Semantic-Preserving Adversarial Text Attacks

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

Abstract—Deep learning models are known immensely brittle to adversarial image examples, yet their vulnerability in text classification is insufficiently explored. Existing text adversarial attack strategies can be roughly divided into three categories, i.e., character-level attack, word-level attack, and sentence-level attack. Despite the success brought by recent text attack methods, how to induce misclassification with the minimal text modifications while keeping the lexical correctness, syntactic soundness, and semantic consistency simultaneously is still a challenge. To examine the vulnerability of deep models, 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 constraint the SPO with a semantic Filter (dubbed SPOF) to improve the semantic similarity between the input text and the adversarial example. To estimate the effectiveness of our proposed methods, BU-SPO and BU-SPOF, we attack four victim deep learning models trained on three real-world text datasets. Experimental results demonstrate that our approaches accomplish the highest semantics consistency and attack success rates by making the 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

X. Yang, T. Zhu, and Wei Liu are with the School of Computer Science, University of Technology Sydney, 15 Broadway, Ultimo 2007, NSW, Australia. E-mail: Xinghao.Yang@student.uts.edu.au, Tianqing.Zhu@uts.edu.au, Wei.Liu@uts.edu.au.

Yongshun Gong is with the School of Software, Shandong University, Jinan, China. E-mail: ysgong@sdu.edu.cn.

Weifeng Liu is with the College of Control Science and Engineering, China University of Petroleum (East China), Qingdao 266580, China. E-mail: liuwf@upc.edu.cn.

J. Bailey is with the Department of Computing and Information Systems, University of Melbourne, Australia. E-mail: baileyj@unimelb.edu.au.

D. Tao is with the JD Explore Academy, China, and also with the Digital Science Institute, Faculty of Engineering, The University of Sydney, Darlington, NSW 2008, Australia (e-mail: dacheng.tao@gmail.com).

TABLE I
COMPARISONS BETWEEN UNIGRAM ATTACKS AND BIGRAM ATTACKS.
ONE SUPERIORITY OF BIGRAM SUBSTITUTION IS THAT IT CAN
DISTINGUISH COMMONLY USED BIGRAM PHRASES AND AVOID
GENERATING MEANINGLESS SENTENCES.

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

development of adversarial machine learning that focuses on improving the robustness and interpretability of DNNs [2], [3], [4]. 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 [5], secret sharing [6], sentiment analysis [7], and webpage phishing [8].

Compared to the image attacks, crafting natural text adversarial example is more challenging due to the following three folds. Firstly, 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. Secondly, 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. Thirdly, 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 [9]. However, character-based and sentence-based alterations usually lead to the wrong representation of the original content. For example, character-level attack modifies predict → perdict, 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 [10]. 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 [11] and

Fig. 1. The workflow of our BU-SPOF method with an 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).

probability weighted word saliency (PWWS) [12], etc. The common shortcoming of following the static WIS order is the word order 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 $\{top1, top3\}$ sometimes.

To address above-mentioned drawbacks, we present a novel word-level attack algorithm entitled Bigram and Unigram based Semantic Preservation Optimization (BU-SPO) in this paper. 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 → mental doctor and high school → 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. 1 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 an 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 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 three categories, including character-level attack [13], [14], [15], sentence-level attack [16], [17] and word-level attack [18], [19], [12], [11].

Firstly, character-level attack [15], [13], [14] generates adversarial text via three functions, i.e., inserting, swapping or deleting characters. Belinkov and Bisk [15] 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* $\rightarrow$ *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.* [13] 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.* [14] 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.

Additionally, sentence-level attack methods [16], [17] aim to concatenate an adversarial sentence before or more commonly after the clean input text to confuse deep learning methods. For example, Jia and Liang [16] 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.* [17] 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.

Finally, 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.* [18] 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.* [19] 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 $\{\textit{pricey}, \textit{cheaper}, \textit{costly}\}$ according to the GloVe distance, while *cheaper* is the opposite in semantics. Thus, a counter-fitting post-process is a necessary approach to guarantee the semantic constraint [20]. Differ from GloVe, collecting substitution words from well-organized linguistic thesaurus is a relatively straightforward way, such as, synonym-based WordNet [21] and sememe-based HowNet [22]. Ren *et al.* [12] 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. Zang *et al.* [23] 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. Recently, Li *et al.* [24] and Garg *et al.* [25] 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 words 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. It is worth mentioning that all these works focus on unigram attacks and ignore the widely used bigram phrases.

