BERT is Robust! A Case Against Synonym-Based Adversarial Examples in Text Classification

Jens Hauser **Zhao Meng*** **Damián Pascual*** **Roger Wattenhofer**
ETH Zurich, Switzerland
{jehauser, zhmeng, dpascual, wattenhofer}@ethz.ch

Abstract

Deep Neural Networks have taken Natural Language Processing by storm. While this led to incredible improvements across many tasks, it also initiated a new research field, questioning the robustness of these neural networks by attacking them. In this paper, we investigate four word substitution-based attacks on BERT. We combine a human evaluation of individual word substitutions and a probabilistic analysis to show that between 96% and 99% of the analyzed attacks do not preserve semantics, indicating that their success is mainly based on feeding poor data to the model. To further confirm that, we introduce an efficient data augmentation procedure and show that many adversarial examples can be prevented by including data similar to the attacks during training. An additional post-processing step reduces the success rates of state-of-the-art attacks below 5%. Finally, by looking at more reasonable thresholds on constraints for word substitutions, we conclude that BERT is a lot more robust than research on attacks suggests.

1 Introduction

Research in computer vision (Szegedy et al., 2014; Goodfellow et al., 2015) and speech recognition (Carlini and Wagner, 2018) has shown that neural networks are sensitive to changes that are imperceptible to humans. These insights led to extensive research on attacks for creating these so-called adversarial examples, especially in the field of computer vision. Looking for similar issues in NLP is natural, and researchers proposed several different attacks over the last years. However, contrary to computer vision, adversarial examples in NLP are never completely invisible, as discrete characters or words have to be exchanged. This brings up the question: How good are these attacks? Do they reveal issues in current models, or are they just introducing nonsense?

In this paper, we show that despite the general consensus that textual adversarial attacks should preserve semantics, striving for ever-higher success rates seems to be more important when implementing them. We combine a human evaluation with a simple probabilistic analysis to show that between 96% and 99% of the adversarial examples on BERT (Devlin et al., 2019) created by four different attacks do not preserve semantics. Additionally, we propose a two-step procedure consisting of data augmentation and post-processing for defending against adversarial examples[1]. While this sounds contradictive at first, the results show that we can eliminate a large portion of the successful attacks by simply including data similar to the adversarial examples and further detect many of the remaining adversarial examples in a post-processing step. Compared to traditional adversarial training strategies, our method is much more efficient and can be used as a baseline defense for researchers looking into new and better attacks.

2 Related Work

Papernot et al. (2016) was the first to introduce adversarial examples in the text domain. In the following years, a range of different attacks have been proposed. Alzantot et al. (2018) use a population-based optimization algorithm for creating adversarial examples, Zhang et al. (2019) use Metropolis Hastings (Metropolis et al., 1953; Hastings, 1970). Further word substitution based attacks were proposed by Ren et al. (2019); Jin et al. (2020); Li et al. (2020) and Garg and Ramakrishnan (2020). They are discussed in more detail in Section 3.1.

Regarding adversarial defense, some papers introducing attacks incorporate the created adversarial examples during training (Alzantot et al., 2018; Ren et al., 2019). However, due to the high cost of running the attacks, they cannot create suffi-

[*] The two authors contributed equally to this paper.

[1] We will release the code with the official publication of this paper.

ciently many adversarial examples and achieve only minor improvements in robustness. Wang et al. (2021a) suggest Synonym Encoding Method (SEM), a method that uses an encoder that maps clusters of synonyms to the same embedding. Such a method works well but also impedes the expressiveness of the network. Wang et al. (2021b) propose a method for fast adversarial training called Fast Gradient Projection Method (FGPM). However, their method is limited to models with non-contextual word vectors as input. On BERT, Meng et al. (2021) use a geometric attack that allows for creating adversarial examples in parallel and therefore leads to faster adversarial training. Another line of work is around certified robustness through Interval Bound Propagation (Jia et al., 2019; Huang et al., 2019), but these approaches currently do not scale to large models and datasets.

There is little work criticizing or questioning current synonym-based adversarial attacks in NLP, Morris et al. (2020a) find that adversarial attacks often do not preserve semantics using a human evaluation. They propose to increase thresholds on frequently used metrics for the similarity of word embeddings and sentence embeddings. However, they only investigate a single attack on BERT.

3 Background

For a classifier $f : \mathcal{S} \rightarrow \mathcal{Y}$ and some correctly classified input $s \in \mathcal{S}$, an adversarial example is an input $s_{pert} \in \mathcal{S}$, such that $sim(s, s_{pert}) \geq t_{sim}$ and $f(s) \neq f(s_{pert})$, where $sim(s, s_{pert}) \geq t_{sim}$ is a constraint on the similarity of s and s_{pert}. For text classification, $s = \{w^1, w^2, ..., w^n\}$ is a sequence of words. Common notions of similarity are the cosine similarity of counter-fitted word vectors (Mrkšić et al., 2016), which we will denote as $cos_{cv}(w^i, w^i_{pert})$ or the cosine similarity of sentence embeddings from the Universal Sentence Encoder (USE) (Cer et al., 2018), which we will denote as $cos_{use}(s, s_{pert})$. Note that this is a slight abuse of notation since s and s_{pert} are just sequences of words. The notation should be interpreted as follows: We first apply USE to s and s_{pert} to get two sentence vectors and then calculate the cosine similarity. The same holds for $cos_{cv}(w^i, w^i_{pert})$, where we first get the counter-fitted word vectors of w^i and w^i_{pert}. Also, note that whenever we talk about the *cosine similarity of words*, it refers to the cosine similarity of words in the counter-fitted embedding. Similarly, *USE score* refers to the cosine similarity of sentence embeddings from the USE.

