Generating watermarked adversarial texts

Mingjie Li, Hanzhou Wu, Zichi Wang, and Xinpeng Zhang*
Shanghai University, School of Communication and Information Engineering, Shanghai, China

Abstract. Adversarial example generation (AEG) has been a hot spot in recent years because it can cause deep neural networks (DNNs) to misclassify the generated adversarial examples, which reveals the vulnerability of DNNs, motivating us to find good solutions to improve the robustness of DNN models. Due to the extensiveness and high liquidity of natural language over the social networks, various natural language-based adversarial attack algorithms have been proposed in the literature. These algorithms generate adversarial text examples with high semantic quality. However, the generated adversarial text examples and the corresponding attack models may be maliciously or illegally used. To tackle this problem, we present a general framework encapsulated in the cloud application programming interfaces (APIs) for generating watermarked adversarial text examples to protect adversarial text examples and corresponding adversarial text attack models. For each word in a given text, a set of candidate words are determined to ensure that all the words in the set can be used to carry secret bits or facilitate the construction of adversarial example. By applying a word-level adversarial text generation algorithm, the watermarked adversarial text example can be finally generated. Experiment results show that the adversarial text examples generated by the proposed method not only successfully fool advanced DNN models, but also carry watermarks that can effectively verify the ownership and trace the source of the adversarial examples and the corresponding attack models. Moreover, the watermark can still survive after attacked with AEG algorithms, which has shown the applicability and superiority. © 2023 SPIE and IS&T [DOI: 10.1117/1.JEI.32.2.023023]

Keywords: adversarial examples; watermarking; text; natural language; deep learning.

1 Introduction

Although deep neural networks (DNNs) have achieved great success in many tasks, such as computer vision and natural language processing (NLP), they are vulnerable to adversarial examples. Adversarial examples are inputs (to DNN models) intentionally designed by an attacker to cause a target model to make incorrect outputs. They can be generated by applying a hardly perceptible perturbation to the original samples. From the perspective of defense, adversarial examples can be used for improving the robustness of DNNs, e.g., by mixing adversarial examples into the training set.

Many advanced adversarial example generation (AEG) algorithms have been proposed in recent years. For example, digital images have been widely adopted for AEG because the majority of DNNs are originally designed for visual tasks, which require the input data to be images. Mainstream image-based AEG algorithms produce adversarial images by modifying the pixels. The adversarial image example will not introduce noticeable artifacts since the modification degree is highly slight. Generating speech adversarial examples has also attracted increasing interest because of the wide application of speech recognition in daily life. For example, by inserting a well-designed noise to a speech signal, the speech adversarial example can cause a target speech recognition system to output any specified sentence. Some representative speech AEG algorithms can be found in references.

In NLP, due to the widespread use of DNNs, there are also increasing concerns about the security of NLP systems, among which research on adversarial natural language (text) examples has become more important. However, due to the discrete nature of text content, gradient
perturbation\(^7,8\) and other AEG algorithms originally designed to images and speech signals cannot be directly applied to texts. In human perception, it is difficult to understand perturbed texts, whereas slight modifications to image pixels can still yield meaningful images. Therefore, generating adversarial texts is a very challenging topic. Mainstream text-based AEG algorithms can be divided to character-level adversarial attacks,\(^18-20\) word-level adversarial attacks,\(^21-23\) and sentence-level adversarial attacks.\(^{24-26}\) Among them, word-level AEG models perform comparatively well on both attack efficiency and adversarial text quality.\(^21,27\)

Adversarial examples can be crafted to cause a target DNN model to misclassify. In contrast, as mentioned above, adversarial examples can be also used for constructing robust DNN models and improving the interpretability of models. For example, they can be employed during DNN training to supply new training data from which the model might benefit.\(^28\) In this sense, adversarial examples should be protected. As AEG models become more and more mature, it will surely raise concerns that adversarial examples may be maliciously or illegally used, such as generating spam text messages that make the e-mail classifier-based DNNs\(^29\) classify wrongly and generating viewing opinions, which misleads sentiment analyzer classification errors. It is therefore necessary to seek solutions to protect adversarial examples.

The aforementioned analysis has motivated us to study how to protect adversarial text attack models. In particular, we will focus on protecting adversarial text examples and the corresponding attack models in this paper. A straightforward idea is to build sophisticated access control protocols to protect adversarial text examples, which, however, has limited control after the adversarial text examples have been shared with authorized users. An alternative option is digital watermarking,\(^30\) which embeds secret data into a cover signal by slightly modifying the noisy component of the cover.\(^31,32\) By extracting secret data from the watermarked signal, we can identify the ownership and trace the source of the watermarked signal. Obviously, it is quite desirable to use digital watermarking for adversarial text examples as long as the watermarked adversarial text examples well maintain the attack efficiency and adversarial text quality.

One may use traditional text watermarking algorithms\(^33-37\) directly to mark adversarial text examples, however, this is not suitable for practice. The reason is, generating adversarial text examples and watermarking texts are two independent tasks. Performing well on one task does not mean it results in good performance on the other task. For example, after embedding a secret watermark, the watermarked “adversarial text example” may be no longer adversarial. In contrast, comparing with watermarking, “adversarial” watermarking requires more redundancy of the original text (which is often highly encoded), which makes it more difficult to embed a watermark. Therefore, generating watermarked adversarial text examples is an important yet very challenging task.

In this paper, we first present a novel general framework encapsulated in the cloud application programming interfaces (APIs) to embed a secret watermark into a given text combining an adversarial text generation algorithm to protect adversarial text example and the corresponding adversarial text attack model. In the proposed work, given the original text that is not adversarial, for each word to be probably modified, a set of candidate words are determined to ensure that all the words in the set can be used to replace the present word to carry secret data or construct the adversarial text. By applying a word-level adversarial text generation algorithm, the generated text is not only adversarial, but also carries a watermark that can verify the ownership and trace the source of the adversarial text. Experimental results have demonstrated the superiority and applicability.

The contributions of the paper are as follows:

1. We first present a novel general framework that can be encapsulated in the cloud APIs to protect adversarial text examples and the corresponding attack model.
2. By combining a mainstream word-level adversarial text generation algorithm, the generated texts by our proposed method is not only adversarial, but also carries watermarks that can verify the ownership and trace the source of the adversarial text examples and the corresponding adversarial text attack model.
3. We conduct sufficient experiments to demonstrate that the adversarial text examples generated by the proposed method not only successfully fool advanced DNN models, but also carry watermarks that can effectively verify the ownership and trace the source of the adversarial examples.
4. In the experiments, we also demonstrate the strong robustness of embedding watermarks in generated adversarial text examples.

2 Preliminaries

In this section, we briefly review the advances in adversarial text generation and text watermarking so that we can better introduce the proposed work in the subsequent section.

2.1 Adversarial Text Generation

More adversarial text generation models are proposed in recent years, which ranging from character-level\textsuperscript{18–20} to sentence-level.\textsuperscript{24–26} Character-level models usually generate adversarial texts by modifying individual characters in the original texts. Though character-level adversarial text generation models have been proven to be effective, they may cause the perceptibility problem because changing individual characters often leads to invalid words, which can be easily recognized by humans. Moreover, character-level adversarial attacks are easy to be defended, e.g., preprocessing the input text with a spell-checking tool can resist against such attacks.\textsuperscript{38} Sentence-level attacks can be realized by generating new sentences, such as inserting an additional sentence to the original text\textsuperscript{39} and rewriting the original sentence by an encoder-decoder model.\textsuperscript{26}

Different from character-level/sentence-level attacks, word-level attacks generate adversarial text examples by replacing some individual words with new words in the original texts, resulting in new texts that have the same semantics to the original ones. On the one hand, synonym replacement keeps the semantic distortion between the adversarial text and the original text within a low level. On the other hand, because only a part of words are to be replaced, the computational complexity is low as well by applying an efficient optimization algorithm. In addition, it has been demonstrated that word-level attacks achieve superior performance in fooling target models,\textsuperscript{21} which motivates us to focus on word-level attacks in this paper.