III. ALGORITHM

We discuss the proposed BU-SPO and BU-SPOF methods in this section. Suppose we have N sentences $\mathcal{X} = \{\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_N\}$ belonging to K labels $\mathcal{Y} = \{\mathbf{Y}_1, \mathbf{Y}_2, \dots, \mathbf{Y}_K\}$. 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. Text Adversarial Attack under Black-box Settings

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 $\mathbf{X} \in \mathcal{X}$, a well-performed DNNs model F targets to output the true label $\mathbf{Y}_{true} \in \mathcal{Y}$, i.e., $F(\mathbf{X}) = \mathbf{Y}_{true}$, by optimizing the posterior probability:

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

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

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

or a user-specified target label $\mathbf{Y}_{target}$

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

where $\mathbf{Y}_{target} \neq \mathbf{Y}_{true}$. These two attack strategies are formally defined as untargeted attack Eq. (2) and targeted attack Eq. (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) [26] 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 $\mathbf{X}$ and its adversarial variation $\mathbf{X}_{adv}$ is defined as

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

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 [27]), 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 [21] 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 synonym space $\mathbb{W}$. HowNet [22] 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 $\mathbb{H}$.

1) *Create Candidate Set*: Given an input sentence with n words, i.e., $\mathbf{X} = \{w_1, w_2, \dots, w_n\}$, 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 $\mathbb{W}$. If yes, we create the candidates set $\mathbb{B}_i$ for the bigram (w_i, w_{i+1}) and jump to search substitution words for w_i and w_{i+1} individually¹. Otherwise, we create the candidate set $\mathbb{S}_i$ for w_i by collecting the synonym candidates from $\mathbb{W}$ and the sememe candidates from $\mathbb{H}$, i.e., $\mathbb{S}_i \subset \mathbb{W} \cup \mathbb{H}$. Particularly, we design a candidate filter to ensure that all the candidates in $\mathbb{S}_i$ 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 $\mathbb{S}_i$ 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 [28]. The selected NE candidate (denoted as NE_{COMP}) should keep the same NE type with w_i . Specifically, NE_{COMP} is the one that appears most frequently in the complementary NE set $\mathbb{W} - \mathbb{W}_{\mathbf{Y}_{true}}$ where $\mathbb{W}_{\mathbf{Y}_{true}}$ includes all the NEs belonging to the true label. Thereafter, the synonym set can be enlarged as $\mathbb{S}_i \leftarrow \mathbb{S}_i \cup NE_{COMP}$.

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 $\mathbb{S}_i$ to achieve effective modifications.

2) *Select the Best Candidate*: The candidate set $\mathbb{S}_i$ (or $\mathbb{B}_i$)² usually contains more than one word, and every $w'_i \in \mathbb{S}_i$ is a potential candidate for w_i . To pick the best candidate, the candidate importance score $I_{w'_i}$ for each w'_i is defined as the true label score reduction:

$$I_{w'_i} = P(\mathbf{Y}_{true} | \mathbf{X}) - P(\mathbf{Y}_{true} | \mathbf{X}'_i), \forall w'_i \in \mathbb{S}_i \quad (5)$$

where

$$\mathbf{X} = w_1, w_2, \dots, w_i, \dots, w_n \quad (6)$$

$$\mathbf{X}'_i = w_1, w_2, \dots, w'_i, \dots, w_n \quad (7)$$

Then the best substitution word w_i^* is picked as the w'_i that attains the highest $I_{w'_i}$. Formally, Eq. (8) gives the function for best synonym candidate selection

$$w_i^* = R(w_i, \mathbb{S}_i) = \arg \max_{w'_i \in \mathbb{S}_i} I_{w'_i} \quad (8)$$

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.

¹This means bigram substitution takes precedence.

²In the remainder of this paper, we abuse the notation slightly by utilizing $\mathbb{S}_i$ to indicate the substitution candidate for w_i . If w_i belongs to a bigram (w_i, w_{i+1}) , then $\mathbb{S}_i$ is equivalent to $\mathbb{B}_i$.

