Are Synonym Substitution Attacks Really *Synonym* Substitution Attacks?

Cheng-Han Chiang
National Taiwan University,
Taiwan
dcml0714@gmail.com

Hung-yi Lee
National Taiwan University,
Taiwan
hungyilee@ntu.edu.tw

Abstract

In this paper, we explore the following question: Are synonym substitution attacks really *synonym* substitution attacks (SSAs)? We approach this question by examining how SSAs replace words in the original sentence and show that there are still unresolved obstacles that make current SSAs generate invalid adversarial samples. We reveal that four widely used word substitution methods generate a large fraction of invalid substitution words that are ungrammatical or do not preserve the original sentence's semantics. Next, we show that the semantic and grammatical constraints used in SSAs for detecting invalid word replacements are highly insufficient in detecting invalid adversarial samples.

1 Introduction

Deep learning-based natural language processing models have been extensively used in different tasks in many domains and have shown strong performance in different realms. However, these models seem to be astonishingly vulnerable in that their predictions can be misled by some small perturbations in the original input (Gao et al., 2018; Tan et al., 2020). These *imperceptible* perturbations, while not changing humans' predictions, can make a well-trained model behave worse than random.

One important type of adversarial attack in natural language processing (NLP) is the **synonym substitution attack** (SSA). In SSAs, an adversarial sample is constructed by substituting some words in the original sentence with their synonyms (Alzantot et al., 2018; Ren et al., 2019; Garg and Ramakrishnan, 2020; Jin et al., 2020; Li et al., 2020; Maheshwary et al., 2021). This ensures that the adversarial sample is semantically similar to the original sentence, thus fulfilling the imperceptibility requirement of a valid adversarial sample. While substituting words with their semantic-related counterparts can retain the semantics of the original sentence, these attacks often utilize constraints to further guarantee that the generated adversarial samples are grammatically correct and semantically similar to the original sentence. These SSAs have all been shown to successfully bring down well-trained text classifiers' performance.

However, some recent works observe, by human evaluations, that the quality of the generated adversarial samples of those SSAs is fairly low and is highly perceptible by human (Morris et al., 2020a; Hauser et al., 2021). These adversarial samples often contain grammatical errors and do not preserve the semantics of the original samples, making them difficult to understand. These characteristics violate the fundamental criteria of a ***valid adversarial sample***: preserving semantics and being imperceptible to humans. This motivates us to investigate what causes those SSAs to generate invalid adversarial samples. Only by answering this question can we move on to design more realistic SSAs in the future.

In this paper, we are determined to answer the following question: Are synonym substitution attacks in the literature really *synonym* substitution attacks? We explore the answer by scrutinizing the key components in several important SSAs and why they fail to generate valid adversarial samples. Specifically, we conduct a detailed analysis of how the word substitution sets are obtained in SSAs, and we look into the semantic and grammatical constraints used to filter invalid adversarial samples. We have the following astonishing observations:

- When substituting words by WordNet synonym sets, current methods neglect the word sense differences within the substitution set. (Section 3.1)
- When using counter-fitted GloVe embedding space or BERT to generate the substitution set, the substitution set only contains a teeny-tiny fraction of synonyms. (Section 3.2)

Findings of the Association for Computational Linguistics: ACL 2023, pages 1853–1878
July 9-14, 2023 ©2023 Association for Computational Linguistics

- Using word embedding cosine similarity or sentence embedding cosine similarity to filter words in the substitution set does not necessarily exclude semantically invalid word substitutions. (Section 4.1 and Section 4.2)
- The grammar checker used for filtering ungrammatical adversarial samples fails to detect most erroneous verb inflectional forms in a sentence. (Section 4.3)

2 Backgrounds

In this section, we provide an overview of SSAs and introduce some related notations that will be used throughout the paper.

2.1 Synonym Substitution Attacks (SSAs)

Given a victim text classifier trained on a dataset D_{train} and a clean testing data $\mathbf{x}_{ori}$ sampled from the same distribution of D_{train}; $\mathbf{x}_{ori} = \{x_1, \cdots, x_T\}$ is a sequence with T tokens. An SSA attacks the victim model by constructing an adversarial sample $\mathbf{x}_{adv} = \{x_1^{'}, \cdots, x_T^{'}\}$ by swapping the words in $\mathbf{x}_{ori}$ with their semantic-related counterparts. For $\mathbf{x}_{adv}$ to be considered as a **valid** adversarial sample of $\mathbf{x}_{ori}$, a few requirements must be met (Morris et al., 2020a): (0) $\mathbf{x}_{adv}$ should make the model yield a wrong prediction while the model can correctly classify $\mathbf{x}_{ori}$. (1) $\mathbf{x}_{adv}$ should be semantically similar with $\mathbf{x}_{ori}$. (2) $\mathbf{x}_{adv}$ should not induce new grammar errors compared with $\mathbf{x}_{ori}$. (3) The word-level overlap between $\mathbf{x}_{adv}$ and $\mathbf{x}_{ori}$ should be high enough. (4) The modification made in $\mathbf{x}_{adv}$ should be natural and non-suspicious. **In our paper, we will refer to the adversarial samples that fail to meet the above criteria as invalid adversarial samples.**

SSAs rely on heuristic procedures to ensure that $\mathbf{x}_{adv}$ satisfies the preceding specifications. Here, we describe a canonical pipeline of generating $\mathbf{x}_{adv}$ from $\mathbf{x}_{ori}$ (Morris et al., 2020b). Given a clean testing sample $\mathbf{x}_{ori}$ that the text classifier correctly predicts, an SSA will first generate a candidate word substitution set $\mathbb{S}_{x_i}$ for each word x_i. The process of generating the candidate set $\mathbb{S}_{x_i}$ is called **transformation**. Next, the SSA will determine which word in $\mathbf{x}_{ori}$ should be substituted first, and which word should be the next to swap, etc. After the word substitution order is decided, the SSA will iteratively substitute each word x_i in $\mathbf{x}_{ori}$ using the candidate words in $\mathbb{S}_{x_i}$ according to the predetermined order. In each substitution step, an x_i is replaced by a word in $\mathbb{S}_{x_i}$, and a new $\mathbf{x}_{swap}$ is obtained. When an $\mathbf{x}_{swap}$ is obtained, some **constraints** are used to verify the validity of $\mathbf{x}_{swap}$. The iterative word substitution process will end if the model's prediction is successfully corrupted by a substituted sentence that sticks to the constraints, yielding the desired $\mathbf{x}_{adv}$ eventually.

Clearly, the transformations and the constraints are critical to the quality of the final $\mathbf{x}_{adv}$. In the remaining part of the paper, we will look deeper into the transformations and constraints used in SSAs and their role in creating adversarial samples[1]. Next, we briefly introduce the transformations and constraints that have been used in SSAs.

2.2 Transformations

Transformation is the process of generating the substitution set $\mathbb{S}_{x_i}$ for a word x_i in $\mathbf{x}_{ori}$. There are four representative transformations in the literature.

WordNet Synonym Transformation constructs $\mathbb{S}_{x_i}$ by querying a word's synonym using WordNet (Miller, 1995; University, 2010), a lexical database containing the word sense definition, synonyms, and antonyms of the words in English. This transformation is used in PWWS (Ren et al., 2019) and LexicalAT (Xu et al., 2019).

Word Embedding Space Nearest Neighbor Transformation constructs $\mathbb{S}_{x_i}$ by looking up the word embedding of x_i in a word embedding space, and finding its k nearest neighbors (kNN) in the word embedding space. Using kNN for word substitution is based on the assumption that semantically related words are closer in the word embedding space. Counter-fitted GloVe embedding space (Mrkšić et al., 2016) is the embedding space obtained from post-processing the GloVe embedding space (Pennington et al., 2014). Counter-fitting refers to the process of pulling away antonyms and narrowing the distance between synonyms. This transformation is adopted in TextFooler (Jin et al., 2020), Genetic algorithm attack (Alzantot et al., 2018), and TextFooler-Adj (Morris et al., 2020a).

[1] In our paper, we do not discuss the relationship between the validity of an SSA and how an SSA determines which word in $\mathbf{x}_{ori}$ should be substituted. Most SSAs use word importance scores to determine what the most salient words are and substitute the most salient words. Since most SSAs use similar methods to determine what word should be replaced, our analyses are generalizable to those SSAs.

Masked Language Model (MLM) Mask-Infilling Transformation constructs $\mathbb{S}_{x_i}$ by masking x_i in $\mathbf{x}_{ori}$ and asking an MLM to predict the masked token; MLM's top-k prediction of the masked token forms the word substitution set of x_i. Widely adopted MLMs includes BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019). Using MLM mask-infilling to generate a candidate set relies on the belief that MLMs can generate fluent and semantic-consistent substitutions for $\mathbf{x}_{ori}$. This method is used in BERT-ATTACK (Li et al., 2020) and CLARE (Li et al., 2021).

MLM Reconstruction Transformation also uses MLMs. When using MLM reconstruction transformation to generate the candidate set, one just feeds the MLM with the original sentence $\mathbf{x}_{ori}$ **without masking** any tokens in the sentence. Here, the MLM is not performing mask-infilling but reconstructs the input tokens from the unmasked inputs. For each word x_i, one can take its top-k token reconstruction prediction as the candidates. This transformation relies on the intuition that reconstruction can generate more semantically similar words than using mask-infilling. This method is used in BAE (Garg and Ramakrishnan, 2020).

2.3 Constraints

When an $\mathbf{x}_{ori}$ is perturbed by swapping some words in it, we need to use some constraints to check whether the perturbed sentence, $\mathbf{x}_{swap}$, is semantically or grammatically valid or not. We use $\mathbf{x}_{swap}$ instead of $\mathbf{x}_{adv}$ here as $\mathbf{x}_{swap}$ does not necessarily flip the model's prediction and thus not necessarily an adversarial sample.

Word Embedding Cosine Similarity requires a word x_i and its perturbed counterpart $x_i^{'}$ to be close enough in the counter-fitted GloVe embedding space, in terms of cosine similarity. A substitution is valid if its word embedding's cosine similarity with the original word's embedding is higher than a pre-defined threshold. This is used in Genetic Algorithm Attack (Alzantot et al., 2018) and TextFooler (Jin et al., 2020).

Sentence Embedding Cosine Similarity demands that the sentence embedding cosine similarity between $\mathbf{x}_{swap}$ and $\mathbf{x}_{ori}$ are higher than a pre-defined threshold. Most previous works (Jin et al., 2020; Li et al., 2020; Garg and Ramakrishnan, 2020; Morris et al., 2020a) use Universal Sentence Encoder (USE) (Cer et al., 2018) as the sentence encoder; A2T (Yoo and Qi, 2021) use a DistilBERT (Sanh et al., 2019) fine-tuned on STS-B (Cer et al., 2017) as the sentence encoder.

In some previous work (Li et al., 2020), the sentence embedding is computed using the whole sentence $\mathbf{x}_{ori}$ and $\mathbf{x}_{swap}$. But most previous works (Jin et al., 2020; Garg and Ramakrishnan, 2020) only extract a context around the currently swapped word in $\mathbf{x}_{ori}$ and $\mathbf{x}_{swap}$ to compute the sentence embedding. For example, if x_i is substituted in the current substitution step, one will compute the sentence embedding between $\mathbf{x}_{ori}[i-w:i+w+1]$ and $\mathbf{x}_{adv}[i-w:i+w+1]$, where w determines the window size. w is set to 7 in Jin et al. (2020) and Garg and Ramakrishnan (2020).

