# BAE: BERT-based Adversarial Examples for Text Classification

**Siddhant Garg**[*] [†]
Amazon Alexa AI Search
Manhattan Beach, CA, USA
sidgarg@amazon.com

**Goutham Ramakrishnan**[*] [†]
Health at Scale Corporation
San Jose, CA, USA
gouthamr@cs.wisc.edu

## Abstract

Modern text classification models are susceptible to adversarial examples, perturbed versions of the original text indiscernible by humans which get misclassified by the model. Recent works in NLP use rule-based synonym replacement strategies to generate adversarial examples. These strategies can lead to out-of-context and unnaturally complex token replacements, which are easily identifiable by humans. We present BAE, a black box attack for generating adversarial examples using contextual perturbations from a BERT masked language model. BAE replaces and inserts tokens in the original text by masking a portion of the text and leveraging the BERT-MLM to generate alternatives for the masked tokens. Through automatic and human evaluations, we show that BAE performs a stronger attack, in addition to generating adversarial examples with improved grammaticality and semantic coherence as compared to prior work.

## 1 Introduction

Recent studies have exposed the vulnerability of ML models to adversarial attacks, small input perturbations which lead to misclassification by the model. Adversarial example generation in NLP (Zhang et al., 2019) is more challenging than in commonly studied computer vision tasks (Szegedy et al., 2014; Kurakin et al., 2017; Papernot et al., 2017) because of (i) the discrete nature of the input space and (ii) the need to ensure semantic coherence with the original text. A major bottleneck in applying gradient based (Goodfellow et al., 2015) or generator model (Zhao et al., 2018) based approaches to generate adversarial examples in NLP is the backward propagation of the perturbations from the continuous embedding space to the discrete token space.

[*] Equal contribution by authors
[†] Work completed as a graduate student at UW-Madison

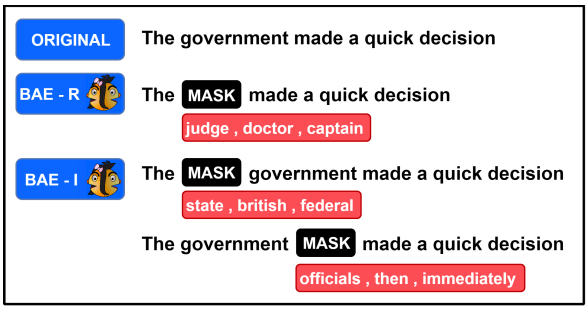



Figure 1: We use BERT-MLM to predict masked tokens in the text for generating adversarial examples. The MASK token replaces a word (BAE−R attack) or is inserted to the left/right of the word (BAE−I).

Initial works for attacking text models relied on introducing errors at the character level (Ebrahimi et al., 2018; Gao et al., 2018) or adding and deleting words (Li et al., 2016; Liang et al., 2017; Feng et al., 2018) for creating adversarial examples. These techniques often result in unnatural looking adversarial examples which lack grammatical correctness, thereby being easily identifiable by humans.

Rule-based synonym replacement strategies (Alzantot et al., 2018; Ren et al., 2019) have recently lead to *more* natural looking adversarial examples. Jin et al. (2019) combine both these works by proposing TextFooler, a strong black-box attack baseline for text classification models. However, the adversarial examples generated by TextFooler solely account for the token level similarity via word embeddings, and not the overall sentence semantics. This can lead to *out-of-context* and *unnaturally complex* replacements (see Table 3), which are easily human-identifiable. Consider a simple example: "The restaurant service was *poor*". Token level synonym replacement of 'poor' may lead to an inappropriate choice such as 'broke', while a context-aware choice such as 'terrible' leads to better retention of semantics and grammaticality.

Therefore, a token replacement strategy contingent on retaining sentence semantics using a pow-



*Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 6174–6181,
November 16–20, 2020. ©2020 Association for Computational Linguistics

erful language model (Devlin et al., 2018; Radford et al., 2019) can alleviate the errors made by existing techniques for homonyms (tokens having multiple meanings). In this paper, we present BAE (BERT-based Adversarial Examples), a novel technique using the BERT masked language model (MLM) for word replacements to better fit the overall context of the English language. In addition to replacing words, we also propose inserting new tokens in the sentence to improve the attack strength of BAE. These perturbations in the input sentence are achieved by masking a part of the input and using a LM to fill in the mask (See Figure 1).