The goal of a word-level adversarial text generation model is to craft adversarial text examples using a limited vocabulary that can successfully fool the victim models, which actually requires us to solve a combinatorial optimization problem. The implementation of word-level adversarial text generation models is generally divided into two key steps: (1) reducing search space and (2) searching for adversarial texts.\textsuperscript{21} In this subsection, we briefly introduce three mainstream word-level adversarial text generation models, which are “Embedding/LM + Genetic,”\textsuperscript{22} “Synonym + Greedy,”\textsuperscript{40} and “Sememe + PSO.”\textsuperscript{21} These models will be used in our experiments.

1. Embedding/LM + Genetic: Alzantot et al.\textsuperscript{22} used a population-based black box optimization algorithm to generate adversarial texts with semantics and syntax similar to the original texts, which are capable of fooling well-trained victim models. In the algorithm, the restriction on vocabulary embedding distance and the restriction on language model prediction score are combined to reduce the search space. For searching suitable words, a popular metaheuristic population evolution algorithm is used. Experimental results show that they achieve good performance in attacking well-trained victim models.

2. Synonym + Greedy: Ren et al.\textsuperscript{40} proposed a new word replacement order determined by word saliency and classification probability based on the synonyms\textsuperscript{41} replacement strategy and present a probability-weighted word saliency greedy algorithm to form the adversarial text generation model. Experiments show that the generated adversarial texts greatly reduce the accuracy of the text classification model under the condition of using low replacement rates that are difficult for humans to perceive.

3. Sememe + PSO: due to inappropriate word space reduction methods and undesirable adversarial text search methods, word-level adversarial text generation models can be further improved. By combining the sememe-based word substitution method with the particle swarm optimization based search algorithm, Zang et al.\textsuperscript{21} proposed a new word-level adversarial text generation model. Through experiments, it is known that the model obtains the higher attack success rates, and the generated adversarial texts are of the higher quality.
2.2 Text Watermarking

Text watermarking (or called linguistic watermarking) can be modeled as a communication problem. A data encoder embeds secret information in a cover text by slightly modifying the cover. The resulting text containing hidden information (also called marked text) should be sent to a data decoder and may be attacked during transmission. After receiving the probably attacked and marked text, the data decoder is able to reliably extract the embedded information for ownership verification or other purposes.

Text watermarking is often evaluated by the rate-distortion performance. For a payload, the distortion between the cover text and the marked text should be as low as possible. The term “distortion” often measures the semantic difference between the cover text and the marked text. In contrast, when the distortion is kept within a fixed level, it is desirable to embed information as much as possible. In addition, because the marked text may be altered by an adversary, it is further required that a text watermarking technique should be robust to malicious attacks so that secret information can be still extracted from the attacked marked text for ownership protection. Text watermarking can be realized by modifying text format, e.g., secret bits can be embedded by adjusting the white space between two words or lines. However, these format-based algorithms have limited robustness because the text format can be easily changed by an intentional attacker, resulting in a high bit error rate. Mainstream algorithms exploit the syntactic and semantic nature of texts for watermark embedding. They emphasize that the critical semantic information of the marked texts are consistent with the original ones. For example, secret bits can be embedded into a text by replacing specified words in the text with semantically similar words. In this paper, we will study text watermarking along this direction to adapt to the construction of adversarial texts.

3 Proposed Method

As shown in Fig. 1, the proposed work embeds a secret watermark during generating the adversarial text example, which considers text watermarking and adversarial text generation as a whole and achieves superior performance in text watermarking and adversarial text generation. For data embedding, the proposed framework first processes the cover text (i.e., the original text) to construct a substitution set for each word in the text. The substitution set of a word in the specific position of the cover text includes an indefinite number of candidate words that can be used to replace the original word to construct the watermarked adversarial text example. In this way, according to the key and the substitution sets, the secret watermark can be successfully embedded into the cover text through a word-level adversarial text generation procedure.

**Fig. 1** Sketch for the proposed adversarial text watermarking system. (a) Watermarked adversarial text generation and (b) watermark extraction.
The resulting new text not only fools the specific DNN model, but also carries a secret watermark identifying its ownership. With the secret key, the secret watermark can be easily reconstructed from the watermarked adversarial text, without the need of constructing the substitution sets. In the following section, we show more details.

### 3.1 Adversarial Text Watermarking

Given a cover text $x = \{x_1, x_2, \ldots, x_n\}$, $n \geq 1$, the proposed watermarked adversarial text generation procedure produces a watermarked adversarial text $y = \{y_1, y_2, \ldots, y_n\}$ from which a secret watermark $w = \{w_1, w_2, \ldots, w_m\} \subset \{0, 1\}^m$, $m \geq 1$ can be retrieved according to a secret key. To achieve this goal, we first preprocess the cover text and then apply a word-level adversarial text generation algorithm to the cover text.

#### 3.1.1 Word-wise preprocessing

The proposed word-wise preprocessing procedure aims to construct a substitution set $S(x_i)$ for each $x_i \in x$. All words in $S(x_i)$ are semantically close to $x_i$ so that $x_i$ can be changed to $y_i \in S(x_i)$ in the subsequent word-level adversarial text generation procedure. It is possible that $y_i = x_i$, i.e., $x_i$ can be unchanged. Let $I = \{1, 2, \ldots, n\}$ be an index set. Thus, each word $x_i \in x$ can be uniquely indexed by $i \in I$. It is possible for two different $i \neq j$ that $x_i = x_j$, i.e., $x_i$ and $x_j$ are corresponding to the same word. To construct $S(x_1), S(x_2), \ldots, S(x_n)$, $I$ is partitioned into three disjoint subsets $I_1$, $I_2$ and $I_3$. In other words, we always have $I_1 \cap I_2 = \emptyset$, $I_1 \cap I_3 = \emptyset$, $I_2 \cap I_3 = \emptyset$ and $I_1 \cup I_2 \cup I_3 = I$.

Assuming that, $I_1$, $I_2$, and $I_3$ have been previously determined. The words corresponding to $I_1$ will be unchanged. It indicates that $S(x_i) = \{x_i\}$ for all $i \in I_1$. For each $i \in I_2 \cup I_3$, a list of candidate words $T(x_i) = \{t_{i,1}, t_{i,2}, \ldots, t_{i,n}\}$ can be collected from a large-scale vocabulary so that each word in $T(x_i)$ is semantically consistent with $x_i$. For example, one may determine $T$ (“see”) = {“see”, “look”, “watch”}. For each $i \in I_2$, we set $S(x_i) = T(x_i)$. For each $i \in I_3$, we determine $S(x_i)$ based on $T(x_i)$ and the secret data to be embedded.

Specifically, for each $i \in I_3$, we use a mapping function $f$ to map each word $e \in T(x_i)$ to a binary string that has a length of $l > 0$. In this way, $T(x_i)$ can be partitioned into a total of $2^l$ disjoint subsets according to the mapped binary strings. Let $T_0(x_i), T_1(x_i), \ldots, T_{2^l-1}(x_i)$ be the $2^l$ subsets. $T_v(x_i)$ contains all words that are mapped to a binary string whose decimal value is $v$, e.g., if $l = 3$ and $f$ (“see”) = (101)2, “see” will be an element of $T_3$ (“see”) since ((101)2)10 = 5. Obviously, we have $T_a(x_i) \cap T_b(x_i) = \emptyset$ for any $0 \leq a < b \leq 2^l-1$ and $T(x_i) = \bigcup_{v=0}^{2^l-1} T_v(x_i)$. Only those words (in $x$) corresponding to $I_1$ will be used to carry the secret data. Assuming that we need to embed a binary stream $w_i$ (that has a length of $l$) into some $x_i (i \in I_3)$, we set $S(x_i) = T_{d(w_i)}(x_i)$, where $d(w_i)$ is the decimal value of $w_i$, e.g., $d$ (“011”) = 3. Algorithm 1 shows the pseudocode to construct $S(x_1), S(x_2), \ldots, S(x_n)$ and an example of the proposed word-wise preprocessing is shown in the Fig. 2.

