Scientific Research Publishing

Journal of Information Security, 2023, 14, 181-194
https://www.scirp.org/journal/jis
ISSN Online: 2153-1242
ISSN Print: 2153-1234

FastAttacker: Semantic Perturbation Functions via Three Classifications

Meng Lu

Institute of Automation, Chinese Academy of Sciences, Beijing, China
Email: meng.lv@nipr.ia.ac.cn

How to cite this paper: Lu, M. (2023) FastAttacker: Semantic Perturbation Functions via Three Classifications. *Journal of Information Security*, **14**, 181-194.
https://doi.org/10.4236/jis.2023.142011

Received: December 30, 2022
Accepted: April 25, 2023
Published: April 28, 2023

Abstract

Deep neural networks (DNNs) have achieved great success in tasks such as image classification, speech recognition, and natural language processing. However, they are susceptible to false predictions caused by adversarial exemplars, which are normal inputs with imperceptible perturbations. Adversarial samples have been widely studied in image classification, but not as much in text classification. Current textual attack methods often rely on low-success-rate heuristic replacement strategies at the character or word level, which cannot search for the best solution while maintaining semantic consistency and linguistic fluency. Our framework, FastAttacker, generates natural adversarial text efficiently and effectively by constructing different semantic perturbation functions. It optimizes perturbations constrained in generic semantic spaces, such as the typo space, knowledge space, contextualized semantic space, or a combination. As a result, the generated adversarial texts are semantically close to the original inputs. Experiments show that FastAttacker generates adversarial texts from different levels of spatial constraints, making the problem of finding synonyms an optimal solution problem. Our approach is not only robust in terms of attack generation, but also in terms of adversarial defense. Experiments have shown that state-of-the-art language models and defense strategies are still vulnerable to FastAttack attacks.

Keywords

FastAttack, Text Learning, Deep Neural Network

1. Introduction

The growing trend of social textual information dissemination, coupled with proper sharing with others via the Internet, has led to a huge demand for verification, called fact-checking. Adversarial robustness assessment of machine learn-

ing (ML) models is receiving increasing research attention because of their vulnerability to adversarial input perturbations (called adversarial attacks). In other words, as the coverage of potentially misleading information and the number of false statements increase, automated fact-checking research based on deep learning applications such as machine learning for text classification, sentiment analysis and textual entailment becomes promising.

Deep learning [1] techniques have created a huge demand for research in natural language processing. It has achieved extraordinary success in many areas. While existing work on adversarial samples has been successful in the image and speech domains, processing textual data remains challenging due to its discrete nature. In recent studies, modern state-of-the-art methods employed on text documents have proven vulnerable to adversarial examples of many data patterns. These intentionally artificially crafted examples visually resemble the original examples they can still fool state-of-the-art deep classifiers by making small modifications to the test input, leading to misclassification of the text input. Contrary well-crafted examples can attack state-of-the-art models formulated by placing very small pixels on the image and often with imperceptible perturbations to humans. This phenomenon has drawn significant attention to the robustness of new SOTA deep learning systems such as BERT, which has inspired a set of large-scale pre-trained language models for many NLP tasks. BERT is a state-of-the-art pre-trained language model nested in a Transformer framework with multiple networks. Generating adversarial samples in NLP domains is more complex and challenging than in computer vision domains due to the discontinuous nature of the input space, which does not have explicit gradients as image inputs and the need to maintain semantic consistency of the original text.

Current successful textual adversarial generation and attack methods mainly use heuristic word- or character-level replacement strategies, which makes finding the optimal solution challenging. Formally, in addition to the ability to deceive the target model, the output of a natural language attack system should satisfy three key utility-preserving properties: 1) human prediction consistency with human predictions should be maintained, 2) semantic similarity—the carefully crafted exemplars should have the same meaning as the source, as judged by humans, and 3) linguistic fluency—the generated exemplars should look natural and grammatical. Previous works hardly met these three requirements. They are unable to find optimal solutions for the huge space of possible substitution combinations while maintaining semantic consistency and linguistic fluency. The approach of synonym substitution requires arbitrary words in the vocabulary to replace input words in the space, which is prone to failure in considering semantic perturbation constraints and prone to creating invalid adversarial samples. Practical exploitation failures of other works do not handle large search spaces well because in practice a large number of queries are required to generate an adversarial example and it grows exponentially with the input length.