LanguageTool (language-tool python, 2022) is an open-source grammar tool that can detect spelling errors and grammar mistakes in an input sentence. It is used in TextFooler-Adj (Morris et al., 2020a) to evaluate the grammaticality of the adversarial samples.

3 Problems with the Transformations in SSAs

In this section, we show that the transformations introduced in Section 2.2 are largely to blame for the invalid adversarial samples in SSAs. This is because the substitution set $\mathbb{S}_{x_i}$ for x_i is mostly invalid, either semantically or grammatically.

3.1 WordNet Synonym Substitution Set Ignores Word Senses

In WordNet, each word is associated with one or more word senses, and each word sense has its corresponding synonym sets. Thus, the substitution set $\mathbb{S}_{x_i}$ proposed by WordNet is the union of the synonym sets of different senses of x_i. When swapping x_i with its synonym using WordNet, it is more sensible to first identify the word sense of x_i in $\mathbf{x}_{ori}$, and use the synonym set of the very sense as the substitution set. However, current attacks using WordNet synonym substitution neglect the sense differences within the substitution set (Ren et al., 2019), which may result in adversarial samples that semantically deviate from the original input.

As a working example, consider a movie review that reads "I highly recommend it". The word "recommend" here corresponds to the word sense of "*express a good opinion of*" according to WordNet and has the synonym set {recommend, commend}. Aside from the above word sense, "recommend" also have another word sense: "push for something", as in "The travel agent recommends not to travel amid the pandemic". This second word sense has the synonym set {recommend, urge, advocate}[2]. Apparently, the only valid substitution is "commend", which preserves the semantics of the original movie review. While "urge" is the synonym of "recommend", it obviously does not fit in the context and should not be considered as a possible substitution. We call substituting x_i with a synonym that matches the word sense of x_i in $\mathbf{x}_{ori}$ a *matched sense substitution*, and we use *mismatched sense substitution* to refer to swapping words with the synonym which belongs to the synonym set of a different word sense.

3.1.1 Experiments

To illustrate that mismatched sense substitution is a problem existing in practical attack algorithms, we conduct the following analysis. We examine the adversarial samples generated by PWWS (Ren et al., 2019), which substitutes words using WordNet synonym set. We use a benchmark dataset (Yoo et al., 2022) that contains the adversarial samples generated by PWWS against a BERT-based classifier fine-tuned on AG-News (Zhang et al., 2015). AG-News is a news topic classification dataset, which aims to classify a piece of news into four categories: world, sports, business, and sci/tech news. The attack success rate on the testing set composed of 7.6K samples is 57.25%. More statistics about the datasets can be found in Appendix B. We categorize the words replaced by PWWS into three disjoint categories: *matched sense substitution*, *mismatched sense substitution*, and *morphological substitution*. The last category, morphological substitution, refers to substituting words with a word that only differs in inflectional morphemes[3] or derivational morphemes[4] with the original word. We specifically isolate *morphological substitution* since it is hard to categorize it into either matched or mismatched sense substitution.

The detailed procedure of categorizing a replaced word's substitution type is as follows: Given a pair of $(\mathbf{x}_{ori}, \mathbf{x}_{adv})$, we first use NLTK (Bird et al., 2009) to perform word sense disambiguation on each word x_i in $\mathbf{x}_{ori}$. We use LemmInflect and NLTK to generate the morphological substitution set $\mathbb{ML}_{x_i}$ of x_i. The matched sense substitution set $\mathbb{M}_{x_i}$ is constructed using the WordNet synonym set of the word sense of x_i in $\mathbf{x}_{ori}$; since this synonym set includes the original word x_i and may also include some words in the $\mathbb{ML}_{x_i}$, we remove x_i and words that are already included in the $\mathbb{ML}_{x_i}$ from the synonym set, forming the final matched sense substitution set, $\mathbb{M}_{x_i}$. The mismatched sense substitution set $\mathbb{MM}_{x_i}$ is constructed by first collecting all synonyms of x_i that belong to the different word sense(s) of x_i in $\mathbf{x}_{ori}$ using WordNet, and then removing all words that have been included in $\mathbb{ML}_{x_i}$ and $\mathbb{M}_{x_i}$.

After inspecting 4140 adversarial samples produced by PWWS, we find that among **26600** words that are swapped by PWWS, only **5398 (20.2%)** words fall in the category of matched sense substitution. A majority of **20055 (75.4%)** word substitutions are mismatched sense substitutions, which should be considered invalid substitutions since using mismatched sense substitution cannot preserve the semantics of $\mathbf{x}_{ori}$ and makes $\mathbf{x}_{adv}$ incomprehensible. Last, about **3.8%** of words are substituted with their morphological related words, such as converting the part of speech (POS) from verb to noun or changing the verb tense. These substitutions, while maintaining the semantics of the original sentence and perhaps human readable, are mostly ungrammatical and lead to unnatural adversarial samples. The aforementioned statistics illustrate that only about 20% word substitutions produced by PWWS are *real* synonym substitutions, and thus the high attack success rate of 57.25% should not be surprising since most word replacements are highly questionable.

3.2 Counter-fitted Embedding kNN and MLM Mask-Infilling/Reconstruction Contain Few Matched Sense Synonym

As shown in Section 3.1.1, even when using WordNet synonyms as the candidate sets, the proportion of the valid substitutions is unthinkably low. This makes us more concerned about the word substitution quality of the other three heuristic transformations introduced in Section 2.2. These three word substitution methods mostly rely on assumptions about the quality of the embedding space or the

[2] The word senses and synonyms are from WordNet.

[3] Inflectional morphemes are the suffixes that change the grammatical property of a word but do not create a new word, such as a verb's tense or a noun's number. For example, recommends→recommend.

[4] Derivational morphemes are affixes or suffixes that change the form of a word and create a new word, such as changing a verb into a noun form. For example, recommend→recommendation.

Transformations	Syn. (matched)	Syn. (mismatched)	Antonyms	Morphemes	Others
GloVe-kNN	0.22	1.01	0	1.55	27.22
BERT mask-infill	0.08	0.36	0.06	0.57	28.93
BERT reconstruction	0.14	0.58	0.09	1.19	27.99

Table 1: The average words of different substitution types in the candidate word set of k =30 words. Syn. is short for Synonym.

ability of the MLM and require setting a hyperparameter k for the size of the substitution set. To the best of our knowledge, no previous work has systematically studied what the candidate sets proposed by the three transformations are like; still, they have been widely used in SSAs.

3.2.1 Experiments

To understand what those substitution sets are like, we conduct the following experiment. We use the benchmark dataset generated by Yoo et al. (2022) that attacks 7.6k samples in the AG-News testing data using TextFooler. For each word x_i in $\mathbf{x}_{ori}$ that is perturbed into another $x_i^{'}$ in $\mathbf{x}_{adv}$, we use the following three transformations to obtain the candidate substitution set: counter-fitted GloVe embedding space, BERT mask-infilling, and BERT reconstruction. [5] We only consider the substitution set of x_i that are perturbed in $\mathbf{x}_{adv}$ because not all words in $\mathbf{x}_{ori}$ will be perturbed by an SSA, and it is thus more reasonable to consider only the words that are really perturbed by an SSA. We set the k in kNN of counter-fitted GloVe embedding space transformation and top-k prediction in BERT mask-infilling/reconstruction to 30, a reasonable number compared with many previous works.

We categorize the candidate words into five disjoint word substitution types. Aside from the three word substitution types discussed in Section 3.1.1, we include two other substitution types. The first one is *antonym substitution*, which is obtained by querying the antonyms of a word x_i using WordNet. Different from synonym substitutions, we do not separate antonyms into antonyms that matched the word sense of x_i in $\mathbf{x}_{ori}$ and the sense-mismatched antonyms, since neither of them should be considered a valid swap in SSAs. The other substitution type is *others*, which simply consists of the candidate words not falling in the category of synonyms, antonyms, or morphological substitutions.

In Table 1, we show how different substitution types comprise the 30 words in the candidate set for different transformations on average. It is easy to tell that only a slight proportion of the substitution set is made up of synonym substitution for all three transformation methods, with counter-fitted GloVe embedding substitution containing the most synonyms among the three methods, but still only a sprinkle of about 1 word on average. Moreover, synonym substitution is mostly composed of mismatched sense substitution. When using BERT mask-infilling as a transformation, there are only 0.08 matched sense substitutions in the top 30 predictions. While using BERT reconstruction for producing the candidate set, the matched sense substitution slightly increases, compared with mask-infilling, it still only accounts for less than 1 word in the top-30 reconstruction predictions of BERT. Within the substitution set, there is on average about 1 word which is the morphological substitution of the original word. Surprisingly, using MLM mask-infilling or reconstruction as transformation, there is a slight chance that the candidate set consists of antonyms of the original word. It is highly doubtful whether the semantics is preserved when swapping the original sentence with antonyms.

The vast majority of the substitution set composes of words that do not fall into the previous four categories. We provide examples of how the substitution sets proposed by different transformations are like in Table 6 in the Appendix, showing that the candidate words in the *others* substitution types are mostly unrelated words that should not be used for word replacement. It is understandable that words falling to the *other* substitution types are invalid candidates; this is because the core of SSAs is to replace words with their semantically close counterparts to preserve the semantics of the original sentence. If a substitution word does not belong to the synonym set proposed by WordNet, it is unlikely that swapping the original word with this word can preserve the semantics of $\mathbf{x}_{ori}$. We also show some randomly selected adversarial samples generated by different SSAs that use different transformations in Table 5 in the Appendix,

[5] For BERT mask-infilling and reconstruction substitution, we remove punctuation and incomplete subword tokens.

which also show that when a word substitution is not a synonym nor a morphological swap, there is a high chance that it is semantically invalid. Hauser et al. (2021) uses human evaluation to show that the adversarial samples generated from TextFooler, BERT-Attack, and BAE do not preserve the meaning of $\mathbf{x}_{ori}$, which also backs up our statement.

When decreasing the number of k, the number of invalid substitution words may possibly be reduced. However, a smaller k often leads to lower attack success rates, as shown in Li et al. (2020), so it is not very common to use a smaller k to ensure the validity of the words in the candidate sets. In practical attacks, whether these words in the candidate sets can be considered valid depends on the constraints. But can those constraints really filter invalid substitutions? We show in the next section that, sadly, the answer is no.

4 Problems with the Constraints in SSAs

In this section, we show that the constraints commonly used in SSAs cannot fully filter invalid word substitutions proposed by the transformations.

4.1 Word Embedding Similarity Cannot Distinguish Valid/Invalid Swaps Well

Setting a threshold on word embedding cosine similarity to filter invalid word substitutions relies on the hypothesis that valid word swaps indeed have higher cosine similarity with the word to be substituted, compared with invalid word replacements. We investigate whether the hypothesis holds with the following experiment. We reuse the 7.6K AG-News testing samples attacked by TextFooler used in Section 3.2, and we gather all pairs of $(\mathbf{x}_{ori}, \mathbf{x}_{adv})$. For each word x_i in $\mathbf{x}_{ori}$ that is perturbed in $\mathbf{x}_{adv}$, we follow the same procedure in Section 3.2 to obtain the morphological substitution set, matched sense substitution set, mismatched sense substitution set, and the antonym set. We then query the counter-fitted GloVe embedding space to obtain the word embeddings of all those words and calculate their cosine similarity with the word embedding of x_i. As a random baseline, we also randomly sample high-frequency words and low-frequency words in the training dataset of AG-News, and compute the cosine similarity between those words and x_i. How these high-frequency and low-frequency words are sampled is detailed in Appendix D.2.