#### 3.1.2 Word-level adversarial text generation

Once the substitution sets are determined, we are able to generate $y$ through an efficient word substitution based adversarial text generation algorithm. That is, the adversarial text generation algorithm is to select exactly one word from each $S(x_i)$, $i \in I$, to construct $y$, so that $y$ is an adversarial text example. In other words, we have $y_i \in S(x_i)$ for all $1 \leq i \leq n$. Because all words in $S(x_i)$ ($i \in I_3$) are mapped to the same secret binary stream, each $y_i (i \in I_3)$ surely carries a secret binary stream, which indicates that, $y$ is also marked after text generation.

#### 3.1.3 Watermark extraction

To verify the ownership, all words corresponding to $I_3$ carrying secret binary streams can be collected from the adversarial text example. Then, all the secret binary streams can be extracted from these words and further concatenated to form the entire watermark $w$ that can be used for
Algorithm 1  Word-wise processing procedure.

Input: cover text \( x \), secret watermark \( w \).

Output: substitution sets \( S(x_1), S(x_2), \ldots, S(x_n) \).

1: determine \( I_1, I_2, I_3 \) (using Algorithm 2)
2: for each \( i \in I_1 \) do
3: set \( S(x_i) = \{ x_i \} \)
4: end for
5: for each \( i \in I_2 \) do
6: Determine \( T(x_i) \) from a vocabulary
7: Set \( S(x_i) = T(x_i) \)
8: end for
9: for each \( i \in I_3 \) do
10: Determine \( T(x_i) \) from a vocabulary
11: Determine \( T_0(x_i), T_1(x_i), \ldots, T_{2^{l-1}}(x_i) \) using \( f \)
12: Determine the present secret binary stream \( w_i \) (that has a length of \( l \)) to be embedded from \( w \)
13: Set \( S(x_i) = T_{d(w_i)}(x_i) \)
14: end for
15: return \( S(x_1), S(x_2), \ldots, S(x_n) \).

Fig. 2 An example of the proposed word-wise preprocessing, which aims to construct a substitution set \( S(x_i) \) for each word \( x_i \) in the input text, and \( I = \{1,2,3,4\} \) be the index set. In phase 1, \( I \) is partitioned into three disjoint subsets \( I_1, I_2, \) and \( I_3 \) according to the Algorithm 2. In phase 2, generate candidate set \( T(x_i) \) for each word in \( I_2 \) and \( I_3 \). In phase 3, generate substitution set \( S \). \( S(x_i) = x_i \) for all \( i \in I_1 \); for each \( i \in I_2 \), set \( S(x_i) = T(x_i) \); and for each \( i \in I_3 \), we determine \( S(x_i) \) based on \( T(x_i) \) and the secret data to be embedded.
ownership verification. Figure 3 shows a watermarked adversarial text example generated by our proposed method and the watermark extraction process.

Remark 1: \( l \) is predetermined according to the secret key. A smaller \( l \) for a fixed \( I \) indicates that the maximum number of embeddable bits is smaller, but the text quality may be better because more candidate words can be used. A larger \( l \) provides a larger embedding capacity,

Algorithm 2 Construction of disjoint subsets.

Input: cover text \( x \), text generation algorithm, secret key, the bit-length of the secret watermark \( L \), parameter \( l \).

Output: disjoint subsets \( I_1, I_2, I_3 \).

1: initialize \( J = I_1 = I_2 = I_3 = \emptyset \) and \( l = \{1,2,\ldots,n\} \)

2: for each \( i \in l \) do

3: \hspace{1cm} if \( x_i \in x \) is modifiable according to text generation algorithm then

4: \hspace{2cm} Set \( J = J \cup \{i\} \)

5: \hspace{1cm} else

6: \hspace{2cm} Set \( I_1 = I_1 \cup \{i\} \)

7: \hspace{1cm} end if

8: end for

9: Randomly select \( L/l \) elements out from \( J \) to constitute \( I_3 \) according to the secret key (notice that \( L/l \leq |J| \))

10: Set \( I_2 = I_2 \setminus J \)

11: return \( I_1, I_2, I_3 \)

Fig. 3 A watermarked adversarial text example generated by our proposed method and watermark extraction. Phase 1 shows proposed word-wise preprocessing, which aim to construct a substitution set \( S(x_i) \) for each word \( x_i \) in the input text, and it is described in detail in Fig. 2. In phase 2, combine the substitution set \( S(x_i) \) and word-level adversarial text generation model generate watermarked adversarial text. In phase 3, we combine the secret and the watermark extraction model to extract watermark from the generated watermarked adversarial text.
but may cause some substitution sets to be empty that may lead to failed embedding. Following previous methods that use a word to carry at most one bit, we will also use \( l = 1 \) throughout this paper.

**Remark 2:** It is free for us to design the mapping function \( f \). For simplicity, we use the Hash function MD5\(^{37} \) to construct \( f \). In detail, given a word, we first compute its MD5 value and then use the \( l \) least bits to denote the mapped binary stream. Obviously, one may also use a secret key to decide what bits should be taken out to constitute the mapped binary stream.

### 3.2 Construction of Disjoint Subsets

It is open for us to determine \( I_1, I_2, \) and \( I_3 \) in advance as long as \( I_3 \) can fully carry the watermark information and the data receiver can determine \( I_1 \) for extracting the watermark. For example, one may use a secret key to randomly produce \( I_1, I_2, \) and \( I_3 \), which, however, does not take advantage of the subsequent word-level adversarial text generation algorithm and therefore may limit the adversarial text quality because word-level adversarial text generation algorithms often require us to take into account the part of speech of the word to be modified to generate adversarial texts with high quality. To this end, we present an efficient strategy for constructing \( I_1, I_2, \) and \( I_3 \). In detail, we first determine the part of speech of each word in \( x \), and then add the index of each word to the corresponding set according to the part of speech of the word. For example, assuming that there are at most \( n_p > 0 \) parts of speech in a text, we therefore initialize \( n_p \) index sets \( P_1, P_2, \ldots, P_{n_p} \) as empty for \( x \), e.g., one may use \( P_1 \) to include all indexes of words (in \( x \)) that are nouns, \( P_2 \) to include all indexes of words (in \( x \)) that are adjectives and so on. Then, for each \( x_i \in x, 1 \leq i \leq n \), we determine the part of speech of \( x_i \) and then add \( i \) to the corresponding index set \( P_{s(i)} \), where \( s(i) \) means the index of the part of speech set of \( x_i \). All words corresponding to \( P_{s(i)} \) have the same part of speech. It can be inferred that, \( P_j \cap P_k = \emptyset \) for all \( j \neq k \) and \( \bigcup_{j=1}^{n_p} P_j = I \).

Suppose that, the used word-level adversarial text generation algorithm \( A \) only allows us to modify words with specific parts of speech, e.g., only modifying verbs and nouns. Let \( J \subset I \), we define \( Q(J) = \bigcup_{i \in J} P_{s(i)} \). Obviously, \( Q(I) = I \). Without the loss of generalization, there must exist exactly one \( J \subset I \) that \( Q(J) \) corresponds to all modifiable words in \( x \) given \( A \). It is inferred that, \( |Q(J)| \leq |I| = n \), where \( | \cdot | \) means the size of a set. Thus, we can set \( I_1 = I \setminus Q(J) \) and \( I_2 \cup I_3 = Q(J) \). Let \( L \) be the bit-length of the secret watermark. If we use each word to carry exactly \( l \) bits, a secret key can be used to randomly select \( L/l \) elements from \( Q(J) \) to constitute \( I_3 \). The rest elements will be then used to constitute \( I_2 \). In this way, the three disjoint subsets are successfully constructed. From the implementation of view, Algorithm 2 shows the pseudocode to construct \( I_1, I_2, \) and \( I_3 \) from which we can find that the time complexity is linear with respect to \( n \).