A major bottleneck of most existing textual adversarial attacks is that they cannot be generalized to other languages because of their unique language-related features and lack of general language resources. Other works, such as the use of gradient backpropagation algorithms for perturbations generated as adversarial samples, are practical computations from continuous embedding space to discrete token space. Previous rule-based synonym replacement strategies lead to more natural adversarial samples from synonym candidate token spaces, such as TextFooler.

To address the above problem, we propose a semantic space query efficient adversarial attack framework, FastAttacker, to generate semantically close adversarial text to the original input to approach our target goal and resolve the current trigger point in the model. Our model achieves a fairly high success rate in both target and non-target scenarios through a black-box setting based on relocatability. We validate the robustness of the model on the textual implication task.

We experiment on large-scale FEVER and FEVER 2.0 datasets and attack the state-of-the-art target models BERT and KernelGAT that are still vulnerable to FastAttacker attacks. Our models outperform two powerful baselines: TextFooler [2] and BERT-Attack [3]. Note that our usage and validation are generic and can be referenced and applied to other adversarial example tasks.

Main Contribution: The results show that our experiments can be used to further evaluate the robustness of NLP models in datasets for statement validation of new datasets. The main contributions of our paper are listed below:

- We assess the significance of attentional connectivity for word embeddings and generate them using BERT. As a result, we introduce FastAttack, a unified and effective adversarial attack framework that builds semantic perturbation functions that limit perturbations within different semantic spaces and their combinations.
- We discover that traditional methods like GloVE and Word2Vec do not effectively capture contextual semantics, impacting model predictions. In contrast, FastAttack generates context-aware perturbations that don't require external knowledge, making it easily adaptable to various languages.
- We observe that the fine-tuned models we attack tend to neglect semantics and give too much weight to certain word patterns for text prediction, resulting in less robust models. To demonstrate this, we carried out comprehensive experiments on various datasets and languages and found that FastAttack generates adversarial texts that are more semantically similar to benign inputs, achieving higher attack success rates compared to existing attack algorithms across different settings.

2. Related Works

Our work is related to fact-checking and validation studies and verifies the robustness weaknesses of current state-of-the-art adversarial attack models.

2.1. Fact Verification

The Fact Extraction and Verification (FEVER) shared task contains 185,445 statements [4] that are manually crafted input based on Wikipedia page validation and output classified as SUPPORTED, REFUTED, or NOTENOUGHINFO. Statements are classified as SUPPORTED and REFUTED, and the exact necessary evidence to support or refute a statement needs to be returned by a combination of the corresponding sentences. Many statements are extracted from Wikipedia by annotators and mutated in various ways, some of which are interpreted and some of which are meaning-altering. Most participants in the shared task model this as a pipeline that consists of an information retrieval component—a search for pages/phrases with content related to a given statement and a natural language inference (NLI) component with the label of that statement. The second iteration of the task—FEVER 2.0—builds on the same dataset of 1174 statements that were manually crafted by participants submitted during the Breaker phase of the 2019 shared task. SemAttack generalizes existing word-level attacks by proposing generic semantic perturbation functions that optimize and constrain perturbations in different semantic spaces so that the generated adversarial text retains its semantics.

The current fact-checking model primarily uses the official FEVER baseline from the workshop, downloading datasets from FEVER (2018) and FEVER 2.0 (2019). FEVER tasks are typically divided into three steps: document retrieval, sentence retrieval, and claim validation. In our study, we focus on the claim validation task. Some previous work performed stringing of evidence to prove the label of a statement. Studying the inference of pairs of claim evidence and assigning them to claim labels is another research direction.