3.1 Attacks

We use four different attacks for our experiments. All of them are based on the idea of exchanging words with other words of similar meaning. The attacks differ in the search method for defining the order of words to replace, in the strategy of choosing the candidate set for replacement words, and in the constraints. To better interpret the results of our analysis, we give a brief summary of the four attacks. Particularly, we are interested in how the attacks build the candidate sets for replacement and in what constraints exist.

TextFooler Jin et al. (2020) propose TextFooler, which builds its candidate set from the 50 nearest neighbors in a vector space of counter-fitted word embeddings. The constraints are $cos_{cv}(w^i, w^i_{pert}) \geq 0.5 \ \forall i$ and $cos_{use}(s, s_{pert}) \geq 0.878$[2].

Probability Weighted Word Saliency (PWWS) PWWS (Ren et al., 2019) uses WordNet[3] synonyms to construct a candidate set. It uses no additional constraints.

BERT-Attack Li et al. (2020) suggest an attack based on BERT itself. BERT-Attack uses a BERT masked-language model (MLM) to propose 48 possible replacements. The constraints are $cos_{use}(s, s_{pert}) \geq 0.2$ and a maximum of 40% of all words can be replaced.

BAE Garg and Ramakrishnan (2020) propose another attack based on a BERT MLM. BAE uses the top 50 candidates of the MLM and tries to enforce semantic similarity by requiring $cos_{use}(s, s_{pert}) \geq 0.936$.

An attack is successful for a given input s, if it finds an adversarial example s_{pert} satisfying all constraints. The *attack success rate* is then defined as the number of successful attacks divided by the number of attempted attacks.

4 Setup

We use the BERT-base-uncased model provided by the Hugging Face Transformers (Wolf et al.,

[2] The official value is 0.841 on the angular similarity between sentence embeddings, which corresponds to a cosine similarity of 0.878

[3] https://wordnet.princeton.edu/

2019) for all our experiments and rely on TextAttack (Morris et al., 2020b) for the implementations of the different attacks. We fine-tuned BERT for two epochs on AG News and Yelp[4] and then randomly sampled 1000 examples from each test-set for running the attacks. The clean accuracies of our models are 94.57% on AG News and 97.31% on Yelp. The attack success rates of the different attacks are shown in Table 1.

It is interesting that BAE, which requires a much higher sentence similarity than BERT-Attack, is a lot less effective despite being otherwise similar. But is a high sentence similarity sufficient to ensure semantic similarity? This is part of what we wanted to investigate using a human evaluation.

5 Quality of Adversarial Examples

To investigate the quality of adversarial examples, we conducted a human evaluation on word substitutions performed by the different attacks. In the following, we call such a word substitution a *perturbation*. A probabilistic analysis is then used to generalize the results on perturbations to attacks.

5.1 Human Evaluation

For the human evaluation, we rely on labor crowd-sourced from Amazon Mechanical Turk[5]. We limited our worker pool to workers in the United States and the United Kingdom who completed over 5000 HITs with over 98% success rate. We collected 100 pairs of [*original word*, *attack word*] for every attack and another 100 pairs for every attack where the context is included with a window size of 11. For the word-pairs, inspired by Morris et al. (2020a), we asked the workers to react to the following claim: *"In general, replacing the first word with the second word preserves the meaning of the sentence."* For the words with context, we presented the two text fragments on top of each other, highlighted the changed word, and asked the workers: *"In general, the change preserves the meaning of the text fragment."* In both cases the workers had seven answers to choose from: "Strongly Disagree", "Disagree", "Somewhat Disagree", "Neutral", "Somewhat Agree", "Agree", "Strongly Agree". We convert these answers to a scale from 1-7.

Table 2 shows the results of this human analysis.

Dataset	**Attack Success Rate (%)**			
	TextFooler	PWWS	BERT-Attack	BAE
AG News	84.99	64.95	79.43	14.27
Yelp	90.47	92.23	93.47	31.50

Table 1: Attack success rates of the different attacks on fine-tuned BERT-base-uncased models.

Contrary to what is suggested in papers proposing the attacks, our results show that humans generally tend to disagree that the newly introduced word preserves the meaning. This holds for all attacks and regardless of whether we show the word with or without context. We believe this difference is mainly due to how the text is shown to the judges and what question is posed. For example, asking *"Are these two text documents similar?"* on two long text documents that only differ by a few words is likely to get a higher agreement because the workers will not bother going into the details. Therefore, we believe it is critical to show the passages that are changed.

Regarding the different attacks, it becomes clear from this evaluation that building a candidate set from the first 48 or 50 candidates proposed by a MLM does not work without an additional constraint on the word similarity. The idea of BERT-based attacks is to only propose words that make sense in the context, however, fitting into the context and preserving semantics is not the same thing. The results on BAE further make it clear that a high sentence similarity according to the USE score is no guarantee for semantic similarity. PWWS and TextFooler receive similar scores for word similarity, but the drop in score for PWWS when going from word similarity to text similarity indicates that while the synonyms retrieved from WordNet are often somewhat related to the original word, the relation is often the wrong one for the given context. TextFooler receives the highest scores in this analysis, but even for TextFooler, just 22% and 24% of the perturbations were rated above 5, which corresponds to "Somewhat Agree".

5.2 Probabilistic Estimation of Valid Attacks

The human evaluation is based on individual perturbations. An attack usually changes multiple words and therefore consists of multiple perturbations. This begs the question: How many of the successful attacks are actually valid attacks? To answer this question, we need to define valid attacks and

[4] We restricted ourselves to examples in Yelp which have fewer than 80 words

[5] https://www.mturk.com/

Attack	**Word Similarity**			**Text Similarity**		
	Avg. (1-7)	Above 5 (%)	Above 6 (%)	Avg. (1-7)	Above 5 (%)	Above 6 (%)
TextFooler	**3.88**	**22**	**7**	**3.47**	**24**	**12**
PWWS	3.83	21	6	2.70	13	6
BERT-Attack	2.27	4	4	2.55	7	3
BAE	1.64	0	0	1.85	3	2

Table 2: Average human scores on a scale from 1-7 and the percentage of scores above 5 and 6 (corresponding to the answers “Somewhat Agree” and “Agree”) for the different attacks and when the words were shown with (text similarity) or without (word similarity) context.

