Semantic-Preserving Adversarial Text Attacks

Xinghao Yang ©, Member, IEEE, Yongshun Gong ©, Member, IEEE, Weifeng Liu ©, Senior Member, IEEE, James Bailey ©, Member, IEEE, Dacheng Tao ©, Fellow, IEEE, and Wei Liu ©, Senior Member, IEEE

Abstract—Deep learning models are known immensely brittle to adversarial text examples. Existing text adversarial attack strategies can be roughly divided into character-level, word-level, and sentence-level attacks. Despite the success brought by recent text attack methods, how to induce misclassification with minimal text modifications while keeping the lexical correctness, syntactic soundness, and semantic consistency is still a challenge. In this paper, we devise a Bigram and Unigram-based adaptive Semantic Preservation Optimization (BU-SPO) approach which attacks text documents not only at a unigram word level but also at a bigram level to avoid generating meaningless sentences. We also present a hybrid attack strategy that collects substitution words from both synonyms and sememe candidates, to enrich the potential candidate set. Besides, a Semantic Preservation Optimization (SPO) method is devised to determine the word substitution priority and reduce the perturbation cost. Furthermore, we constrain the SPO with a semantic Filter (dubbed SPOF) to improve the semantic similarity. To estimate the effectiveness of our proposed methods, BU-SPO and BU-SPOF, we attack four victim deep learning models trained on three text datasets. Experimental results demonstrate that our approaches accomplish the highest semantics consistency and attack success rates by making minimal word modifications compared with competitive methods.

Index Terms—Adversarial machine learning, text attack, natural language processing.

I. INTRODUCTION

DESpite great achievements of Deep Neural Networks (DNNs) in various of Natural Language Processing (NLP) and Computer Vision (CV) tasks, they are proved susceptible to process adversarial examples. This phenomena is initially discovered in the domain of image processing [1], where an adversarial image is an legitimate input via some intentionally crafted pixel perturbations. These perturbations are usually small enough to steer by human vision system but can mislead well-performing DNNs models to erroneous predictions with high confidence. This raises great interest and big concern in the CV community, along with the rapid development of adversarial machine learning that focuses on improving the robustness and interpretability of DNNs [2], [3], [4], [5]. On the contrary, the security threat of DNNs is insufficiently estimated in the field of NLP, especially for some security critical applications, such as spam filtering [6], secret sharing [7], sentiment analysis [8], and webpage phishing [9]. Therefore, researching on effective adversarial attack strategies is significant to understanding the behaviors of DNN models and improving their robustness. The semantic preservation aims to ensure that the generated adversarial samples are difficult to distinguish by humans.

Compared to the image attacks, crafting natural text adversarial example is more challenging due to the following three folds. First, the text adversarial samples should meet some linguistic constraints, such as lexically correct, syntactically sound, and semantically similar to the original text, so that the adversarial sentences are fluent to human readers. Second, optimizing perturbations on continuous image data with gradient descent achieves superior efficiency, while the discrete property of text sequences limits the direct implementation of such first-order techniques. Third, from human perception perspective, adding small perturbations on many pixels of an image can produce a meaningful adversarial image, but making even small changes (e.g., a single character or word) in text document can result in a significant semantic loss.

Typical text attack algorithms usually modify text at three levels, i.e., the character level, the sentence level and the word level [10]. However, character-based and sentence-based alterations usually lead to the wrong representation of the original content. For example, character-level attack modifies predict → perdic, which leads to a lexical mistake; and the sentence-level attack will generate a complete adversarial sentence that often produces unreadable text. To roundabout these issues, many of recent researches have paid more attention to word based substitutions that replace the input word with a maliciously selected candidate [11]. However, most of the existing methods generate substitution candidates by focusing on every single word (i.e., a unigram), which easily destroys some common bigram phrases and issue in semantic incorrect expression (e.g., mad doctor → angry doctor). Besides, most existing methods use the word importance score (WIS) to determine the word substitution order by descending sort of WIS. There exist various definitions of WIS, e.g., the reduction of DNNs’ predictions by deleting a word [12] and probability weighted word saliency (PWWS) [13], etc. The common shortcoming of following the
static WIS order is the word over modification, e.g., replacing the top-3 WIS words \{top1, top2, top3\} consecutively may not fool a DNN model, while it works based only on a two-word combination \{top1, top2\} sometimes.

To address above-mentioned drawbacks, we present a novel black-box word-level attack algorithm entitled Bigram and Unigram based Semantic Preservation Optimization (BU-SPO) in this paper. In the black-box settings, the internal parameters and gradient information of the victim model are unavailable, which is more relevant to the practical scenario compared to white-box attack. Different from the traditional unigram word attack, we examine both unigram substitution and bigram modification. In our approach, we produce a more natural adversarial text by replacing an entire bigram with its synonyms (e.g., mad doctor \to mental doctor and high school \to secondary school). As shown in Table I, we provide three instances to demonstrate the advantages of bigram attacks over unigram attacks. In addition, we propose to alternate original words with both synonym candidates and sememe candidates. By this complementary combination, we have more options to generate successful adversarial samples.

More importantly, we propose an adaptive Semantic Preservation Optimization (SPO) to arrange word substitution order. The SPO inherits the best attack combinations from the previous generation. Then a heuristic search determines which word should be replaced in the next iteration. Specifically, if modifying the \{top1\} token fails to fool a classifier, the existing static algorithms must attack the combination \{top1, top2\} in the second attempt, but the proposed SPO will count for more cases, e.g., \{top1, top2\}, \{top1, top3\}, etc. Therefore, the proposed SPO can fool DNNs models with fewer word substitutions, which is crucial to reduce syntax errors and semantic loss. Besides, we integrate a semantic Filter into the SPO method (dubbed SPOF) so that it can select the best candidate to maximally preserve the semantic consistency between the input text and the adversarial output. Fig. I illustrates the framework of our algorithm with an attack example. We summarize five major contributions of this study as below:

1. We propose to generate fluent text adversarial samples by replacing not only individual words (unigram) in the text but also phrases (bigram) as a whole. This strategy is significant in generating more semantically natural adversarial examples and avoiding meaningless outputs.
2. We propose to collect substitution candidates with a hybrid approach from both synonym set and sememe-consistency words. The complementary combination provides more options to craft meaningful adversarial examples.
3. We design a Semantic Preservation Optimization (SPO) approach to adaptively establish the word substitution order. The SPO is devised to mislead a DNN classifier using the minimal word modifications compared with static word replacement baselines. Making less word replacements is helpful to reduces the syntactic mistakes.
4. We further customize the SPO with a semantic Filter (SPOF), targeting to get adversarial examples with the highest semantic consistency. This step can significantly enhance the quality of the output adversarial example in terms of the sentence naturality and fluency.
5. We conduct extensive experiments on IMDB, AG’s News and Yahoo! Answers dataset by attacking both CNN and LSTM models. Experimental results demonstrate the

Fig. 1  The workflow of our BU-SPOF method with a text example “Study: CEOs rewarded for outsourcing. NEW YORK (CNN/Money) - The CEOs of the top 50 US companies that sent service jobs overseas pulled down far more pay than their counterparts at other large companies last year, a study said Tuesday.” For brevity, in this figure we only show several words from the long text. This example is originally labeled as “Business” (66.68%) by LSTM, but is misclassified as “Sci/Tech” by replacing a bigram “New York” with “Empire State”. We highlight the two successful attacks with green color boxes, as they reduce the true label probability much more than those failure purple ones. Finally, we accept the “Empire State” substitution, because it preserves more semantics (0.9877) than “AFP” (0.9724).

