

7. Conclusion

In this paper, we originally propose a novel paraphraser-based lexical substitution (LS) method named ParaLS, which generates substitute candidates by considering the context and preserving the sentence's meaning without requiring anno-

Table 7
Examples of watermarked sentences on WikiText-2 compared with Para-AWT and BERT-LW. The substituted words are marked in color.

Original	AWT	BERT-NLW	Para-NLW
In 1951, a small airstrip was built at the ruins	In 1951, a small airstrip was built *on* the ruins	In 1951, an small air- strip was built at the ruins	In 1951, a *tiny* airstrip was *constructed* *on* the ruins
, but the complex is broken up by the heat of cooking	, *and* the complex is broken up by the heat of cooking	, but the complex is broken up by the *temperature* of cooking	, but the *complicated* is broken *down* by the *temperature* of *baking*
resulting in a population decline as workers left for other areas	resulting in a population decline *an* workers left for other areas	resulting in a *demographic* decline as *employees* left for other areas	resulting in a *demographic* *decrease* as *employees* left for *another* regions
It is bordered on the east side by identical temples	It is bordered *at* the east side by identical temples	It is *bounded* on the *eastern* side by ident- ical temples	It is bordered on the *eastern* *edge* from identical *shrines*
Blythe, who is <unk>, took off his glasses before entering the stage, which together with the smoke and light effects allegedly left him half	Blythe, who is <unk>, took off his glasses before entering the stage, which together *@-@* the smoke and light effects allegedly left him half	Blythe, who is <unk>, took off his glasses before entering the stage, which *along* with the smoke and light effects allegedly left him half	Blythe, who is <unk>, took off his *eyeglasses* *prior* *entry* the stage, which *combined* with the *fumes* and *illumination* effects allegedly *leaving* *blythe* *one-half*

Table 8
Failure examples without the **Exchangeability Test**, where the words marked in red indicate that the generated candidate sets are different from the embedding stage, while the words labeled with green are the same. **w/o** means the exchangeability test is not deployed.

Original	Embedding Stage **w/o**	Extraction Stage **w/o**	Difference	
the cooperative elements incorporated into the second game were removed,	the co-operative components incorporated into the second play were removed,	the *co-operative* *components* *incorporated* into the *second* *play* were *removed*,	FirstSub(*elements*)=*components* FirstSub(*components*)=parts	✗
as they took up a large portion of memory space needed for the improvements.	as they took up a large part of memory storage required for the improvements.	as they *took* up a *large* *part* of *memory* *storage* *required* for the *improvements*.	FirstSub(*space*)=*storage* FirstSub(*storage*)=stores	✗

tated LS data. Specifically, we design a simple decoding strategy that focuses on lexical variations of the target word during decoding, and propose a substitute candidate ranking method by utilizing the newest text generation evaluation metrics. The experimental results show that ParaLS significantly outperforms the existing supervised and unsupervised LS methods.

Based on the proposed LS method, we propose a novel natural language watermarking method as a potential solution towards tracing text provenance. To embed watermark signals using the proposed LS method, we design the exchangeability test to search for the words capablity of carrying watermark signals in the context. Compared with the baselines, our method can outstandingly preserve the semantics of the original text and significantly improve the payload by more than 2 times. Our work offers an new research area towards natural language watermarking from paraphrase generation models. In the future, we will try to extract high-quality substitute candidates from a paraphraser, and further investigate the method's general availability.

Due to the payload capacity, it is hard to embed long watermarking information into short texts. Additionally, the proposed NLW method is not a reversible NLW method. In other words, we cannot re-obtain original sentences from watermarked sentences. The proposed NLW method is not suitable for these applications that require to recover original sentences.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Jipeng Qiang reports financial support was provided by National Natural Science Foundation of China.

Data availability

I will share our code and data in github.com.

Acknowledgement

This research is partially supported by the National Natural Science Foundation of China under grants 62076217, 62120106008, and 61906060, and the Blue Project of Yangzhou University.