Research on the FEVER 1.0 dataset aims to develop efficient automated fact-checking systems to check the veracity of human statements from Wikipedia. GEAR proposes claim verification as a graph inference task with two types of attention. The FEVER we aim to attack uses the baseline of a kernel graph attention network (KernelGAT) [5] with a BERT-Large model. KGAT introduces node kernels that better measure the importance of evidence nodes and edge kernels that perform fine-grained evidence propagation in the graph into the graph attention network for more accurate fact verification. The model uses graph models to reason about and aggregate claim evidence pairs. The main idea of these methods for creating graph-based models is to create node interactions for joint inference by connecting multiple pieces of evidence.

The FEVER shared task (FEVER 2.0) aims to develop automated fact-checking systems to check the veracity of human-generated statements by extracting evidence from Wikipedia. FEVER 1.0 was held as a competition on Codalab 1 with a blind test set and attracted a lot of attention from the NLP community. Many fact-checking frameworks verify statement evidence and utilize natural language inference (NLI) techniques. Natural Language Inference (NLI), also known as Recognizing Textual Implication (RTE), determines whether a given hypothesis

logically follows an implicit, contradictory, or neutral premise. It is similar to the FEVER task, except that FEVER requires the system to match pairs of claim evidence and usually has various pieces of evidence corresponding to a claim.

2.2. Adversarial Attacks in NLP

Adversarial examples are artificial textual interferences that cause the target model to output incorrect predictions or judgments. Adversarial attacks may occur at the character, lexical, lexical, syntactic, or semantic level. Many studies contribute to the creation of adversarial samples by greedy algorithms that rank the token elements in the input sequence by importance according to their proposed initial scoring equations. The elements are then greedily perturbed based on a precomputed re-ranking to improve query efficiency. The shortcoming of these methods is their inherent limitation of modifying each position at most once, which leads to a strictly limited search space. Ribeiro *et al.* (2018) [6] and Michele *et al.* (2019) propose methods to find adversarial examples by preserving the semantic content, as these elaborate attacks tend to corrupt the semantics of the sentences. Alzanto *et al.* (2018) and Jin *et al.* (2019) introduce synonym substitution to create adversarial examples with a fixed word embedding space to search for the n closest words. Aye *et al.* (2018) introduce the method by conditionally generating adversaries over syntactic templates to interpret adversarial examples.

2.3. Adversarial Training in NLP

The intent of operating adversarial training is to ensure our NLP systems are not left vulnerable to SOTA attacks. The existing adversarial training work includes augmenting the training data by adding the adversaries or replacing the clean samples in the training dataset.

There are three types of semantic spaces taken into account:

- Typo-Space, fool the models with typo words or characters instead of human judges;
- Embedding-Space, exploiting external linguistics as valid perturbation CANDIDATES;
- Semantic-Space, fools the model by utilizing the embedding space of BERT to generate a contextualized perturbation set semantically close to the original word.

The semantic space does not require additional knowledge since it utilizes BERT to produce contextualized tokens; therefore, it can scale to other languages, especially low-resource languages where large datasets are unavailable.

2.4. Factual Verification Literature Review

There are several correlated problems in verifying the truthfulness of one or multiple sentences, such as claim verification, natural language inference (NLI), misinformation detection, rumor identification, sentiment analysis and subjec-

tivity detection, Etc. Previously, numerous pieces of research on rumors and the credibility of information before fake news appeared among researchers. Kumar *et al*. categorized fake news studies into two types: opinion-based and fact-based. Ihsan *et al*. (2022) classified fake news detection methods to detect fake news on a large scale based on the features and methods.

Prior research aims to study factual verification given different data genres as evidence, such as structured, semi-structured, and unstructured data.

3. Definitions and Notations

Given an input $x = [x_0, x_1, \cdots, x_n]$, where x_i is the ith input token, the classifier f maps the input to final logits $z = f(x) \in R^C$, where C is the number of classes, and the outputs a label $y = \arg\max f(x)$.

To evaluate the effectiveness of attack algorithms during the attack, we introduce and calculate two indicators:

Targeted attack success rate (TSR):

$$\text{TSR} = \frac{\sum_{x' \in (D_{adv})} \left[\mathcal{I} * \arg\max f(x') \equiv y^* \right]}{|D_{adv}|}$$

Untargeted attack success rate(USR):

$$\text{USR} = \frac{\sum_{x' \in (D_{adv})} \left[\mathcal{I} * \arg\max f(x') \neq y \right]}{|D_{adv}|}$$

D_{adv}: adversarial datasets composed of an attack algorithm generate one adversarial sentence for each sample by the attack algorithm.

y^*: targeted false class.

y: ground truth label.