To quantify how hard it is to use the word embedding cosine similarity to distinguish a valid substitution (the matched sense substitution) from another type of invalid substitution, we calculate the area under the precision-recall curve (AUPR) of the threshold-based detector that predicts whether a perturbed $x_i^{'}$ is a valid substitution based on its cosine similarity with x_i. Given an x_i and a perturbed $x_i^{'}$, a threshold-based detector measures the word embedding cosine similarity between x_i and $x_i^{'}$, and assigns it as positive (valid substitution) if the cosine similarity is higher than the threshold. A perfect detector should have an AUPR of 1.0, while a random detector will have an AUPR of 0.5. Note that the detector we discuss here will only be presented with two types of substitution, one is the matched sense substitution and the other is a substitution type other than the matched sense substitution.

Substitution Type	AUPR
Synonyms (mismatched)	0.627
Antonym	0.980
Morpheme	0.433
Random high-freq	0.900
Random low-freq	0.919

Table 2: The AUPR when using a threshold-based detector to separate matched sense synonyms from another type of invalid substitution.

We show the AUPR in Table 2. First, we notice that when using the word embedding cosine similarity to distinguish matched sense substitutions from mismatched ones, the AUPR is as low as 0.627. While this is better than random, this is far from a useful detector, showing that word embedding cosine similarity constraints are not useful to remove invalid substitutions like unmatched sense words. The AUPR for morpheme substitutions is even lower than 0.5, implying that the word embedding cosine similarity between x_i and its morphological similar words is higher than the similarity score between matched sense synonyms. This means that when we set a higher cosine similarity threshold, we are keeping more morphological swaps instead of valid matched sense substitutions. While morphological substitutions have meanings similar to or related to the original word, as we previously argued, they are mostly ungrammatical.

The AUPR when using a threshold-based detector to separate matched sense substitutions from antonym substitutions is almost perfect, which is

0.980. This should not be surprising since the counter-fitted word embedding is designed to make synonyms and antonyms have dissimilar word embeddings. Last, the AUPR of separating random substitutions from matched sense substitutions is also high, meaning that it is possible to use a detector to remove random and unrelated substitutions based on word embedding cosine similarity. Based on the result in Table 2, setting a threshold on word-embedding cosine similarity may only filter out the antonyms and random substitutions but still fails to remove the other types of invalid substitutions.

4.2 Sentence Encoder Is Insensitive to Invalid Word Substitutions

To test if sentence encoders really can filter invalid word substitutions in SSA, we conduct the following experiment. We use the same attacked AG-News samples that were used in Section 3.2.1. For each pair of $(\mathbf{x}_{ori}, \mathbf{x}_{adv})$ in that dataset, we first collect the swapped indices set $\mathbb{I} = \{i | x_i \neq x_i^{'}\}$ that represents the positions of the swapped words in $\mathbf{x}_{adv}$. We shuffle the elements in $\mathbb{I}$ to form an ordered list $\mathbb{O}$. Using $\mathbf{x}_{ori}$ and $\mathbb{O}$, we construct a sentence $\mathbf{x}_{swap}^{n}$ by swapping n words in $\mathbf{x}_{ori}$. The n positions where the substitutions are made in $\mathbf{x}_{swap}^{n}$ are the first n elements in the ordered list $\mathbb{O}$; at each substitution position, the word is replaced by a word randomly selected from a type of candidate word set. All the n replaced words in $\mathbf{x}_{swap}^{n}$ are the same type of word substitution. We conduct experiments with six types of candidate word substitution sets: matched sense, mismatched sense, morphological, antonym, random high-frequency, and random low-frequency word substitutions. After obtaining $\mathbf{x}_{swap}^{n}$, we compute the cosine similarity between the sentence embedding between $\mathbf{x}_{swap}^{n}$ with $\mathbf{x}_{ori}$ using USE and set the window size w to 7, following Jin et al. (2020) and Garg and Ramakrishnan (2020). We vary the number of replaced words from 1 to 10.[6] This experiment helps us know how the cosine similarity changes when the words are swapped using different types of candidate word sets. More details on this experiment are in Appendix D.3 and Figure 2 in the Appendix.

The results are shown in Figure 1. While replacing more words in $\mathbf{x}_{ori}$ does decrease its cosine similarity with $\mathbf{x}_{ori}$, the cosine similarity when substituting random high-frequency words is still roughly higher than 0.80. Considering that practical SSAs often set the cosine similarity threshold to around 0.85 or even lower[7], depending on the SSAs and datasets, it is suspicious whether the constraint and threshold can really filter invalid word substitution. We can also observe that when substituting words with antonyms, the sentence embedding cosine similarity with the original sentence closely follows the trend of substituting words using a synonym, regardless of whether the synonym substitution matches the word sense or not. Recalling that we have revealed that the candidate set proposed by BERT can contain antonyms in Table 1, the results here indicates that sentence embedding similarity constraint cannot filter this type of faulty word substitution. For the two different types of synonym substitutions, only matched sense substitutions are valid replacement that follows the semantics of the original sentence. However, the sentence embedding of $\mathbf{x}_{ori}$ and the sentence embedding of the two types of different synonym substitutions are equally similar. The highest cosine similarity is obtained when the words in $\mathbf{x}_{ori}$ are swapped using their morphological substitutions, and this is expected since morphological substitutions merely change the semantics.

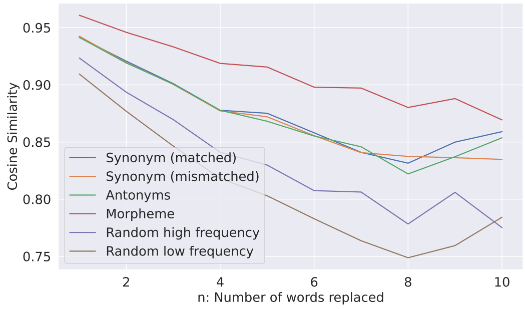


Figure 1: The USE sentence embedding cosine similarity between $\mathbf{x}_{ori}$ and the series of sentences $\mathbf{x}_{swap}^{n}$ obtained by replacing words in $\mathbf{x}_{ori}$ with one type of word substitution.

In Figure 1, we only show the average cosine similarity and do not show the variance of the cosine similarity of each substitution type. In Figure 3 in the Appendix, we show the distribution of the cosine similarity of different substitution types. The main observation from Figure 3 is that the cosine similarity distributions of different substitution types (for the same n) are highly overlapped, and it is impossible to distinguish valid word swaps from

[6] Attacking AG-News using TextFooler perturbs about 9 out of 38.6 words in a benign sample on average.

[7] We include the sentence embedding cosine similarity threshold of prior works in Table 4 in Appendix C.

the invalid ones simply by using a threshold on the sentence embedding cosine similarity.

Overall, the results in Figure 1 demonstrate that USE tends to generate similar sentence embeddings when two sentences only differ in a few tokens, no matter whether the replacements change the sentence meaning or not. While we only show the result of USE, we show in Appendix E that different sentence encoders have similar behavior. Moreover, when we use the whole sentence instead of a windowed subsentence to calculate the sentence embedding, the cosine similarity is even higher than that shown in Figure 1, as shown in Appendix E. Again, these sentence encoders fail to separate invalid word substitutions from valid ones. While frustrating, this result should not be surprising, since most sentence encoders are not trained to distinguish sentences with high word overlapping.

4.3 LanguageTool Cannot Detect False Verb Inflectional Form

LanguageTool is used in TextFooler-Adj (TF-Adj) (Morris et al., 2020a) to prevent the attack to induce grammar errors. TF-Adj also uses stricter word embedding and sentence embedding cosine similarity constraints to ensure the semantics in $\mathbf{x}_{ori}$ are preserved in $\mathbf{x}_{adv}$. However, when browsing through the adversarial samples generated by TF-Adj, we observe that the word substitutions made by TF-Adj are often ungrammatical morphological swaps that convert a verb's inflectional form. This indicates that LanguageTool may not be capable of detecting a verb's inflectional form error.

To verify this hypothesis, we conduct the following experiment. For each sample in the test set of AG-News that LanguageTool reports no grammatical errors, we convert the inflectional form of the verbs in the sample by a hand-craft rule that will always make a grammatical sentence ungrammatical; this rule is listed in Appendix D.4. We then use LanguageTool to detect how many grammar errors are there in the verb-converted sentences.

We summarize the experiment results as follows. For the 1039 grammatical sentences in AG-News, the previous procedure perturbed **4.37** verbs on average. However, the average number of grammar errors identified by LanguageTool is **0.97**, meaning that LanguageTool cannot detect all incorrect verb forms. By this simple experiment and the results from Table 2 and Figure 1, we can understand why the attack results of TF-Adj are often ungrammatical morphological substitutions: higher cosine similarity constraints prefer morphological substitutions, but those often ungrammatical substitutions cannot be detected by LanguageTool. Thus, aside from showing that the text classifier trained on AG-News is susceptible to inflectional perturbations, TF-Adj actually exposes that LanguageTool itself is vulnerable to inflectional perturbations.

5 Related Works

Some prior works also discuss a similar question that we study in this paper. Morris et al. (2020a) uses human evaluation to reveal that SSAs sometimes produce low-quality adversarial samples. They attribute this to the insufficiency of the constraints and use stricter constraints and LanguageTool to generate better adversarial samples. Our work further points out that the problem is not only in the constraints; we show that the transformations are the fundamental problems in SSAs. We further show that LanguageTool used by Morris et al. (2020a) cannot detect ungrammatical verb inflectional forms, and reveal that the adversarial samples generated by TF-Adj exploit the weakness of LanguageTool and are often made up of ungrammatical morphological substitutions. Hauser et al. (2021) uses human evaluations and probabilistic statements to show that SSAs are low quality and do not preserve original semantics. Our work can be seen as an attempt to understand the cause of the observations in Hauser et al. (2021).

Morris (2020) also questions the validity of using sentence encoders as semantic constraints. They attack sentence encoders by swapping words in a sentence with their antonyms and the attack goal is to maximally preserve the swapped sentence's sentence embedding cosine similarity with the original sentence. This is related to our experiments in Section 4.2. The main differences between our experiments and theirs are: (1) When swapping words, we only swap the words that are really swapped by TextFooler; on the contrary, the words swapped in Morris (2020) are not necessarily words that are actually substituted in an SSA. The words swapped when attacking a sentence encoder and attacking a text classifier can be significantly different. Since our goal is to verify how sentence encoders behave when used *in SSAs*, it makes more sense to only swap the words that are really replaced by an SSA. (2) Morris (2020) only uses antonyms for word substitution.

6 Discussion and Conclusion

This paper discusses how the elements in SSAs lead to invalid adversarial samples. We highlight that the candidate word sets generated by all four different word substitution methods contain only a small fraction of semantically matched and grammatically correct word replacements. While these transformations produce inappropriate candidate words, this alone will not contribute to the invalid adversarial samples. The inferiority of those adversarial samples should be largely attributed to the deficiency of the constraints that ought to guarantee the quality of the perturbed sentences: word embedding cosine similarity is not always larger for valid word substitutions, sentence encoder is insensitive to invalid word swaps, and LanguageTool fails to detect grammar mistakes. These altogether bring about the adversarial samples that are human distinguishable, unreasonable, and mostly inexplicable. These adversarial samples are not suitable for evaluating the vulnerability of NLP models because they are not reasonable inputs.