References

[1] S. Abdelnabi, M. Fritz, Adversarial watermarking transformer: towards tracing text provenance with data hiding, in: 2021 IEEE Symposium on Security and Privacy (SP), IEEE, 2021, pp. 121–140.
[2] N. Arefyev, B. Sheludko, A. Podolskiy, A. Panchenko, Always keep your target in mind: studying semantics and improving performance of neural lexical substitution, in: Proceedings of the 28th International Conference on Computational Linguistics, 2020, pp. 1242–1255.
[3] N. Arefyev, B. Sheludko, A. Podolskiy, A. Panchenko, A comparative study of lexical substitution approaches based on neural language models, arXiv preprint arXiv:2006.00031, 2020.
[4] J.T. Brassil, S. Low, N.F. Maxemchuk, Copyright protection for the electronic distribution of text documents, Proc. IEEE 87 (1999) 1181–1196.
[5] T. Brown, B. Mann, N. Ryder, M. Subbiah, J.D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, Adv. Neural Inf. Process. Syst. 33 (2020) 1877–1901.
[6] K. Cho, Noisy parallel approximate decoding for conditional recurrent language model, arXiv preprint arXiv:1605.03835, 2016.
[7] J. Ganitkevitch, B.V. Durme, C. Callison-Burch, Ppdb: the paraphrase database, in: NAACL-HLT, 2013, pp. 758–764.
[8] W. Hao, L. Xiang, Y. Li, P. Yang, X. Shen, Reversible natural language watermarking using synonym substitution and arithmetic coding, Comput. Mater. Continua 55 (2018) 541–559.
[9] S. Hassan, A. Csomai, C. Banea, R. Sinha, R. Mihalcea, Unt: Subfinder: combining knowledge sources for automatic lexical substitution, in: Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007), 2007, pp. 410–413.
[10] G. Hintz, C. Biemann, Language transfer learning for supervised lexical substitution, in: ACL, 2016, pp. 118–129.
[11] J.E. Hu, A. Singh, N. Holzenberger, M. Post, B. Van Durme, Large-scale, diverse, paraphrastic bitexts via sampling and clustering, in: CoNLL, Association for Computational Linguistics, Hong Kong, China, 2019, pp. 44–54, https://aclanthology.org/K19-1005.
[12] S. Kadotani, T. Kajiwara, Y. Arase, M. Onizuka, Edit distance based curriculum learning for paraphrase generation, in: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop, Online, Association for Computational Linguistics, 2021, pp. 229–234, https://aclanthology.org/2021.acl-srw.24, https://doi.org/10.18653/v1/2021.acl-srw.24.
[13] G. Kremer, K. Erk, S. Padó, S. Thater, What substitutes tell us-analysis of an "all-words" lexical substitution corpus, in: Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, 2014, pp. 540–549.
[14] R. Kriz, E. Miltsakaki, M. Apidianaki, C. Callisonburch, Simplification using paraphrases and context-based lexical substitution.
[15] I. Kulikov, A.H. Miller, K. Cho, J. Weston, Contextualized perturbation for textual adversarial attack, in: Proceedings of the 12th International Conference on Natural Language Generation, 2019, pp. 76–87.
[16] C. Lacerra, T. Pasini, R. Tripodi, R. Navigli, Alasca: an automated approach for large-scale lexical substitution, in: Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21, 2021, pp. 3836–3842.
[17] C. Lacerra, R. Tripodi, R. Navigli, Genesis: a generative approach to substitutes in context, in: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, 2021, pp. 10810–10823.
[18] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, L. Zettlemoyer, Bart: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, arXiv preprint arXiv:1910.13461, 2019.
[19] D. McCarthy, R. Navigli, Semeval-2007 task 10: English lexical substitution task, in: Proceedings of the 4th International Workshop on Semantic Evaluations, 2007, pp. 48–53.
[20] O. Melamud, I. Dagan, J. Goldberger, Modeling word meaning in context with substitute vectors, in: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2015, pp. 472–482.
[21] O. Melamud, J. Goldberger, I. Dagan, Context2vec: learning generic context embedding with bidirectional lstm, in: Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, 2016, pp. 51–61.