$\mathcal{I}(\cdot)$: indicator function.

4. Methods

To tackle the issues of adversarial texts semantically inconsistent, we adopt the general form of the semantic perturbation functions and discuss their classifications under different semantic circumstances.

x: one token as input.

S: CANDIDATES perturbation space, $S = x_0^*, x_1^*, \cdots, x_n^*$.

f: perturbation function.

4.1. Algorithm

Whitebox and Blackbox Attack: FastAttacker requires access to the parameters and gradients of the function created by the model but can be used under both white-box and black-box scenarios. Our experiments utilize a zero-query black-box setting, with state-of-the-art large-scale language target models enhanced with cutting-edge defense methods. The framework is inaccessible during the attack progress. We create a common scenario in social media applications and

better verify [7] the generic effectiveness and efficiency across the models.

The algorithm is to propose a genetic function: semantic perturbation function. The advantage of this method is that FastAttacker can produce [8] an adversarial text which keeps the textual semantic to the largest extent, and the problem of searching for the optimal perturbation in different semantic spaces determined by the semantic perturbation function [9] is transferred into the problem of finding the optimal solution. Thus, the query work token can be reduced as little as possible when the targeted Attack is realized. We adopt baseline methods to the setting of zero-query by performing a transferability-based black-box attack, creating adversarial texts by BERT to attack the target models.

In conclusion, FastAttack is: 1) effective, outperforming prior attacks in both success rate and perturbation rate; 2) utility-preserving, maintaining semantic content, grammaticality, and classification accuracy as determined by human evaluators; and 3) efficient, generating adversarial text with computational complexity proportional to text length.

For the detail of utilizing the algorithm, we:

1) First, set English words as tokens from the datasets, $x = \left[x_0, x_1, x_2, \cdots, x_n\right]$.

2) Utilize BERT as a classifier to map each x_i from input space to final output logits *z*. We use BERT to map the input *x* to embedding vector *e*.

$e_i = Matrix_{BERT} * x_i$

3) Initialize perturbation e^*, and add e^* to *e* for *m* iterations. We assign the perturbed embedding to e', which means: $e_i' = e_i + e_k^*$.

4) Initialize the adversarial text x' with *x*.

5) Optimize e^* with construct optimization function:

$L\left(e^*\right) = \left\|e^*\right\|_p + c * g\left(x'\right)$. The p-regularization of e^* aims to limit and control the magnitude of e^*.

6) $g\left(\cdot\right)$ is the attack objective function that divides the attack scenario into targeted and untargeted. In our experiments, we operate datasets on both scenarios. We use *c* to adjust the weights of the attack goal against the attack cost.

7) We construct the semantic space *S* with perturbation function $F\left(\cdot\right)$, $S = F\left(x_i\right)$ for each input token x_i. We select the perturbed token x_i' from the space *S*, which we have: $x_i' = \arg\min\left(\left(\left\|e_i' - M_e x_i'\right\|\right)_p\right), x_i' \in S$.

8) Optimization: We calculate the descent of function $L\left(e^*\right)$, set as $\nabla\left(e^*\right)$. We use α as the step size of the gradient descent and update the e_{k+1}^* with $e_{k+1}^* = e_k^* - \alpha\nabla L\left(e_k^*\right)$.

9) Token Substitution: After optimization, we get the optimal e_{k+1}^* and perturbed embedding $e' = e + e_{k+1}^*$.

10) Finally, we update and obtain the optimal adversarial text x' by calculating the argument of the minimization of the p-norm of perturbed embedding with BERT embedding of adversarial text x' and $x_i' = \arg\min_{x_i' \in S}\left(\left\|e_i' - M_e x_i'\right\|\right)_p$.