The results and observations shown in the main content of our paper are not unique for BERT fine-tuned on AG-News, which is the only attacked model shown in Section 3 and Section 4. We include supplementary analyses in Appendix F for different model types and datasets, which supports all the claims and observations in the main contents. In this paper, we follow previous papers on SSAs to only show the result of attacking the victim model once and not reporting the performance variance due to random seed and hyperparameters used during the fine-tuning of victim model (Ren et al., 2019; Li et al., 2020; Jin et al., 2020). This is because conducting SSA is very time-consuming. In our preliminary experiments, we used TextAttack to attack three BERT models fine-tuned on AG-News and we crafted the adversarial samples for 100 samples in the testing data for each model The three models were fine-tuned with three different sets of hyperparameters. We find that our observation in Section 3.2 and Section 4 do not change for the three victim models. Overall, the observation shown in the paper is not an exception but rather a general phenomenon in SSAs.

By the analyses in the paper, we show that we may still be far away from *real* SSAs, and how to construct valid synonym substitution adversarial samples remains an unresolved problem in NLP. While there is still a long way to go, it is essential to recognize that the prior works have contributed significantly to constructing valid SSAs. Although prior SSAs may not always produce reasonable adversarial samples, they are still valuable since they pave the way for designing better SSAs and help us uncover the inadequacy of the transformations and constraints for constructing *real* synonym substitution adversarial samples. As an initiative to stimulate future research, we provide some possible directions and guidelines for constructing better SSAs, based on the observation in our paper.

1. Simply consider the word senses when making a replacement with WordNet.

2. Use better sentence encoders that are sensitive to token replacements that change the semantics of the original sentence. For example, DiffCSE (Chuang et al., 2022) is shown to be able to distinguish the tiny differences between sentences.

3. When designing transformations, one should always verify the validity of the proposed method through well-controlled experiments. These experiments include recruiting human evaluators to check the quality of the transformations or using experiments as in Section 3 to check what the candidate sets proposed by the transformations are like. It is perilous to solely rely on heuristics or black-box models such as sentence encoders to guarantee the quality of the transformation.

4. Since the sentences crafted by SSAs may largely deviate from normal sentences, one should test if constraint models, e.g., grammar checkers or sentence encoders, work as expected when faced with those abnormal sentences. For example, one can perform stress tests (Ribeiro et al., 2020) to test the behavior of the constraint models. This prevents us from exploiting the vulnerability of the constraints when attacking the text classifier.

The problems outlined in this paper may be familiar to those with experience in lexical substitution (Melamud et al., 2015; Zhou et al., 2019), but they have not yet been widely recognized in the field of SSAs. Our findings on why SSAs fail can serve as a reality check for the field, which has been hindered by overestimating prior SSAs. We hope our work will guide future researchers in cautiously building more effective SSAs.

Limitations

In this paper, we only discuss the SSAs in English, as this has been the most predominantly studied in adversarial attacks in NLP. The authors are not sure whether SSAs in a different language will suffer from the shortcomings discussed in this paper. However, if an SSA in a non-English language uses the transformations or constraints discussed in this paper, there is a high chance that this attack will produce low-quality results for the same reason shown in this paper. Still, the above claim needs to be verified by extensive human evaluation and further language-specific analyses.

In our paper, we use WordNet as the gold standard of the word senses since WordNet is a widely adopted and accepted tool in the NLP community. Chances are that some annotations in WordNet, while very scarce, are not perfect, and this may be a possible limitation of our work. It is also possible that the matched sense synonyms found by WordNet may not always be a valid substitution even if the annotation of WordNet is perfect. For example, the collocating words of the substituted word may not match that of the original word, and the substitution word may not fit in the original context. However, if a word is not even a synonym, it is more unlikely that it is a valid substitution. Thus, being a synonym in WordNet is a minimum requirement and we use WordNet synonym sets to evaluate the validity of a word substitution.

Last, we do not conduct human evaluations on what the *other substitution types* in Table 1 are. As stated in Section 3.2.1, while we do not perform human evaluations on this, the readers can browse through Table 6 in the Appendix to see what the *others* substitutions are. It will be interesting to see what human evaluators think about the *other* substitutions in the future.

Ethics Statement and Broader Impacts

The goal of our paper is to highlight the overlooked details in SSAs that cause their failures. By mitigating the problems pointed out in our paper, there are two possible consequences:

1. One may find that there exist no *real* synonym substitution adversarial samples, and the NLP models currently used are robust. This will cause no ethical concerns since this indicates that no harm will be caused by our work. Previous observations on the vulnerability are just the product of low-quality adversarial samples.
2. There exists *real* synonym substitution adversarial samples, and excluding the issues mentioned in this paper will help malicious users easier to find those adversarial samples. This will become a potential risk in the future. The best way to mitigate the above issue is to construct new defenses for *real* SSAs.

While our goal is to raise attention to whether SSAs are really SSAs, we are not advocating malicious users to attack text classifiers using better SSAs. Instead, we would like to highlight that there is still an unknown risk, the *real* SSAs, against text classifiers, and we researchers should devote more to studying this topic and developing defenses against such attacks before they are adopted by adversarial users.

Another major ethical consideration in our paper is that we challenge prior works on the quality of the SSAs. While we reveal the shortcomings of previously proposed methods, we still highly acknowledge their contributions. As emphasized in Section 6, we do not and try not to devalue those works in the past. We scientifically and objectively discuss the possible risks of those transformations and constraints, and our ultimate goal is to push the research in adversarial attacks in NLP a step forward; from this perspective, we believe that we are in common with prior works.

Acknowledgements

We thank the reviewers for their valuable feedback and actionable suggestions. We've made major revisions based on the reviews and we list the main modification in Appendix A. Cheng-Han Chiang is supported by a Ph.D. scholarship program by Delta Electronics.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium. Association for Computational Linguistics.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*.

Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljačić, Shang-Wen Li, Wen-tau Yih, Yoon Kim, and James Glass. 2022. Diffcse: Difference-based contrastive learning for sentence embeddings. *arXiv preprint arXiv:2204.10298*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE.

Siddhant Garg and Goutham Ramakrishnan. 2020. BAE: BERT-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6174–6181, Online. Association for Computational Linguistics.

Jens Hauser, Zhao Meng, Damián Pascual, and Roger Wattenhofer. 2021. Bert is robust! a case against synonym-based adversarial examples in text classification. *arXiv preprint arXiv:2109.07403*.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8018–8025.

language-tool python. 2022. language_tool_python: a grammar checker for python.

Dianqi Li, Yizhe Zhang, Hao Peng, Liqun Chen, Chris Brockett, Ming-Ting Sun, and Bill Dolan. 2021. Contextualized perturbation for textual adversarial attack. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5053–5069, Online. Association for Computational Linguistics.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. Bert-attack: Adversarial attack against bert using bert. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Rishabh Maheshwary, Saket Maheshwary, and Vikram Pudi. 2021. A strong baseline for query efficient attacks in a black box setting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8396–8409.

Oren Melamud, Omer Levy, and Ido Dagan. 2015. A simple word embedding model for lexical substitution. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 1–7, Denver, Colorado. Association for Computational Linguistics.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

John Morris. 2020. Second-order nlp adversarial examples. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 228–237.

John Morris, Eli Lifland, Jack Lanchantin, Yangfeng Ji, and Yanjun Qi. 2020a. Reevaluating adversarial examples in natural language. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3829–3839, Online. Association for Computational Linguistics.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020b. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gasic, Lina M Rojas Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–148.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1085–1097, Florence, Italy. Association for Computational Linguistics.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Samson Tan, Shafiq Joty, Min-Yen Kan, and Richard Socher. 2020. It's morphin'time! combating linguistic discrimination with inflectional perturbations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2920–2935.

Princeton University. 2010. About wordnet. *WordNet*.

Jingjing Xu, Liang Zhao, Hanqi Yan, Qi Zeng, Yun Liang, and Xu Sun. 2019. LexicalAT: Lexical-based adversarial reinforcement training for robust sentiment classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5518–5527, Hong Kong, China. Association for Computational Linguistics.

Jin Yong Yoo and Yanjun Qi. 2021. Towards improving adversarial training of nlp models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 945–956.

KiYoon Yoo, Jangho Kim, Jiho Jang, and Nojun Kwak. 2022. Detection of adversarial examples in text classification: Benchmark and baseline via robust density estimation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3656–3672, Dublin, Ireland. Association for Computational Linguistics.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NIPS*.

Wangchunshu Zhou, Tao Ge, Ke Xu, Furu Wei, and Ming Zhou. 2019. BERT-based lexical substitution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3368–3373, Florence, Italy. Association for Computational Linguistics.

A Different from the Pre-review Version

We list the main difference between this version and the pre-review version of our paper (the pre-review version is similar to the previous arXiv version). Most modifications are made based on the reviewers' suggestions. We thank the reviewers for their valuable feedback that help us polish and strengthen this paper.

- We change how we present our result in Section 3.2 from a bar chart to a table for easier interpretation.
- We largely reformulate Section 4.1. We change how we present the experiment results: in the previous version, we only qualitatively plot the distribution of the word embedding cosine similarity of different substitution types. In this version, we adopt the reviewers' suggestion to quantitatively show that some types of invalid substitutions cannot be easily detected by the word embedding cosine similarity. We also correct the result of antonym substitutions.
- We add Section 5 to discuss relevant works.
- We discuss the performance variance due to different fine-tuning hyperparameters and random seeds in Section 6.
- We add the links to the victim text classifiers in Appendix B.
- We remove the FAQ section in the Appendix, which is mainly used for rebuttal.
- In this revision, we incorporate some of the answers to the reviewers' questions in the rebuttal.

B Dataset

In our paper, we use benchmark adversarial datasets generated by Yoo et al. (2022). Yoo et al. (2022) generates adversarial samples using the TextAttack (Morris et al., 2020b) module. Yoo and Qi (2021) release the dataset with a view to facilitating the detection of adversarial samples in NLP and reducing the redundant computation resources to re-generate adversarial samples. They thus generate adversarial samples using PWWS (Ren et al., 2019), TextAttack (Jin et al., 2020), BAE (Garg and Ramakrishnan, 2020) and TextFooler-Adj (Morris et al., 2020a) on LSTM, CNN, BERT, and RoBERTa trained/fine-tuned on SST-2 (Socher et al., 2013), IMDB (Maas et al., 2011), and AG-News (Zhang et al., 2015).

In the main content of our paper, we only use two datasets: the adversarial samples obtained using PWWS to attack BERT fine-tuned on AG-News, and the adversarial samples obtained by attacking TextFooler on BERT fine-tuned on AG-News. The testing set of AG-News contains 7.6K samples; the adversarial samples obtained by attacking these datasets will be less than 7.6K since the attack success rates of the two SSAs are not 100%. We summarize the detail of these two datasets in Table 3.

The models they used as victim model to generate classifiers are the fine-tuned by the TextAttack (Morris et al., 2020b) toolkit and are publicly available at `https://textattack.readthedocs.io/en/latest/3recipes/models.html` and Huggingface models. For example, the BERT fine-tuned on AG-News is at `https://huggingface.co/textattack/bert-base-uncased-ag-news`. The hyperparameters used to fine-tune those models can be found from the model cards and `config.json` and we do not list them here to save the space.