Our BAE attack beats the previous baselines by a large margin on empirical evaluation over multiple datasets and models. We show that, surprisingly, just a few replace/insert operations can reduce the accuracy of even a powerful BERT classifier by over $80\%$ on some datasets. Moreover, our human evaluation reveals the improved grammaticality of the adversarial examples generated by BAE over the baseline TextFooler, which can be attributed to the BERT-MLM. To the best of our knowledge, we are the first to use a LM for generating adversarial examples. We summarize our contributions as:

- We propose BAE, an adversarial example generation technique using the BERT-MLM.
- We introduce 4 BAE attack modes by replacing and inserting tokens, all of which are almost always stronger than previous baselines on 7 text classification datasets.
- Through human evaluation, we show that BAE yields adversarial examples with improved grammaticality and semantic coherence.

## 2 Methodology

**Problem Definition.** We are given a dataset $(S, Y) = \{(\mathbb{S}_1, y_1), \ldots (\mathbb{S}_m, y_m)\}$ and a trained classification model $C : \mathbb{S} \rightarrow Y$. We assume the soft-label black-box setting where the attacker can only query the classifier for output probabilities on a given input, and does not have access to the model parameters, gradients or training data. For an input pair $(\mathbb{S}{=}[t_1, \ldots, t_n], y)$, we want to generate an adversarial example $\mathbb{S}_{adv}$ such that $C(\mathbb{S}_{adv}){\neq}y$. Additionally we would like $\mathbb{S}_{adv}$ to be grammatically correct and semantically similar to $\mathbb{S}$.

**BAE.** For generating an adversarial example $\mathbb{S}_{adv}$, we introduce 2 types of token-level perturbations: (i) Replace a token $t \in \mathbb{S}$ with another and (ii) Insert a new token $t'$ in $\mathbb{S}$. Some tokens in the input contribute more towards the final prediction by $C$ than others. Replacing these tokens or inserting a new token adjacent to them can thus have a stronger effect on altering the classifier prediction. This intuition stems from the fact that the replaced/inserted tokens changes the local context around the original token. We estimate token importance $I_i$ of each $t_i \in \mathbb{S}$, by deleting $t_i$ from $\mathbb{S}$ and computing the decrease in probability of predicting the correct label $y$, similar to Jin et al. (2019); Ren et al. (2019).

**Algorithm 1**: BAE-R Pseudocode

**Input:** Sentence $\mathbb{S} = [t_1, \ldots, t_n]$, ground truth label $y$, classifier model C
**Output:** Adversarial Example $\mathbb{S}_{adv}$
**Initialization:** $\mathbb{S}_{adv} \leftarrow \mathbb{S}$
Compute token importance $I_i \; \forall \; t_i \in \mathbb{S}$
**for** $i$ *in descending order of* $I_i$ **do**
  $\mathbb{S}_M \leftarrow \mathbb{S}_{adv[1:i-1]}[M]\mathbb{S}_{adv[i+1:n]}$
  Predict top-K tokens $\mathbb{T}$ for mask M $\in \mathbb{S}_M$
  $\mathbb{T} \leftarrow$ FILTER($\mathbb{T}$)
  $\mathbb{L} = \{\}$ `// python-style dict`
  **for** $t \in \mathbb{T}$ **do**
    $\mathbb{L}[t] = \mathbb{S}_{adv[1:i-1]}[t]\mathbb{S}_{adv[i+1:n]}$
  **end**
  **if** $\exists \; t \in \mathbb{T}$ s.t $C(\mathbb{L}[t]) \neq y$ **then**
    **Return:** $\mathbb{S}_{adv} \leftarrow \mathbb{L}[t']$ where $C(\mathbb{L}[t']) \neq y$, $\mathbb{L}[t']$ has maximum similarity with $\mathbb{S}$
  **else**
    $\mathbb{S}_{adv} \leftarrow \mathbb{L}[t']$ where $\mathbb{L}[t']$ causes maximum reduction in probability of $y$ in $C(\mathbb{L}[t'])$
  **end if**
**end**
**Return:** $\mathbb{S}_{adv} \leftarrow None$

The Replace (R) and Insert (I) operations are performed on a token $t$ by masking it and inserting a mask token adjacent to it respectively. The pre-trained BERT-MLM is used to predict the mask tokens (See Figure 1). BERT-MLM is a powerful LM trained on a large training corpus ($\sim$ 2 billion words), and hence the predicted mask tokens fit well into the grammar and context of the text.