4.2. Typo-Based Perturbation Function

Perturbation at a word or character level determined by function f_t constrains

the search space in the typo space. Utilize typo words or characters to replace original tokens to fool the model but keep the original meaning perceived by a human. We employ the TextBugger (Li *et al*., 2018), which can attack deep text understanding systems under both white-box and black-box scenarios, as a reference for generating typos. Since our experiments were conducted under the black-box setting, gradients of the model are not directly available, and we need to change the input sequence without the influence of gradients. Thus, instead of directly selecting important words based on gradient information from white-box attacks, we first find important sentences and then the important words within them under the black-box settings. We find the important sentences, then utilize a scoring function to determine the importance of every word according to the classification result, and rank the words according to their scores. Finally, we use the bug selection algorithm to change the selected words.

In order to illustrate how the proposed framework can be effectively and efficiently adapted to the English setting, we generate word/character-level semantic space. We modify characters such as letters, special symbols, or numbers for character-level perturbation. For word-level perturbation, we modify the words by synonyms, misspellings, specific types of keywords, etc.

For vulnerable words, we use word replacement via BERT. We iteratively replace the words in the list for each word to search for the perturbations that can fool the target model. Compared with previous work, which usually uses multiple human-crafted approaches to ensure the semantic consistency of the generated example, FastAttacker can search the optimal perturbations from different typo-level spaces determined by the semantic perturbation function.

4.3. Embedding-Based Perturbation Function

The embedding level-based perturbation determined by the function f_e limits the embedding perturbation search space. The perturbation function uses tagged embeddings to construct candidate perturbation sets in the study for all possible replacements of the selected words as synonym sets. We initiate CANDIDATES using the N closest synonyms as their cosine similarity to each other in context. We use word embeddings to represent the words from to improve the quality of the adversarial examples and manually tag words with their semantic relations so that synonyms queried from the synonym set will maintain the semantic meaning of the queried words. We select these synonyms queried from CANDIDATES, construct the search space with these synonyms, and select these synonyms returned from CANDIDATES as the search space. FastAttacker utilizes this set of embedding vectors to select the top N synonyms with a cosine similarity to word "w" greater than the threshold value of. It is important to note that the list of candidates may contain both superlative and comparative words, which can result in issues with synonym substitution. To address this, the candidate search space is limited to synonym sets only.

Furthermore, even though BERT [10] is used to generate context-based word

embedding tags, the same tag for a word may have different lexical (POS) tags for different synonyms (e.g. "experienced" as a verb and "skilled" as an adjective) [11]. This issue could result in meaningless substitutions. To avoid this, FastAttacker only retains words with identical lexical tags. The frequency of POS is calculated in the set of synonyms and the most frequent ones are selected. Finally, after filtering the words using the created synonym set, the goal of generating adversarial input tokens that trick the model yet still maintain human interpretability is achieved.

4.4. BERT-Based Semantic Perturbation Function

The function f_b determines the context-level perturbation and leverages the contextual semantic space to avoid polysemy issues present in non-contextual embedding spaces like GLoVE or Word2Vec. By processing data with the BERT embedding space, FastAttacker can preserve contextualization both semantically and syntactically as much as possible. Additionally, to make the POS checking process easier, f_b can handle any language model, as long as pre-trained BERT models for that language exist.

5. Experiments

This section describes our experiments' dataset, evaluation metrics, baselines, and implementation details.

In our experiments, we construct BERT-base embedding space (Yuan *et al.*, 2019) and BERT-base-case (Devlin *et al.*, 2019). The number of BERT layers' hidden size is 768, the number of attention heads in each layer is 12, the hidden layer dropout probability is 0.1, the size of hidden embeddings is 12, and the maximum position of embeddings is 512. The computation can be parallelized by batching multiple input word tokens to increase throughput. We operate our experiments in a zero-query black-box setting. The target models are the state-of-the-art large-scale language models "Bert (Devlin *et al.*, 2019) and KernelGAT (Liu *et al.*, 2019)" enhanced with cutting-edge defense methods. This setting is a common scenario in real-world applications and better demonstrates the algorithm's ability to generalize across models. We adopt FastAttacker and baselines to this setting by performing a transferability-based black-box attack [12].