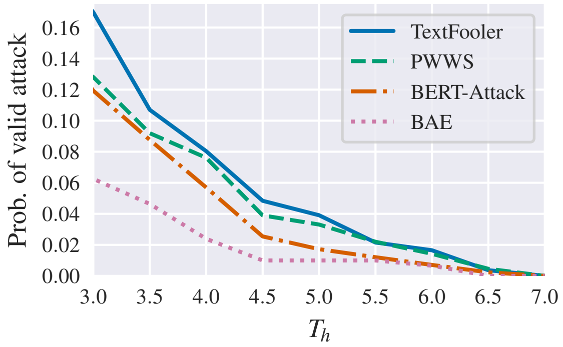


Figure 1: Probability that an attack is valid according to our probabilistic analysis, for the different attacks and for different thresholds T_h.

valid perturbations.

Definition 5.1 (Valid Perturbation)**.** A *valid perturbation* is a perturbation that receives a human score above some threshold T_h.

Definition 5.2 (Valid Attack)**.** A *valid attack* is an attack consisting of valid perturbations only.

Sensible values for T_h are in the range 5-6, which corresponds to “Somewhat Agree” to “Agree”. In order to get an estimate for the percentage of valid attacks, we perform a simple probabilistic analysis. Let A_{val}, P_{val} and A^i_{val} denote the events of a valid attack, a valid perturbation and a valid attack consisting of exactly i perturbations. Further, let $p(i)$ denote the probability that an attack perturbs i words. Using that notation, we can approximate the probability that a successful attack is valid as

$$\begin{aligned} p(A_{val}) &= \sum_{i=1}^{N} p(i)p(A^i_{val}) \\ &\approx \sum_{i=1}^{N} p(i)p(P_{val})^i, \end{aligned} \qquad (1)$$

where N is the maximal number of allowed perturbations. With the data from Amazon Mechanical Turk and the collected adversarial examples, we can get an unbiased estimate for this probability as

$$\hat{p}(A_{val}) = \sum_{i=1}^{N} \hat{p}(i) \left(\frac{\text{count}[S_h \geq T_h]}{n_{pert}} \right)^i, \qquad (2)$$

where S_h is the average score of the workers for a perturbation, n_{pert} is the total number of perturbations analyzed by the workers for any given attack, and $\hat{p}(i)$ can be estimated using counts. The results of this analysis are shown in Figure 1 as a function of the threshold T_h. It can be seen that if we require an average score of 5 for all perturbations, we can expect around 4% of the successful attacks from TextFooler to be valid, slightly less for PWWS, below 2% for BERT-Attack, and just around 1% for BAE. In other words, between 96% and 99% of the successful attacks can not be considered valid according to the widely accepted requirement that adversarial examples should preserve semantics.

This analysis assumes that perturbations are independent of each other, which is not true because every perturbation impacts the following perturbations. Nevertheless, we argue that this approximation tends to result in optimistic estimates on the true number of valid attacks for the following reasons: 1) When an attack is already almost successful, all attacks except for PWWS try to maximize sentence similarity on the last perturbation, making the last perturbation generally weaker. 2) We strongly assume that in a sentence with multiple changes, a human is generally less likely to say that the meaning is preserved, even if the individual perturbations are considered valid.

5.3 Metrics vs. Human

Figure 2 shows the probability that a perturbation is considered valid (for $T_h = 5$) as a function of cosine similarity of words and as a function of USE score. The plots are based on the 400 words with context from the different attacks which were

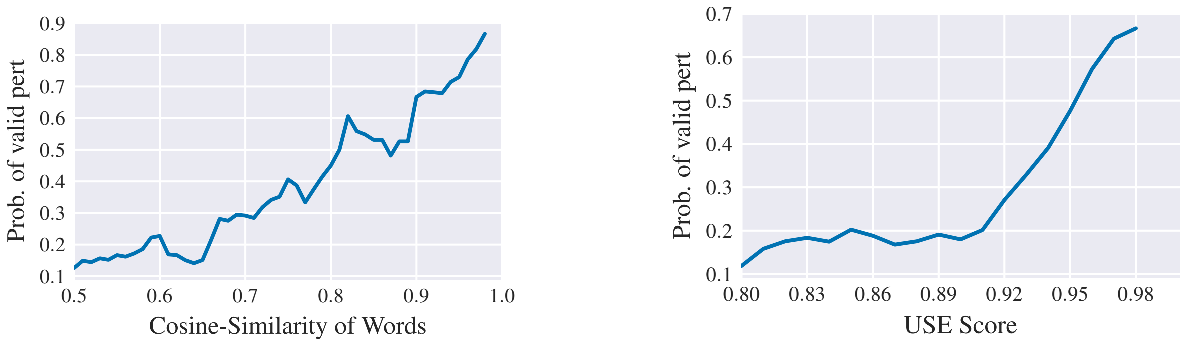


Figure 2: The probability that a perturbation is considered valid by a human, as a function of cosine similarity of words (left) and USE score (right). T_h is set to 5, i.e. an average score of 5 is required to be considered valid.

judged by humans. We use left-aligned buckets of size 0.05, i.e., the probability of a valid perturbation for a given cosine similarity x and metric $m \in \{cos_{cv}(\cdot,\cdot), cos_{use}(\cdot,\cdot)\}$, is estimated as

$$\frac{\text{count}[(S_h \geq T_h) \wedge (m \in [x, x+0.05))]}{\text{count}[m \in [x, x+0.05)]}. \tag{3}$$

It can be observed that there is a strong positive correlation between both metrics and the probability that a perturbation is considered valid, confirming both the validity of such metrics and the quality of our human evaluation. However, the exact probabilities have to be interpreted with care, as the analysis based on one variable does not consider the conditional dependence between the two metrics.