**TABLE I**

| Original Input       | Unigram Attack     | Bigram Attack     |
|----------------------|--------------------|-------------------|
| Machine Learning     | Device Learning    | Data Mining       |
| New York             | Fresh York         | Empire State      |
| Primary School       | Major School       | Elementary School |

Fig. 1. The workflow of our BU-SPOF method with a text example “Study: CEOs rewarded for outsourcing. NEW YORK (CNN/Money) - The CEOs of the top 50 US companies that sent service jobs overseas pulled down far more pay than their counterparts at other large companies last year, a study said Tuesday.” For brevity, in this figure we only show several words from the long text. This example is originally labeled as “Business” (66.68%) by LSTM, but is misclassified as “Sci/Tech” by replacing a bigram “New York” with “Empire State”. We highlight the two successful attacks with green color boxes, as they reduce the true label probability much more than those failure purple ones. Finally, we accept the “Empire State” substitution, because it preserves more semantics (0.9877) than “AFP” (0.9724).
effectiveness of our proposed model that can not only achieve the high attack success rate with low perturbation costs, but also can keep a high semantic consistency when generating adversarial examples. Besides, our BU-SPOF also shows superiorities in transfer attack, adversarial retrain, and targeted attack compared with baselines.

The rest of this paper is organized as follows: Section II includes a literature review of generating text adversarial samples. In Section III, we formalize our algorithm in detail. Evaluation metrics and all experimental results are reported in Section IV. Finally, conclusions are drawn in Section V.

II. RELATED WORK

We briefly review typical text attack methods in this section. Relevant studies can be divided into four categories, including character-level attack [14, 15, 16], sentence-level attack [17, 18], word-level attack [12, 13, 19, 20], and multi-level attack [21, 22, 23].

First, character-level attack [14, 15, 16] generates adversarial text via three functions, i.e., inserting, swapping or deleting characters. Belinkov et al. [16] devised four types of synthetic noise: swap, middle random, fully random, and keyboard typo that can mislead the neural machine translation (NMT) models in a large degree. However, they modify every word of an input sentence as they can, which leads to a high perturbation loss. For example, the “swap” of two letters (e.g., noise → nosie) is applied to all words with length ≥ 4, as it does not alter the first and last letters. To reduce the distortion degree, Ebrahimi et al. [14] proposed HotFlip, which represents every character as a one-hot vector. Then it estimates the best character change by computing directional derivatives with respect to vector operations. Gao et al. [15] designed a model named black-box DeepWordBug to evaluate word importance scores by directly removing words one by one and comparing the prediction changes. However, character-level attack breaks the lexical constraint that generates misspelled words. Such words can be easily detected and removed by a spell check machine installed before the classifier.

Second, sentence-level attack methods [17, 18] aim to concatenate an adversarial sentence before or more commonly after the clean input text to confuse deep learning methods. For example, Jia et al. [17] appended a compatible sentence to the end of paragraph to fool reading comprehension models (RCM). The adversarial sentence looks similar to the original question by combining altered question and fake answers, aiming to mislead RCM into wrong answer location. Nevertheless, this strategy requires a lot of human intervention and cannot be fully automated, e.g., it relies on about 50 manually-defined rules to ensure the adversarial sentence in a declarative form. Recently, Wallace et al. [18] sought for the universal adversarial triggers, i.e., input-agnostic sequences, which causes a specific target prediction when it is concatenated to any input from the same data set. The universal sequence is randomly initialized and iteratively updated to increase the likelihood of the target prediction using token replacement gradient as HotFlip. Nevertheless, sentence-level attacks usually generate sentences that are incomprehensible to humans and thus reduce the naturalness of the adversarial sample.

Additionally, word-level attack algorithms replace original words in the input sentence with deliberately selected words. The main challenges are twofold (1) how to pick appropriate candidates for each input word and (2) how to optimize the word substitution priority to achieve a high fooling rate with small changes. At the early stage, Papernot et al. [19] mapped words into a latent embedding space and explored the effects of each input on model output via the Jacobian matrix. However, a very small change in the projection space can produce completely unrelated words because there is no strong constraint that two neighbor words in the embedding space are also close in semantics. Therefore, subsequent researches concentrated on the synonym attack that collect synonyms from several word embedding methods, e.g., existing thesaurus (e.g., WordNet and HowNet), GloVe embedding model, and BERT Masked Language Model (MLM). Alzantot et al. [20] employed the GloVe embedding to select candidate words and completed the substitution via a genetic algorithm (GA). However, the GloVe embedding relies on the co-occurrence matrix to represent words, which cannot properly identify synonyms and antonyms. For example, the nearest tokens for expensive include {pricey, cheaper, costly} according to the GloVe distance, while cheaper is the opposite in semantics. Thus, a counter-fitting post-process is a necessary approach to enhance the semantic constraint [24]. Differ from GloVe, collecting substitution words from well-organized linguistic thesaurus is a relatively straightforward way, such as, synonym-based WordNet [25] and sememe-based HowNet [26]. Ren et al. [13] explored synonyms via the WordNet synsets and proposed the probability weighted word saliency (PWWS) to rank word replacement order. However, the PWWS is intrinsically a static algorithm, i.e., it sorts and attacks words at once based on the descending order of word important score. This strategy usually suffers from the local optimal solution and leads to excessive word substitution. Zhu et al. [27] utilized an improved beam search algorithm with multiple random perturbing positions to search the optima. Zhang et al. [28] indicated that the HowNet can return more sememe-based candidates than WordNet and presented a discrete Particle Swarm Optimization (PSO) to optimize the word replacement priority. Li et al. [29] and Garg et al. [30] utilized the BERT-MLM to generate context-aware candidates and proposed BERT-Attack and BERT-based Adversarial Examples (BAE), respectively. The main difference between them is the definition of WIS - the BERT-Attack deletes each word and defines the probability reduction as WIS, while BAE replaces each word with [MASK] to evaluate the word importance. Nevertheless, both of these algorithms belong to the static word replacement strategy by the WIS descending order. Recently, Lee et al. [31] developed the Blockwise Bayesian Attack (BBA) to accomplish the black-box attack with limited number of queries. Dong et al. [32] constructed a domain-specific word embedding with domain knowledge to improve the attack performance on domain specific models. It is worth mentioning that all these works focus on unigram attacks and ignore the widely used bigram phrases.
Finally, the perturbed targets of multi-level attacks are usually two or three combinations of characters, words, and sentences [21], [22], [23]. For example, Blohm et al. [21] implemented four adversarial text generation methods with different edits at the level of words and sentences. Li et al. [22] computed the importance of each word in the case of white box through the Jacobian matrix, then queries the victim model in the black box setting to determine the importance of the split sentence. Vijayaraghavan et al. [23] utilized a self-critical approach to achieve word-level and character-level perturbations via a hybrid encoder-decoder based adversarial example generators.

III. ALGORITHM

We discuss the proposed BU-SPO and BU-SPOF methods in this section. Suppose we have $N$ sentences $\mathcal{X} = \{X_1, X_2, \ldots, X_N\}$ belonging to $K$ labels $\mathcal{Y} = \{Y_1, Y_2, \ldots, Y_K\}$. A well-trained DNNs model $F$ is a non-linear function projecting from the text space to the label space $F: \mathcal{X} \rightarrow \mathcal{Y}$.

A. Black-Box Text Adversarial Attack

Our attack methods are designed under black-box settings where network structure, internal parameters or gradient information are all unavailable. The only capability is to obtain the output of the victim model (confidence scores), performing as a regular user. Formally, given an input text $X \in \mathcal{X}$, a well-performed DNNs model $F$ targets to output the true label $Y_{true} \in \mathcal{Y}$, i.e., $F(X) = Y_{true}$, by optimizing the posterior probability

$$\arg \max_{Y \in \mathcal{Y}} P(Y | X) = Y_{true} \quad (1)$$

A reasonable text attacker pursues to generate a subtle perturbation $\Delta X$ on the input example $X$ so that the classifier $F$ can be fooled by the adversarial example $X^* = X + \Delta X$. Specifically, a successful adversarial example can mislead a victim model $F$ into any incorrect label

$$\arg \max_{Y \in \mathcal{Y}} P(Y | X^*) \neq Y_{true} \quad (2)$$

or a user-specified target label $Y_{target}$

$$\arg \max_{Y \in \mathcal{Y}} P(Y | X^*) = Y_{target} \quad (3)$$

where $Y_{target} \neq Y_{true}$. These two attack strategies are formally defined as untargeted attack (2) and targeted attack (3), respectively. To generate natural and fluent adversarial sentences, the crafted perturbations should be lexical correct, grammatical correct, and semantic consistent. The lexical constraint can be satisfied if we focus on word level modification instead of the character level attack. Besides, we design the bigram substitution to generate meaningful texts, and propose an adaptive SPO to reduce word substitutions while maintaining semantic similarity and syntactic coherence.