5.1. Datasets

We construct a purpose-built large public [13] fact verification dataset FEVER and FEVER 2.0 for this task. FEVER contains 185,445 human-generated claims, annotated claims with 5,416,537 Wikipedia documents from the June 2017 Wikipedia dump. Annotators classify all claims as SUPPORTS, REFUTES, or NOT ENOUGH INFO. We pre-trained and fine-tuned BERT with all FEVER train datasets and leveraged 999 claims for testing the BERT model. The dataset partition of the experiments re-mains the same with the FEVER Shared Task as shown in For the FEVER shared task (2.0) dataset, which contains 1774 claims.

We test the FEVER 2.0 datasets on the pre-trained model with FEVER datasets. The datasets were constructed in two stages, and we partitioned the dataset and kept the same with the FEVER Shared Task, as shown in Table 1.

5.1.1. Claim Generation

Information is extracted from Wikipedia and statements are generated. These statements are generated by interpreting facts and changing them in various ways, including mean changes.

5.1.2. Claim Labeling

Classify whether a claim is supported or refuted by Wikipedia, select the evidence for or against it, or determine whether there is not enough information to make a decision. Annotators select evidence from Wikipedia in the form of sentences without knowing where the statement was generated from.

5.1.3. Evaluation Metrics

Traditional metrics for evaluating [14] statement validation include primarily FEVER scores and label accuracy. One of our baseline methods, KernelGAT, utilizes the FEVER score and considers the Golden FEVER score.

5.2. Attack Baselines

We evaluate our model on Fever 1.0 and Fever 2.0 datasets and compare our method with two black-box settings: TextFooler and BERT-Attack, on adversarial attack success rate as our baselines, under both targeted and untargeted scenarios.

- TextFooler is a simple but powerful baseline for natural language attacks in the black box setting, and it can quickly create high-profile utility-preserving adversarial examples that force the target model to make incorrect predictions in the black box setting.
- BERT-Attack is a simple and effective method for generating adversarial samples using BERT as a language model, which can effectively generate fluent and semantically preserved adversarial samples that can successfully mislead state-of-the-art models in NLP, such as fine-tuned BERT for various downstream tasks.

5.3. Models

We evaluated the robustness of the BERT and KernelGAT models. We show their test accuracies in Table 2 and Table 3. Hyperparameter settings and training details are discussed below. The selected large-scale models and methods

Table 1. FEVER 1.0.

SPLIT	SUPPORTED	REFUTED	NOTENOUGHINFO
TRAIN	80,035	29,775	35,639
DEV	6666	6666	6666
TEST	6666	6666	6666

represent the SOTA performance on the RTE task and achieve the highest robustness. Athene, KGAT, and UNC NLP encode declarative evidence pairs using ESIM.

BERT and KernelGAT are our two main target models, and our approach generates adversarial attacks that significantly outperform previous SOTA approaches without pre-training. TextFooler and BERT-Attack are the two baselines in our experiments. They implement BERT word embedding classification and text entailment tasks to obtain better performance. For KernelGAT, they implement a version of GAT using dot product instead of kernel, similar to GEAR, to evaluate the effectiveness of kernel. GEAR uses graphical attention networks to extract complementary information from other evidence and aggregates all evidence through the attention layer. For our FastAttacker, the input is a long sentence connecting the statement and the evidence, and we perform a training task on the statement and pass its one-dimensional output to the attack model. For KernelGAT, the input is each statement-evidence pair, which has a different data dimension than our framework. The KernelGAT input needs to combine each statement's synonym substitution with all its evidence to form a two-dimensional array and pass them to the attack model. After training the statements using our framework and searching the set of synonyms for each

Table 2. Zero-query blackbox attack success rate for different attacks under targeted/untargeted attacks (TSR/USR) and corresponding word perturbation percentage against large-scale LMs and defense methods on FEVER 1.0 datasets.