C Synonym Substitution Attacks

We list the transformations and constraints of the SSAs that are discussed or mentioned in our paper in Table 4. We only include the semantic and grammaticality constraints in Table 4 and omit other constraints such as the word-level overlap constraints. The "window" in the sentence encoder cosine similarity constraint indicates whether use a window around the current substitution word or use the whole sentence. The "compare with $\mathbf{x}_{ori}$" indicates that $\mathbf{x}^n_{swap}$ will be compared against the sentence embedding of $\mathbf{x}_{ori}$, and "compared with $\mathbf{x}^{n-1}_{swap}$" means that $\mathbf{x}^n_{swap}$ will be compared against the sentence embedding of $\mathbf{x}^{n-1}_{swap}$, that is, the sentence before the current substitution step.

C.1 Random Adversarial Samples

To illustrate that the adversarial samples generated by SSAs are largely made up of invalid word replacements, we randomly sample two adversarial samples generated by PWWS (Ren et al., 2019), TextFooler (Jin et al., 2020), BAE (Garg and Ramakrishnan, 2020), and TextFooler-Adj (Morris et al., 2020a). To avoid the suspicion of cherry-picking the adversarial samples to support our claims, we simply select the first and the last successfully attacked samples in AG-News using the four SSAs in the dataset generated by Yoo et al. (2022). Since the dataset is not generated by us, we cannot control which sample is the first one and which sample is the last one in the dataset, meaning that we will not be able to cherry-pick the adversarial samples that support our claims.

The adversarial samples are listed in Table 5. The blue words in $\mathbf{x}_{ori}$ are the words that will be perturbed in $\mathbf{x}_{adv}$. The red words are the swapped words. The readers can verify the claims in our paper using those adversarial samples. We recap some of our claims as follows:

- PWWS uses mismatched sense substitution: This can be observed in all the word substitutions of PWWS in Table 5. For example, the word "world" in the second example of PWWS have the word sense "the 3rd planet from the sun; the planet we live on". But it is swapped with the word "cosmos", which is the synonym of the word sense "everything that exists anywhere".

- Counter-fitted embedding substitution set contains a large proportion of *others* substitution types, which are mostly invalid: This can be observed in literally all word substitutions in TextFooler.

- BERT reconstruction substitution set contains a large proportion of *others* substitution types, which are mostly invalid: This can be observed in literally all word substitutions in BAE.

- Morphological substitutions are mostly ungrammatical: This can be observed in the first adversarial sample of TextFooler-Adj.

	PWWS	TextFooler
Success attacks	4140	5885
Attack success rate	57.25%	81.39%
Average words per sample	38.57	38.57
Average perturbed words percentage	17.63%	23.38%

Table 3: Details of the adversarial sample datasets obtained by attacking a BERT fine-tuned on AG-News using PWWS and TextFooler.

- TextFooler-Adj prefers morphological swap due to its strict constraints: This can be observe in almost all substitutions in TextFooler-Adj, excluding goods→wares.

C.1.1 Example of the Word Substitution Sets of Different Transformations

In this section, we show the substitution sets using different transformations. We only show one example here, and this example is the second successful attack example in the adversarial sample datasets (Yoo et al., 2022) that attacks a BERT fine-tuned classifier trained on AG-News using TextFooler. We do not use the first sample in Table 5 because we would like to show the readers a different adversarial sample in the datasets.

$\mathbf{x}_{ori}$: The Race is On: Second Private Team Sets Launch Date for Human Spaceflight (SPACE.com) SPACE.com - TORONTO, Canada – A second team of rocketeers competing for the #36;10 million Ansari X Prize, a contest for privately funded suborbital space flight, has officially announced the first launch date for its manned rocket.

$\mathbf{x}_{adv}$: The Race is Around: Second Privy Remit Set Lanza Timeline for Hummanitarian Spaceflight (SEPARATION.com) SEPARATION.com - CANADIENS, Countries – para second squad of rocketeers suitors for the #36;10 billion Ansari X Nobel, a contestant for convertly championed suborbital spaceship plane, had solemnly proclaim the first began timeline for its desolate bomb.

We show the substitution set for the first four words that are substituted by TextFooler in Table 6. We do not show that substitution set for all the attacked words simply because it will occupy too much space, and our claim in the main content that "*others* substitution sets of counter-fitted embedding substitution and BERT mask-infilling/reconstruction mostly consist of invalid swaps" can already be observed in Table 6.

D Implementation Details

D.1 Experiment Details of Section 3

In this section, we give details on how we obtain different word substitution types for a $\mathbf{x}_{ori}$. The whole process is summarized in Algorithm 1. In Algorithm 1, the reader can also find how the perturbed indices list $\mathbb{I}$ used in Section 4.2 is obtained.

An important detail that is not mentioned in the main content is that when computing how many synonyms are in the substitution set of BERT MLM substitution, we actually perform lemmatization on the top-30 predictions of BERT. This is because, for example, if BERT proposes to use the word "defines" to replace the original word "sets" (the third person present tense of the verb "set") in the original sentence, and the word "define" happens to a synonym according to WordNet; in this case, the word "defines" will not be considered as a synonym substitution. But "defines" should be considered as a synonym substitution since it is the third person present tense of "define". Lemmatizing the prediction of BERT can partially solve the problem. However, if the lemmatized word is already in the top-30 prediction of BERT, we do not perform lemmatization. This process is detailed on Line 6 on Algorithm 2. This can ensure that words can be considered as synonyms while words that should be considered as morphological swaps are mostly not affected.

D.2 Experiment Details of Section 4.1

Here, we explain how the random high/low-frequency words are sampled in Section 4.1. First, we use the tokenizer of BERT-base-uncased to tokenize all the samples in the training dataset of AG-News. Next, we count the occurrence of each token in the vocabulary of the BERT-base-uncased, and sort the tokens based on their occurrence in the training set in descending order. The vocabulary size of BERT-base-uncased is 30522, including five special tokens, some subword tokens, and some unused tokens. We define the high-frequency

Attack	Transformation	Constraints
Genenetic Algorithm Attack (Alzantot et al., 2018)	Counter-fitted GloVe embedding kNN substitution with $k = 8$	Word embedding mean square error distance with threshold 0.5; language model perplexity (as a grammaticality constraint)
PWWS (Ren et al., 2019)	WordNet synonym set substitution	None
TextFooler (Jin et al., 2020)	Counter-fitted GloVe embedding kNN substitution with $k = 50$	USE sentence embedding cosine similarity with threshold 0.878, window size $w = 7$, compare with $\mathbf{x}_{ori}$; word embedding cosine similarity with threshold 0.5; disallow swapping words with different POS but allow swapping verbs with nouns or the reverse
BERT-Attack (Li et al., 2020)	BERT mask-infilling substitution with $k =$ 48	Sentence embedding cosine similarity with different thresholds for different dataset, and the highest threshold is 0.7, no window, compare with $\mathbf{x}_{ori}$
BAE (Garg and Ramakrishnan, 2020)	BERT reconstruction substitution	USE sentence embedding cosine similarity with threshold 0.936, window size $w = 7$, compare with $\mathbf{x}_{swap}^{n-1}$
TextFooler-Adj (Morris et al., 2020a)	Counter-fitted GloVe embedding kNN substitution with $k = 50$	USE sentence embedding cosine similarity with threshold 0.98, window size $w = 7$, compare with $\mathbf{x}_{ori}$; word embedding cosine similarity with threshold 0.9; disallow swapping words with different POS but allow swapping verbs with nouns or the reverse; adversarial sample should not introduce new grammar errors, checked by LanguageTool
A2T (Yoo and Qi, 2021)	Counter-fitted GloVe embedding kNN substitution with $k = 20$ or BERT reconstruction with $k = 20$	Word embedding cosine similarity with threshold 0.8; DistilBERT fine-tuned on STS-B sentence embedding cosine similarity with threshold 0.9, window size $w = 7$, compare with $\mathbf{x}_{ori}$; disallow swapping words with different POS
CLARE (Li et al., 2021)	DistilRoBERTa mask-infilling substitution, instead of using top-k, they select the predictions whose probability is larger than 5×10^{-3}; this set contains 42 tokens on average	USE sentence embedding cosine similarity with threshold 0.7, window size $w = 7$, compare with $\mathbf{x}_{ori}$

Table 4: Detailed transformations and constraints of different SSAs mentioned in our paper.

Attack	$\mathbf{x}_{ori}$	$\mathbf{x}_{adv}$
PWWS	Ky. Company Wins Grant to Study Peptides (AP) AP - A company founded by a chemistry researcher at the University of Louisville won a grant to develop a method of producing better peptides, which are short chains of amino acids, the building blocks of proteins.	Ky. Company profits yield to bailiwick Peptides (AP) AP - amp company founded by a chemistry researcher at the University of Louisville won a grant to develop a method of producing better peptides, which are short chains of amino acids, the building blocks of proteins.
PWWS	Around the world Ukrainian presidential candidate Viktor Yushchenko was poisoned with the most harmful known dioxin, which is contained in Agent Orange, a scientist who analyzed his blood said Friday.	Around the cosmos Ukrainian presidential candidate Viktor Yushchenko was poisoned with the most harmful known dioxin, which is contained in Agent Orange, a scientist who analyzed his lineage said Friday.
Text-Fooler	Fears for T N pension after talks Unions representing workers at Turner Newall say they are 'disappointed' after talks with stricken parent firm Federal Mogul.	Fears for T percent pension after debate Syndicates portrayal worker at Turner Newall say they are 'disappointed' after chatter with bereaved parenting corporations Canada Mogul.
Text-Fooler	5 of arthritis patients in Singapore take Bextra or Celebrex &lt; b&gt;...&lt;/b&gt; SINGAPORE : Doctors in the United States have warned that painkillers Bextra and Celebrex may be linked to major cardiovascular problems and should not be prescribed.	5 of bursitis patients in Malaysia taken Bextra or Celebrex &lt;seconds&gt;...&lieutenants;/iii&gt; SINGAPORE : Medecine in the United Nations get reminding that sedatives Bextra and Celebrex may pose link to enormous cardiovascular woes and planned not be planned.
BAE	Fears for T N pension after talks Unions representing workers at Turner Newall say they are 'disappointed' after talks with stricken parent firm Federal Mogul.	Fears for T pl pension after talks Unions representing workers at Turner network say they are 'disappointed' after talks with stricken parent firm Federal Mogul.
BAE	5 of arthritis patients in Singapore take Bextra or Celebrex &lt;b&gt;...&lt;/b&gt; SINGAPORE : Doctors in the United States have warned that painkillers Bextra and Celebrex may be linked to major cardiovascular problems and should not be prescribed.	5 of arthritis patients in Singapore take cd or i &m;x&gt;...&lt;/b&gt; SINGAPORE : doctors in the United state have warned that painkillers used and Celebrex may be linked to major cardiovascular harm and should not be prescribed.
Text-Fooler -Adj	Venezuela Prepares for Chavez Recall Vote Supporters and rivals warn of possible fraud; government says Chavez's defeat could produce turmoil in world oil market.	Venezuela Prepares for Chavez Recall Voted Supporters and rivals warn of possible fraud; government says Chavez's defeat could produce turmoil in world oil marketed.
Text-Fooler -Adj	EU to Lift U.S. Sanctions Jan. 1 BRUSSELS (Reuters) - The European Commission is sticking with its plan to lift sanctions on $4 billion worth of U.S. goods on Jan. 1 following Washington's repeal of export tax subsidies in October, a spokeswoman said on Thursday.	EU to Lift U.S. Sanctions Jan. 1 BRUSSELS (Reuters) - The European Commission is sticking with its plan to lift sanctions on $4 billion worth of U.S. wares on Jan. 1 following Washington's repeal of export taxation subsidies in October, a spokeswoman said on Thursday.