### 4 Experimental Results and Analysis

In this section, we are to conduct extensive experiments and we evaluate proposed model according to the analysis of the experimental results.

#### 4.1 Datasets and Models

We choose two popular benchmark datasets, i.e., IMDB\(^{48} \) and SST-2\(^{49} \) to evaluate the performance of the proposed method. The IMDB dataset includes 25,000 training texts and 25,000 testing texts for movie reviews, labeled as positive and negative. The SST-2 dataset includes 6920 training texts, 872 validation texts and 1821 testing texts. Both datasets are commonly used in binary sentiment classification. However, the average sentence length of the texts in the SST-2 dataset is much shorter than that in the IMDB dataset. Therefore, it is more difficult to generate adversarial texts on SST-2. We adopt two widely used sentence encoding models bidirectional LSTM (BiLSTM)\(^{50} \) and BERT\(^{51} \) as the victim models. In our simulation, the hidden
states of the BiLSTM is 128-dimensional and the 300-dimensional pretraining GloVe\textsuperscript{52} was used for word embedding. Additionally, we use three above-mentioned adversarial text generation models, namely Embedding/LM + Genetic\textsuperscript{22}, Synonym + Greedy\textsuperscript{40}, and Sememe + PSO\textsuperscript{21} for generating the watermarked adversarial texts. Obviously, these algorithms can be used to generate the corresponding adversarial texts without any hidden information, i.e., the adversarial texts are non-watermarked.

4.2 Basic Settings and Evaluation Metrics

In experiments, we empirically set the population size to 60 and the maximum number of iterations to 20 for the two population-based models “Embedding/LM + Genetic” and “Sememe + PSO.” The other hyper-parameters for the three word-level adversarial text generation models are consistent with the recommended hyper-parameters in the original papers. For training BiLSTM, we set the epoch to 20 and the batch size to 64. We use TensorFlow for simulation and NVIDIA TITAN RTX 24GB GPU to accelerate model training.

We evaluate the proposed method in terms of two aspects, i.e., adversarial text attack performance and text watermarking performance if not otherwise specified. The adversarial attack performance is measured on the attack success rates on victim models. The attack success rate is defined as the percentage of attack attempts that make the victim model output the target label. The watermarking performance is mainly evaluated by the quality of the generated watermarked adversarial texts, the watermark payload and the watermark detection accuracy. We use four metrics to evaluate the quality of the watermarked adversarial texts, i.e., modification rate, grammaticality, fluency and human evaluation. The modification rate is defined as the percentage of the modified words in the watermarked adversarial text. We measure the grammaticality by counting the increase rate of grammatical errors in the watermarked adversarial text (compared with the original text) with the LanguageTool\textsuperscript{53}. The fluency of each watermarked adversarial text will be measured by the language model perplexity (PPL) with the help of GPT-2\textsuperscript{54}. Notice that, both adversarial attack and watermarking require that the generated texts should have good quality.

4.3 Performance Evaluation on Adversarial Attack

Due to the limited computational resource and large search space for adversarial text construction, for each experiment, we randomly select 500 correctly classified texts from the corresponding test sets to generate watermarked adversarial texts to evaluate our algorithm. During the experiment, we limit the length of each original text in IMDB dataset to the range [10, 120] for efficient evaluation. We set the watermark payload size to the range [0.05, 0.2] to ensure that both the attack success rate and the quality of the watermarked adversarial text are high. Here, the payload size is defined as the ratio between the number of watermark bits and the length of the original text, i.e., $L/n$. The corresponding non-watermarked adversarial texts are generated as well under the same experimental condition for fair comparison. The running time of the proposed method based on different attack models are shown in Table 1. We observed that the running time of IMDB is usually less than the running time of SST-2, this demonstrates that it is more difficult to generate watermarked adversarial text on SST-2 because the average length of this dataset is shorter.

Tables 2 and 3 show examples of using the proposed method for text generation. In Tables 2 and 3, “0.05 bpw” represents the payload size, i.e., each word carries 0.05 bits. Here, “bpw” is short for bits per word. From Tables 2 and 3, we can find that the original text can be modified with a few words to become an adversarial text, which verifies the superiority of the used AEG algorithm. In contrast, we can find that the word-modification rates become higher when the payload size becomes larger, which is reasonable because a larger payload size means that more words are likely to be modified to carry more secret bits.

The attack success rates of the proposed method based on different attack models are shown in Table 4. Comparing with the non-watermarked adversarial texts (i.e., the payload size is 0), the success rates for the watermarked adversarial texts are relatively lower. We observe that with the increase of the watermark payload size, the attack success rates of the proposed method based on
Table 1  Comparison of the running time of proposed method based on different attack models with different watermarking payload when attacking the victim model BiLSTM and BERT.

| Victim model | Dataset | Attack model                  | Running time (h) |
|--------------|---------|-------------------------------|-----------------|
| BiLSTM       | IMDB    | Embedding/LM + Genetic        | 13.6            |
|              |         | Synonym + Greedy              | 12.8            |
|              |         | Sememe + PSO                  | 7.3             |
|              | SST-2   | Embedding/LM + Genetic        | 16.2            |
|              |         | Synonym + Greedy              | 18.5            |
|              |         | Sememe + PSO                  | 9.6             |
| BERT         | IMDB    | Embedding/LM + Genetic        | 13.9            |
|              |         | Synonym + Greedy              | 12.9            |
|              |         | Sememe + PSO                  | 9.2             |
|              | SST-2   | Embedding/LM + Genetic        | 18.0            |
|              |         | Synonym + Greedy              | 18.4            |
|              |         | Sememe + PSO                  | 10.5            |

Table 2  Examples of applying the proposed method (with “Sememe + PSO”) to attack BiLSTM on the IMDB dataset. The modified words in the watermarked adversarial texts are highlighted with underlines. “bpw” means “bits per word.”

Original text prediction = “negative”
This movie was perhaps the biggest waste of 2 hrs of my life. From the opening 10 min, I was ready to leave. The cliches there slapping you in the face, and the plot was not only predictably stupid, but full of more holes than swiss cheese. I am considering suing for that lost 2 hrs, and $6.25 along with the fact that I am now stupider for watching this waste of film. The T-Rex’s must be flipping in their graves, so to speak.