The BERT-MLM, however, does not guarantee semantic coherence to the original text as demonstrated by the following simple example. Consider the sentence: 'the food was good'. For replacing the token 'good', BERT-MLM may predict the token 'bad', which fits well into the grammar and context of the sentence, but changes the original sentiment of the sentence. To achieve a high semantic similarity with the original text on introducing perturbations, we filter the set of top K tokens (K is a pre-defined constant) predicted by BERT-MLM for the masked token, using a Universal Sentence Encoder (USE) based sentence similarity scorer (Cer et al., 2018). For the R operation, we additionally filter out predicted tokens that do not form the same part of speech (POS) as the original token.

| Model | Adversarial Attack | Datasets | | | |
|---|---|---|---|---|---|
| | | Amazon | Yelp | IMDB | MR |
| **wordLSTM** | Original | 88.0 | 85.0 | 82.0 | 81.16 |
| | TextFooler | 31.0 (0.747) | 28.0 (0.829) | 20.0 (0.828) | 25.49 (0.906) |
| | BAE−R | 21.0 (0.827) | 20.0 (0.885) | 22.0 (0.852) | 24.17 (0.914) |
| | BAE−I | 17.0 (0.924) | 22.0 (0.928) | 23.0 (0.933) | 19.11 (0.966) |
| | BAE−R/I | 16.0 (0.902) | 19.0 (0.924) | 8.0 (0.896) | 15.08 (0.949) |
| | BAE−R+I | **4.0 (0.848)** | **9.0 (0.902)** | **5.0 (0.871)** | **7.50 (0.935)** |
| **wordCNN** | Original | 82.0 | 85.0 | 81.0 | 76.66 |
| | TextFooler | 42.0 (0.776) | 36.0 (0.827) | 31.0 (0.854) | 21.18 (0.910) |
| | BAE−R | 16.0 (0.821) | 23.0 (0.846) | 23.0 (0.856) | 20.81 (0.920) |
| | BAE−I | 18.0 (0.934) | 26.0 (0.941) | 29.0 (0.924) | 19.49 (0.971) |
| | BAE−R/I | 13.0 (0.904) | 17.0 (0.916) | 20.0 (0.892) | 15.56 (0.956) |
| | BAE−R+I | **2.0 (0.859)** | **9.0 (0.891)** | **14.0 (0.861)** | **7.87 (0.938)** |
| **BERT** | Original | 96.0 | 95.0 | 85.0 | 85.28 |
| | TextFooler | 30.0 (0.787) | 27.0 (0.833) | 32.0 (0.877) | 30.74 (0.902) |
| | BAE−R | 36.0 (0.772) | 31.0 (0.856) | 46.0 (0.835) | 44.05 (0.871) |
| | BAE−I | 20.0 (0.922) | 25.0 (0.936) | 31.0 (0.929) | 32.05 (0.958) |
| | BAE−R/I | **11.0 (0.899)** | 16.0 (0.916) | 22.0 (0.909) | 20.34 (0.941) |
| | BAE−R+I | 14.0 (0.830) | **12.0 (0.871)** | **16.0 (0.856)** | **19.21 (0.917)** |

Table 1: Automatic evaluation of adversarial attacks on 4 Sentiment Classification tasks. We report the test set accuracy. The average semantic similarity, between the original and adversarial examples, obtained from USE are reported in parentheses. Best performance, in terms of maximum drop in test accuracy, is highlighted in **boldface**.

If multiple tokens can cause $C$ to misclassify $\mathbb{S}$ when they replace the mask, we choose the token which makes $\mathbb{S}_{adv}$ most similar to the original $\mathbb{S}$ based on the USE score. If no token causes misclassification, then we choose the one that decreases the prediction probability $P(C(\mathbb{S}_{adv})=y)$ the most. We apply these token perturbations iteratively in decreasing order of token importance, until either $C(\mathbb{S}_{adv})\neq y$ (successful attack) or all the tokens of $\mathbb{S}$ have been perturbed (failed attack).