6 Adversarial Defense

We have shown that current attacks use lenient constraints and, therefore, mostly produce adversarial examples that should not be considered valid, but finding suitable thresholds on the constraints is difficult. Morris et al. (2020a) try to find these thresholds by choosing the value where humans "Agree" (on a slightly different scale) on average and find thresholds of 0.90 on the word similarity and 0.98 on the sentence similarity score. However, this misses all the perturbations which were considered valid by the workers at lower scores (see Figure 2). Before discussing other thresholds, we show that we can avoid many adversarial examples even for low thresholds.

Our procedure consists of two steps, where the first step prepares for the second. The first step is a data augmentation procedure and looks as follows:

Step 1

a) Initialize thresholds $t_{rr} \in (0, 100]$ for the maximal percentage of words to augment, and $t_{cv} \in (0, 1)$ for a threshold on cosine similarity of words.

b) During training of the model, for every batch, calculate the gradients to get the t_{rr} percent of most important words for every input. The union of the words considered as stop-words by the four attacks is filtered out.

c) Then, for every word marked as important according to b), a candidate set $\mathcal{C}$ is built out of all words in a counter-fitted embedding with cosine similarity greater than t_{cv}.

d) To account for the fact that all attacks tend to favor words with low cosine similarity (see Appendix D), the replacement $v_i \in \mathcal{C}$ for the original word w is chosen with probability:

$$p(v_i) = \frac{1 - cos_{cv}(w, v_i)}{\sum_{v_j \in \mathcal{C}} 1 - cos_{cv}(w, v_j)}. \tag{4}$$

This skews the probability towards words with lower cosine similarity.

e) Finally, the perturbed batch with the changed words is concatenated to the original batch

The data augmentation procedure makes the model more robust against attack words with cosine similarity greater t_{cv}. If we expect BERT to be robust against these kinds of replacements, this is the least we should do. Otherwise, we cannot expect the model to generalize to the attack's input space, which is significantly larger than the input space during fine-tuning.

We can further improve the robustness with a post-processing step that builds on this robustness to random substitutions.

Dataset	Method	Clean Acc. (%)	Attack Success Rate (%)			
			TextFooler	PWWS$_{cv50}$	BERT-Attack$_{cv50}$	BAE$_{cv50}$
AG News	Normal	94.57	84.99	16.38	20.72	0.32
	DA	**94.82**	52.37	10.73	18.61	–
	DA+PP	93.84 ± 0.07	**3.93** ± 0.41	**2.55** ± 0.31	**3.73** ± 0.29	–
	DA+MA$_5$	93.72 ± 0.12	14.11 ± 0.48	4.61 ± 0.41	7.52 ± 0.48	–
	Normal+PP	87.89 ± 0.16	10.32 ± 0.48	5.0 ± 0.31	5.59 ± 0.36	–
Yelp	Normal	**97.31**	90.47	33.26	49.53	0.41
	DA	97.10	29.79	10.52	16.49	–
	DA+PP	96.59 ± 0.06	**4.37** ± 0.39	**2.54** ± 0.15	**4.86** ± 0.33	–
	DA+MA$_5$	95.40 ± 0.10	10.23 ± 0.59	4.62 ± 0.36	7.38 ± 0.38	–
	Normal+PP	94.50 ± 0.08	6.07 ± 0.47	5.22 ± 0.48	7.35 ± 0.61	–

Table 3: Effectiveness of defense procedure for different attacks modified with constraint on cosine-similarity of words.

Step 2

a) For every text that should be classified, N versions are created where $t_{rr}\%$ of the words (which are not stop-words) are selected uniformly at random and are exchanged by another uniformly sampled word from a candidate set $\mathcal{C}$ consisting of all words with cosine-similarity above t_{cv}.

b) The outputs of the model (logits) are added up for the N versions and the final prediction is made according to the maximum value. Formally, let $l_j(s)$ denote the value of the j-th logit for some input s. Then the prediction y_{pred} is made according to

PREDICTION OF CLASS j

$$y_{pred} = \arg\max_j \sum_{i=1}^{N} l_j(s_i). \tag{5}$$

This procedure can be applied for any threshold $t_{cv} \in (0, 1)$, but it only makes sense if we expect an attack to use the same or a higher threshold. We always set t_{cv} to the same value as the attack uses. Further, we set $t_{rr} = 40$ and $N = 8$ in all our experiments, and we use the same thresholds for both steps.

7 Defense Results

In Table 3, we show the effect of the procedure on the different attacks modified with the constraint that the cosine-similarity between original word and attack word should be above 0.5. The notation is the following: Normal stands for a model fine-tuned normally. DA stands for a model fine-tuned with data augmentation, and PP stands for post-processing. MA$_5$ is a baseline for our post-processing procedure that replaces 5% of all tokens with the [MASK] token (see Appendix B). The results show that up to two-thirds of the attacks can be prevented using data augmentation. This indicates that adversarial examples for text classification are closely related to the data on which the model is fine-tuned. The attacks try to create examples that are out-of-distribution with respect to the training data. Additionally, between 70% and 92% of the attacks can be reverted using our post-processing procedure, resulting in attack success rates below 5% for all attacks. For TextFooler, this corresponds to a decrease in attack success rate of more than 95%. Because the post-processing step is probabilistic, we ran it ten times for every combination of dataset and attack. We show the mean and standard deviation of the ten resulting attack success rates. Compared to the mask-baseline, our post-processing procedure can revert significantly more attacks while having a smaller impact on the clean accuracy. Table 3 also shows that the post-processing step should always be preceded by data augmentation. While applying post-processing in isolation still reverts many attacks, the clean accuracy drops significantly, especially on AG News.