Table 5: Adversarial samples from the benchmark dataset generated by Yoo and Qi (2021).

x_i	Counter-fitter GloVe embedding	BERT MLM	BERT reconstruction
On	Orn, Pertaining, Per, Toward, Dated, Towards, Circa, Dates, Relating, Pour, Relative, Sur, Into, Date, Concerning, Onto, Around, About, In, To, Sobre, Relate, During, Respecting, For, Regarding, At, Days, Throughout, Relation	following, completed, ongoing, over, in, included, contested, followed, this, now, below, announced, after, split, for, therefore, concluded, titled, currently, follows, planned, listed, thus, held, on, to, that, scheduled, called, where	around, round, a, here, ongoing, over, in, the, involved, pending, at, next, now, under, for, ahead, set, off, currently, onto, given, considered, about, held, on, of, to, by, time, with
Private	Confidentiality, Camera, Personal, Clandestine, Privately, Hoc, Undercover, Confidential, Secretive, Secrets, Dedicated, Secret, Surreptitiously, Confidentially, Belonged, Peculiar, Personally, Specially, Fenced, Owned, Covert, Particular, Especial, Covertly, Own, Deprived, Secretly, Privy, Soldier, Special	google, my, o, a, from, hs, the, 1, chapter, 1st, in, this, mv, md, ukrainian, le, facebook, baltimore, hr, of, th, to, that, donald, and, by, gma, where, with	personal, vr, 2012, my, a, from, own, official, local, the, vc, small, for, national, billionaire, social, private, 2014, 2010, pv, facebook, public, independent, of, privately, to, new, family, and, by
Team	Panels, Grouping, Machine, Equipments, Tasks, Task, Devices, Pc, Group, Appliance, Cluster, Computers, Groups, Teams, Tooling, Accoutrements, Remit, Pcs, Appliances, Grupo, Teamwork, Chore, Apparatus, Squad, Computer, Device, Machines, Panel, Squads, Equipment	fund, label, launch, google, team, sponsor, investor, project, citizen, investigator, sector, plane, foundation, company, helicopter, website, line, platform, rocket, and, group, blog, planet, computer, charity, to, jet, pilot, party, fan	firm, one, weekend, partnership, round, team, committee, teams, number, couple, country, site, button, company, line, side, crew, ballot, group, nation, winner, division, club, boat, of, to, family, party, time
Sets	Defines, Stake, Matches, Provides, Prescribes, Determine, Set, Betting, Establishes, Stipulates, Jeu, Gambling, Staking, Stipulated, Toys, Determines, Defined, Game, Defining, Playing, Gaming, Games, Determining, Define, Jeux, Gamble, Identifies, Stipulate, Plays, Play	google, a, from, estimated, first, larsen, the, 1, 1st, 3, at, next, announced, top, named, def, or, possible, predicted, 3rd, facebook, 000, online, about, on, of, to, and, no, with	reaches, established, announce, places, records, official, announcing, begins, forms, indicates, announced, declares, sets, starts, estimates, determines, set, details, draws, lays, lists, specifies, calls, setting, stages, of, gives, establishes, announces, names

Table 6: Candidate substitutions proposed by different transformations. We use green to denote matched sense substitution, orange to denote mismatched sense substitution, brown to denote morpheme substitution, and purple to denote antonyms. The *other* type substitution is denoted using the default black.

Algorithm 1 Process of obtaining the substitution set

```
Require: $\mathbf{x}_{ori}, \mathbf{x}_{adv}$
$\mathbb{I} \leftarrow []$    ▷ Initialize the perturbed indices list
for $x_i \in \mathbf{x}_{ori}$ do
    if $x_i = x_i^{'}$ then
        continue
    end if
    $x_i \leftarrow x_i$.lower()    ▷ Get the lower case of $x_i$
    $x_i^{'} \leftarrow x_i^{'}$.lower()    ▷ Get the lower case of $x_i^{'}$
    $\mathbb{S}_{ml} \leftarrow$ GetMorph$(x_i, \mathbf{x}_{ori})$    ▷ Get morphological substitutions
    $\mathbb{S}_{ms} \leftarrow$ GetMatchedSense$(x_i, \mathbf{x}_{ori})$    ▷ Get matched sense synonym by first using word sense disambiguation then WordNet synonym sets
    $\mathbb{S}_{mms} \leftarrow$ GetMismatchedSense$(x_i, \mathbf{x}_{ori})$    ▷ Get mismatched sense synonym by first using word sense disambiguation then WordNet synonym sets
    $\mathbb{A} \leftarrow$ GetAntonym$(x_i)$    ▷ Get antonyms by WordNet
    $\mathbb{S}_{ml} \leftarrow \mathbb{S}_{ml} \setminus \{x_i\}$
    $\mathbb{S}_{ms} \leftarrow \mathbb{S}_{ms} \setminus \mathbb{S}_{ml} \setminus \{x_i\}$
    $\mathbb{S}_{mms} \leftarrow \mathbb{S}_{mms} \setminus \mathbb{S}_{ms} \setminus \mathbb{S}_{ml} \setminus \{x_i\}$    ▷ Remove overlapping elements to make $\mathbb{S}_{ml}, \mathbb{S}_{ms}, \mathbb{S}_{mms}$ disjoint
    $\mathbb{S}_{embed} \leftarrow$ GetEmbeddingSwaps$(x_i)$
    $\mathbb{S}_{MLM} \leftarrow$ GetMLMSwaps$(x_i, \mathbf{x}_{ori})$
    $\mathbb{S}_{recons} \leftarrow$ GetReconsSwaps$(x_i, \mathbf{x}_{ori})$
    if $x_i^{'} \in \mathbb{S}_{ml}$ then
        The substitution is a morphological substitution
    else if $x_i^{'} \in \mathbb{S}_{ms}$ then
        The substitution is a matched sense substitution
    else if $x_i^{'} \in \mathbb{S}_{mms}$ then
        The substitution is a mismatched sense substitution
    else if $x_i^{'} \in \mathbb{A}$ then
        The substitution is an antonym substitution
    else
        This substitution is a other type
    end if
    Check the substitution types of each word in $\mathbb{S}_{embed}$ by comparing with $\mathbb{S}_{ml}, \mathbb{S}_{ms}, \mathbb{S}_{mms}, \mathbb{A}$
    Check the substitution types of each word in $\mathbb{S}_{MLM}$ by comparing with $\mathbb{S}_{ml}, \mathbb{S}_{ms}, \mathbb{S}_{mms}, \mathbb{A}$
    Check the substitution types of each word in $\mathbb{S}_{recons}$ by comparing with $\mathbb{S}_{ml}, \mathbb{S}_{ms}, \mathbb{S}_{mms}, \mathbb{A}$
    if $\mathbb{S}_{ml}, \mathbb{S}_{ms}, \mathbb{S}_{mms}, \mathbb{A} \neq \emptyset$ then
        $\mathbb{I}$.append$(i)$    ▷ We only include the words whose have morphological substitutions, matched sense substitutions, mismatched sense substitutions
    end if
end for
$\mathbb{O} \leftarrow$shuffle.$(\mathbb{I})$
```

Algorithm 2 **GetMLMSwaps**$x_i, \mathbf{x}_{ori}$

```
Require: $x_i, \mathbf{x}_{ori}$, BERT, Lemmatizer
$\mathbf{x}_{mask} \leftarrow \{x_1, \cdots, x_{i-1}, [\text{MASK}], x_{i+1}, \cdots, x_n\}$    ▷ Get masked input
Candidates← Top-k prediction of $\mathbf{x}_{mask}$ using BERT
New_Candidates ← []
for $w$ ∈Candidates do
    $w_{lemmatized} \leftarrow$ Lemmatizer($w$)
    if $w_{lemmatized}$ ∉Candidates and $w_{lemmatized}$ ∉New_Candidates then
        New_Candidates.append($w_{lemmatized}$)
    else
        New_Candidates.append($w$)
    end if
end for
return New_Candidates
```

words as the top-50 to top-550 words in the training dataset. The reason that we omit the top 50 words as the high-frequency token is that these words are often stop words, and they are seldom used as word substitutions in SSAs. The low-frequency words are the top-10K to top-10.5K occurring words in AG-News' training set.

D.3 Experiment Details of Section 4.2

Here, we give more details on the sentence embedding similarity experiment in Section 4.2. The readers can refer to Algorithm 1 to see how we obtain the different types of word substitution sets, the substituted indices set $\mathbb{I}$ and the ordered list $\mathbb{O}$ from a pair of $(\mathbf{x}_{ori}, \mathbf{x}_{adv})$.

We also use a figurative illustration to show how we obtain $\mathbf{x}^n_{swap}$ in Figure 2. In Figure 2, we show how to use the *same sense substitution set* to replace the words in $\mathbf{x}_{ori}$ based on the ordered list $\mathbb{O}$. As can be seen in the figure, we swap the words in $\mathbf{x}_{ori}$ according to the order determined by $\mathbb{O}$; since the first element in $\mathbb{O}$ is 5, we will first replace x_5 in $\mathbf{x}_{ori}$ with one of the same sense synonyms of x_5. We thus obtain the $\mathbf{x}^1_{swap}$. In order to compute the sentence embedding similarity between $\mathbf{x}^1_{swap}$ and $\mathbf{x}_{ori}$, we extract a context around the word just replaced; in this case, we will extract the context around the fifth word in $\mathbf{x}^1_{swap}$ and $\mathbf{x}_{ori}$. Different from what we really use in our experiment, we set the window size w to 1 in Figure 2; this is because using $w = 7$ is too large for this example. Thus, we should extract $\mathbf{x}^1_{swap}[4:7]$ and $\mathbf{x}_{ori}[4:7]$; however, since the sentences only have 5 words, the context to be extracted will exceed the length of the sentences. In this case, we simply extract the context until the end of both sentences.[8] The parts that will be used for computing the sentence embeddings in each sentence are outlined with a dark blue box in Figure 2. Next, we follow a similar process to obtain $\mathbf{x}^2_{swap}$ and $\mathbf{x}^3_{swap}$ and compare their sentence embedding cosine similarity with $\mathbf{x}_{ori}$.

D.4 Experiment Details of Section 4.3

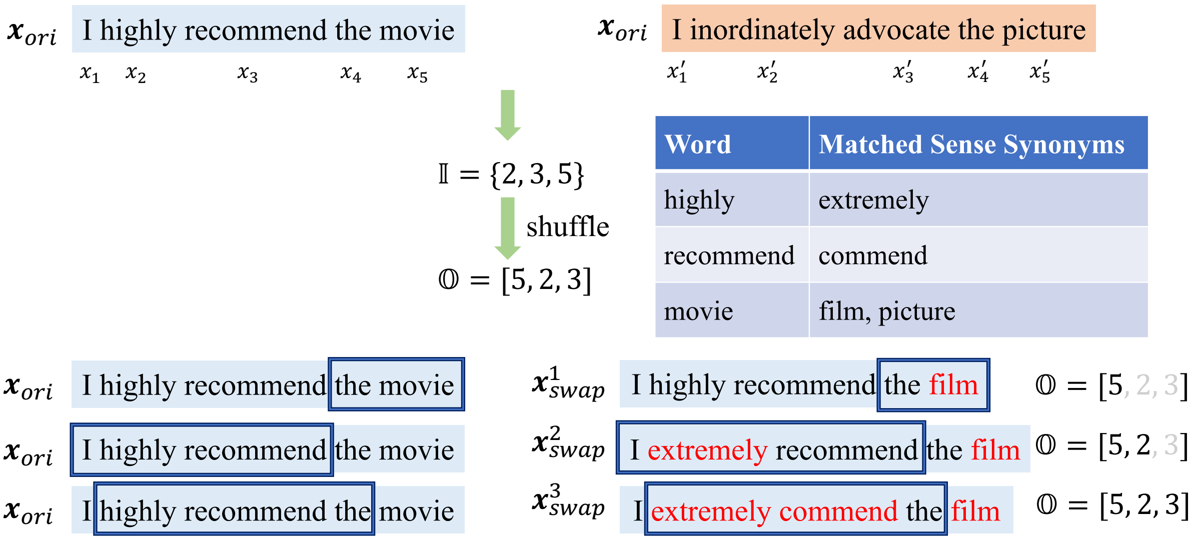
In this experiment, we usethe POS tagger in NLTK to identify the verb form of the verbs. The inflectional form of the verbs are obtained using LemmInflect. Here, we list the verb inflectional form conversion rules:

- For each third-person singular present verb, it is converted to the verb's base form.