We present 4 attack modes for BAE based on the R and I operations, where for each token $t$ in $\mathbb{S}$:

- **BAE−R:** Replace token $t$ (See Algorithm 1)
- **BAE−I:** Insert a token to the left or right of $t$
- **BAE−R/I:** Either replace token $t$ or insert a token to the left or right of $t$
- **BAE−R+I:** First replace token $t$, then insert a token to the left or right of $t$

## 3 Experiments

**Datasets and Models.** We evaluate BAE on different text classification tasks. Amazon, Yelp, IMDB are sentiment classification datasets used in recent works (Sarma et al., 2018) and MR (Pang and Lee, 2005) contains movie reviews based on sentiment polarity. MPQA (Wiebe and Wilson, 2005) is a dataset for opinion polarity detection, Subj (Pang and Lee, 2004) for classifying a sentence as subjective or objective and TREC (Li and Roth, 2002) for question type classification.

We use 3 popular text classification models: word-LSTM (Hochreiter and Schmidhuber, 1997), word-CNN (Kim, 2014) and a fine-tuned BERT (Devlin et al., 2018) base-uncased classifier. We train models on the training data and perform the adversarial attack on the test data. For complete model details, refer to Appendix A.

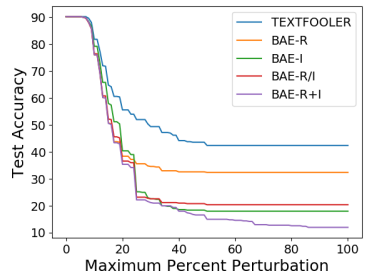
As a baseline, we consider TextFooler (Jin et al., 2019) which performs synonym replacement using a fixed word embedding space (Mrkšić et al., 2016). We only consider the top K=50 synonyms from the BERT-MLM predictions and set a threshold of 0.8 for the cosine similarity between USE based embeddings of the adversarial and input text.

**Automatic Evaluation Results.** We perform the 4 BAE attacks and summarize the results in Tables 1 and 2. Across datasets and models, our BAE attacks are almost always more effective than the baseline attack, achieving significant drops of 40-80% in test accuracies, with higher average semantic similarities as shown in parentheses.

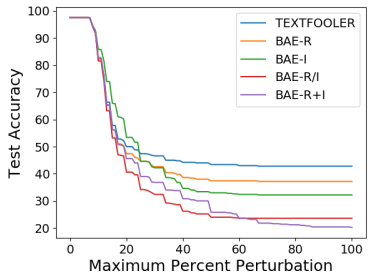
With just one exception, BAE−R+I is the strongest attack since it allows both replacement and insertion at the same token position. We observe a general trend that the BAE−R and BAE−I attacks often perform comparably, while the BAE−R/I and BAE−R+I attacks are much stronger. We observe that the BERT classifier is more robust to BAE and TextFooler attacks than

| Model | Adversarial Attack | Datasets | | |
|---|---|---|---|---|
| | | MPQA | Subj | TREC |
| **wordLSTM** | Original | 89.43 | 91.9 | 90.2 |
| | TextFooler | 48.49 (0.745) | 58.5 (0.882) | 42.4 (0.834) |
| | BAE−R | 45.66 (0.748) | 50.2 (0.899) | 32.4 (0.870) |
| | BAE−I | 40.94 (0.871) | 49.8 (0.958) | 18.0 (0.964) |
| | BAE−R/I | 31.60 (0.820) | 43.1 (0.946) | 20.4 (0.954) |
| | BAE−R+I | **25.57 (0.766)** | **29.0 (0.929)** | **11.8 (0.874)** |
| **wordCNN** | Original | 89.06 | 91.3 | 93.2 |
| | TextFooler | 48.77 (0.733) | 58.9 (0.889) | 47.6 (0.812) |
| | BAE−R | 44.43 (0.735) | 51.0 (0.899) | 29.6 (0.843) |
| | BAE−I | 44.43 (0.876) | 49.8 (0.958) | 15.4 (0.953) |
| | BAE−R/I | 32.17 (0.818) | 41.5 (0.940) | 13.0 (0.936) |
| | BAE−R+I | **27.83 (0.764)** | **31.1 (0.922)** | **8.4 (0.858)** |
| **BERT** | Original | 90.66 | 97.0 | 97.6 |
| | TextFooler | 36.23 (0.761) | 69.5 (0.858) | 42.8 (0.866) |
| | BAE−R | 43.87 (0.764) | 77.2 (0.828) | 37.2 (0.824) |
| | BAE−I | 33.49 (0.862) | 74.6 (0.918) | 32.2 (0.931) |
| | BAE−R/I | 24.53 (0.826) | 64.0 (0.903) | 23.6 (0.908) |
| | BAE−R+I | **24.34 (0.766)** | **58.5 (0.875)** | **20.2 (0.825)** |

Table 2: Automatic evaluation of adversarial attacks on MPQA, Subj and TREC datasets. Other details follow those from Table 1. All 4 modes of BAE attacks almost always outperform TextFooler.