Algorithm 1: The proposed BU-SPO Algorithm

Input: Sample sentence containing n words
 $\mathbf{X} = (w_1, \dots, w_n)$
Input: The upper threshold of word replacement M
Input: Well-trained classification model F
Output: Adversarial text sample $\mathbf{X}_{adv}$
 /* Select candidates for input words */

```

1 for  $i = 1$  to  $n$  do
2   Connect  $w_i$  with its next word as  $(w_i, w_{i+1})$ ;
3   Collect bigram candidate set  $\mathbb{B}_i$  for  $(w_i, w_{i+1})$  from
     synonym space  $\mathbb{W}$ ;
4   if  $\mathbb{B}_i \neq \emptyset$  then
5     Select the best candidate bigram from  $\mathbb{B}_i$ ;
6      $i += 1$ ;  $\triangleright$  skip attacking  $w_{i+1}$ 
7   else
8     Collect a synonym candidate and sememe candidate
       set  $\mathbb{S}_i$  for  $w_i$  from  $\mathbb{W} \cup \mathbb{H}$ ;
9     if  $w_i$  is a NE then
10       $\mathbb{S}_i \leftarrow \mathbb{S}_i \cup NE_{COMP}$ ;
11    Select the best candidate unigram from  $\mathbb{S}_i$ ;
12 Create an empty set  $\mathbb{G}^0 = \emptyset$  as the initial generation;
13 Set the modification threshold  $M = \min(M, n)$ ;
   /* The SPO search starts */
14 for  $m = 1$  to  $M$  do
15    $\mathbb{G}^m = \mathcal{F}(\mathbb{G}^{m-1}, \{w_1^*, \dots, w_n^*\})$ ;  $\triangleright$  Alg. 2
16   for  $Candidate \in \mathbb{G}^m$  do
17     Craft  $\mathbf{X}_{adv}$  by replacing  $Candidate$  in  $\mathbf{X}$ ;
18     if  $F(\mathbf{X}) \neq F(\mathbf{X}_{adv})$  then
19       break;  $\triangleright$  successful attack
20     else
21        $\Delta P_{adv} = P(\mathbf{Y}_{true}|\mathbf{X}) - P(\mathbf{Y}_{true}|\mathbf{X}_{adv})$ ;
22 return  $\mathbf{X}_{adv}$ 

```

Algorithm 2: Generation Creation Function ($\mathcal{F}$)

Input: The previous generation population $\mathbb{G}^{m-1}$
Input: The best replacement candidates $\{w_1^*, \dots, w_n^*\}$
Output: Current generation combinations $\mathbb{G}^m$

```

1 Initialize  $\mathbb{G}^m = \emptyset$  as the current generation;
2 if  $\mathbb{G}^{m-1} = \emptyset$  then
3    $\mathbb{G}^m = \{w_1^*, \dots, w_n^*\}$ ;  $\triangleright$  1st generation
4 else
5   Search for the most effective element  $\mathbb{G}_{best}^{m-1}$  from  $\mathbb{G}^{m-1}$ 
     that achieves the biggest  $\Delta P_{adv}$ ;
6   Delete all words in  $\mathbb{G}_{best}^{m-1}$  from  $\{w_1^* \dots w_n^*\}$ , remaining
      $s$  ( $1 \leq s \leq n$ ) elements left;
7   for  $i = 1$  to  $s$  do
8     Combine  $\mathbb{G}_{best}^{m-1}$  with  $i^{th}$  remaining element  $w_i^*$  to
       create the current generation element  $\mathbb{G}^m(i)$ ;
9 return  $\mathbb{G}^m$ 

```

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 $\{\mathbf{X}_1^*, \dots, \mathbf{X}_n^*\}$, where each sentence contains one replaced word, i.e., $\mathbf{X}_i^* = \{w_1, \dots, w_i^*, \dots, w_n\}$. The true label score reduction from $\mathbf{X}$ to $\mathbf{X}_i^*$ denotes the

maximal attack effect which is induced by attacking w_i :

$$\Delta P_i^* = P(\mathbf{Y}_{true}|\mathbf{X}) - P(\mathbf{Y}_{true}|\mathbf{X}_i^*) \quad (9)$$

A straightforward strategy of sorting word replacement is by their ΔP_i^* 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.

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 $\mathbb{G}^0$ 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 $\mathcal{F}$ function defined in Algorithm 2 to create the population set for the current generation, i.e., $\mathbb{G}^m$. When $m = 1$, the $\mathcal{F}$ directly returns the first generation as the n best substitutions $\{w_1^*, \dots, w_n^*\}$. Then we iteratively craft adversarial example $\mathbf{X}_{adv}$ and check whether the classifier is deceived. If yes, the SPO stops the optimization and returns the successful adversarial example $\mathbf{X}_{adv}$. Otherwise, we count the true label score reduction ΔP_{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 ΔP_{adv} .

In every next generation, the function $\mathcal{F}$ constructs the population set $\mathbb{G}^m$ with three major phases, as shown in Algorithm 2 lines 5-8. We first locate the best element that attains the highest ΔP_{adv} from the previous generation $\mathbb{G}^{m-1}$ and denote it as $\mathbb{G}_{best}^{m-1}$. Then we remove all the candidates that are in the best element $\mathbb{G}_{best}^{m-1}$ from the whole set $\{w_1^*, \dots, w_n^*\}$, leaving s ($1 \leq s \leq n$) elements. Finally, the previous best element $\mathbb{G}_{best}^{m-1}$ is combined with every remaining candidate to constitute the new population member $\mathbb{G}^m(i)$ in lines 7-8 of Algorithm 2. Given the new generation set $\mathbb{G}^m$, 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 bond 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 $SucAdv$ to collect all possible adversarial examples that can make