Watermarked adversarial texts prediction = “positive”

| Watermarked adversarial texts | bpw | Original text prediction = “negative” | Running text |
|-------------------------------|-----|---------------------------------------|--------------|
| 0.05 bpw                      | This movie was perhaps the biggest waste of 2 hrs of my life. From the opening 10 min, I was ready to leave. The cliches there slapping you in the face, and the plot was not only predictably stupid, but full of more holes than swiss cheese. I am considering suing for that lost 2 hrs, and $6.25 along with the fact that I am now stupider for watching this waste of film. The T-Rex’s must be flipping in their graves, so to speak. |
| 0.10 bpw                      | This movie was perhaps the biggest waste of 2 hrs of my life. From the opening 10 min, I was ready to leave. The cliches there slapping you in the face, and the plot was either only predictably stupid, but full of more holes than swiss cheese. I am rapprochement suing for that lost 2 hrs, and $6.25 along with the fact that I am now stupider for watching this waste of film. The T-Rex’s must be flipping in their graves, so to speak. |
| 0.15 bpw                      | This film was perhaps the biggest waste of two yorker of my life. From the opening 10 min, I was ready to leave. The cliches there civilian you in the face, and the clown was not only predictably stupid, but full of more holes than swiss cheese. I am considering suing for that lost 2 hrs, and $6.25 along with the fact that I am now stupider for watching this waste of film. The T-Rex’s must be flipping in their graves, so to speak. |
| 0.20 bpw                      | This movie was perhaps the biggest waste of two yorker of my life. From the opening 10 min, I was ready to leave. The cliches there civilian you in the face, and the clown was not only predictably stupid, but full of more holes than swiss cheese. I am rapprochement suing for that lost 2 hours, and $6.25 along with the sub that I am now stupider for watching this waste of film. The T-Rex’s must be flipping in their graves, so to speak. |
Different attack models gradually decrease when attacking the victim models on different datasets. The attack success rates on IMDB are generally higher than that on SST-2 under the same conditions. It reveals that it is more difficult to generate watermarked adversarial texts on SST-2, which is due to the shorter average length of the text examples. We can easily find that the attack success rates of the proposed method based on different attack models can achieve higher than 30% in all cases when attacking the victim models, which reveals the vulnerability of DNNs and demonstrates the superiority of the AEG procedure. It also indicates that the proposed watermarking strategy does not significantly impair the attack performance.

Table 3 Examples of applying the proposed method (with “Sememe + PSO”) to attack BiLSTM on the SST dataset.

| Victim model | Dataset | Attack model      | Non-watermarked | Watermarked with our watermarking method (bpw) |
|--------------|---------|------------------|-----------------|-----------------------------------------------|
| BiLSTM       | IMDB    | Embedding/LM + Genetic | 87.30           | 86.90 86.50 71.70 69.10                     |
|              |         | Synonym + Greedy  | 90.20           | 70.10 52.60 38.40 38.60                      |
|              |         | Sememe + PSO     | 100.00          | 88.40 82.70 64.50 55.20                      |
| SST-2        | Embedding/LM + Genetic | 69.10           | 53.30 55.60 50.90 45.20                     |
|              | Synonym + Greedy  | 70.20           | 60.90 58.60 53.20 50.40                      |
|              | Sememe + PSO     | 89.80           | 75.70 55.40 51.30 50.70                      |

Table 4 Comparison of the attack success rates (%) of the non-watermarked adversarial texts and watermarked with our watermarking method based on different attack models with different watermarking payload when attacking the victim model BiLSTM and BERT.

| Victim model | Dataset | Attack model      | Non-watermarked | Watermarked with our watermarking method (bpw) |
|--------------|---------|------------------|-----------------|-----------------------------------------------|
| BiLSTM       | IMDB    | Embedding/LM + Genetic | 88.40           | 82.10 85.20 71.80 68.40                     |
|              |         | Synonym + Greedy  | 70.60           | 60.60 46.90 34.70 30.50                      |
|              |         | Sememe + PSO     | 94.50           | 86.30 79.30 58.70 48.50                      |
| SST-2        | Embedding/LM + Genetic | 60.10           | 53.50 46.20 40.50 36.10                     |
|              | Synonym + Greedy  | 62.80           | 59.70 58.10 49.40 48.20                      |
|              | Sememe + PSO     | 85.60           | 73.10 52.90 47.60 48.10                      |
4.4 Performance Evaluation on Text Watermarking

The (mean) modification rates for non-watermarked adversarial texts and watermarked adversarial texts with different watermark payload sizes are shown in Table 5. Comparing with the non-watermarked adversarial texts, the modification rates for the watermarked adversarial texts are relatively higher. Moreover, the modification rates increase as the watermark payload size becomes larger. The modification rates for the watermarked adversarial texts generated on IMDB are generally lower than that on SST-2 under the same conditions. This is also mainly due to the fact that the average length of texts of IMDB is much longer than that of SST-2. Overall, the modification rates are all <14% and the maximum difference between the (mean) modification rate for watermarked texts and that for non-watermarked texts is less than 6%, indicating that the modification degree can be kept within a low level. It means that the proposed framework does not significantly modify the original text, which implies that the adversarial text quality after watermarking can be maintained.

The average increase rates of grammatical errors of the non-watermarked adversarial texts and the watermarked adversarial texts generated by the proposed method are shown in Table 6. Comparing with the non-watermarked adversarial texts, the average increase rates of grammatical errors of the watermarked adversarial texts are relatively higher. And we can clearly see that the average increase rates of grammatical errors of the watermarked adversarial texts gradually increase with the increase of the payload size, and the average increase rates of grammatical errors for adversarial texts generated on the IMDB dataset are generally lower than that on the SST-2 dataset. It is related to the fact that the modification rates increase with the increase of the watermark payload size, and the modification rates on the IMDB dataset are generally lower than that on the SST-2 dataset.

We use PPL to measure the fluency of the generated adversarial texts. It is generally believed that the larger the PPL, the worse the quality of the corresponding text given the original text. The mean PPLs for non-watermarked adversarial texts and watermarked adversarial texts under different conditions are shown in Table 7. Comparing with the non-watermarked adversarial

---

**Table 5** The mean modification rates (%) of the non-watermarked adversarial texts and watermarked with our watermarking method based on different attack models with different watermarking payload when attacking the victim model BiLSTM and BERT.

| Victim model | Dataset | Attack model            | Non-watermarked | Watermarked with our watermarking method (bpw) |
|--------------|---------|-------------------------|-----------------|-----------------------------------------------|
|              |         |                         |                 | 0.05  | 0.1  | 0.15 | 0.2  |
| BiLSTM       | IMDB    | Embedding/LM + Genetic  | 7.01            | 7.14  | 9.26 | 12.03 | 12.97 |
|              |         | Syonym + Greedy         | 6.59            | 6.77  | 6.61 | 9.62  | 10.08 |
|              |         | Sememe + PSO            | 3.32            | 3.50  | 5.05 | 8.26  | 11.14 |
|              | SST-2   | Embedding/LM + Genetic  | 10.28           | 11.99 | 11.31 | 12.37 | 13.32 |
|              |         | Syonym + Greedy         | 9.08            | 10.18 | 11.70 | 12.23 | 13.18 |
|              |         | Sememe + PSO            | 9.29            | 9.32  | 9.74 | 12.21 | 13.10 |
| BERT         | IMDB    | Embedding/LM + Genetic  | 7.07            | 7.09  | 7.13 | 9.08  | 10.15 |
|              |         | Syonym + Greedy         | 3.82            | 4.76  | 6.22 | 7.34  | 8.82  |
|              |         | Sememe + PSO            | 3.31            | 3.34  | 4.75 | 7.26  | 10.06 |
|              | SST-2   | Embedding/LM + Genetic  | 9.27            | 9.35  | 10.06 | 10.04 | 11.06 |
|              |         | Syonym + Greedy         | 8.12            | 8.14  | 9.20 | 9.88  | 10.75 |
|              |         | Sememe + PSO            | 7.76            | 8.02  | 9.16 | 8.34  | 12.03 |
### Table 6
The average increase rates of grammatical errors (%) of the non-watermarked adversarial texts and watermarked with our watermarking method based on different attack models with different watermarking payload when attacking the victim model BiLSTM and BERT.