B. Semantic Similarity

The semantic similarity between the original input sentence and the adversarial output sentence is vitally important to ensure that the modifications are imperceptible to human. To measure the semantic similarity between text examples, we employ the Universal Sentence Encoder (USE) [33] method in this paper. The USE model encodes different input sentences into 512 dimensional embedded features, and calculates their cosine similarity scores subsequently. Specifically, the USE encoder is trained on variety of web text with general purpose, e.g., web news, discussion forums, Wikipedia and web question-answer pages. Therefore, it is capable of feeding multiple downstream tasks. Formally, denote an USE encoder by Encoder, then the USE score between an example $X$ and its adversarial variation $X_{adv}$ is defined as

$$USE_{score} = \text{Cosine} (\text{Encoder}(X), \text{Encoder}(X_{adv})) \quad (4)$$

One major advantage of USE sentence embedding is that it can imply how much the selected candidate word fits the original sentence. On the contrary, alternative word embedding methods (e.g., word2vec [34]), which maps each word to the embedding space, fails to generate context aware representations.

C. Bigram and Unigram Candidate Selection

Before elaborating on the candidate selection procedure, we first briefly introduce the WordNet and HowNet and give the definition of the synonym space and sememe space. WordNet [25] groups word relations into 117,000 unordered synonym sets (synsets). Different synsets are interlinked by super-subordinate relation, e.g., the “furniture” synset includes the “bed” synset. In this work, we collect synonym candidates from the WordNet synonm space $\mathcal{W}$. HowNet [26] explains words with sememes, where a sememe represents a minimum semantic unit in linguistics. For example, the word “apple” contains several sememe tags, e.g., “fruit”, “Smartphone”, etc. We denote that words sharing the same sememe tag can be exchangeable in generating adversarial examples. Then, the HowNet sememe candidates space is expressed as $\mathcal{H}$.

1) Create Candidate Set: Given an input sentence with $n$ words, i.e., $X = \{w_1, w_2, \ldots, w_n\}$, we first combine each word $w_i$ with the next word $w_{i+1}$ and examine if the bigram $(w_i, w_{i+1})$ has candidates in the synonym space $\mathcal{W}$. If yes, we create the candidates set $\mathcal{B}_i$ for the bigram $(w_i, w_{i+1})$ and jump to search substitution words for $w_i$ and $w_{i+1}$ individually. Otherwise, we create the candidate set $\mathcal{S}_i$ for $w_i$ by collecting the synonym candidates from $\mathcal{W}$ and the sememe candidates from $\mathcal{H}$, i.e., $\mathcal{S}_i \subset \mathcal{W} \cup \mathcal{H}$. Particularly, we design a candidate filter to ensure that all the candidates in $\mathcal{S}_i$ share the same part-of-speech (POS) tags with the input word $w_i$. Through this process, we can avoid creating grammatical errors because all substituting words have the same POS tags (e.g., nouns).

For any named entity (NE) $w_i$, we extend its candidate set $\mathcal{S}_i$ by joining one more NE candidate. The NE is commonly known as a real-world object, e.g., person names, organizations, and locations, which can be symbolized by proper nouns [35]. The selected NE candidate (denoted as $N_{ECOMP}$) should keep the same NE type with $w_i$. Specifically, $N_{ECOMP}$ is the one

\(^1\)This means bigram substitution takes precedence.
that appears most frequently in the complementary NE set \( W - W_{Y_{true}} \), where \( W_{Y_{true}} \) includes all the NEs belonging to the true label. Thereafter, the synonym set can be enlarged as \( S_i \leftarrow S_i \cup NE_{COMP} \).

Considering polysemy, one word often contains more than one sememe tag defined by HowNet. Therefore, we add only words that have at least one common sememe with the original word \( w_i \) into its candidate set \( S_i \) to achieve effective modifications.

2) Select the Best Candidate: The candidate set \( S_i \) (or \( B_i \)) usually contains more than one word, and every \( w'_i \in S_i \) is a potential candidate for \( w_i \). To pick the best candidate, the candidate importance score \( I_{w'_i} \) for each \( w'_i \in S_i \) is defined as the true label score reduction
\[
I_{w'_i} = P(Y_{true} | X) - P(Y_{true} | X'_i), \forall w'_i \in S_i \quad (5)
\]
where
\[
X = w_1, w_2, \ldots, w_i, \ldots, w_n
\]
\[
X'_i = w_1, w_2, \ldots, w'_i, \ldots, w_n \quad (6)
\]
Then the best substitution word \( w^*_i \) is picked as the \( w'_i \) that attains the highest \( I_{w'_i} \). Formally, (7) gives the function for best synonym candidate selection
\[
w^*_i = R(w_i, S_i) = \arg \max_{w'_i \in S_i} I_{w'_i} \quad (7)
\]
Sequentially implementing this process on every input word addresses the first problem of our model, which is shown in lines 1 to 11 of the Algorithm 1.

D. Semantic Preservation Optimization

The Semantic Preservation Optimization (SPO) is designed to optimize the word replacement priority with three objectives: 1) achieve a successful attack, 2) make minimal substitutions, and 3) keep the sentence semantic unchanged. Specifically, after the best candidate selection, we can get \( n \) adversarial sentences \( \{X^*_1, \ldots, X^*_n\} \), where each sentence contains one replaced word, i.e., \( X^*_i = \{w_1, \ldots, w'_i, \ldots, w_n\} \). The true label score reduction from \( X \) to \( X^*_i \) denotes the maximal attack effect which is induced by attacking \( w_i \)
\[
\Delta P_{adv} = P(Y_{true} | X) - P(Y_{true} | X^*_i) \quad (8)
\]
A straightforward strategy of sorting word replacement is by their \( \Delta P_{adv} \) descending order. However, substituting words in a fixed sequence easily results in a local optimal solution and triggers the problem of word over substitution. Accordingly, the top-\( k \) important words are not necessarily the best combination in fooling DNNs.

Algorithm 1: The Proposed BU-SPO Algorithm.

Input: Sample sentence containing \( n \) words
\( X = (w_1, \ldots, w_n) \)
Input: The upper threshold of word replacement \( M \)
Input: Well-trained classification model \( F \)
Output: Adversarial text sample \( X_{adv} \)

1 // Select candidates for input words * /
2 for \( i = 1 \) to \( n \) do
3 Connect \( w_i \) with its next word as \( (w_i, w_{i+1}) \);
4 Collect bigram candidate set \( B_i \) for \( (w_i, w_{i+1}) \) from synonym space \( W \);
5 if \( B_i \neq \emptyset \) then
6 Select the best candidate bigram from \( B_i \);
7 else
8 Collect a synonym candidate and sememe candidate set \( S_i \) for \( w_i \) from \( W \cup B_i \);
9 if \( w_i \) is a NE then
10 \( S_i \leftarrow S_i \cup NE_{COMP} \);
11 Select the best candidate unigram from \( S_i \);
12 Create an empty set \( G^0 = \emptyset \) as the initial generation;
13 Set the modification threshold \( M = \min(M, n) \);
14 // The SPO search starts *
15 for \( m = 1 \) to \( M \) do
16 if \( F(X) \neq F(X_{adv}) \) then
17 break; // successful attack
18 else
19 \( \Delta P_{adv} = P(Y_{true} | X) - P(Y_{true} | X_{adv}); \)
20 return \( X_{adv} \)
21 \]

Algorithm 2: Generation Creation Function \( (F) \).