Algorithm 3: The proposed BU-SPOF Algorithm

Input: Sample sentence containing n words
 $\mathbf{X} = (w_1, \dots, w_n)$
Input: The upper threshold of word replacement M
Input: Well-trained classification model F
Output: Adversarial example $\mathbf{X}_{adv}$
 /* Select candidates for input words */
 1 **for** $i = 1$ **to** n **do**
 2 Connect w_i with its next word as (w_i, w_{i+1}) ;
 3 Collect bigram candidate set $\mathbb{B}_i$ for (w_i, w_{i+1}) from
 synonym space $\mathbb{W}$;
 4 **if** $\mathbb{B}_i \neq \emptyset$ **then**
 5 Select the best candidate bigram from $\mathbb{B}_i$;
 6 $i += 1$; ▷ skip attacking w_{i+1}
 7 **else**
 8 Collect a synonym candidate and sememe candidate
 set $\mathbb{S}_i$ for w_i from $\mathbb{W} \cup \mathbb{H}$;
 9 **if** w_i is a NE **then**
 10 $\mathbb{S}_i \leftarrow \mathbb{S}_i \cup NE_{COMP}$;
 11 Find the best unigram candidate from $\mathbb{S}_i$;
 12 Create the initial generation with empty $\mathbb{G}^0 = \emptyset$;
 13 Set the upper bound $M = \min(M, n)$;
 /* The SPOF search starts */
 14 **for** $m = 1$ **to** M **do**
 15 $SucAdv = \emptyset$;
 16 $\mathbb{G}^m = \mathcal{F}(\mathbb{G}^{m-1}, \{w_1^*, \dots, w_n^*\})$; ▷ Alg. 2
 17 **for** $Candidate \in \mathbb{G}^m$ **do**
 18 Craft $\mathbf{X}_{adv}$ by replacing $Candidate$ in $\mathbf{X}$;
 19 **if** $F(\mathbf{X}) \neq F(\mathbf{X}_{adv})$ **then**
 20 Calculate the USE_{score} between $\mathbf{X}$ and $\mathbf{X}_{adv}$
 by Eq. (4);
 21 $SucAdv = SucAdv \cup \mathbf{X}_{adv}$;
 ▷ successful attack
 22 **else**
 23 $\Delta P_{adv} = P(\mathbf{Y}_{true}|\mathbf{X}) - P(\mathbf{Y}_{true}|\mathbf{X}_{adv})$;
 /* The Semantic Filter */
 24 **if** $SucAdv \neq \emptyset$ **then**
 25 Sort adversarial examples in $SucAdv$ by their
 USE_{score} descending order;
 26 $\mathbf{X}_{adv} = SucAdv\{0\}$;
 27 **break**; ▷ stop the next iterations
 28 **return** $\mathbf{X}_{adv}$

successful attacks (Algorithm 3 line 15). If a population member $\mathbf{X}_{adv}$ changes the classifier's prediction (Algorithm 3 line 19), we calculate the semantic similarity score between $\mathbf{X}$ and $\mathbf{X}_{adv}$ (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 $\mathbf{Y}_{target}$. In this section, we show our BU-SPO and BU-SPOF algorithms can

TABLE II
DATASET INFORMATION SUMMARIZATION. “# AVG. WORDS” IS THE AVERAGE NUMBER OF WORDS FOR ALL SAMPLES.

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

TABLE III
THE ORIGINAL TEST ACCURACY BEFORE ATTACKS.

Dataset	Model	Original Test Accuracy
IMDB	CNN	87.97%
	LSTM	87.94%
	Bi-LSTM	85.71%
AG's News	CNN	90.75%
	LSTM	91.62%
	Ch-CNN	89.24%
Yahoo! Answers	CNN	71.21%
	Bi-LSTM	71.60%
	Ch-CNN	64.62%

be easily adapted to conduct targeted attacks by making the following three modifications. Firstly, we change the successful attack condition in Algorithm 1 line 18 and Algorithm 3 line 19 from $F(\mathbf{X}) \neq F(\mathbf{X}_{adv})$ to $F(\mathbf{X}_{adv}) = \mathbf{Y}_{target}$. This means we only count adversarial examples that can mislead the classifier to the target label as successful attacks. Secondly, 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 Eq. (5) is reformulated to Eq. (10)

$$I_{w'_i} = P(\mathbf{Y}_{target}|\mathbf{X}'_i) - P(\mathbf{Y}_{target}|\mathbf{X}), \forall w'_i \in \mathbb{S}_i \quad (10)$$

and Eq. (9) is transformed to Eq. (11)

$$\Delta P_i^* = P(\mathbf{Y}_{target}|\mathbf{X}_i^*) - P(\mathbf{Y}_{target}|\mathbf{X}) \quad (11)$$

Additionally, line 21 of Algorithm 1 and line 23 of Algorithm 3 become Eq. (12)

$$\Delta P_{adv} = P(\mathbf{Y}_{target}|\mathbf{X}_{adv}) - P(\mathbf{Y}_{target}|\mathbf{X}). \quad (12)$$

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 Github³ to ensure reproducibility.