7.1 Adjusted Thresholds

Table 3 shows that with the constraint on cosine similarity of words added, TextFooler is by far the most effective attack, at least before post-processing. There is a simple reason for this, TextFooler already has that constraint and is the only attack out of the four to choose its candidate set directly from the counter-fitted embedding used to calculate the cosine similarity. On the other end of the spectrum, BAE's attacks success rate drops close to zero. This is because the intersection of the set of words proposed by the MLM, the set

Dataset	**Method**	**Attack Success Rate (%)**				
		TF_{cv50}	TF^{use88}_{cv50}	TF^{use85}_{cv70}	TF^{use90}_{cv70}	TF^{use90}_{cv80}
AG News	Normal	88.79	24.95	22.52	11.63	7.51
	DA	55.58	16.11	10.79	7.12	4.50
	DA+PP	4.49 ± 0.39	3.31 ± 0.28	2.07 ± 0.16	1.91 ± 0.17	0.99 ± 0.17
Yelp	Normal	91.40	49.22	42.59	25.18	11.09
	DA	38.46	13.74	10.34	7.78	2.87
	DA+PP	5.04 ± 0.35	3.9 ± 0.34	2.12 ± 0.21	2.28 ± 0.17	0.71 ± 0.13

Table 4: Effectiveness of defense procedure for different combinations of thresholds.

of words with cosine similarity greater than 0.5, and the set of words keeping the USE score above 0.936 is small and leaves the attack not much room. A similar observation can be made for PWWS, although not as pronounced.

However, there is one more reason why TextFooler is more effective compared to the other attacks, despite an additional constraint on the USE score. While attacking a piece of text, this constraint on the USE score is not checked between the current perturbed text s_{pert} and the original text s, but instead between the current perturbed text s_{pert} and the previous version s'_{pert}. This means that by perturbing one word at a time, the effective USE score between s and s_{pert} can be a lot lower than the threshold suggests. When discussing the effect of raising thresholds to higher levels, we do so by relying on TextFooler as the underlying attack because it is the most effective, but we adjust the constraint on the USE score to always compare to the original text. We believe this is the right way to implement this constraint, and more importantly, it is consistent with how we gathered data from Amazon Mechanical Turk.

Table 4 shows the results from our defense procedure when the thresholds on TextFooler are adjusted. TF_{cv50} corresponds to TextFooler without the constraint on the USE score. Comparing with Table 3 confirms that the original implementation of the USE constraint only had a small impact. TF^{use88}_{cv50} corresponds to TextFooler with $cos_{cv}(w^i, w^i_{pert}) \geq 0.5 \ \forall i$ and $cos_{use}(s, s_{pert}) \geq 0.88$ (0.878 to be precise), the same thresholds as in the original implementation, but without allowing to drift away from the original text as discussed above. This already decreases the attack success rate significantly. Using data augmentation, we can decrease the attack success rate by more than a factor of 5 compared to what we saw originally (84.99 to 16.11 and 90.47 to 13.74). This shows that by preventing TextFooler from using that little trick and some data augmentation, we can decrease the attack success rate to values far from the ones suggested in their paper. When increasing the thresholds on the constraints (compare to Figure 2 to see that these are still not particularly strong constraints), it becomes even more evident that BERT is a lot more robust than work on attacks suggests. Especially if we allow for post-processing.

7.2 Comparing data augmentation with adversarial training

While adversarial training provides the model with data from the true distribution generated by an attack, our data augmentation procedure only approximates that distribution. The goal is to trade robustness for speed. However, it turns out that our procedure can even be superior to true adversarial training in some cases. We compare to two different strategies for adversarial training. ADV_{naive} denotes the simplest procedure for adversarial training in text classification: collect adversarial examples on the training set and then train a new model on the extended dataset consisting of both adversarial examples and original training data. We used TextFooler to collect these adversarial examples. On the complete training set, this resulted in 103’026 adversarial examples on AG News and 179’335 adversarial examples on Yelp. For a more sophisticated version for adversarial training, we follow Meng et al. (2021) by creating adversarial examples on-the-fly during training. We denote this method as ADV (corresponds to ADV in their paper).

A comparison of the results on AG News and Yelp is shown in Table 5. Interestingly, ADV_{naive} did not result in an improvement on Yelp. We hypothesize that this is because Yelp is easier to attack, resulting in weaker training data for the extended dataset. For example, 26% of the created adversarial examples on Yelp differ by only one or two words from the original text, on AG News

Dataset	Method	Clean Acc. (%)	Training Time (h:min)	Epochs	Attack Success Rate (%)		
					TextFooler	PWWS$_{cv50}$	BERT-Attack$_{cv50}$
AG News	Normal	94.57	0:19	2	84.99	16.38	20.72
	DA	**94.82**	5:33	12	52.37	10.73	18.61
	ADV	92.83	160:15	12	**34.54**	**6.50**	**9.38**
	ADV$_{naive}$	94.26	45:14	2	56.20	12.50	17.44
Yelp	Normal	**97.31**	0:32	2	90.47	33.26	49.53
	DA	97.10	9:08	12	**29.79**	**10.52**	**16.49**
	ADV	95.94	107:56	5	59.52	14.64	25.52
	ADV$_{naive}$	96.65	56:53	2	95.12	33.09	47.61

Table 5: Comparison of data augmentation and adversarial training.

this holds for just 11% of the adversarial examples. Furthermore, the average word replace rate on Yelp is 16% compared to 24% on AG News. The same argument would also explain why, quite surprisingly, we reach higher robustness on Yelp with our data augmentation procedure compared to ADV. To be fair, it must be said that we did not train ADV until convergence on Yelp due to computational constraints. Overall, lower computation time is precisely the biggest advantage of our method. Considering that the training data increases by a factor of two, the overhead per epoch is only around 50% compared to normal training.