Input: The previous generation population \( G^{m-1} \)
Input: The best replacement candidates \( \{w^*_1, \ldots, w^*_n\} \)
Output: Current generation combinations \( G^m \)

1 Initialize \( G^m = \emptyset \) as the current generation;
2 if \( G^{m-1} = \emptyset \) then
3 \( G^m = \{w^*_1, \ldots, w^*_n\}; \quad // \) The first generation
4 else
5 Search for the most effective element \( G^{m-1}_{\text{best}} \) from \( G^{m-1} \) that achieves the biggest \( \Delta P_{adv} \);
6 Delete all words in \( G^{m-1}_{\text{best}} \) from \( \{w^*_1, \ldots, w^*_n\} \), remaining \( s \) \( (1 \leq s \leq n) \) elements left;
7 for \( i = 1 \) to \( s \) do
8 Combine \( G^{m-1}_{\text{best}} \) with \( i^{th} \) remaining element \( w^*_i \) to create the current generation element \( G^m(i) \);
9 return \( G^m \)

In this study, we design the Semantic Preservation Optimization (SPO) method to optimize the word replacement priority in an adaptive way. Particularly, we define the initial generation \( G^0 \) with an empty set in Algorithm 1 (line 12). Then a threshold \( M \) is used to limit the number of words to be modified, i.e., \( M = \min(M, n) \). If the input can not make a successful attack after \( M \) words substitution, this strategy can force to stop the
loop. Algorithm 1 lines 14-21 list the SPO search. In each iteration, we first call the $F$ function defined in Algorithm 2 to create the population set for the current generation, i.e., $G^{m}$. When $m = 1$, the $F$ directly returns the first generation as the $n$ best substitutions $\{w'_1, \ldots, w'_n\}$. Then we iteratively craft adversarial example $X_{adv}$ and check whether the classifier is deceived. If yes, the SPO stops the optimization and returns the successful adversarial example $X_{adv}$. Otherwise, we count the true label score reduction $\Delta P_{adv}$ in line 21. If no successful attack can be found in the first generation, we need to run the second generation while updating the $\Delta P_{adv}$.

In every next generation, the function $F$ constructs the population set $G^{m}$ with three major phases, as shown in Algorithm 2 lines 5–8. We first locate the best element that attains the highest $\Delta P_{adv}$ from the previous generation $G^{m-1}$ and denote it as $G_{best}^{m-1}$. Then we remove all the candidates that are in the best element $G_{best}^{m-1}$ from the whole set $\{w'_1, \ldots, w'_n\}$, leaving $s$ ($1 \leq s \leq n$) elements. Finally, the previous best element $G_{best}^{m-1}$ is combined with every remaining candidate to constitute the new population member $G^{m}(i)$ in lines 7–8 of Algorithm 2. Given the new generation set $G^{m}$, we can repeat the same greedy search defined in lines 16-21 to find a successful attack but modifies one more word/bigram in every next generation. This optimization procedure continues until it finds a successful adversarial example or reaches the upper bound of the threshold $M$. Briefly, the SPO keeps the most effective population member from each previous generation and adaptively enlarges the attack combination in the next generation. Compared with the static baselines, the SPO achieves a higher successful attack rate with a low word substitution cost. This solves the second key issue.

E. SPO With Semantic Filter (SPOF)

To further enhance the semantic similarity of SPO, we further improve it by using a semantic filter with the proposed SPOF algorithm (Algorithm 3). The SPOF employs the same strategy to collect bigram and unigram candidates but improves the word priority determination procedure. Specifically, for each generation, we first create an empty set of successful adversarial examples that can make successful attacks (Algorithm 3 line 15). If a population member $X_{adv}$ changes the classifier’s prediction (Algorithm 3 line 19), we calculate the semantic similarity score between $X$ and $X_{adv}$ (Algorithm 3 line 20) and append it into the successful adversarial example set $SucAdv$ (Algorithm 3 line 21). We traverse this procedure for every population member in each generation. Then we discover the adversarial example that preserves the highest semantic similarity from the successful adversarial example subset $SucAdv$ if it is not empty (Algorithm 3 line 24-26). Line 27 breaks the optimization to ensure that our SPOF rephrases minimal words in achieving successful attack.

F. Targeted Attack Strategy

Targeted attack is the scenario where attackers aim to misdirect the classifier to a pre-specified target label $Y_{target}$. In this section, we show our BU-SPO and BU-SPOF algorithms can be easily adapted to conduct targeted attacks by making the following three modifications. First, we change the successful attack condition in Algorithm 1 line 18 and Algorithm 3 line 19 from $F(X) \neq F(X_{adv})$ to $F(X_{adv}) = Y_{target}$. This means we only count adversarial examples that can mislead the classifier to the target label as successful attacks. Second, we evaluate the attack strength by calculating how much the target label probability increased rather than how much the true label score decreased. Therefore, the (5) is reformed to (9)

$$I_{w_i} = P(Y_{target}|X'_i) - P(Y_{target}|X), \forall w'_i \in S_i$$

(9)

and (8) is transformed to (10)

$$\Delta P^*_{adv} = P(Y_{target}|X_{adv}) - P(Y_{target}|X)$$

(10)

Additionally, line 21 of Algorithm 1 and line 23 of Algorithm 3 become (11)

$$\Delta P_{adv} = P(Y_{target}|X_{adv}) - P(Y_{target}|X)$$

(11)
Finally, we select the most frequent NE substitution from the target class, i.e., $NE_{Target}$, instead of from the complementary set $NE_{COMP}$. This is helpful to increase the target label score and improve the success rate of targeted attacks.

IV. EXPERIMENTS

We assess the effectiveness of the proposed BU-SPO and BU-SPOF on popular text classification datasets. The code and data are shared at Github\(^3\) to ensure reproducibility.

A. Datasets

We conduct experiments on three publicly available benchmarks, including IMDB dataset, AG’s News dataset, and Yahoo! Answers dataset. Statistical properties of these three datasets are reported in Table II.

| Dataset          | # Train | # Test | # Avg. Words | # Classes |
|------------------|---------|--------|--------------|-----------|
| IMDB             | 2,5000  | 2,5000 | 227.11       | 2         |
| AG’s News        | 120,000 | 7,600  | 37.84        | 4         |
| Yahoo! Answers   | 1,400,000 | 60,000 | 32.39        | 10        |

We implement the proposed attack methods on four victim methods, including two Recurrent Neural Networks (RNNs) and two Convolutional Neural Networks. The above methods are effective tools for text classification in either character level or word level.

Word-based CNN (CNN) [38] is composed of a word embedding layer with 50 dimensions, a convolutional layer with 250 kernels, a maximum pooling layer, two pairs of densely-connected layer and nonliner activation layer. Besides, it contains two dropout layers with 0.2 dropout rate to prevent overfitting. This Word CNN model is implemented on all the three datasets.

Character-based CNN (Ch-CNN) [37] is stacked by a character embedding layer with 69 dimensions, six convolutional layers and three fully-connected layers. Each convolutional layer employs 256 filter kernels with filter size varying from 3 to 7. It also inserts one dropout layer after every densely-connected layer with 0.1 dropout frequency. We evaluate this Ch-CNN model on AG’s News dataset.

Word-based LSTM (LSTM) consists of a 100-dimensional word embedding layer, concatenating a long short-term memory layer with 128-units, followed by a 0.5 fraction dropout layer. The LSTM structure prevents gradient vanishing by utilizing a memory cell, and is effective for sequence text classification [19]. This LSTM model is applied on the AG’s News dataset.

Bidirectional LSTM (Bi-LSTM) passes the input text through a 128 dimensional word embedding layer, a 64-unit bidirectional LSTM layer, combining a 0.5 proportion dropout layer and a fully-connected layer for classification. We run this bidirectional LSTM model on both IMDB and Yahoo! Answer datasets.

Table III shows the classification accuracy of the above four well-trained methods on the original test set.