³<https://github.com/AdvAttack/BU-SPO>

TABLE IV

THE ATTACK SUCCESS RATE (ASR) OF DIFFERENT ATTACK METHODS. IN EACH COLUMN, THE BEST ASR IS DENOTED AS BOLD, THE SECOND BEST ASR IS HIGHLIGHTED IN UNDERLINE, AND THE THIRD BEST ASR IS DISTINGUISHED WITH ITALIC FONT.

Methods	IMDB			AG's News			Yahoo! Answers		
	CNN	Bi-LSTM	LSTM	CNN	LSTM	Ch-CNN	CNN	Bi-LSTM	Ch-CNN
BEAT	91.02%	90.15%	<u>99.77%</u>	86.91%	77.52%	N/A	N/A	N/A	N/A
PSO	100%	100%	100%	85.48%	79.87%	N/A	N/A	N/A	N/A
TEFO	100%	100%	100%	82.75%	85.17%	N/A	N/A	N/A	N/A
PWWS	<u>94.60%</u>	<u>99.76%</u>	90.01%	85.48%	79.87%	75.48%	68.39%	67.76%	97.59%
WSA	38.85%	81.70%	31.65%	77.98%	74.73%	68.33%	66.82%	66.20%	96.36%
PWWS ($M = 20$)	90.56%	97.09%	82.60%	81.98%	76.81%	75.25%	68.23%	67.29%	96.54%
RAND	81.13%	94.79%	74.33%	82.42%	79.76%	75.26%	51.96%	51.64%	95.50%
U-SPO	<u>95.58%</u>	<u>98.90%</u>	90.80%	85.76%	81.07%	<u>80.70%</u>	70.27%	67.92%	97.59%
HU-SPO	100%	100%	98.65%	<u>92.77%</u>	<u>88.40%</u>	92.17%	90.14%	87.95%	98.91%
BU-SPO	100%	100%	98.32%	<u>91.77%</u>	<u>86.21%</u>	<u>91.49%</u>	<u>88.58%</u>	<u>85.60%</u>	<u>98.50%</u>
BU-SPOF	100%	100%	100%	93.44%	89.06%	<u>91.49%</u>	<u>87.17%</u>	<u>85.45%</u>	<u>98.29%</u>

TABLE V

THE AVERAGE WORD REPLACEMENT (AWR) NUMBER OF DIFFERENT ATTACK ALGORITHMS. IN EACH COLUMN, THE BEST AWR IS REPRESENTED WITH BOLD, THE SECOND BEST AWR IS HIGHLIGHTED IN UNDERLINE, AND THE THIRD BEST AWR IS DENOTED AS ITALIC FONT.

Methods	IMDB			AG's News			Yahoo! Answers		
	CNN	Bi-LSTM	LSTM	CNN	LSTM	Ch-CNN	CNN	Bi-LSTM	Ch-CNN
BEAT	7.50	7.89	9.88	5.97	6.24	N/A	N/A	N/A	N/A
PSO	3.42	5.22	5.91	5.02	5.82	N/A	N/A	N/A	N/A
TEFO	8.01	8.13	7.43	7.42	8.49	N/A	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	<i>1.04</i>
HU-SPO	<u>2.07</u>	<u>2.28</u>	2.67	4.16	<u>5.74</u>	3.41	1.85	2.28	1.02
BU-SPO	<u>2.11</u>	<u>2.33</u>	<u>2.58</u>	<u>4.32</u>	5.99	<u>3.52</u>	<u>1.89</u>	<u>2.40</u>	<i>1.04</i>
BU-SPOF	2.06	1.62	2.38	<u>4.17</u>	5.65	3.62	<u>1.96</u>	<u>2.32</u>	<u>1.03</u>

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.

IMDB [29] is a sentiment analysis dataset containing 50,000 movie reviews, belonging to either positive class or negative class. The average text length is 227 words (without punctuation). AG's News [30] is a news classification dataset with 4 topic classes, i.e., Business, Sports, World, and Sci/Tech. Each class consists of 30,000 train examples and 1,900 test documents. It totally contains 120,000 training sample and 7,600 testing samples. Yahoo! Answers [30] is a 10-class topic categorization dataset with 1,400,000 text samples for training and 60,000 for testing.

B. Victim Models

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) [31] 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 nonlinear 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) [30] 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 [18]. 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

TABLE VI

THE AVERAGE UNIVERSAL SENTENCE ENCODER (USE) SCORE OF VARIOUS METHODS. IN EACH COLUMN, THE BEST USE SCORE IS DENOTED AS BOLD, THE SECOND BEST USE SCORE IS HIGHLIGHTED IN UNDERLINE, AND THE THIRD BEST USE SCORE IS REPRESENTED WITH ITALIC.