- For each third past tense verb, it is converted to the verb's gerund or present participle form (V+ing).

- For all verbs whose form is not third-person singular present and is not past tense verb, we convert them into the third-person singular present. We provide three random examples from the test set in AG-News that are perturbed using the above rules in Table 7. It can be easily seen that all the perturbed sentences are ungrammatical. Interestingly, LanguageTool detects no grammar errors in all the six samples in Table 7.

[8] Similarly, if the context to be extracted starts from a position that is on the left-hand side of the sentence, we simply extract the context starting from the first word in the sentence.


x_{ori} I highly recommend the movie
x_1 x_2 x_3 x_4 x_5
x_{ori} I inordinately advocate the picture
x'_1 x'_2 x'_3 x'_4 x'_5
$\mathbb{I} = \{2, 3, 5\}$
shuffle
$\mathbb{O} = [5, 2, 3]$
x_{ori} I highly recommend the movie
x_{ori} I highly recommend the movie
x_{ori} I highly recommend the movie
x^1_{swap} I highly recommend the film
x^2_{swap} I extremely recommend the film
x^3_{swap} I extremely commend the film
$\mathbb{O} = [5, 2, 3]$
$\mathbb{O} = [5, 2, 3]$
$\mathbb{O} = [5, 2, 3]$

Word	Matched Sense Synonyms
highly	extremely
recommend	commend
movie	film, picture

Figure 2: An example for illustrating process of obtaining $\mathbb{I}, \mathbb{O}$ and $\mathbf{x}^n_{swap}$ from a pair of $(\mathbf{x}_{ori}, \mathbf{x}_{adv})$. Here, the substitution type used for constructing $\mathbf{x}^n_{swap}$ is the matched sense synonyms. The subsentences outlined by dark blue in the bottom three $\mathbf{x}_{ori}$ and $\mathbf{x}^n_{swap}$ are the parts that are used to compute the sentence embedding by the sentence encoder. In the figure, we set the window size w of the sentence encoder to 1 for the ease of illustration.

E Supplementary Materials for Experiments of Sentence Encoders

E.1 Distribution of the Sentence Embedding Cosine Similarity of Different Substitution Types

In Figure 3, we show the distribution of the USE sentence embedding cosine similarity of different word replacement types using different numbers of word replacements n. The left subfigure shows the distribution of the cosine similarity between $\mathbf{x}_{ori}$ and $\mathbf{x}^1_{swap}$ and the right subfigure is the similarity distribution between $\mathbf{x}_{ori}$ and $\mathbf{x}^8_{swap}$. While in Figure 1, we can see that the sentence embedding cosine similarity of different word substitution types is sometimes separable on average, we still cannot separate valid and invalid word substitution simply using one threshold. This is because the word embedding cosine similarity scores of different word substitution types are highly overlapped, which is evident from Figure 3. This is true for different n of $\mathbf{x}^n_{swap}$, and we only show $n = 1$ and $n = 8$ for simplicity.

E.2 Different Methods For Computing Sentence Embedding Similarity

In this section, we show some supplementary figures of the experiments in Section 4.2. Recall that in the main content, we only show the sentence embedding cosine similarity results when we compare $\mathbf{x}^n_{swap}$ with $\mathbf{x}_{ori}$ around a 15-word window around the n-th substituted word. But we have mentioned in Section 2.3 that this is not what is always done. In Figure 4, we show the result when we compare $\mathbf{x}^n_{swap}$ with $\mathbf{x}_{ori}$ using **the whole sentence**. It can be easily observed that it is still difficult to separate valid swaps from the invalid ones using a threshold on the cosine similarity. One can also observe that the similarity in Figure 4 is a lot higher than that in Figure 1.

Another important implementation detail about sentence encoder similarity constraint is that some previous work does not calculate the similarity of the current $\mathbf{x}_{swap}$ with $\mathbf{x}_{ori}$. Instead, they calculate the similarity between the current $\mathbf{x}_{swap}$ and the $\mathbf{x}_{swap}$ in the previous substitution step (Garg and Ramakrishnan, 2020). That is, if in the previous substitution step, 6 words in $\mathbf{x}_{ori}$ are swapped, and in this substitution step, we are going to make the 7th substitution. Then the sentence embedding similarity is computed between the 6-word substituted

Original sentence	Verb-perturbed sentence
Storage, servers bruise HP earnings update Earnings per share rise compared with a year ago, but company misses analysts' expectations by a long shot.	Storage, servers bruises HP earnings update Earnings per share rise compares with a year ago, but company miss analysts' expectations by a long shot.
IBM to hire even more new workers By the end of the year, the computing giant plans to have its biggest headcount since 1991.	IBM to hires even more new workers By the end of the year, the computes giant plans to has its biggest headcount since 1991.
Giddy Phelps Touches Gold for First Time Michael Phelps won the gold medal in the 400 individual medley and set a world record in a time of 4 minutes 8.26 seconds.	Giddy Phelps Touches Gold for First Time Michael Phelps winning the gold medal in the 400 individual medley and sets a world record in a time of 4 minutes 8.26 seconds.

Table 7: Examples of the verb-perturbed sentences. The perturbed verbs are highlighted in red, and their unperturbed counterparts are highlighted in blue.

sentence and the 7-word substituted sentence.

In Figure 5, we show the result when we we compare $\mathbf{x}_{swap}^{n}$ with $\mathbf{x}_{swap}^{n-1}$ around a 15-word window around the $n-th$ substituted word. This is adopted in Garg and Ramakrishnan (2020), according to TextAttack (Morris et al., 2020b). Last, we show the result when we compare $\mathbf{x}_{swap}^{n}$ with $\mathbf{x}_{swap}^{n-1}$ with the whole sentence; this is not used in any previous works, and we include this for completeness of the results. All the sentence encoders used in Figure 1, 4, 5, 6 are USE.

E.3 Different Sentence Encoders

We show in Figure 7 the result when we compare $\mathbf{x}_{swap}^{n}$ with $\mathbf{x}_{ori}$ around a 15-word window around the n-th substituted word using a DistilBERT fine-tuned on STS-B, which is the sentence encoder used in Yoo and Qi (2021). Figure 7 shows that DistilBERT fine-tuned model better distinguishes between antonyms and synonym swaps, compared with the USE in Figure 1. However, it still cannot distinguish between the matched and mismatched synonym substitutions very well. Interestingly, this model is flagged as deprecated on huggingface for it produces sentence embeddings of low quality. We also show the result when we use a DistilRoBERTa fine-tuned on STS-B in Figure 8. Interestingly, this sentence encoder can also better distinguish antonym substitutions and synonym substitutions on average. This might indicate that the models only fine-tuned on STS-B can have the ability to distinguish valid and invalid swaps.

In Figure 9, we show the result when we compare $\mathbf{x}_{swap}^{n}$ with $\mathbf{x}_{ori}$ around a 15-word window around the $n-th$ substituted word using sentence-transformers/all-MiniLM-L12-v2. This model has 110M parameters and is the 4th best sentence encoder in the pre-trained models on sentence-transformer package (Reimers and Gurevych, 2019). It is trained on 1 billion text pairs. We report the result when using this sentence encoder because it is the best model that is smaller than USE, which has 260M parameters. We can see that the trend in Figure 9 highly resembles that in Figure 1, indicating that even a very strong sentence encoder is not suitable to be used as a constraint in SSAs.

We also include the result when we use the best sentence encoder on sentence-transformer package, the all-mpnet-base-v2. It has 420M parameters. The result is in Figure 10, and it is obvious that it is still quite impossible to use this sentence encoder to filter invalid swaps.

F Statistics of Other Victim Models and Other Datasets

In this section, we show some statistics on adversarial samples in the datasets generated by Yoo et al. (2022). The main takeaway in this is part is: Our observation in Section 3 holds across different types of victim models (LSTM, CNN, BERT, RoBERTa), different SSAs, and different datasets.

F.1 Proportion of Different Types of Word Replacement

First, we show how different word substitution types consist of the adversarial samples of AG-News. We show the result of four models and four SSAs in Table 8, 9, 10, 11. This is done by a similar procedure as in Section 3.1.1.

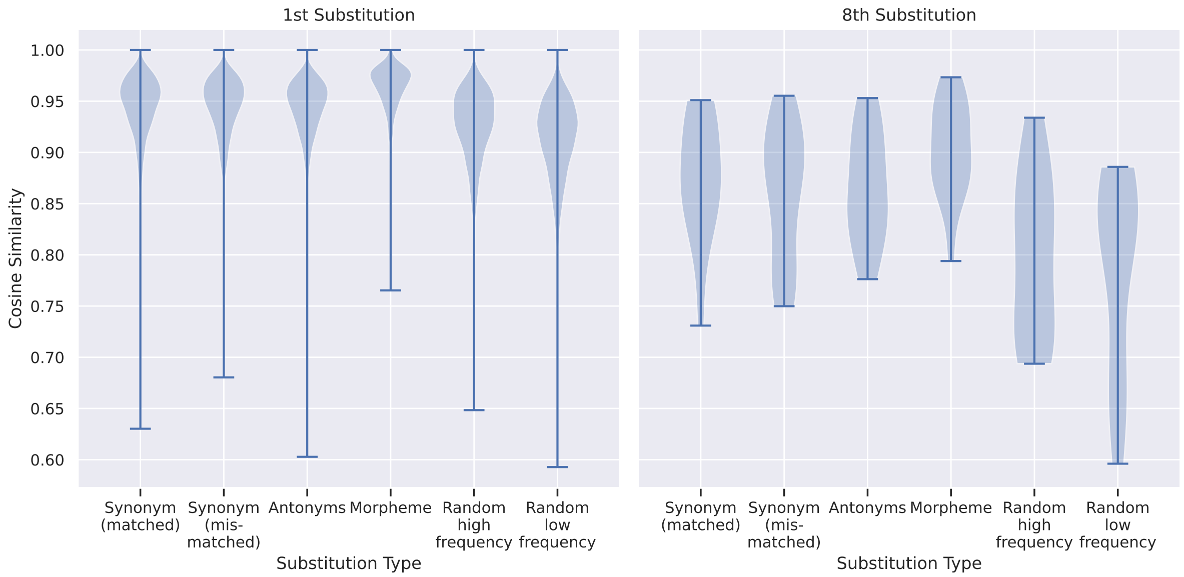


Figure 3: The USE sentence embedding cosine similarity distribution between $\mathbf{x}_{ori}$ and the series of sentences obtained by replacing words in $\mathbf{x}_{ori}$ with one type of word substitution. The window size is the same as Figure 1. The left subfigure shows the distribution of the cosine similarity between $\mathbf{x}_{ori}$ and $\mathbf{x}^1_{swap}$ and the right subfigure is the similarity distribution between $\mathbf{x}_{ori}$ and $\mathbf{x}^8_{swap}$.