(a) Word-LSTM




(b) BERT

Figure 2: Graphs comparing attack effectiveness on the TREC dataset, as a function of maximum % perturbation to the input.

the word-LSTM and word-CNN possibly due to its large size and pre-training on a large corpus.

The TextFooler attack is sometimes stronger than the BAE−R attack for the BERT classifier. We attribute this to the shared parameter space between the BERT-MLM and the BERT classifier before fine-tuning. The predicted tokens from BERT-MLM may not be able to drastically change the internal representations learned by the BERT classifier, hindering their ability to adversarially affect the classifier prediction.

Additionally, we make some interesting observations pertaining to the average semantic similarity of the adversarial examples with the original sentences (computed using USE). From Tables 1, 2 we observe that across different models and datasets, all BAE attacks have higher average semantic similarity than TextFooler. Notably, the BAE−I attack achieves the highest semantic similarity among all the 4 modes. This can be explained by the fact that all tokens of the original sentence are retained, in the original order, in the adversarial example generated by BAE−I. Interestingly, we observe that the average semantic similarity of the BAE−R+I attack is always higher than the BAE−R attack. This lends support to the importance of the 'Insert' operation in ameliorating the effect of the 'Replace' operation. We further investigate this through an ablation study discussed later.

**Effectiveness.** We study the effectiveness of BAE on limiting the number of R/I operations permitted on the original text. We plot the attack performance as a function of maximum % perturbation (ratio of number of word replacements and insertions to the length of the original text) for the TREC dataset. From Figure 2, we clearly observe that the BAE attacks are consistently stronger than TextFooler. The classifier models are relatively robust to perturbations up to 20%, while the effectiveness saturates at 40-50%. Surprisingly, a 50% perturbation for the TREC dataset translates to replacing or inserting just 3-4 words, due to the short text lengths.

**Qualitative Examples.** We present adversarial examples generated by the attacks on sentences from the IMDB and Yelp datasets in Table 3. All attack strategies successfully changed the classification to negative, however the BAE attacks produce more natural looking examples than TextFooler. The tokens predicted by the BERT-MLM fit well in the sentence context, while TextFooler tends to replace words with complex synonyms, which can be easily detected. Moreover, BAE's additional degree of freedom to insert tokens allows for a successful attack with fewer perturbations.

**Human Evaluation.** We perform human evaluation of our BAE attacks on the BERT classifier. For 3 datasets, we consider 100 samples from each test set shuffled randomly with their successful ad-

| | |
|---|---|
| **Original [Positive Sentiment]:** This film offers many delights and surprises. | |
| **TextFooler:** This flick citations disparate revel and surprises. | |
| **BAE-R:** This movie offers enough delights and surprises | |
| **BAE-I:** This lovely film platform offers many pleasant delights and surprises | |
| **BAE-R/I:** This lovely film serves several pleasure and surprises . | |
| **BAE-R+I:** This beautiful movie offers many pleasant delights and surprises . | |
| **Original [Positive Sentiment]:** Our server was great and we had perfect service. | |
| **TextFooler:** Our server was tremendous and we assumed faultless services. | |
| **BAE-R:** Our server was decent and we had outstanding service. | |
| **BAE-I:** Our server was great enough and we had perfect service but. | |
| **BAE-R/I:** Our server was great enough and we needed perfect service but. | |
| **BAE-R+I:** Our server was decent company and we had adequate service. | |

Table 3: Qualitative examples of each attack on the BERT classifier (Replacements: Red, Inserts: Blue)

| Dataset | Sentiment Accuracy (%) | | | |
|---|---|---|---|---|
| | Original | TF | R | R+I |
| Amazon | 95.7 | 79.1 | **85.2** | 83.8 |
| IMDB | 90.3 | 83.1 | **84.3** | 79.3 |
| MR | 93.3 | 82.0 | **84.6** | 82.4 |
| **Dataset** | **Naturalness (1-5)** | | | |
| | Original | TF | R | R+I |
| Amazon | 4.26 | 3.17 | **3.91** | 3.71 |
| IMDB | 4.35 | 3.41 | **3.89** | 3.76 |
| MR | 4.19 | 3.35 | **3.84** | 3.74 |