| Victim model | Dataset | Attack model       | Non-watermarked | Watermarked with our watermarking method (bpw) |
|--------------|---------|-------------------|----------------|----------------------------------------------|
|              |         |                   |                | 0.05  | 0.1  | 0.15 | 0.2  |
| BiLSTM       | IMDB    | Embedding/LM + Genetic | 6.21           | 6.60  | 7.85 | 8.31 | 8.36 |
|              |         | Synonym + Greedy   | 4.96           | 5.07  | 5.01 | 6.27 | 7.43 |
|              |         | Sememe + PSO       | 1.82           | 2.03  | 2.05 | 2.96 | 3.06 |
| SST-2        | Embedding/LM + Genetic | 6.88           | 7.14           | 7.03  | 8.89 | 9.13 |
|              |         | Synonym + Greedy   | 4.98           | 5.15  | 5.82 | 6.90 | 7.22 |
|              |         | Sememe + PSO       | 4.10           | 4.31  | 4.86 | 4.39 | 4.97 |
| BERT         | IMDB    | Embedding/LM + Genetic | 5.65           | 5.73  | 6.98 | 7.19 | 7.68 |
|              |         | Synonym + Greedy   | 4.23           | 4.35  | 4.79 | 5.02 | 6.47 |
|              |         | Sememe + PSO       | 1.75           | 2.13  | 2.16 | 2.73 | 2.85 |
| SST-2        | Embedding/LM + Genetic | 6.17           | 6.26           | 7.01  | 7.36 | 8.72 |
|              |         | Synonym + Greedy   | 4.52           | 4.81  | 5.93 | 6.48 | 6.51 |
|              |         | Sememe + PSO       | 3.26           | 3.28  | 4.20 | 3.79 | 4.65 |

### Table 7
The mean language model PPL of the non-watermarked adversarial texts and watermarked with our watermarking method based on different attack models with different watermarking payload when attacking the victim model BiLSTM and BERT.

| Victim model | Dataset | Attack model       | Non-watermarked | Watermarked with our watermarking method (bpw) |
|--------------|---------|-------------------|----------------|----------------------------------------------|
|              |         |                   |                | 0.05  | 0.1  | 0.15 | 0.2  |
| BiLSTM       | IMDB    | Embedding/LM + Genetic | 112.41         | 113.50 | 129.86 | 150.96 | 154.57 |
|              |         | Synonym + Greedy   | 103.57         | 111.83 | 109.49 | 130.17 | 135.43 |
|              |         | Sememe + PSO       | 81.25          | 99.28  | 101.26 | 130.17 | 135.43 |
| SST-2        | Embedding/LM + Genetic | 287.12         | 345.16         | 315.23 | 380.51 | 472.55 |
|              |         | Synonym + Greedy   | 281.63         | 308.63 | 355.32 | 376.57 | 406.29 |
|              |         | Sememe + PSO       | 256.58         | 316.14 | 323.65 | 349.28 | 375.40 |
| BERT         | IMDB    | Embedding/LM + Genetic | 95.17          | 106.51 | 113.27 | 130.24 | 145.16 |
|              |         | Synonym + Greedy   | 93.54          | 109.30 | 120.65 | 125.13 | 141.67 |
|              |         | Sememe + PSO       | 80.49          | 98.14  | 105.42 | 121.58 | 127.25 |
| SST-2        | Embedding/LM + Genetic | 288.62         | 310.06         | 336.14 | 358.60 | 454.38 |
|              |         | Synonym + Greedy   | 280.70         | 307.35 | 323.61 | 342.93 | 356.32 |
|              |         | Sememe + PSO       | 247.58         | 295.67 | 308.15 | 300.70 | 322.07 |
texts, the (mean) PPLs for the adversarial texts are generally higher. It is seen that the PPLs gradually increase with the increase of the watermark payload size, meaning that the text quality gradually declines. The PPLs tested on the SST-2 dataset are generally higher than that tested on the IMDB dataset. Overall, the (mean) PPLs difference between the watermarked adversarial texts and the non-watermarked adversarial texts can be kept low for low payload sizes, which means that the text quality is satisfactory.

To further measure the naturality of the watermarked adversarial texts generated by the proposed method, we apply the current popular Amazon Mechanical Turk for human evaluation. Specifically, we randomly sample 200 generated watermarked adversarial texts under each condition, and let the annotators to perform a binary sentiment classification and provide a naturality score from 1 to 3 (the higher, the more natural the adversarial text) on each generated watermarked adversarial text and the corresponding original text. The classification accuracy and average naturality score of watermarked adversarial texts and the corresponding original texts of human on IMDB dataset are shown in Tables 8 and 9, respectively. From the human evaluation results in Tables 8 and 9, we observe that the watermarked adversarial texts generated by our proposed method are not much different from the original texts. This also verifies that the quality of the watermarked adversarial texts generated by our proposed method is guaranteed to a certain extent.

**Table 8** The classification accuracy of watermarked adversarial texts and the corresponding original texts of human on IMDB dataset.

| Victim model | Attack model       | Non-watermarked | Watermarked with our watermarking method (bpw) |
|--------------|-------------------|-----------------|-----------------------------------------------|
| N/A          | Original          | 97.5            | 0.05  0.1  0.15  0.2                          |
| BiLSTM       | Embedding/LM + Genetic | 83.0            | 81.5  80.5  76.0  72.5                        |
|              | Synonym + Greedy  | 84.5            | 87.0  84.0  82.5  76.0                        |
|              | Sememe + PSO      | 86.5            | 84.5  85.0  81.5  80.0                        |
| BERT         | Embedding/LM + Genetic | 89.0            | 84.0  82.5  79.5  74.0                        |
|              | Synonym + Greedy  | 80.5            | 77.5  74.0  76.0  73.5                        |
|              | Sememe + PSO      | 87.0            | 85.5  83.5  82.0  78.0                        |

**Table 9** The average naturality score of watermarked adversarial texts and the corresponding original texts of human on IMDB dataset. The higher, the more natural the adversarial text.

| Victim model | Attack model       | Non-watermarked | Watermarked with our watermarking method (bpw) |
|--------------|-------------------|-----------------|-----------------------------------------------|
| N/A          | Original          | 2.425           | 0.05  0.1  0.15  0.2                          |
| BiLSTM       | Embedding/LM + Genetic | 2.290            | 2.265  2.230  2.245  2.235                   |
|              | Synonym + Greedy  | 2.245           | 2.220  2.185  2.150  2.180                   |
|              | Sememe + PSO      | 2.395           | 2.280  2.240  2.260  2.235                   |
| BERT         | Embedding/LM + Genetic | 2.275            | 2.250  2.255  2.230  2.215                   |
|              | Synonym + Greedy  | 2.270           | 2.230  2.210  2.215  2.160                   |
|              | Sememe + PSO      | 2.375           | 2.260  2.235  2.220  2.195                   |
Table 10  The mean accuracy (%) of watermark extraction after attacking watermarked adversarial texts.

| Victim Model | Dataset | Watermark attacking method | Proposed method |  |
|--------------|---------|---------------------------|----------------|---|
|              |         |                           | Embedding/LM + Genetic | Synonym + Greedy | Sememe + PSO |
|              |         |                           | Watermarked with our watermarking method (bpw) | Watermarked with our watermarking method (bpw) | Watermarked with our watermarking method (bpw) |
| BILSTM       | IMDB    | Embedding/LM + Genetic    | 97.51 93.90 92.91 92.46 | 98.16 97.93 94.97 92.70 | 98.53 96.56 91.43 90.40 |
|              |         | Synonym + Greedy          | 98.68 94.22 93.29 92.87 | 98.02 97.81 93.55 92.64 | 99.71 97.26 92.08 91.62 |
|              |         | Sememe + PSO              | 97.71 95.17 93.34 94.41 | 99.20 98.82 96.13 93.14 | 98.42 96.23 90.23 87.30 |
|              | SST-2   | Embedding/LM + Genetic    | 96.83 93.87 92.80 92.32 | 97.61 97.50 92.68 92.02 | 98.40 94.31 90.67 90.28 |
|              |         | Synonym + Greedy          | 96.86 94.72 92.93 92.47 | 97.12 98.73 93.50 91.93 | 98.44 96.24 93.83 91.51 |
|              |         | Sememe + PSO              | 97.05 95.13 91.22 93.26 | 98.70 97.81 94.77 92.84 | 98.27 94.15 89.65 90.04 |
| BERT         | IMDB    | Embedding/LM + Genetic    | 98.46 96.01 93.95 92.87 | 99.83 97.61 94.66 92.80 | 99.94 96.72 93.93 91.76 |
|              |         | Synonym + Greedy          | 98.71 96.62 94.04 92.93 | 98.89 97.10 94.54 92.68 | 99.91 98.80 93.72 90.83 |
|              |         | Sememe + PSO              | 98.82 97.30 94.57 93.26 | 98.92 98.94 92.71 93.91 | 98.86 96.48 93.05 88.43 |
|              | SST-2   | Embedding/LM + Genetic    | 97.36 95.73 94.05 92.34 | 98.72 97.67 94.28 91.71 | 99.57 96.45 92.17 91.72 |
|              |         | Synonym + Greedy          | 97.55 95.94 96.01 92.42 | 98.17 96.80 94.50 92.53 | 99.70 97.27 92.96 91.81 |
|              |         | Sememe + PSO              | 97.80 96.26 94.47 93.60 | 99.85 97.91 95.05 92.98 | 98.98 95.94 91.64 90.35 |
4.5 Robustness Evaluation