C. Evaluation Metrics

We use three metrics to evaluate the text attack performance, i.e., Attack Success Rate (ASR), the average word replacement (AWR) number, and the semantic similarity score (USE score).

The ASR indicates how much an adversary can mislead the victim model. Formally, an successful attack is when the classifier $F$ can correctly classify the original legitimate input $F(X) = Y_{true}$ but makes a wrong prediction on the corresponding attacked input $F(X + \Delta X) = Y^*$. Therefore, the ASR is defined as

\[
ASR = \frac{\sum_{X \in \mathcal{X}} \{F(X) = Y_{true} \land F(X + \Delta X) = Y^*\}}{\sum_{X \in \mathcal{X}} \{F(X) = Y_{true}\}}
\]  

(12)

where $Y^*$ can be any label different from the $Y_{true}$ (untargeted attack) or equal to a user specified label (targeted attack). The $\Delta X$ denotes the modifications for the legitimate text sample. To accurately represent the text perturbation loss, we directly compute the number of word replacements for each sample and calculate the average word replacement (AWR) number to get the adversarial attack cost. The semantic similarity score is also an important indicator to estimate the quality of the crafted adversarial, as it measures whether the adversarial example reads natural and fluent. Intuitively, a rational hacker hopes to attain high ASR and semantic similarity score while modifying a small number of words.

\(^3\)https://github.com/AdvAttack/BU-SPO
TABLE IV
THE ATTACK SUCCESS RATE (ASR) OF DIFFERENT ATTACK METHODS

| Methods  | IMDB          | AG’s News          | Yahoo! Answers          |
|----------|---------------|--------------------|-------------------------|
|          | CNN | Bi-LSTM | LSTM | CNN | LSTM | Ch-CNN | CNN | Bi-LSTM | LSTM | Ch-CNN |
| BEAT     | 91.02% | 90.15% | 99.72% | 86.91% | 77.52% | N/A | N/A | N/A |
| PSO      | 100% | 100% | 100% | 85.48% | 79.87% | N/A | N/A | N/A |
| TEO      | 94.60% | 99.78% | 90.01% | 82.75% | 85.17% | N/A | N/A | N/A |
| PWWS     | 38.85% | 81.70% | 81.65% | 77.98% | 84.73% | 88.33% | 68.23% | 69.22% | 96.36% |
| WSA      | 90.56% | 97.96% | 82.60% | 81.98% | 76.83% | 75.25% | 68.23% | 69.22% | 96.36% |
| PWWS (M = 20) | 81.13% | 94.79% | 74.33% | 82.42% | 79.76% | 75.26% | 51.96% | 51.64% | 95.50% |
| RAND     | 95.58% | 98.99% | 90.80% | 85.76% | 81.07% | 80.70% | 70.27% | 67.92% | 97.58% |
| U-SPO    | 100% | 100% | 100% | 92.77% | 88.49% | 92.17% | 90.14% | 87.95% | 98.91% |
| BU-SPO   | 100% | 100% | 98.32% | 91.77% | 86.21% | 91.49% | 88.58% | 85.60% | 98.50% |
| BU-SPOF  | 100% | 100% | 100% | 93.44% | 89.06% | 91.52% | 87.17% | 85.45% | 98.29% |

The bold entities indicate the optimal values of algorithm evaluation metrics for different attack algorithms on victim model attacks under IMDB, AG’s News, and Yahoo! Answers datasets.

TABLE V
THE AVERAGE WORD REPLACEMENT (AWR) NUMBER OF DIFFERENT ATTACK ALGORITHMS

| Methods  | IMDB          | AG’s News          | Yahoo! Answers          |
|----------|---------------|--------------------|-------------------------|
|          | CNN | Bi-LSTM | LSTM | CNN | LSTM | Ch-CNN | CNN | Bi-LSTM | LSTM | Ch-CNN |
| BEAT     | 7.50 | 7.89 | 9.88 | 5.97 | 6.24 | N/A | N/A | N/A |
| PSO      | 3.42 | 5.22 | 5.91 | 5.02 | 5.82 | N/A | N/A | N/A |
| TEO      | 8.01 | 8.13 | 7.43 | 7.42 | 8.49 | N/A | N/A | N/A |
| PWWS     | 8.63 | 5.53 | 14.02 | 6.29 | 8.05 | 5.59 | 3.34 | 3.47 | 1.05 |
| WSA      | 16.15 | 10.55 | 16.85 | 9.51 | 10.12 | 8.23 | 3.95 | 3.83 | 1.21 |
| PWWS (M = 20) | 5.87 | 5.16 | 7.84 | 6.03 | 7.72 | 5.44 | 3.15 | 3.20 | 1.23 |
| RAND     | 7.31 | 5.67 | 8.79 | 5.23 | 6.19 | 5.06 | 3.65 | 3.74 | 1.12 |
| U-SPO    | 4.50 | 4.24 | 6.21 | 4.82 | 5.97 | 4.39 | 2.78 | 2.98 | 1.04 |
| BU-SPO   | 2.02 | 2.28 | 2.67 | 4.16 | 5.24 | 3.41 | 1.85 | 2.28 | 1.02 |
| BU-SPOF  | 2.71 | 2.35 | 2.58 | 4.22 | 5.99 | 3.52 | 1.89 | 2.49 | 1.04 |
| BU-SPOF  | 2.06 | 1.62 | 2.38 | 4.17 | 5.65 | 3.62 | 1.96 | 2.32 | 1.13 |

The bold entities indicate the optimal values of algorithm evaluation metrics for different attack algorithms on victim model attacks under IMDB, AG’s News, and Yahoo! Answers datasets.

D. Baselines

We validate our method by comparing it with typical word-level attack algorithms under black-box settings.
- RAND randomly picks a synonym for each word from WordNet and follows our SPO algorithm to determine the word replacement priority.
- Word saliency attack (WSA) [39] obtains candidates from WordNet and runs the attack by the word saliency (WS) descending order. The strategy of WS is essentially analogous to (5) but substitutes $w_i$ with unknown.
- PWWS [13] selects substitution candidates from WordNet and weights the word importance by multiplying the probability reduction and word saliency.
- PSO [28] chooses sememe candidates via HowNet and designs the PSO to adversarial text. The PSO regards every sample as a particle and optimizes its space location to accomplish word modifications.
- TextFooler (TEFO) [12] seeks neighbor candidates from the Glove space and calculates the WIS by iteratively deleting each input words and querying the DNNs score changes.
- BERT-ATTACK (BEAT) [29] employs the BERT MLM to produce candidates and implements text attack along a static WIS descending order. The strategy of WIS is essentially analogous to (5) but replaces $w_i$ by [MASK].

E. Experimental Settings

All the victim DNNs models are trained by utilizing the ADAM optimizer [40], where parameters settings are: learning rate $= 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-7}$. We run our BU-SPO, RAND, WSA and PWWS on Keras. The rest baselines (i.e., PSO, TEFO, and BEAT) are deployed on the TextAttack toolbox [41], where the Yahoo! Answers dataset and the Ch-CNN model are currently unavailable. Therefore, these results are infeasible to achieve (i.e., N/A) in Tables IV, V, and VI.

In our methods, the upper bond number of word replacements is restricted to $M = 20$. This means we will stop the attack iteration if a text sample does not admit the adversarial attack after 20 substitutions. For the baseline methods, we use their recommended parameters for fair comparison. Particularly, for the most related baseline PWWS, we also report its performance with the constraint, i.e., $M = 20$, for fair comparison. For efficiency, we evaluate the attack performance on 1000 samples on the test set as the conventional setting [12], [28].