Table 4: Human evaluation results (TF: TextFooler and R(R+I): BAE-R(R+I)).

versarial examples from BAE-R, BAE-R+I and TextFooler. We calculate the sentiment accuracy by asking 3 annotators to predict the sentiment for each sentence in this shuffled set. To evaluate the *naturalness* of the adversarial examples, we first present the annotators with 50 other original data samples to get a sense of the data distribution. We then ask them to score each sentence (on a Likert scale of 1-5) in the shuffled set on its grammar and likelihood of being from the original data. We average the 3 scores and present them in Table 4.

Both BAE-R and BAE-R+I attacks almost always outperform TextFooler in both metrics. BAE-R outperforms BAE-R+I since the latter inserts tokens to strengthen the attack, at the expense of naturalness and sentiment accuracy. Interestingly, the BAE-R+I attacks achieve higher average semantic similarity scores than BAE-R, as discussed in Section 3. This exposes the shortcomings of using USE for evaluating the retention of semantics of adversarial examples, and reiterates the importance of human-centered evaluation. The gap between the scores on the original data and the adversarial examples speaks for the limitations of the attacks, however BAE represents an important step forward towards improved adversarial examples.

**Replace vs. Insert.** Our BAE attacks allow insertion operations in addition to replace. We analyze the benefits of this flexibility of R/I operations in Table 5. From Table 5, the splits $\mathbb{A}$ and $\mathbb{B}$ are the % of test points which *compulsorily* need I and R operations respectively for a successful attack. We can observe that the split $\mathbb{A}$ is larger than $\mathbb{B}$ thereby indicating the importance of the I operation over R. Test points in split $\mathbb{C}$ require both R *and* I operations for a successful attack. Interestingly, split $\mathbb{C}$ is largest for Subj, which is the most robust to attack (Table 2) and hence needs both R/I operations. Thus, this study gives positive insights towards the importance of having the flexibility to both replace and insert words.

| Dataset | Word-LSTM | | | Word-CNN | | | BERT | | |
|---|---|---|---|---|---|---|---|---|---|
| | $\mathbb{A}$ | $\mathbb{B}$ | $\mathbb{C}$ | $\mathbb{A}$ | $\mathbb{B}$ | $\mathbb{C}$ | $\mathbb{A}$ | $\mathbb{B}$ | $\mathbb{C}$ |
| **MR** | 15.1 | 10.1 | 3.1 | 12.4 | 9.6 | 2.8 | 24.3 | 12.9 | 5.7 |
| **Subj** | 14.4 | 12.3 | 5.1 | 16.2 | 13.8 | 7.4 | 13.9 | 11.4 | 7.5 |
| **TREC** | 16.6 | 1.6 | 0.2 | 20.0 | 5.0 | 1.4 | 14.0 | 8.6 | 2.4 |

Table 5: Analyzing relative importance of 'Replace' and 'Insert' perturbations for BAE. $\mathbb{A}$ denotes % of test instances which are successfully attacked by BAE-R/I, but not BAE-R, i.e. $\mathbb{A}$ : (R/I) $\cap \overline{\text{R}}$. Similarly, $\mathbb{B}$ : (R/I) $\cap \overline{\text{I}}$ and $\mathbb{C}$ : (R/I) $\cap \overline{\text{R}} \cap \overline{\text{I}}$.

We present complete effectiveness graphs and details of human evaluation in Appendix B and C. BAE is implemented[1] in TextAttack (Morris et al., 2020), a popular suite of NLP adversarial attacks.

## 4 Conclusion

In this paper, we have presented a new technique for generating adversarial examples (BAE) through contextual perturbations based on the BERT Masked Language Model. We propose inserting and/or replacing tokens from a sentence, in their order of importance for the text classification task, using a BERT-MLM. Automatic and human evaluation on several datasets demonstrates the strength and effectiveness of our attack.


## Acknowledgments

The authors thank Arka Sadhu, Kalpesh Krishna, Aws Albarghouthi, Yingyu Liang and Justin Hsu for providing in-depth feedback for this research. The authors thank Jack Morris and Jin Yong Yoo for integrating BAE in the TextAttack framework. This work is supported, in part, by the National Science Foundation CCF under award 1652140.