In practice, an attacker may attempt to remove the hidden watermark by modifying a watermarked adversarial text. It is reasonably assumed that the modification should not impair the adversarial characteristics of the marked text. Therefore, it is necessary for the proposed method to resist against such kind of modification, which is referred to as robustness. To evaluate the robustness, we mimic a real-world attack scenario that is: given a watermarked adversarial text, the attacker applies an AEG algorithm to the watermarked adversarial text so that the watermark can be removed while the text is still adversarial.

In Sec. 4.3, we have generated 500 watermarked adversarial texts for each experiment. We use the three word-level AEG algorithms to attack these watermarked adversarial texts. After attacking, these texts are still adversarial, but the watermark is distorted. The distortion can be measured by the percentage of correctly reconstructed watermark bits, which is defined as accuracy. Obviously, a higher accuracy means that more bits are correctly recovered, which reveals better robustness. Table 10 shows the mean (watermark) accuracy after attacking watermarked adversarial texts with different watermark attacking methods. From Table 10, we can find that the accuracy reaches $>90\%$ for all cases though it gradually decreases with the increase of the payload size. Moreover, in most cases, the watermark attacking method used to attack the watermarked adversarial texts based on the watermark attacking method itself makes the mean accuracy lower than the other watermark attacking methods, which is reasonable because different attacking methods use words of different parts of speech for AEG and obviously the watermark attacking method used to attack the watermarked adversarial texts based on the watermark attacking method itself is more likely to erase the watermark information. Nevertheless, overall, the proposed method shows satisfactory robustness.

4.6 Future Work

In this work, we explore a novel general framework that can be encapsulated in the cloud APIs to embed a secret watermark into a given text combining an adversarial text generation algorithm to protect mainstream adversarial text attack models. In the proposed work, once the candidate set for each word in the original text has been determined, a watermarked adversarial text example will be generated by combining the mainstream adversarial text generation algorithms. Ideally, we want the generated watermarked adversarial text examples to be as similar as possible to the original texts. In recent years, semisupervised learning methods are commonly used to improve the performance of model recognition in recognition fields including video recognition, human activity recognition, human intention recognition, etc. Luo et al. present a novel semisupervised feature selection approach from a new perspective. Instead of employing a predetermined graph, this approach incorporates the exploration of local structure into the process of joint feature selection so that the optimal graph is learned simultaneously. Inspired by this approach, in the future we intend to incorporate the influence of candidates for each word on the semantic similarity between the generated watermarked adversarial text example and the original text in order to generate the optimal candidate word $S$. In this way, the semantic similarity between the final generated watermarked adversarial text example and the original text should be improved.

5 Conclusion

In this paper, aiming to protect adversarial text attack models from being abused, we propose a general framework encapsulated in the cloud APIs to embed secret watermarks into texts while keeping its adversarial attack ability. By comparing the watermarked adversarial texts with the non-watermarked adversarial texts, our experiments show that the proposed method well maintains the adversarial attack ability and achieves satisfactory text quality. In other words, the watermarked adversarial texts generated by the proposed method not only successfully fool the target models but also enable us to retrieve a secret watermark for identifying the ownership and source information. In the future, we will improve the adversarial attack performance and the watermarking performance. Though the proposed work is designed for texts, it can be extended
to other media objects. We hope this attempt can make a contribution to the interdisciplinary field involving AEG and digital watermarking.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (NSFC) (Grant Nos. U22B2047 and U1936214); and in part by the Shanghai “Chen Guang” Program (Grant No. 19CG46; and in part by the Science and Technology Commission of Shanghai Municipality (STCSM) (Grant No. 21010500200).