F. Experimental Results and Analysis

Tables IV, V, and VI report the attack performance of ASR, AWR, and the semantic similarity, respectively. We answer the following four questions to elaborate the first four contributions as mentioned in the Introduction.
Q1: Is the proposed SPO better than static baselines? To answer Q1, we present the U-SPO algorithm that collects synonyms from only WordNet and modifies only the individual word (unigram) - the same as WSA and PWWS, while adopts the adaptive SPO to optimize the word replacement order. We can see from Tables IV–VI that U-SPO obtains higher ASR, changes a much smaller number of words, and keeps more semantic similarity comparing with other static methods (WSA, and PWWS). In addition, even the RAND achieves higher SAR than WSA on two datasets (i.e., AG’s News and IMDB), which still indicates the superiority of the proposed adaptive SPO.

Q2: Is the mixing of synonyms and sememes beneficial? We update the U-SPO to a Hybrid version, namely HU-SPO. HU-SPO is the same as U-SPO but integrates WordNet and HowNet to find complementary synonym-sememe candidates. Table IV elaborates that HU-SPO attains the best ASR in most scenarios and exceeds U-SPO by a large margin. Intriguingly, Tables V and VI demonstrate that HU-SPO accomplishes such high ASR by making fewer word substitutions and improving the semantic similarity than U-SPO. This strongly validates the advantage of incorporating HowNet to select the sememe candidates.

Q3: What’s the advantage of bigram substitutions? The bigram attack is vitally significant in generating meaningful sentence and improving semantic smoothness. To validate this, we propose the BU-SPO method, which considers both bigram and unigram attack. Compared with HU-SPO, the BU-SPO achieves higher USE score even if it changes more words. This means bigram substitution can avoid produce meaningless sentences. Intuitively, we provide two adversarial examples from IMDB dataset in Table VIII, which outperforms competitive methods.

Q4: Can the semantic filter really improve semantic similarity? A straightforward way to answer this question is to compare the BU-SPO and BU-SPOF, since the only difference between the two algorithms is whether they use semantic filter. From Table VI we can see that BU-SPOF attains higher semantic similarity than BU-SPO and often achieves the highest USE score compared with all baselines. This confirms our expectation that the semantic filter is significant in improving the naturalness and fluency of the generated adversarial examples.

Q5: Is the SPO more efficient than other baselines? Let T denote the number of elements in the candidate set S. The generation size and swarm size of the PSO are g and p. The static baselines i.e., BEAT, TEFO, PWWS, and WSA all require \( O(n \times T) \) forward queries to the victim model for achieving a successful attack. The computational cost for the RAND and PSO is \( O(M \times T) \) and \( O(g \times p \times n \times T) \), respectively. The proposed SPO requires \( O(n \times T) \) forward queries for the victim model to select the best candidates, and requires \( O(M) \) queries for the optimization steps, therefore the total computational cost of the SPO is \( O(n \times M \times T) \). Compared with the static baselines and RAND, the SPO needs more additional queries. However, the complexity of our SPO is much smaller than another heuristic search PSO.

Overall, Tables IV–VI elaborate that our proposed algorithms (U-SPO, HU-SPO, BU-SPO, and BU-SPOF) almost take the top three spots in all settings, which outperforms competitive methods.

G. Transferability

Transferability denotes that the adversarial samples generated to mislead a given model \( F \) can also be used to mislead other well-trained models \( F' \) - even if their network structures greatly differ [42]. To evaluate whether our adversarial samples are transferable between models, we build three new convolutional models, including Word CNN2, Word CNN3 and Word CNN4. Based on the basic Word CNN model (described in Section IV-B), Word CNN2 stacks one more densely connected layer, Word CNN3 changes the nonlinear function from Relu to Tanh, and Word CNN4 concatenates another convolutional layer. Then we employ 1000 adversarial samples generated on Word CNN to cheat Word CNN2, Word CNN3, Word CNN4, and the LSTM model. The transfer attack results are reported in Fig. 2. It is apparent that our BU-SPOF achieves the best performance in the
TABLE VII
IMDB Adversarial Examples by Attacking Word CNN

| IMDB Example 1 |
|----------------|
| PWWS (Successful attack. True label score: 70.5% → 1.21%) |
| Nikki Finn is the kind variety of girl I would marry. Never boring, always thinking positively, good with animals. Okay, as one reviewer wrote, a bit too much peroxide, lipstick, and eyebrows (Only Madonna could get away with that). But that’s why I love Nikki Finn, she’s not your ordinary girl. She makes things happen, always exciting to be around, and always honest. |
| BU-SPOF (Successful attack. True label score: 70.5% → 1.91%) |
| Nikki Finn is the kind of girl I would marry. Never boring, always thinking positively, good with animals. Okay, as one reviewer wrote, a bit too much peroxide, lipstick, and eyebrows (Only Madonna could get away with that). But that’s why I love Nikki Finn, she’s not your ordinary girl. She makes things happen, always exciting to be around, and always honest. |

| IMDB Example 2 |
|----------------|
| PWWS (Successful attack. True label score: 97.41% → 43.81%) |
| Alfred Lee Hitchcock Lee shows originality in the remake of his own 1934 British film, "The Man Who Knew Too Much". This 1956 take on the same story is much lighter than the previous one. Mr. Hitchcock was lucky in having collaborators that went with him from one film to the next, thus keeping a standard in his work. Robert Burks did an excellent splendid job problem with the cinematography and George Tomasinis's editing shows his talent. Ultimately, Bernard Hermann is seen conducting at the magnificent Royal Albert Hall in London at the climax of the picture. James Stewart was an actor that worked well with Mr. Hitchcock. In this version, he plays a doctor from Indiana on vacation with his wife and son. When we meet him, they are on their way to Marrakesh in one local bus and the intrigue begins. |
| BU-SPOF (Successful attack. True label score: 97.41% → 47.84%) |
| Alfred Hitchcock Alfred Joseph Hitchcock shows originality in the remake of his own 1934 British film, "The Man Who Knew Too Much". This 1956 take on the same story is much lighter than the previous one. Mr. Hitchcock was lucky in having collaborators that went with him from one film to the next, thus keeping a standard in his work. Robert Burks did an excellent job duty with the cinematography and George Tomasinis's editing shows his talent. Ultimately, Bernard Hermann is seen conducting at the magnificent Royal Albert Hall in London at the climax of the picture. James Stewart was an actor that worked well with Mr. Hitchcock. In this version, he plays a doctor from Indiana on vacation with his wife and son. When we meet him, they are on their way to Marrakesh in one local bus and the intrigue begins. |

TABLE VIII
AG’s News Adversarial Examples by Attacking Word LSTM Model

| AG’s New Example 1 |
|-------------------|
| PWWS (Successful attack. True label score: 90.86% → 37.17%) |
| Afghan women make arrive brief Olympic defeat introduction. Afghan women made a short-lived debut in the Olympic Games on Wednesday as 18-year-old judo wildcard Friba Razayee was defeated after 45 seconds of her first match peer in the under-70kg middleweight. |
| BU-SPOF (Successful attack. True label score: 90.86% → 29.69%) |
| Afghan women make brief Olympic defeat. Afghan women made a short-lived debut in the Olympic Games on Wednesday as 18-year-old judo wildcard Friba Razayee was defeated after 45 seconds of her first match in the under-70kg middleweight. |

| AG’s News Example 2 |
|-------------------|
| PWWS (Successful attack. True label score: 66.13% → 45.21%) |
| Internosis Will Relocate To Greenbelt in October. Internosis Inc., an information entropy technology company in Arlington, plans to move its headquarters to Greenbelt in October. The relocation will bring 170 jobs to Prince George’s County. |
| BU-SPOF (Successful attack. True label score: 66.13% → 18.42%) |
| Internosis Will Relocate To Greenbelt in October. Internosis Inc., an information technology IT company in Arlington, plans to move its headquarters to Greenbelt in October. The relocation will bring 170 jobs to Prince George’s County. |

transferability test, illustrating the effectiveness of the proposed algorithm in transfer attack.