[1] https://github.com/QData/TextAttack/blob/master/textattack/attack_recipes/bae_garg_2019.py

## 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*. Association for Computational Linguistics.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

Shi Feng, Eric Wallace, Mohit Iyyer, Pedro Rodriguez, Alvin Grissom II, and Jordan L. Boyd-Graber. 2018. Right answer for the wrong reason: Discovery and mitigation. *CoRR*, abs/1804.07781.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. *CoRR*, abs/1801.04354.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Com.*, 9(8):1735–1780.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is bert really robust? natural language attack on text classification and entailment. *arXiv preprint arXiv:1907.11932*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2017. Adversarial examples in the physical world. *ICLR Workshop*.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding neural networks through representation erasure. *CoRR*, abs/1612.08220.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2017. Deep text classification can be fooled. *CoRR*, abs/1704.08006.

John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp.

Nikola Mrkšić, Ivan Vulić, Diarmuid Ó Séaghdha, Ira Leviant, Roi Reichart, Milica Gašić, Anna Korhonen, and Steve Young. 2017. Semantic specialization of distributional word vector spaces using monolingual and cross-lingual constraints. *Transactions of the Association for Computational Linguistics*.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints. In *Proceedings of HLT-NAACL*.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity. In *Proceedings of ACL*, pages 271–278.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of ACL*.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '17, pages 506–519, New York, NY, USA. ACM.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

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*. Association for Computational Linguistics.

Prathusha K Sarma, Yingyu Liang, and Bill Sethares. 2018. Domain adapted word embeddings for improved sentiment classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 37–42.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *International Conference on Learning Representations*.

Janyce Wiebe and Theresa Wilson. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*.

Wei Emma Zhang, Quan Z. Sheng, and Ahoud Abdulrahmn F. Alhazmi. 2019. Generating textual adversarial examples for deep learning models: A survey. *CoRR*, abs/1901.06796.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2018. Generating natural adversarial examples. In *International Conference on Learning Representations*.

## Broader Ethical Impact

Our work addresses the important problem of adversarial vulnerabilities of modern text classification models. While we acknowledge the possibility of its misuse to maliciously attack publicly available text classifiers, we believe our work represents an important step forward in analyzing the robustness of NLP models. We hope our work inspires improved defenses against adversarial attacks on text classification models.

## A Experimental Reproducibility

**Dataset and Models** The dataset statistics are reported in Table 5 and we give a brief overview of the dataset and the task for which it is used along with public links to download the datasets.

- *Amazon*: Amazon product reviews dataset [2].
- *Yelp*: A restaurant reviews dataset from Yelp[2].
- *IMDB*: IMDB movie reviews dataset[2].
- *MR*: A movie reviews dataset based on subjective rating and sentiment polarity [3].
- *MPQA*: An unbalanced dataset for polarity detection of opinions [4].
- *TREC*: A dataset for classifying types of questions with 6 classes [5].
- *SUBJ*: A dataset for classifying a sentence as objective or subjective. [2]

| Dataset | # Classes | Train | Test | Avg Length |
|---|---|---|---|---|
| Amazon | 2 | 900 | 100 | 10.29 |
| Yelp | 2 | 900 | 100 | 11.66 |
| IMDB | 2 | 900 | 100 | 17.56 |
| MR | 2 | 9595 | 1067 | 20.04 |
| MPQA | 2 | 9543 | 1060 | 3.24 |
| Subj | 2 | 9000 | 1000 | 23.46 |
| TREC | 6 | 5951 | 500 | 7.57 |

Table 5: Summary statistics for the datasets

[2] https://archive.ics.uci.edu/ml/datasets/Sentiment+Labelled+Sentences

[3] https://www.cs.cornell.edu/people/pabo/movie-review-data/

[4] http://mpqa.cs.pitt.edu/

[5] http://cogcomp.org/Data/QA/QC/

**Training Details** On the sentence classification task, we target three models: word-based convolutional neural network (WordCNN), word-based LSTM, and the state-of-the-art BERT. We use 100 filters of sizes 3,4,5 for the WordCNN model with a dropout of 0.3. Similar to (Jin et al., 2019) we use a 1-layer bi-directional LSTM with 150 hidden units and a dropout of 0.3. For both models, we use the 300 dimensional pre-trained counter fitted word embeddings (Mrkšić et al., 2017). For the BERT classifier, we used the BERT base uncased model which has 12-layers, 12 attention heads and 768 hidden dimension size. Across all models and datasets, we use the standard BERT uncased vocabulary of size 30522. We first train all three models on the training data split and use early stopping on the test dataset. For BERT fine-tuning, we use the standard setting of an Adam classifier having a learning rate of $2 \times 10^{-5}$ and 2 fine-tuning epochs.