References

1. C. Szegedy et al., “Going deeper with convolutions,” in Proc. IEEE Conf. Comp. Vis. Pattern Recognit., pp. 1–9 (2015).
2. O. Ronneberger, P. Fischer, and T. Brox, “U-Net: convolutional networks for biomedical image segmentation,” Lect. Notes Comput. Sci. 9351, 234–241 (2015).
3. K. He et al., “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., pp. 770–778 (2016).
4. R. Sennrich, B. Haddow, and A. Birch, “Neural machine translation of rare words with subword units,” in Proc. Annu. Meet. of the Assoc. for Computational Linguistics (ACL), pp. 1715–1725 (2016).
5. Y. Wu et al., “Google’s neural machine translation system: bridging the gap between human and machine translation,” arXiv:1609.08144 (2016).
6. D. Chen and C. D. Manning, “A fast and accurate dependency parser using neural networks,” in Proc. Empirical Methods in Nat. Lang. Process., pp. 740–750 (2014).
7. C. Szegedy et al., “Intriguing properties of neural networks,” Comput. Sci. (2013).
8. I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” Comput. Sci. (2014).
9. E. Wallace et al., “Universal adversarial triggers for attacking and analyzing NLP,” in Proc. Conf. Empirical Methods in Nat. Lang. Process. and the 9th Int. Joint Conf. Nat. Lang. Process., pp. 2153–2162 (2019).
10. A. Kurakin, I. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” in Int. Conf. Learn. Represent. (2017).
11. N. Papernot et al., “The limitations of deep learning in adversarial settings,” in Proc. IEEE Eur. Symp. Security & Privacy, pp. 372–387 (2016).
12. S. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, “DeepFool: a simple and accurate method to fool deep neural networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., pp. 2574–2582 (2016).
13. N. Carlini and D. Wagner, “Audio adversarial examples: targeted attacks on speech-to-text,” in Proc. IEEE Security & Privacy Workshop, pp. 1–7 (2018).
14. N. Carlini et al., “Hidden voice commands,” in Proc. USENIX Conf. Secur. Symp., pp. 513–530 (2016).
15. Y. Qin et al., “Imperceptible, robust, and targeted adversarial examples for automatic speech recognition,” in Proc. 36th Int. Conf. Mach. Learn. (2019).
16. M. Alzantot, B. Balaji, and M. Srivastava, “Did you hear that? Adversarial examples against automatic speech recognition,” in 31st Conf. Neural Inf. Process. Syst. (2018).
17. Y. Gong and C. Poellabauer, “Crafting adversarial examples for speech paralinguistics applications,” arXiv:1711.03280 (2017).
18. S. Eger et al., “Text processing like humans do: visually attacking and shielding NLP systems,” in Proc. North Am. Chapter of the Assoc. for Computational Linguistics, pp. 1634–1647 (2019).
19. J. Ebrahimi et al., “HotFlip: white-box adversarial examples for text classification,” in Proc. Annu. Meet. of the Assoc. for Computational Linguistics, pp. 31–36 (2018).
20. Y. Belinkov and Y. Bisk, “Synthetic and natural noise both break neural machine translation,” arXiv:1711.02173 (2017).
21. Y. Zang et al., “Word-level textual adversarial attacking as combinatorial optimization,” in Proc. Annu. Meet. of the Assoc. for Computational Linguistics, pp. 6066–6080 (2020).
22. M. Alzantot et al., “Generating natural language adversarial examples,” in Proc. Empirical Methods in Nat. Lang. Process., pp. 2890–2896 (2018).
23. N. Papernot et al., “Crafting adversarial input sequences for recurrent neural networks,” in Proc. IEEE Military Commun. Conf., pp. 49–54 (2016).
24. M. T. Ribeiro, S. Singh, and C. Guestrin, “Semantically equivalent adversarial rules for debugging NLP models,” in Proc. Annu. Meet. of the Assoc. for Computational Linguistics, pp. 856–865 (2018).
25. J. Dong et al., “A sentence-level text adversarial attack algorithm against IIoT based smart grid,” Comput. Netw. 190, 107956 (2021).
26. M. Iyyer et al., “Adversarial example generation with syntactically controlled paraphrase networks,” arXiv:1804.06059 (2018).
27. X. Wang, H. Jin, and K. He, “Natural language adversarial attacks and defenses in word level,” arXiv:1909.06723 (2019).
28. P. Bělohlávek et al., “Using adversarial examples in natural language processing,” in Proc. Int. Conf. Lang. Resour. and Eval., pp. 3693–3700 (2018).
29. G. Stringhini, C. Kruegel, and G. Vigna, “Detecting spammers on social networks,” in Proc. 26th Annu. Comput. Security Appl. Conf., pp. 1–9 (2020).
30. I. Cox et al., Digital Watermarking and Steganography, 2nd ed., Morgan Kaufmann (2007).
31. H. Wu et al., “Separable reversible data hiding for encrypted palette images with color partitioning and flipping verification,” IEEE Trans. Circuits Syst. Video Technol. 27, 1620–1637 (2017).
32. H. Wu et al., “Watermarking neural networks with watermarked images,” IEEE Trans. Circuits Syst. Video Technol. 31, 2591–2601 (2021).
33. A. M. Alattar and O. M. Alattar, “Watermarking electronic text documents containing justified paragraphs and irregular line spacing,” Proc. SPIE 5306, 685–695 (2004).
34. Q. Li and Z. Dong, “Novel text watermarking algorithm based on Chinese characters structure,” in Proc. Int. Symp. Comput. Sci. Computat. Technol., pp. 348–351 (2008).
35. J. Chen et al., “Text watermarking algorithm based on semantic role labeling,” in Int. Conf. Digital Inf. Process., Data Mining, & Wireless Commun., pp. 117–120 (2016).
36. T. Fang, M. Jaggi, and K. Argyraki, “Generating steganographic text with LSTMs,” arXiv:1705.10742 (2017).
37. H. Kang, H. Wu, and X. Zhang, “Generative text steganography based on LSTM network and attention mechanism with keywords,” in Proc. IS&T Electron. Imaging, Media Watermarking, Security and Forensics (2020).
38. Z. Meng and R. Wattenhofer, “A geometry-inspired attack for generating natural language adversarial examples,” arXiv:2010.01345 (2020).
39. R. Jia and P. Liang, “Adversarial examples for evaluating reading comprehension systems,” in Proc. Empirical Methods in Nat. Lang. Process., pp. 2021–2031 (2017).
40. S. Ren et al., “Generating natural language adversarial examples through probability weighted word saliency,” in Proc. Annu. Meet. of the Assoc. for Computational Linguistics, pp. 1085–1097 (2019).
41. G. A. Miller, “WordNet: a lexical database for English,” Commun. ACM 38, 39–41 (1995).
42. N. S. Kamaruddin et al., “A review of text watermarking: theory, methods, and applications,” IEEE Access 6, 8011–8028 (2018).
43. M. L. Mali, N. N. Patil, and J. B. Patil, “Implementation of text watermarking technique using natural language watermarks,” in Proc. Int. Conf. Commun. Syst. Netw. Technol., pp. 482–486 (2013).
44. M. J. Atallah et al., “Natural language watermarking: design, analysis, and a proof-of-concept implementation,” Lect. Notes Comput. Sci. 2137, 185–200 (2001).
45. M.-Y. Kim, “Text watermarking by syntactic analysis,” in Proc. WSEAS Int. Conf. Comput., pp. 904–909 (2008).
46. M.-Y. Kim and R. Goebel, “Adaptive-capacity and robust natural language watermarking for agglutinative languages,” Security Commun. Netw. 5, 301–310 (2012).
47. R. Rivest, “The MD5 message-digest algorithm,” https://en.wikipedia.org/wiki/MD5 (1992).
48. A. L. Maas et al., “Learning word vectors for sentiment analysis,” in Annu. Meet. of the Assoc. for Computational Linguistics: Human Lang. Technol., pp. 142–150 (2011).
49. R. Socher et al., “Recursive deep models for semantic compositionality over a sentiment treebank,” in Proc. Empirical Methods in Nat. Lang. Process., pp. 1631–1642 (2013).
50. A. Conneau et al., “Supervised learning of universal sentence representations from natural language inference data,” arXiv:1705.02364 (2017).
51. J. Devlin et al., “BERT: Pre-training of deep bidirectional transformers for language understanding,” arXiv:1810.04805 (2018).
52. J. Pennington, R. Socher, and C. Manning, “GloVe: global vectors for word representation,” in Proc. Empirical Methods in Nat. Lang. Process., pp. 1532–1543 (2014).
53. D. Naber, “A rule-based style and grammar checker,” https://languagetool.org/ (2003).
54. A. Radford et al., “Language models are unsupervised multitask learners,” OpenAI Blog, Vol. 1, p. 9 (2019).
55. O. Schwartz, “Untold history of AI: how Amazon’s mechanical Turkers got squeezed inside the machine,” https://www.mturk.com (2005).
56. M. Luo et al., “An adaptive semisupervised feature analysis for video semantic recognition,” IEEE Trans. Cybern. 48, 2 (2018).
57. K. Chen et al., “A semisupervised recurrent convolutional attention model for human activity recognition,” IEEE Trans. Neural Netw. Learn. Syst. 31, 5 (2020).
58. D. Zhang et al., “Making sense of spatio-temporal preserving representations for EEG-based human intention recognition,” IEEE Trans. Cybern. 50, 7 (2020).

Mingjie Li received his BS degree from the School of Electronic Engineering from Xi’an Aeronautical University, China, and received his MS degree from the School of Electronics and Electrical Engineering, Henan Normal University, China. Currently, he is studying at the School of Information and Communication Engineering, Shanghai University. His research interests include watermarking, steganography, and steganalysis.

Hanzhou Wu received his BS and PhD degrees from Southwest Jiaotong University, Chengdu, China, in 2011 and 2017, respectively. He was working as a researcher at the Institute of Automation, Chinese Academy of Sciences, Beijing, China, from 2017 to 2019. He has published more than 40 research papers in peer journals and conferences and authored 3 book chapters. He has received around 800 citations from Google Scholar. His research interests include AI security, steganography, steganalysis, watermarking, and graph theory.

Zichi Wang received his BS degree in electronics and information engineering from Shanghai University, China, in 2014; his MS degree in signal and information processing in 2017; and his PhD in information and communication engineering from the same university in 2020. His research interests include steganography, steganalysis, and reversible data hiding. He has published over 30 papers in these areas.

Xinpeng Zhang received his BS degree from Jilin University, China, in 1995 and MS and PhD degrees from Shanghai University. Since 2004, he has been with the faculty of the School of Communication and Information Engineering, Shanghai University. He has published over 200 research papers. He was working as an associate editor at IEEE Transactions on Information Forensics and Security from 2014 to 2017. His research interests include multimedia security, image processing, and digital forensics.