Methods	IMDB			AG's News			Yahoo! Answers		
	CNN	Bi-LSTM	LSTM	CNN	LSTM	Ch-CNN	CNN	Bi-LSTM	Ch-CNN
BEAT	0.9825	0.9764	0.9786	0.859	0.8412	N/A	N/A	N/A	N/A
PSO	<u>0.9940</u>	0.9889	0.9849	0.9175	0.9056	N/A	N/A	N/A	N/A
TEFO	0.9878	0.9851	<i>0.9859</i>	0.8773	0.8530	N/A	N/A	N/A	N/A
PWWS	0.9905	0.9852	0.9660	0.8557	0.8079	0.9223	0.8012	0.7915	0.9019
WSA	0.9514	0.9325	0.9192	0.7770	0.7565	0.8481	0.7558	0.7542	0.8965
PWWS ($M = 20$)	0.9907	0.9868	0.9761	0.8593	0.8206	0.9241	0.8014	0.7923	0.8993
RAND	0.9893	0.9839	0.9711	0.8874	0.8519	0.9326	<i>0.8045</i>	<i>0.7940</i>	<i>0.8994</i>
U-SPO	0.9921	0.9881	0.9804	0.8928	0.8585	0.9350	0.8176	0.8048	0.9020
HU-SPO	<u>0.9945</u>	<u>0.9934</u>	<u>0.9922</u>	0.9028	0.8622	<i>0.9416</i>	0.7969	0.7735	0.8861
BU-SPO	<u>0.9945</u>	<u>0.9935</u>	<u>0.9922</u>	<i>0.9044</i>	<i>0.8647</i>	<u>0.9431</u>	0.8017	0.7752	0.8877
BU-SPOF	0.9970	0.9958	0.9941	<u>0.9122</u>	<u>0.8749</u>	0.9526	0.8292	0.8108	0.9344

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 $F(\mathbf{X}) = \mathbf{Y}_{true}$ but makes a wrong prediction on the corresponding attacked input $F(\mathbf{X} + \Delta\mathbf{X}) = \mathbf{Y}^*$. Therefore, the ASR is defined as

$$ASR = \frac{\sum_{\mathbf{X} \in \mathcal{X}} \{F(\mathbf{X}) = \mathbf{Y}_{true} \wedge F(\mathbf{X} + \Delta\mathbf{X}) = \mathbf{Y}^*\}}{\sum_{\mathbf{X} \in \mathcal{X}} \{F(\mathbf{X}) = \mathbf{Y}_{true}\}} \quad (13)$$

where $\mathbf{Y}^*$ can be any label different from the $\mathbf{Y}_{true}$ (untargeted attack) or equal to a user specified label (targeted attack). The $\Delta\mathbf{X}$ 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.

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) [32] obtains candidates from WordNet and runs the attack by the word saliency (WS) descending order. The strategy of WS is essentially analogous to Eq. (5) but substitutes w_i with *unknown*.
- PWWS [12] selects substitution candidates from WordNet and weights the word importance by multiplying the probability reduction and word saliency.
- PSO [23] chooses sememe candidates via HowNet and designs the PSO to optimize adversarial text. The PSO regards every sample as a particle and optimizes its space location to accomplish word modifications.
- TextFooler (TEFO) [11] 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) [24] 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 Eq. (5) but replaces w_i by [MASK].

E. Experimental Settings

All the victim DNNs models are trained by utilizing the ADAM optimizer [33], 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 [34], 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 Table IV, Table V, and Table 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 [23], [11].

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 a little 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 Tomasini's editing shows his talent. Ultimately, Bernard Herrmann 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 Tomasini's editing shows his talent. Ultimately, Bernard Herrmann 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.

Fig. 2. IMDB adversarial examples by attacking Word CNN. Blue and red texts denote input words and substitutions, respectively.

F. Experimental Results and Analysis

Table IV and Table V, and Table 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 Table 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, Table V and Table 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 Fig. 2, AG's News dataset in Fig. 3, and Yahoo! Answers dataset in Fig. 4 for qualitative analysis. These adversarial examples demonstrate that the proposed bigram substitution strategy can largely reduce the semantic changes. For example, Fig. 3 shows that our method replaces two words (i.e., information technology → IT) but causes less semantic variation than PWWS only changing one word (information → entropy).

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 naturality and fluency of the generated adversarial examples.

Overall, Table IV - VI elaborate that our proposed algorithms (U-SPO, HU-SPO, BU-SPO, and BU-SPOF) almost

AG's News Example 1	
PWWS (Successful attack. True label score: 90.86% → 37.17%)	
Afghan women make arrive brief Olympic debut introduction . Afghan women made a short-lived debut in the Olympic Games on Wednesday as 18-year-old judo wildcard Friha 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 debut. Afghan women made a short-lived debut in the Olympic Games Olympiad on Wednesday as 18-year-old judo wildcard Friha 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.	