H. Adversarial Training

As an effective method to improve model robustness, adversarial training add adversarial samples into training set and retraining models. This study can make the model adapt to the perturbation of adversarial samples, i.e., improving the defense performance of the model. In this part of the experiment, we randomly select four groups of samples {500, 1000, 1500, 2000} from the training set of AG’s New database to generate adversarial samples, and join the generated adversarial samples to the training set to re-train the victim model (Word CNN). We evaluate if the adversarially retrained model becomes more robust by checking the classification accuracy on the test set. Fig. 3 exhibits the five-run average test accuracy of the Word CNN after adversarial training. From Fig. 3 we obtain following two results: (1) With the increase number of training adversarial examples, the robustness of the retrained model is gradually
TABLE IX
YAHOO! ANSWERS ADVERSARIAL EXAMPLES BY ATTACKING BI-LSTM MODEL

| Yahoo! Answers Example 1 | PWWS (Successful attack. True label score: 92.54% → 43.65%) |
|-------------------------|----------------------------------------------------------|
| What are exist good honorable resources to learn memorize about treatments for prostate cancer? | BU-SPOF (Successful attack. True label score: 92.54% → 30.42%) |
| What are good resources to learn about treatments for prostate cancer prostatic adenocarcinoma? |

| Yahoo! Answers Example 2 | PWWS (Successful attack. True label score: 81.52% → 40.85%) |
|-------------------------|----------------------------------------------------------|
| Why did president Bush Equine get his Masters degree? | BU-SPOF (Successful attack. True label score: 81.52% → 2.37%) |
| Why did president Bush Dubyah get his Masters degree? |

TABLE X
TARGETED ATTACK RESULTS ON AG’S NEWS DATASET

| Model | Target | ASR | AWR |
|-------|--------|-----|-----|
|       | PWWS   | BU-SPOF | PWWS | BU-SPOF |
| CNN   |        |       |     |
| 0     | 87.21% | 95.59% | 4.78 | 3.58 |
| 1     | 70.41% | 91.14% | 8.3  | 5.47 |
| 2     | 73.66% | 93.79% | 7.52 | 4.38 |
| 3     | 82.58% | 93.41% | 6.98 | 4.41 |
| Ch-CNN|        |       |     |
| 1     | 36.88% | 66.18% | 10.91 | 8.79 |
| 2     | 46.58% | 81.03% | 9.11  | 5.49 |
| 3     | 81.76% | 96.95% | 4.53  | 2.81 |
| LSTM  |        |       |     |
| 0     | 77.92% | 90.79% | 7.09  | 5.21 |
| 1     | 72.27% | 91.28% | 7.88  | 5.37 |
| 2     | 81.31% | 90.99% | 5.58  | 4.64 |
| 3     | 79.50% | 88.83% | 6.74  | 5.11 |

The bold entities indicate the highest attack success rate (ASR) and the lowest average number of word substitutions (AWR) of targeted attacks under the AG’s News dataset.

Fig. 2. Transfer attack results on Yahoo! Answers database. The lower the classification accuracy, the higher the transferability (lower is better).

Fig. 3. Adversarial retraining results. The higher the accuracy, the more robust of the model after retraining.

I. Targeted Attack Evaluations

Targeted attack is usually regarded as a more dangerous attack strategy, as it can arbitrarily mislead the victim model to misclassify any label to a pre-specified label. We conduct the targeted attack experiments on AG’s News dataset by attacking Word-CNN, Ch-CNN and Word-LSTM models. For each model, we attack 1000 legitimate samples to the four target labels: 0 (World), 1 (Sports), 2 (Business) and 3 (Sci/Tech). Table X exhibits the targeted attack experimental results. From Table X we can see that our BU-SPOF attains a much higher ASR than PWWS for all target labels and victim models, especially for the Ch-CNN model. Besides, our BU-SPOF replaces less words than PWWS. These results demonstrate that our BU-SPOF is effective for both targeted and untargeted attacks.

V. CONCLUSION

We proposed a novel text attack algorithm in this paper, i.e., Bigram and Unigram based Semantic Preservation Optimization (BU-SPO), to generate fluent text adversarial examples. The BU-SPO exploits not only unigram modifications but also bigram substitutions to avoid breaking commonly used bigram phrases. Besides, the hybrid synonym-sememe candidate selection approach provides better candidate options to produce high quality adversarial examples. More importantly, we design an adaptive SPO algorithm to determine the word substitution priority, which is significant in reducing the perturbation cost. We also proposed a semantic filter (BU-SPOF) in SPO to further improve the semantic preservation performance. Extensive experimental results exhibit that our BU-SPO and BU-SPOF methods achieve high attack success rates (ASR) and high semantic similarity with low numbers of word modifications. The proposed BU-SPOF also shows its superiority on transfer attack, adversarial retraining, and targeted attack. In future, research on defense methods via using a n-gram strategy where n > 2 will be a promising research direction.
REFERENCES

[1] C. Szegedy et al., “Intriguing properties of neural networks,” in Proc. Int. Conf. Learn. Representations, 2014.

[2] P. Liu et al., “Point adversarial self-mining: A simple method for facial expression recognition,” IEEE Trans. Cybern., vol. 52, no. 12, pp. 12649–12660, Dec. 2022.

[3] D. Wang et al., “Daedalus: Breaking nonmaximum suppression in object detection via adversarial examples,” IEEE Trans. Cybern., vol. 52, no. 8, pp. 7427–7440, Aug. 2022.

[4] H. Liu, W. Zhang, F. Liu, H. Wu, and L. Shen, “Fingerprint presentation attack detection using global-local model,” IEEE Trans. Cybern., vol. 52, no. 11, pp. 12315–12328, Nov. 2022.

[5] C. Xu, H. Liu, Z. Guan, X. Wu, J. Tan, and B. Ling, “Adversarial incomplete multiview subspace clustering networks,” IEEE Trans. Cybern., vol. 52, no. 10, pp. 5006–5019, Oct. 2022.

[6] F. Yang, Z. Chen, and A. Gangopadhyay, “Using randomness to improve robustness of tree-based models against evasion attacks,” IEEE Trans. Knowl. Data Eng., vol. 34, no. 2, pp. 969–982, Feb. 2022.

[7] W. Zheng, K. Wang, and F.-Y. Wang, “GAN-based key secret-sharing scheme in blockchain,” IEEE Trans. Cybern., vol. 51, no. 1, pp. 393–404, Jan. 2021.

[8] T. Zhang, X. Gong, and C. L. P. Chen, “BMT-Net: Broad multitask transformer network for sentiment analysis,” IEEE Trans. Cybern., vol. 52, no. 7, pp. 6232–6243, Jul. 2022.

[9] M. He et al., “An efficient phishing webpage detector,” Expert Syst. Appl., vol. 38, no. 10, pp. 12 018–12 027, 2011.

[10] W. Wang, L. Wang, R. Wang, Z. Wang, and A. Ye, “Towards a robust deep neural network in text: A survey,” 2019, arXiv:1902.07285.

[11] W. E. Zhang, Q. Z. Zheng, A. F. Alhazmi, and C. Li, “Generating textual adversarial examples for deep learning models: A survey,” 2019, arXiv:1901.06796.

[12] D. Jin, Z. Jin, J. Tianyi Zhou, and P. Szolovits, “Is bert really robust? a strong baseline for natural language text on classification and entailment,” in Proc. 34th AAAI Conf. Artif. Intell., 2020, pp. 8018–8025.

[13] S. Ren, Y. Deng, K. He, and W. Che, “Generating natural language adversarial examples through probabilistic weight word saliency,” in Proc. 57th Annu. Meet. Assoc. Comput. Linguistics, 2019, pp. 1085–1097.

[14] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, “HotFlip: White-box adversarial examples for text classification,” in Proc. 56th Annu. Meet. Assoc. Comput. Linguistics, 2018, pp. 31–36.

[15] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi, “Black-box generation of adversarial text sequences to evade deep learning classifiers,” in Proc. IEEE Secur. Privacy Workshops, 2018, pp. 50–56.

[16] Y. Belinkov and Y. Bisk, “Synthetic and natural noise both break neural machine translation,” in Proc. Int. Conf. Learn. Representations, 2018.