Model	Attack Method	USR%/TSR%	Perturbation%
	TextFooler	69.5/24.0	13.9/43.9
BERT	FastAttacker(+f_t)	42.4/9.3	4.7/9.1
Acc:	FastAttacker(+f_e)	74.4/59.3	5.7/10.9
	FastAttacker(+f_b)	84.3/79.7	4.4/9.1
	FastAttacker(+f_{all})	94.6/90.8	4.3/10.2
	BERT-Attack	80.3/44.9	25.0/30.3

Table 3. Zero-query blackbox attack success rate for different attacks under targeted/untargeted attacks (TSR/USR) and corresponding word perturbation percentage against large-scale LMs and defense methods on FEVER 2.0 datasets.

Model	Attack Method	USR%/TSR%	Perturbation%
	TextFooler	32.3/17.8	11.6/13.4
KernelGAT	FastAttacker(+f_t)	53.8/33.4	23.9/29.1
Acc:	FastAttacker(+f_e)	40.7/23.2	21.4/22.2
	FastAttacker(+f_b)	76.5/63.8	30.9/36.3
	FastAttacker(+f_{all})	86.2/68.5	39.0/36.9
	BERT-Attack	80.3/44.9	25.0/30.3

word, we transfer the evidence to the generated statements, construct new statement-evidence pairs and transfer them to KernelGAT. Therefore, to evaluate our attack approach for the KernelGAT setup, we adjust the dimensionality of the input array from 1 to 2 dimensions when solving this data processing problem.

Adversarial Attack

To verify and prove the robustness of the model in realistic scenarios, we consider performing a black-box attack: a zero-query setup. We validate and evaluate the robustness of BERT and KernelGAT and show their testing accuracy and perturbations in Table 2 and Table 3.

- BERT-Attack (Li *et al*., 2020): a high-quality and effective strong black-box attack method that generates adversarial samples utilizing pre-trained masked language models exemplified by BERT. The core algorithm of BERT-Attack is straightforward and consists of two stages: finding the vulnerable words in one given input sequence for the target model; then applying BERT in a semantic-preserving way to generate substitutes for the vulnerable words. With the capability of BERT, the perturbations are generated considering the context around it.
- TextFooler: a black box attack method for generating adversarial text performs synonym replacement using a fixed word embedding space. Applying it to two fundamental natural language tasks can successfully attack three target models, including the powerful pre-trained BERT and widely used convolutional and recurrent neural networks.

5.4. Implementation Details

We apply this method to attack different types of models in the textual implication pipeline. We conducted a comprehensive experiment to evaluate the attack algorithm we set up, FastAttacker. We first fine-tuned the BERT model on the FEVER 1.0 dataset and took as input the declared crosstabs and their corresponding evidence. Second, we attack the pre-trained BERT using our model on different datasets using the TextFooler and BERT-Attack baselines. Next, third, we attack the KernelGAT model and compare the results with the two SOTA baselines. Finally, we evaluate and compare the experimental results of different baselines in different scenarios.

5.5. Adversarial Attack Evaluation

In the black-box attack scenarios in Table 2 and Table 3, FastAttacker was able to make the model incorrectly classify almost all sentences with only a small number of characters in both the target and non-target settings. The untargeted attack achieves a success rate of 94.6% by replacing a small number of tokens on the FEVER dataset.

6. Conclusion

In this paper, we propose a novel framework for semantic space-constrained

adversarial attacks, FastAttacker, which uses perturbation functions within different semantics to generate synonym substitutions. Our comprehensive experiments show that FastAttacker can generate natural adversarial texts in different semantic spaces and achieve higher attack success rates than existing textual attacks. Experiments show that FastAttacker leads to more accurate fact verification. Our study shows that four perturbation functions play different roles in generating adversarial texts in different semantic spaces and contribute to different aspects that are crucial for fact verification. We also demonstrate that existing SOTA LM and defense methods are still vulnerable to FastAttacker attacks. In the future, we will further investigate FastAttacker in more realistic scenarios and demonstrate that it is more general and can generate natural adversarial texts with high attack rates for different languages (e.g., English and Chinese). We hope that our research will throw light on future research to evaluate and enhance the robustness of LM to different languages.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Alzantot, M., Sharma, Y., Elgohary, A., Ho, B.J., Srivastava, M. and Chang, K.-W. (2018) Generating Natural Language Adversarial Examples. *Proceedings of the* 2018 *Conference on Empirical Methods in Natural Language Processing*, 2890-2896. https://doi.org/10.18653/v1/D18-1316