[22] O. Melamud, O. Levy, I. Dagan, A simple word embedding model for lexical substitution, in: Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing, 2015, pp. 1–7.
[23] Y. Meng, X. Ao, Q. He, X. Sun, Q. Han, F. Wu, C. Fan, J. Li, ConRPG: paraphrase generation using contexts as regularizer, in: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Online and Punta Cana, Dominican Republic, Association for Computational Linguistics, 2021, pp. 2551–2562, https://aclanthology.org/2021.emnlp-main.199, https://doi.org/10.18653/v1/2021.emnlp-main.199.
[24] G. Michalopoulos, I. McKillop, A. Wong, H. Chen, Lexsubcon: Integrating knowledge from lexical resources into contextual embeddings for lexical substitution, 2022.
[25] E. Pavlick, C. Callison-Burch, Simple ppdb: a paraphrase database for simplification, in: ACL, 2016, pp. 143–148.
[26] E. Pavlick, P. Rastogi, J. Ganitkevitch, B. Van Durme, C. Callison-Burch, Ppdb 2.0: better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification, in: ACL, 2015, pp. 425–430.
[27] W. Qi, W. Guo, T. Zhang, Y. Liu, Z. Guo, X. Fang, Robust authentication for paper-based text documents based on text watermarking technology, Math. Biosci. Eng. 16 (2019) 2233–2249.
[28] J. Qiang, Y. Li, Y. Zhu, Y. Yuan, Y. Shi, X. Wu, Lsbert: lexical simplification based on bert, IEEE/ACM Trans. Audio Speech Lang. Process. 29 (2021) 3064–3076.
[29] J. Qiang, Y. Li, Y. Zhu, Y. Yuan, X. Wu, Lexical simplification with pretrained encoders, AAAI (2020) 8649–8656.
[30] J. Qiang, X. Lu, Y. Li, Y. Yuan, X. Wu, Chinese lexical simplification, IEEE/ACM Trans. Audio Speech Lang. Process. 29 (2021) 1819–1828.
[31] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P.J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, arXiv preprint, arXiv:1910.10683, 2019.
[32] S.G. Rizzo, F. Bertini, D. Montesi, Content-preserving text watermarking through unicode homoglyph substitution, in: Proceedings of the 20th International Database Engineering & Applications Symposium, 2016, pp. 97–104.
[33] T. Sellam, D. Das, A.P. Parikh, Bleurt: learning robust metrics for text generation, in: Proceedings of ACL, 2020.
[34] G. Szarvas, R. Busa-Fekete, E. Hüllermeier, Learning to rank lexical substitutions, in: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, 2013, pp. 1926–1932.
[35] Y.C. Tam, Cluster-based beam search for pointer-generator chatbot grounded by knowledge, Comput. Speech Lang. 64 (2020) 101094.
[36] M. Topkara, C.M. Taskiran, E.J. Delp III, Natural language watermarking, in: Security, Steganography, and Watermarking of Multimedia Contents VII, SPIE, 2005, pp. 441–452.
[37] U. Topkara, M. Topkara, M.J. Atallah, The hiding virtues of ambiguity: quantifiably resilient watermarking of natural language text through synonym substitutions, in: Proceedings of the 8th Workshop on Multimedia and Security, 2006, pp. 164–174.
[38] A.K. Vijayakumar, M. Cogswell, R.R. Selvaraju, Q. Sun, S. Lee, D. Crandall, D. Batra, Diverse beam search for improved description of complex scenes, in: AAAI, 2018.
[39] J. Wieting, K. Gimpel, Paranmt-50m: pushing the limits of paraphrastic sentence embeddings with millions of machine translations, arXiv preprint, arXiv:1711.05732, 2017.
[40] C. Xiao, C. Zhang, C. Zheng, Fontcode: embedding information in text documents using glyph perturbation, ACM Trans. Graph. 37 (2018) 1–16.

[41] X. Yang, J. Zhang, K. Chen, W. Zhang, Z. Ma, F. Wang, N. Yu, Tracing text provenance via context-aware lexical substitution, in: AAAI, 2022.
[42] W. Yuan, G. Neubig, P. Liu, Bartscore: evaluating generated text as text generation, in: M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, J.W. Vaughan (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc., 2021, pp. 27263–27277, https://proceedings.neurips.cc/paper/2021/file/e4d2b6e6fdeca3e60e0f1a62fee3d9dd-Paper.pdf.
[43] D. Yuret, Ku: word sense disambiguation by substitution, in: Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007), 2007, pp. 207–214.
[44] W. Zhou, T. Ge, K. Xu, F. Wei, M. Zhou, Bert-based lexical substitution, in: ACL, 2019, pp. 3368–3373.