[17] R. Iia and P. Liang, “Adversarial examples for evaluating reading comprehension systems,” in Proc. Conf. Empir. Methods Natural Lang. Process., 2017, pp. 2021–2031.

[18] E. Wallace, S. Feng, N. Kandpal, M. Gardner, and S. Singh, “Universal trigger sequences for attacking and analyzing NLP,” in Proc. Conf. Empir. Methods Natural Lang. Process., 2019, pp. 2153–2162.

[19] N. Papernot, P. McDaniel, A. Swami, and R. Harang, “Crafting adversarial input sequences for recurrent neural networks,” in Proc. IEEE Mil. Commun. Conf., 2016, pp. 49–54.

[20] M. Alzantot, Y. Sharma, A. Elghoary, B.-J. Ho, M. Srivastava, and K.-W. Chang, “Generating natural language adversarial examples,” Proc. Conf. Empir. Methods Natural Lang. Process., 2018, pp. 2890–2896.

[21] M. Blohm, G. Jagfeld, E. Sood, X. Yu, and N. T. Vu, “Comparing attention-based convolutional and recurrent neural networks: Success and limitations in machine reading comprehension,” in Proc. 22nd Conf. Comput. Natural Lang. Learn., 2018, pp. 108–118.

[22] J. Li, S. Ji, T. Du, B. Li, and T. Wang, “TextBugger: Generating adversarial text against real-world applications,” in Proc. 26th Annu. Netw. Distrib. Syst. Secur. Symp., 2019.

[23] P. Vijayaraghavan and D. Roy, “Generating black-box adversarial examples for text classifiers using a deep reinforced model,” in Proc. Mach. Learn. Knowl. Discov. Databases: Eur. Conf., Würzburg, Germany, Springer, 2020, pp. 711–726.

[24] N. Mikic et al., “Counter-fitting word vectors to linguistic constraints,” in Proc. 15th Annu. Conf. North Amer. Chapter Assoc. Comput. Linguistics, 2016, pp. 142–148.

[25] G. A. Miller, WordNet: An Electronic Lexical Database. Cambridge, MA: MIT Press, 1998.

[26] Z. Dong and Q. Dong, Hownet and the Computation of Meaning. Singapore: World Scientific, 2006.

[27] B. Zhu, Z. Gu, Y. Qian, F. Lau, and Z. Tian, “Leveraging transferability and improved beam search in textual adversarial attacks,” Neurocomputing, vol. 500, pp. 135–142, 2022.

[28] Y. Zhang et al., “Word-level textual adversarial attacking as combinatorial optimization,” in Proc. 58th Annu. Meet. Assoc. Comput. Linguistics, 2020, pp. 6066–6080.

[29] L. Li, R. Ma, Q. Guo, X. Xue, and X. Qiu, “BERT-ATTACK: Adversarial attack against BERT using BERT,” in Proc. Conf. Empir. Methods Natural Lang. Process., 2020, pp. 6193–6202.

[30] S. Garg and G. Ramakrishnan, “BAE: BERT-based adversarial examples for text classification,” 2020, arXiv:2004.01970.

[31] D. Lee, S. Moon, J. Lee, and H. O. Song, “Query-efficient and scalable black-box adversarial attacks on discrete sequential data via Bayesian optimization,” in Proc. Int. Conf. Mach. Learn., PMLR, 2022, pp. 12478–12497.

[32] J. Dong, S. Wang, L. Wu, H. Dong, and Z. Guan, “A textual adversarial attack scheme for domain-specific models,” in Proc. Int. Conf. Mach. Learn. Cyber Secur., Springer, 2023, pp. 104–117.

[33] D. Cer et al., “Universal sentence encoder,” 2018, arXiv:1803.11175.

[34] X. Rong, “word2vec parameter learning explained,” 2014, arXiv:1411.2738.

[35] D. Nouvel, M. Ehrmann, and S. Rosset, Named Entities for Computational Linguistics. Hoboken, NJ, USA: Wiley Online Library, 2016.

[36] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, “Learning word vectors for sentiment analysis,” in Proc. 49th Annu. Meet. Assoc. Comput. Linguistics, 2011, pp. 142–150.

[37] X. Zhang, J. Zhao, and Y. LeCun, “Character-level convolutional networks for text classification,” in Adv. Neural Inf. Process. Syst., 2015, pp. 649–657.

[38] Y. Kim, “Convolutional neural networks for sentence classification,” in Proc. Conf. Empir. Methods Natural Lang. Process., 2014, pp. 1746–1751.

[39] J. Li, W. Monroe, and D. Jurafsky, “Understanding neural networks through representation erasure,” 2016, arXiv:1612.08220.

[40] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. Int. Conf. Learn. Representations, 2015.

[41] J. X. Morris, E. Lilblind, J. Y. Yoo, and Y. Qi, “TextAttack: A framework for adversarial attacks in natural language processing,” 2020, arXiv:2005.05909.

[42] N. Papernot, P. McDaniel, and I. Goodfellow, “Transferability in machine learning: From phenomena to black-box attacks using adversarial samples,” 2016, arXiv:1605.07277.
Weifeng Liu (Senior Member, IEEE) received the double BS degrees in automation and business administration, and the PhD degree in pattern recognition and intelligent systems from the University of Science and Technology of China, Hefei, China, in 2002 and 2007, respectively. He is currently a full professor with the College of Control Science and Engineering, China University of Petroleum, Qingdao, China. He has authored or co-authored a dozen papers in top journals and prestigious conferences, including four Essential Science Indicators (ESI) highly cited papers and two ESI hot papers. His research interests include computer vision, pattern recognition, and machine learning. Prof. Liu serves as an associate editor for the Neural Processing Letters, the co-chair for the IEEE SMC Technical Committee on Cognitive Computing, and a guest editor for the special issue of the Signal Processing, the IET Computer Vision, the Neurocomputing, and the Remote Sensing.

James Bailey (Member, IEEE) received the PhD degree from the University of Melbourne, in 1998. He is a professor with the School of Computing and Information Systems, the University of Melbourne. He was an Australian Research Council Future fellow from 2012–2015. His research interests are in the area of data mining and machine learning, particularly perturbation analysis, clustering, correlation assessment and anomaly detection and explanation. His research has been translated to systems in the area of health, partnering with both hospitals (real time medical emergency prediction for patients) and industry (cognitive systems for immersive simulation training). He has received the best paper award at conferences such as IEEE ICDM, PAKDD and SIAM SDM. He was co-PC chair of PAKDD 2016 and co-general chair of ACM CIKM 2015. He is a member of several Editorial Boards, including ACM Transactions on Data Science, IEEE Transactions on Big Data, and Knowledge and Information Systems.

Dacheng Tao (Fellow, IEEE) is currently the president of the JD Explore Academy and a senior vice president of JD.com, and an advisor and chief scientist of the Digital Science Institute with The University of Sydney. He mainly applies statistics and mathematics to artificial intelligence and data science, and his research is detailed in one monograph and more than 200 publications in prestigious journals and proceedings at leading conferences. Dacheng received the 2015/2020 Australian Eureka Prize, the 2018 IEEE ICDM Research Contributions Award, and the 2021 IEEE Computer Society McCluskey Technical Achievement Award. He is a fellow of the Australian Academy of Science, AAAS, and ACM.

Wei Liu (Senior Member, IEEE) received the PhD degree in machine learning research from the University of Sydney, in 2011. He is currently an Associate Professor and the director of Future Intelligence Lab, School of Computer Science, Faculty of Engineering and Information Technology, the University of Technology Sydney (UTS). Before joining UTS, he was a research fellow with the University of Melbourne and then a machine learning researcher with NICTA. He works in the areas of machine learning and has published more than 90 papers in tensor factorization, adversarial learning, multimodal machine learning, graph mining, causal inference, and anomaly detection. He received the Best Paper Award at AusAI 2012 and the Most Influential Paper Award at PAKDD 2021.