8 Limitations

In practice, the post-processing step cannot be decoupled from a black-box attack. It would be interesting to see how successful an attack is when the whole system, including post-processing, is regarded as a single black-box model. We hypothesize that it would remain challenging because the attacker can rely much less on its search method for finding the right words to replace.

The method is also not applicable if a deterministic answer is required. However, in many applications such as spam filters or fake news detection, we are only interested in making a correct decision as often as possible while being robust to a potential attack.

9 Discussion & Conclusion

Using a human evaluation, we have shown that most perturbations introduced through adversarial attacks do not preserve semantics. This is contrary to what is generally claimed in papers introducing these attacks. We believe the main reason for this discrepancy is that researchers working on attacks have not paid enough attention to preserving semantics because attacks with new state-of-the-art success rates are easier to publish. However, in order to find meaningful adversarial examples that could help us better understand current models, we need to get away from that line of thinking. For example, 10-20% attack success rate with valid adversarial examples and a good analysis on them is much more valuable than 80-90% attack success rate by introducing nonsensical words. We hope this work encourages researchers to think more carefully about appropriate perturbations to text which do not change semantics.

Our results on data augmentation show that a significant amount of adversarial examples can be prevented when including perturbations during training that could stem from an attack. It is debatable whether changing 40% of the words with a randomly chosen word from a candidate set still constitutes a valid input, but this is only necessary because the attacks have that amount of freedom. The more appropriate the allowed perturbations for an attack, the more appropriate is our data augmentation procedure, which can easily be adapted for other candidate sets. Compared to adversarial training, our method scales to large datasets and multiple epochs of training, making it an excellent baseline defense method for researchers working on new attacks and defenses. The post-processing step completes our defense procedure and shows that attacks can largely be prevented in a probabilistic setting without a severe impact on the clean accuracy. In practice, this means that most attacks can at least be detected. Whether or not this two-step procedure will prevent the same amount of attacks when the whole model is considered a probabilistic black-box is up for future investigation.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *EMNLP*.

Nicholas Carlini and David Wagner. 2018. Audio adversarial examples: Targeted attacks on speech-to-text. In *IEEE Security and Privacy Workshops (SPW)*.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for english. In *EMNLP*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Siddhant Garg and Goutham Ramakrishnan. 2020. Bae: Bert-based adversarial examples for text classification. In *EMNLP*.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *ICLR*.

W Keith Hastings. 1970. Monte carlo sampling methods using markov chains and their applications.

Po-Sen Huang, Robert Stanforth, Johannes Welbl, Chris Dyer, Dani Yogatama, Sven Gowal, Krishnamurthy Dvijotham, and Pushmeet Kohli. 2019. Achieving verified robustness to symbol substitutions via interval bound propagation. In *EMNLP-IJCNLP*.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. In *EMNLP-IJCNLP*.

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 *AAAI*.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. Bert-attack: Adversarial attack against bert using bert. In *EMNLP*.

Zhao Meng, Yihan Dong, Mrinmaya Sachan, and Roger Wattenhofer. 2021. Self-supervised contrastive learning with adversarial perturbations for robust pretrained language models. *arXiv preprint arXiv:2107.07610*.

Zhao Meng and Roger Wattenhofer. 2020. A geometry-inspired attack for generating natural language adversarial examples. In *COLING*.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. 1953. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092.

John Morris, Eli Lifland, Jack Lanchantin, Yangfeng Ji, and Yanjun Qi. 2020a. Reevaluating adversarial examples in natural language. In *EMNLP Findings*.

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 *EMNLP System Demonstrations*.

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 *NAACL-HLT*.

Nicolas Papernot, Patrick McDaniel, Ananthram Swami, and Richard Harang. 2016. Crafting adversarial input sequences for recurrent neural networks. In *MILCOM IEEE Military Communications Conference*.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *ACL*.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *ICLR*.

Xiaosen Wang, Yichen Yang, Yihe Deng, and Kun He. 2021a. Adversarial training with fast gradient projection method against synonym substitution based text attacks. In *UAI*.

Xiaosen Wang, Yichen Yang, Yihe Deng, and Kun He. 2021b. Adversarial training with fast gradient projection method against synonym substitution based text attacks. In *AAAI*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Huangzhao Zhang, Hao Zhou, Ning Miao, and Lei Li. 2019. Generating fluent adversarial examples for natural languages. In *ACL*.

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

Dataset	N	Reverted Attacks (Mean/Std) (%)		
		TextFooler	PWWS$_{cv50}$	BERT-Att$_{cv50}$
AG News	4	92.13 / 0.65	75.39 / 3.35	78.7 / 1.94
	8	92.49 / 0.79	76.27 / 2.87	79.94 / 1.54
	16	92.81 / 0.53	78.24 / 1.95	80.17 / 0.85
	32	92.97 / 0.24	76.57 / 1.61	81.07 / 0.88
Yelp	4	83.94 / 1.49	74.31 / 3.28	68.56 / 3.02
	8	85.33 / 1.32	75.88 / 1.4	70.5 / 1.97
	16	85.81 / 1.26	76.37 / 1.88	70.81 / 1.12
	32	86.26 / 0.74	76.96 / 0.79	71.31 / 2.16

Table 6: Effectiveness of post-processing for different number of versions.

Dataset	Method	Clean Acc. (%)	Reverted (%)
AG News	MA$_5$	93.62	63.24
	MA$_{10}$	92.14	62.76
	MA$_{20}$	87.30	57.34
	MA$_{30}$	76.25	50.01
Yelp	MA$_5$	95.19	59.0
	MA$_{10}$	93.98	61.42
	MA$_{20}$	90.53	60.83
	MA$_{30}$	86.91	59.25

Table 7: By masking random tokens instead of exchanging words, more than half of the attacks can be reverted. However, the clean accuracy drops.