Fig. 3. AG's News adversarial examples by attacking Word LSTM model. Blue and red texts denote input words and substitutions, respectively.

Yahoo! Answers Example 1	
PWWS (Failure. 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 Dubyuh get his Masters degree?	

Fig. 4. Yahoo! Answers adversarial examples by attacking Bi-LSTM model. Blue and red texts denote input words and substitutions, respectively.

take the top three spots on all settings, which outperforms than 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 [35]. 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. 5. It is apparent that our BU-SPOF achieves the best performance in the transferability test, illustrating the effectiveness of the proposed algorithm in transfer attack.

Fig. 5. Transfer attack results on Yahoo! Answers database. The lower the classification accuracy, the higher the transferability (lower is better).

H. Adversarial Training

As an effective method to improve model robustness, adversarial training add adversarial samples into training set and retraining models. 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

Fig. 6. Adversarial retraining results. The higher the accuracy, the more robust of the model after retraining.

TABLE VII
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%	6.08	4.41
Ch-CNN	0	32.61%	65.99%	11.56	9.11
	1	36.88%	66.18%	10.91	8.79
	2	46.58%	81.03%	9.11	5.49
	3	81.76%	96.05%	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

robust by checking the classification accuracy on the test set. Fig. 6 exhibits the five-run average test accuracy of the Word CNN after adversarial training. From Fig. 6 we obtain following two results: (1) With the increase number of training adversarial examples, the robustness of the retrained model is gradually improved; (2) Compared with the baseline PWWS, our BU-SPO and BU-SPOF methods produce more effective adversarial examples in improving the robustness of victim model.

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 VII exhibits the targeted attack experimental results. From Table VII 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-sesame 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 show 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, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” in *International Conference on Learning Representations*, 2014.
- [2] P. Liu, Y. Lin, Z. Meng, L. Lu, W. Deng, J. T. Zhou, and Y. Yang, “Point adversarial self-mining: A simple method for facial expression recognition,” *IEEE Transactions on Cybernetics*, pp. 1–12, 2021.
- [3] D. Wang, C. Li, S. Wen, Q.-L. Han, S. Nepal, X. Zhang, and Y. Xiang, “Daedalus: Breaking nonmaximum suppression in object detection via adversarial examples,” *IEEE Transactions on Cybernetics*, pp. 1–14, 2021.
- [4] H. Liu, W. Zhang, F. Liu, H. Wu, and L. Shen, “Fingerprint presentation attack detector using global-local model,” *IEEE Transactions on Cybernetics*, pp. 1–14, 2021.
- [5] F. Yang, Z. Chen, and A. Gangopadhyay, “Using randomness to improve robustness of tree-based models against evasion attacks,” *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2020.
- [6] W. Zheng, K. Wang, and F.-Y. Wang, “Gan-based key secret-sharing scheme in blockchain,” *IEEE Transactions on Cybernetics*, vol. 51, no. 1, pp. 393–404, 2021.
- [7] T. Zhang, X. Gong, and C. L. P. Chen, “Bmt-net: Broad multitask transformer network for sentiment analysis,” *IEEE Transactions on Cybernetics*, pp. 1–12, 2021.
- [8] M. He, S.-J. Horng, P. Fan, M. K. Khan, R.-S. Run, J.-L. Lai, R.-J. Chen, and A. Sutanto, “An efficient phishing webpage detector,” *Expert systems with applications*, vol. 38, no. 10, pp. 12 018–12 027, 2011.
- [9] W. Wang, L. Wang, R. Wang, Z. Wang, and A. Ye, “Towards a robust deep neural network in texts: A survey,” *arXiv preprint arXiv:1902.07285*, 2019.
- [10] W. E. Zhang, Q. Z. Sheng, A. A. F. Alhazmi, and C. Li, “Generating textual adversarial examples for deep learning models: A survey,” *CoRR*, abs/1901.06796, 2019.
- [11] D. Jin, Z. Jin, J. Tianyi Zhou, and P. Szolovits, “Is bert really robust? a strong baseline for natural language attack on text classification and entailment,” in *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.
- [12] S. Ren, Y. Deng, K. He, and W. Che, “Generating natural language adversarial examples through probability weighted word saliency,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 1085–1097.
- [13] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, “Hotflip: White-box adversarial examples for text classification,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018, pp. 31–36.
- [14] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi, “Black-box generation of adversarial text sequences to evade deep learning classifiers,” in *IEEE SPW*, 2018, pp. 50–56.