[2] Li, L.Y., Ma, R.T., Guo, Q.P., Xue, X.Y. and Qiu, X.P. (2020) Bert-Attack: Adversarial Attack against Bert Using Bert. *Proceedings of the* 2020 *Conference on Empirical Methods in Natural Language Processing* (*EMNLP*), 6193-6202. https://doi.org/10.18653/v1/2020.emnlp-main.500

[3] Jin, D., Jin, Z.J., Zhou, J.T. and Szolovits, P. (2020) Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment. *Proceedings of the AAAI Conference on Artificial Intelligence,* **34**, 8018-8025. https://doi.org/10.1609/aaai.v34i05.6311

[4] Wang, B.X., Xu, C.J., Liu, X.Y., Cheng, Y. and Li, B. (2022) SemAttack: Natural Textual Attacks via Different Semantic Spaces. *Findings of the Association for Computational Linguistics*: *NAACL* 2022, 176-205. https://doi.org/10.18653/v1/2022.findings-naacl.14

[5] Hou, B.R., Jia, J.H., Zhang, Y.H., Zhang, G.H., Zhang, Y., Liu, S.J. and Chang, S.Y. (2022) TextGrad: Advancing Robustness Evaluation in NLP by Gradient-Driven Optimization. https://arxiv.org/abs/2212.09254

[6] Bengio, Y., Leonard, N. and Courville, A. (2013) Estimating or Propagating Gradients through Stochastic Neurons for Conditional Computation. https://arxiv.org/abs/1308.3432

[7] Feng, S., Wallace, E., Grissom, I., Iyyer, M., Rodriguez, P., Boyd-Graber, J., *et al.* (2018) Pathologies of Neural Models Make Interpretations Difficult. *Proceedings of the* 2018 *Conference on Empirical Methods in Natural Language Processing*, 3719-3728. https://doi.org/10.18653/v1/D18-1407

[8] Carlini, N. and Wagner, D. (2017) Towards Evaluating the Robustness of Neural

Networks. *IEEE Symposium on Security and Privacy*, 39-57. https://doi.org/10.1109/SP.2017.49

[9] Carmon, Y., Raghunathan, A., Schmidt, L., Duchi, J.C. and Liang, P.S. (2019) Unlabeled Data Improves Adversarial Robustness. *Advances in Neural Information Processing Systems*. https://arxiv.org/abs/1905.13736

[10] Ebrahimi, J., Rao, A., Lowd, D. and Dou, D. (2017) Hotflip: White-Box Adversarial Examples for Text Classification. *Proceedings of the* 56*th Annual Meeting of the Association for Computational Linguistics*, 31-36. https://doi.org/10.18653/v1/P18-2006

[11] Jang, E., Gu, S.X. and Poole, B. (2016) Categorical Reparameterization with Gumbel-Softmax. *International Conference on Learning Representations*. https://openreview.net/pdf?id=rkE3y85ee

[12] Kurakin, A., Goodfellow, I. and Bengio, S. (2016) Adversarial Examples in the Physical World. Chapman and Hall/CRC eBooks, 99-112. https://doi.org/10.1201/9781351251389-8

[13] Chi, P., Chung, P., Wu, T., Hsieh, C., Chen, Y., Li, S. and Lee, H. (2021) Audio Albert: A Lite Bert for Self-Supervised Learning of Audio Representation. 2021 *IEEE Spoken Language Technology Workshop* (*SLT*), Shenzhen, 19-22 January 2021, 344-350. https://doi.org/10.1109/SLT48900.2021.9383575

[14] Li, Z.Y., Xu, J.H., Zeng, J.H., Li, L.Y., Zheng, X.Q., Zhang, Q., Chang, K.W. and Hsieh, C.-J. (2021) Searching for an Effective Defender: Benchmarking Defense against Adversarial Word Substitution. *Proceedings of the* 2021 *Conference on Empirical Methods in Natural Language Processing*, 3137-3147. https://doi.org/10.18653/v1/2021.emnlp-main.251