Model	Matched sense	Mismatched sense	Morphological	Antonym	Others
CNN	5449 (16.8%)	23727 (73.2%)	788 (2.43%)	0 (0.0%)	2434 (7.51%)
LSTM	5185 (15.7%)	24621 (74.5%)	788 (2.38%)	0 (0.0%)	2467 (7.46%)
BERT	4319 (16.2%)	19467 (73.2%)	1026 (3.86%)	0 (0.0%)	1788 (6.72%)
RoBERTa	5057 (16.3%)	21741 (70.2%)	1253 (4.05%)	0 (0.0%)	2905 (9.38%)

Table 8: Attack statistics of other models on AG-News. The SSA use to attack the models is PWWS.

F.2 Statistics of Different Datasets

In this section, we show the statistics of types of word substitution of another two datasets in Yoo et al. (2022). The result is in Table 12. Clearly, our observation that valid word substitutions are scarce can also be observed in both SST-2 and IMDB.

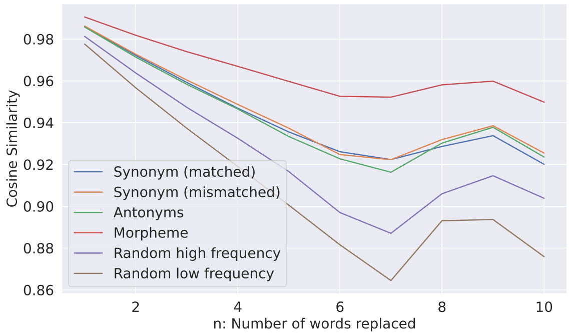


Figure 4: The USE sentence embedding cosine similarity between $\mathbf{x}_{ori}$ and the series of sentences obtained by replacing words in $\mathbf{x}_{ori}$ with one type of word substitution. Different from Figure 1, we use the whole sentence (without using window) to compute the sentence embedding of $\mathbf{x}_{ori}$ and $\mathbf{x}^n_{swap}$.

Model	Matched sense	Mismatched sense	Morphological	Antonym	Others
CNN	319 (0.891%)	897 (2.5%)	1464 (4.09%)	0 (0.0%)	33138 (92.5%)
LSTM	304 (0.752%)	1125 (2.78%)	1662 (4.11%)	0 (0.0%)	37350 (92.4%)
BERT	399 (0.806%)	1632 (3.3%)	2471 (4.99%)	0 (0.0%)	45008 (90.9%)
RoBERTa	391 (0.783%)	1613 (3.23%)	2276 (4.56%)	2 (0.004%)	45656 (91.4%)

Table 9: Attack statistics of other models on AG-News. The SSA use to attack the models is TextFooler.

Model	Matched sense	Mismatched sense	Morphological	Antonym	Others
CNN	34 (1.21%)	73 (2.6%)	232 (8.25%)	5 (0.178%)	2468 (87.8%)
LSTM	30 (0.998%)	62 (2.06%)	234 (7.78%)	7 (0.233%)	2674 (88.9%)
BERT	21 (0.88%)	39 (1.6%)	184 (7.7%)	8 (0.34%)	2128 (89.4%)
RoBERTa	25 (0.755%)	61 (1.84%)	304 (9.18%)	6 (0.181%)	2914 (88.0%)

Table 10: Attack statistics of other models on AG-News. The SSA use to attack the models is BAE.

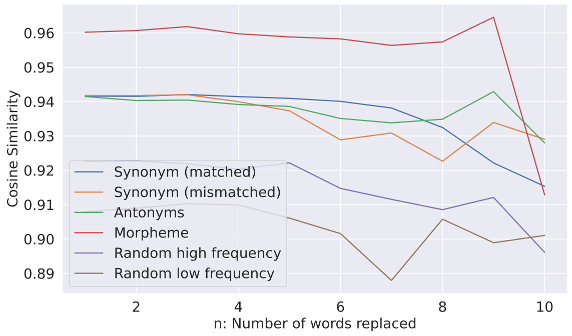


Figure 5: The USE sentence embedding cosine similarity between $\mathbf{x}_{ori}$ and the series of sentences obtained by replacing words in $\mathbf{x}_{ori}$ with one type of word substitution. Different from Figure 1, we compare $\mathbf{x}_{swap}^{n}$ with $\mathbf{x}_{swap}^{n-1}$ for $n \geq 2$. The sentence embedding is calculated using a 15-word window around the n-th substituted word, as in Figure 1.

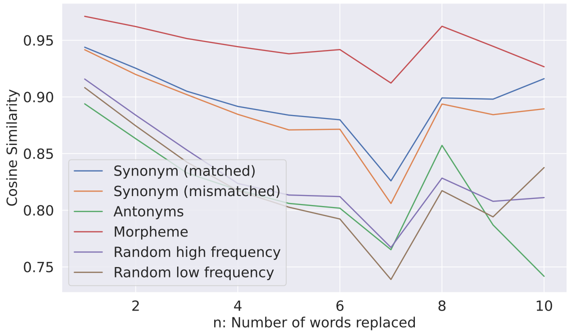


Figure 7: Using the DistilBERT fine-tuned on STS-B as the sentence encoder. Sentence embedding cosine similarity between $\mathbf{x}_{ori}$ and the series of sentences obtained by replacing words in $\mathbf{x}_{ori}$ with one type of word substitution.

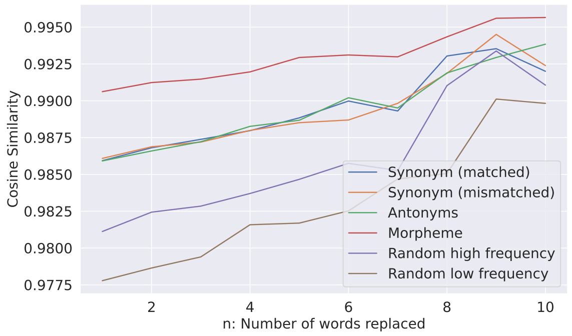


Figure 6: The USE sentence embedding cosine similarity between $\mathbf{x}_{ori}$ and the series of sentences obtained by replacing words in $\mathbf{x}_{ori}$ with one type of word substitution. The sentence embedding similarity shown in this figure is calculated by the whole sentence without windowing and the cosine similarity is calculated between $\mathbf{x}_{swap}^{n}$ and $\mathbf{x}_{swap}^{n-1}$.

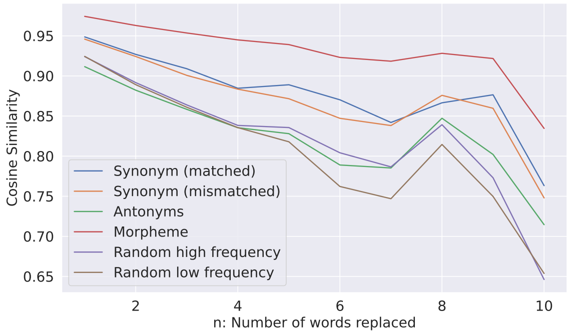


Figure 8: Using the DistilRoBERTa fine-tuned on STS-B as the sentence encoder. Sentence embedding cosine similarity between $\mathbf{x}_{ori}$ and the series of sentences obtained by replacing words in $\mathbf{x}_{ori}$ with one type of word substitution.

Model	Matched sense	Mismatched sense	Morphological	Antonym	Others
CNN	65 (3.86%)	176 (10.5%)	706 (42.0%)	0 (0.0%)	735 (43.7%)
LSTM	70 (3.9%)	208 (11.6%)	698 (38.9%)	0 (0.0%)	820 (45.7%)
BERT	53 (4.32%)	118 (9.62%)	530 (43.2%)	0 (0.0%)	526 (42.9%)
RoBERTa	59 (4.21%)	137 (9.79%)	581 (41.5%)	0 (0.0%)	623 (44.5%)

Table 11: Attack statistics of other models on AG-News. The SSA use to attack the models is TextFooler-Adj.

Model	Matched sense	Mismatched sense	Morphological	Antonym	Others
SST-2	34 (0.945%)	118 (3.28%)	206 (5.72%)	0 (0.0%)	3241 (90.1%)
IMDB	1881 (1.43%)	4825 (3.66%)	8708 (6.6%)	21 (0.0159%)	116479 (88.3%)

Table 12: Attack statistics of other BERT fine-tuned on other datasets. The SSA use to attack the models is TextFooler.

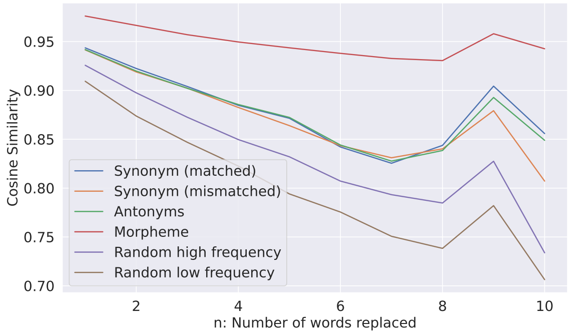


Figure 9: The sentence-transformers/all-MiniLM-L12-v2 as the sentence encoder. Sentence embedding cosine similarity between $\mathbf{x}_{ori}$ and the series of sentences obtained by replacing words in $\mathbf{x}_{ori}$ with one type of word substitution.

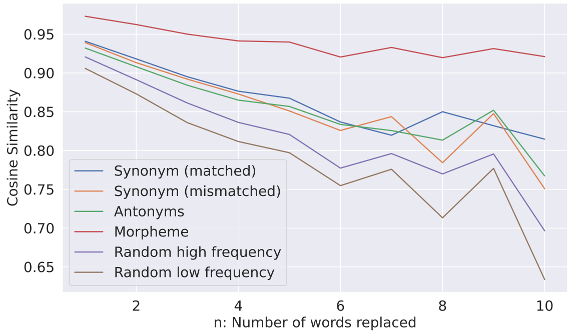


Figure 10: The sentence-transformers/all-mpnet-base-v2 as the sentence encoder. Sentence embedding cosine similarity between $\mathbf{x}_{ori}$ and the series of sentences obtained by replacing words in $\mathbf{x}_{ori}$ with one type of word substitution.

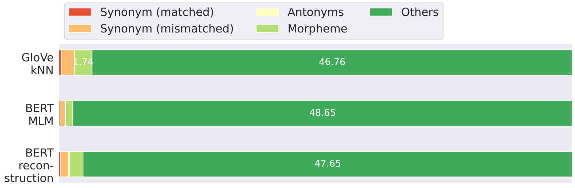


Figure 11: The average words of different substitution types in the candidate word set with 50 words for each transformation. If the average number of words of a substitution type is less than 1.7, we do not show the average number in the bar.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Limitations

☑ A2. Did you discuss any potential risks of your work?
Limitations, Ethical Statement and Broader Impacts

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Sec 3.1.1, App D

☑ B1. Did you cite the creators of artifacts you used?
Sec 3.1.1, App B, App D

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
They do not provide licenses

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Sec 3.1.1, App B

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Removing name entities in AG-News causes the news to be unreadable.

□ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Sec 3.1.1, App B

C ☑ Did you run computational experiments?

Sec 3

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
App F.3.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

□ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Not applicable. Left blank.

□ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Not applicable. Left blank.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
App E

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.