A Number of versions in post-processing

In order to understand the impact of the number of versions N created during the post-processing step, we can make the following analysis: Let us consider the augmented inputs as instances of a discrete random variable X. For $x \in X$ and a classification problem with K classes, let $l_{correct}(x)$ denote the value of the logit corresponding to the correct label and $l_j(x)$ denote the value of the j-th logit corresponding to a wrong label, such that $j \in \{1, ..., K-1\}$. We are only interested in the differences $g_j(x) = l_{correct}(x) - l_j(x)$. Ideally, we would like to make a decision based on the expectations of $g_j(X)$. An attack should be reverted if and only if

$$\mathrm{E}[g_j(X)] = \sum_{x \in X} g_j(x) p_X(x) \geq 0 \quad \forall j, \qquad (6)$$

where $p_X(x) = \frac{1}{|X|}$. Because we cannot enumerate over all instances x, we approximate this with sums over just N instances

$$\sum_{i=1}^{N} \frac{g_j(x_i)}{N} \geq 0 \quad \forall j. \qquad (7)$$

These are unbiased estimates of the expectations in (6) for any choice of N. By multiplying with N and plugging in the definition of $g_j(x)$, it can be verified that a decision based on (7) reverts the same attacks as a decision based on (5). The expectation estimates become more and more accurate as we increase N. Since we are making a discrete decision based on whether the expectations are ≥ 0, the estimate is more likely to be correct with more samples. If we assume that the true expectation is positive in most cases, this means we can generally expect a higher number of reverted attacks for higher N. Being more precise on the estimate also means we generally tend to make the same decision every time on the same example, therefore reducing the variance in the reverted attack rate. Table 6 shows results on reverted attacks for 4, 8, 16 and 32 versions and generally confirms this. However, the results are already quite good with just four versions, so this is a trade-off between speed and accuracy, as creating N versions increases the batch size during inference by a factor N.

B Baseline for post-processing

Instead of replacing words with other words in Step 2 of our defense procedure, one could also think of other ways of slightly perturbing the adversarial examples to flip the label back to the correct one. To show that our method is superior to such simple perturbations, Table 7 shows the results of a baseline procedure in which we replace randomly chosen words with the `[MASK]` token. Indeed, averaged over TextFooler, PWWS, and BERT-Attack, up to 63% of the adversarial examples on AG News can be reverted by masking just 5% of the words. However, further improving on that by masking more tokens fails, and the clean accuracy drops substantially. This is contrary to our procedure, in which we exchange 40% of the words with just a minimal decrease in accuracy.

C Word Frequencies

We observe that attacks frequently introduce words that rarely occur during training. Table 8 shows median word occurrences (Occ. column) of original words and attack words in the training set for different attacks. The results are quite striking and a further justification for using data augmentation. It is also interesting to see that BERT-Attack acts differently in that regard. We assume this is because BERT-Attack has the weakest constraints (no

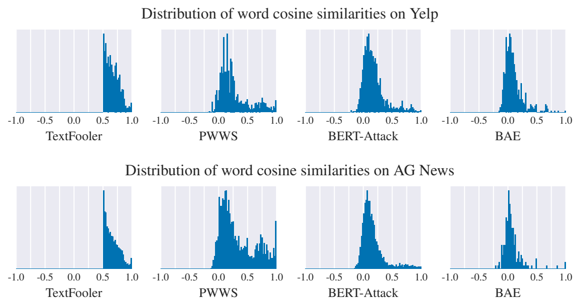


Figure 3: Distribution of cosine similarities of words.

constraint on cosine similarity of words and a weak constraint on USE). This could allow BERT-Attack to find more effective perturbations than other attacks that have to choose from a set of more similar words and then rely on the ones the model does not know.

Table 8 further shows that attacks often use words with higher relative frequency in other classes. Column GT reveals the percentage of times that the original words and attack words have the highest relative frequency (word occurrences in class divided by the total number of words in the same class) in the ground truth class. It can be observed that attacks often introduce words with higher relative frequency in a different class. This is an interesting observation as no one would be surprised by the success of such perturbations if we were dealing with a bag-of-words model.

D Cosine Similarities of Words

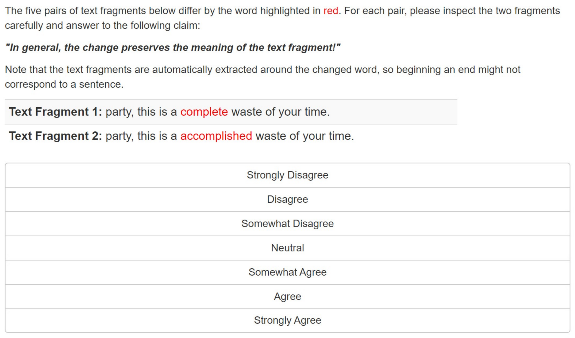
In a counter-fitted embedding, perfect synonyms are supposed to have a cosine similarity of 1 and perfect antonyms are supposed to have a cosine similarity of 0. Figure 3 shows the distribution of cosine similarities for the four attacks on both datasets.

E Details on Human Evaluation

We relied on workers with at least 5000 HITs and over 98% success rate. For the word-pairs, we showed the workers 100 pairs of words in a google form. In order to ensure a good quality of work, we included some hand-designed test cases at several places and rejected workers with strange answers on these word-pairs. These test cases were [*good*, *bad*], [*help*, *hindrance*] (expected answer “Strongly Disagree” or “Disagree”) and [*sofa*, *couch*], [*seldom*, *rarely*] (expected answer “Strongly Agree” or “Agree”). In a first test run, surprisingly, many workers agreed on antonyms like good and bad, which is why we added a note with an example and emphasized that this is about whether the meaning is preserved and not about whether both words fit into the same context. Workers were paid 2.0$ for one HIT with 100 pairs and 4 test cases. We showed every pair of words to ten workers and calculated the mean. A screenshot of the form can be found in Figure 5. For the words with context, we used the amazon internal form because it allowed for a clearer presentation of the two text fragments (see Figure 4). We always presented five pairs of text fragments in one HIT and rejected workers that submitted the hit within less than 60s to ensure quality. Workers were paid 0.5$ for one HIT with five pairs. We showed every pair of text fragments to five workers and calculated the mean.