- [15] Y. Belinkov and Y. Bisk, "Synthetic and natural noise both break neural machine translation," in *International Conference on Learning Representations*, 2018.
- [16] R. Jia and P. Liang, "Adversarial examples for evaluating reading comprehension systems," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2021–2031.
- [17] E. Wallace, S. Feng, N. Kandpal, M. Gardner, and S. Singh, "Universal trigger sequences for attacking and analyzing nlp," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019, pp. 2153–2162.
- [18] N. Papernot, P. McDaniel, A. Swami, and R. Harang, "Crafting adversarial input sequences for recurrent neural networks," in *IEEE Military Communications Conference*, 2016, pp. 49–54.
- [19] M. Alzantot, Y. Sharma, A. Elgohary, B.-J. Ho, M. Srivastava, and K.-W. Chang, "Generating natural language adversarial examples," *Proceedings of the 2018 conference on Empirical Methods in Natural Language Processing*, pp. 2890–2896, 2018.
- [20] N. Mrkšić, D. O. Séaghdha, B. Thomson, M. Gašić, L. Rojas-Barahona, P.-H. Su, D. Vandyke, T.-H. Wen, and S. Young, "Counter-fitting word vectors to linguistic constraints," in *The 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2016, pp. 142–148.
- [21] G. A. Miller, *WordNet: An electronic lexical database*. MIT press, 1998.
- [22] Z. Dong and Q. Dong, *HowNet and the computation of meaning*. World Scientific, 2006.
- [23] Y. Zang, F. Qi, C. Yang, Z. Liu, M. Zhang, Q. Liu, and M. Sun, "Word-level textual adversarial attacking as combinatorial optimization," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 6066–6080.
- [24] L. Li, R. Ma, Q. Guo, X. Xue, and X. Qiu, "Bert-attack: Adversarial attack against bert using bert," *arXiv preprint arXiv:2004.09984*, 2020.
- [25] S. Garg and G. Ramakrishnan, "Bae: Bert-based adversarial examples for text classification," *arXiv preprint arXiv:2004.01970*, 2020.
- [26] D. Cer, Y. Yang, S. yi Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, Y.-H. Sung, B. Strope, and R. Kurzweil, "Universal sentence encoder," 2018.
- [27] X. Rong, "word2vec parameter learning explained," *arXiv preprint arXiv:1411.2738*, 2014.
- [28] D. Nouvel, M. Ehrmann, and S. Rosset, *Named entities for computational linguistics*. Wiley Online Library, 2016.
- [29] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, 2011, pp. 142–150.
- [30] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, 2015, pp. 649–657.
- [31] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 2014, pp. 1746–1751.
- [32] J. Li, W. Monroe, and D. Jurafsky, "Understanding neural networks through representation erasure," *arXiv preprint arXiv:1612.08220*, 2016.
- [33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.
- [34] J. X. Morris, E. Lifland, J. Y. Yoo, and Y. Qi, "Textattack: A framework for adversarial attacks in natural language processing," *arXiv preprint arXiv:2005.05909*, 2020.
- [35] N. Papernot, P. McDaniel, and I. Goodfellow, "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples," *arXiv preprint arXiv:1605.07277*, 2016.

Xinghao Yang received the B.Eng. degree in electronic information engineering and M.Eng. degree in information and communication engineering from the China University of Petroleum (East China), Qingdao, China, in 2015 and 2018, respectively. Currently, he is a PhD student in School of Computer Science, University of Technology Sydney, Australia. His research interests include multi-view learning and adversarial machine learning with publications on AAAI, IJCAI, TKDE, Information Fusion and Information Sciences.

Yongshun Gong (M'21) is an Associate Professor at the School of Software, Shandong University, China. He received his Ph.D. degree from the University of Technology Sydney, Australia. His principal research interest covers the data mining and machine learning. He has published above 20 papers in top journals and refereed conference proceedings, including the IEEE T-KDE, IEEE T-NNLS, IEEE T-CYB, IEEE T-MM, Pattern Recognition, NeurIPS, KDD, CIKM, AAAI and IJCAI.

Weifeng Liu (M'12-SM'17) received the double B.S. degrees in automation and business administration and the Ph.D. 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 received the PhD degree from the University of Melbourne, in 1998. He is a Professor in the School of Computing and Information Systems at 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.

Tianqing Zhu received the B.Eng. and M.Eng. degrees from Wuhan University, China, in 2000 and 2004, respectively, and the Ph.D. degree in computer science from Deakin University, Australia, in 2014. She served as a Lecturer with Wuhan Polytechnic University, China, from 2004 to 2011. She is currently an Associate Professor at the School of Computer Science, University of Technology Sydney, Australia. Her research interests include privacy preserving, cyber security, and network security.

Dr. Tianqing received the Best Student Paper

Award in PAKDD 2014.

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 in 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 over 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, ACM, and IEEE.

Wei Liu (M'15-SM'20) 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 at the School of Computer Science, Faculty of Engineering and Information Technology, the University of Technology Sydney (UTS). Before joining UTS, he was a Research Fellow at the University of Melbourne and then a Machine Learning Researcher at 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.

Wei received the Best Paper Award at AusAI 2012 and the Most Influential Paper Award at PAKDD 2021.