Dataset	**Attack**	**Orig. Word**		**Att. Word**	
		Occ.	GT (%)	Occ.	GT (%)
AG News	TextFooler	736	67.31	18	24.63
	PWWS	889	60.04	24	16.06
	BERT-Att.	585	65.92	344	22.91
	BAE	617	52.66	4	9.31
Yelp	TextFooler	4240	72.79	19	44.60
	PWWS	5715	74.56	13	33.76
	BERT-Att.	4521	75.27	3398	35.55
	BAE	4601	76.03	44	41.87

Table 8: Median word occurrences of original words and attack words in training set (Occ.) and percentage of times that words have the highest relative frequency in ground truth class (GT).


The five pairs of text fragments below differ by the word highlighted in red. For each pair, please inspect the two fragments carefully and answer to the following claim:

"In general, the change preserves the meaning of the text fragment!"

Note that the text fragments are automatically extracted around the changed word, so beginning an end might not correspond to a sentence.

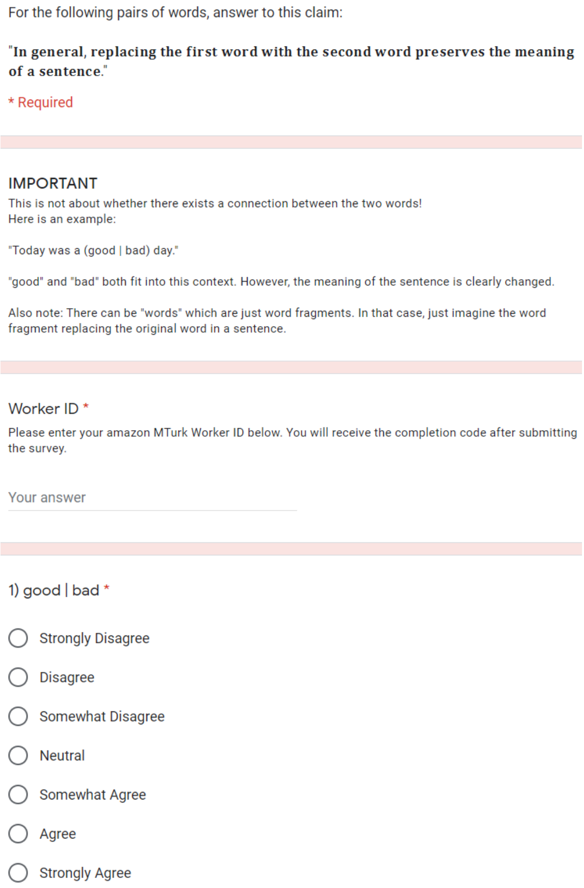
Text Fragment 1: party, this is a complete waste of your time.

Text Fragment 2: party, this is a accomplished waste of your time.

Figure 4: Screenshot of the human evaluation used to evaluate words with context.

F Datasets

For our experiments, we use two different text classification datasets: AG News and Yelp. On Yelp, we only used the examples consisting of 80 words or less. Especially comparing to ADV would have



For the following pairs of words, answer to this claim:

"**In general, replacing the first word with the second word preserves the meaning of a sentence.**"

* Required

IMPORTANT
This is not about whether there exists a connection between the two words!
Here is an example:

"Today was a (good | bad) day."

"good" and "bad" both fit into this context. However, the meaning of the sentence is clearly changed.

Also note: There can be "words" which are just word fragments. In that case, just imagine the word fragment replacing the original word in a sentence.

Worker ID *
Please enter your amazon MTurk Worker ID below. You will receive the completion code after submitting the survey.

Your answer

1) good | bad *

○ Strongly Disagree

○ Disagree

○ Somewhat Disagree

○ Neutral

○ Somewhat Agree

○ Agree

○ Strongly Agree

Figure 5: Screenshot of the Google form used to evaluate similarity of words.

been much harder otherwise. Statistics of the two datasets are displayed in Table 9.

Dataset	**Labels**	**Train**	**Test**	**Avg Len**
AG News	4	120'000	7'600	43.93
Yelp	2	199'237	13'548	45.69

Table 9: Statistics of the two datasets.

AG News (Zhang et al., 2015) is a topic classification dataset. It is contructed out of titles and headers from news articles categorized into the four classes World, Sports, Business, and Sci/Tech.
Yelp (Zhang et al., 2015) is a binary sentiment classification dataset. It contains reviews from Yelp, reviews with one or two stars are considered negative, reviews with 3 or 4 stars are considered positive.

G Implementation

Training We use bert-base-uncased from huggingface[6] for all our experiments. The normal models were fine-tuned for two epochs with a learning rate of 2e-5. We restrict the maximum input length to 128 tokens. For the training with data-augmentation, we train for 12 epochs with a starting learning rate of 2e-5 and linear schedule. We evaluate the robustness on an additional held-out dataset after every epoch. For a threshold of 0.5 on the cosine similarity of words, the robustness reaches its peak after the last epoch. However, we find that two or three epochs are already enough for larger thresholds on cosine similarity of words. All our experiments are conducted on a single RTX 3090.

Attacks We use TextAttack[7] for the implementations of all attacks, including the ones with adjusted thresholds. For adversarial training, we adapt the code from Meng and Wattenhofer (2020).

[6] https://huggingface.co/transformers/

[7] https://textattack.readthedocs.io/en/latest/