For our BAE attacks, we use a pre-trained BERT Base-uncased MLM to predict the masked tokens. We only consider the top K=50 synonyms from the BERT-MLM predictions and set a threshold of 0.8 for the cosine similarity between USE based embeddings of the adversarial and input text.

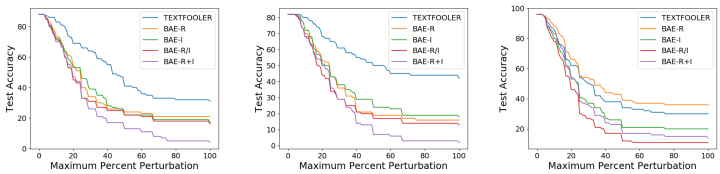
For R operations, we filter out predicted tokens which form a different POS than the original token in the sentence. For both R and I operations, we filter out stop words using NLTK from the set of predicted tokens. Additionally we filter out antonyms using synonym embeddings (Mrkšić et al., 2016) for sentiment analysis tasks.

## B Results

Figures 3 - 8 are the complete set of graphs showing the attack effectiveness for all seven datasets.

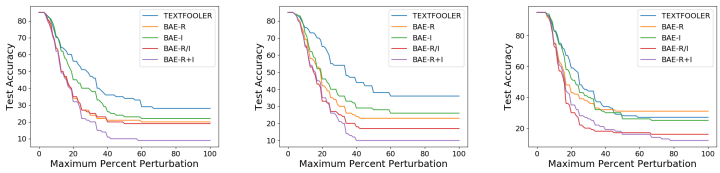
## C Human Evaluation

We ask the human evaluators to judge the *naturalness* of texts presented to them, i.e. whether they think they are adversarial examples or not. They were instructed to do so on the basis of grammar and how likely they think it is from the original dataset, and rate each example on the following Likert scale of 1-5: 1) Sure adversarial sample, 2) Likely an adversarial example, 3) Neutral, 4) Likely an original sample, 5) Sure original sample. From the results of Table 3, it is clear that BAE-R always beats the sentiment accuracy and *naturalness* score of TextFooler. The latter is due to unnaturally long and complex synonym replacements on using TextFooler.



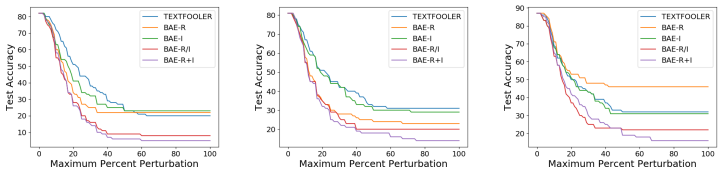

(a) Word-LSTM (b) Word-CNN (c) BERT

Figure 3: Amazon



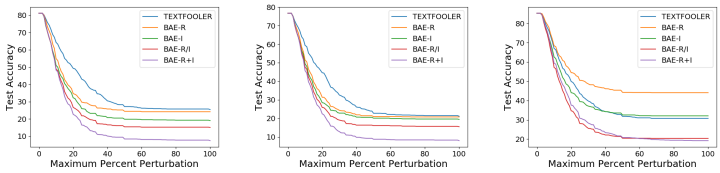

(a) Word-LSTM (b) Word-CNN (c) BERT

Figure 4: Yelp



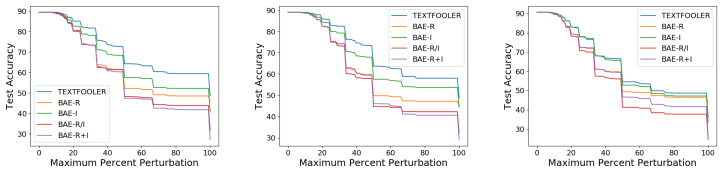

(a) Word-LSTM (b) Word-CNN (c) BERT

Figure 5: IMDB



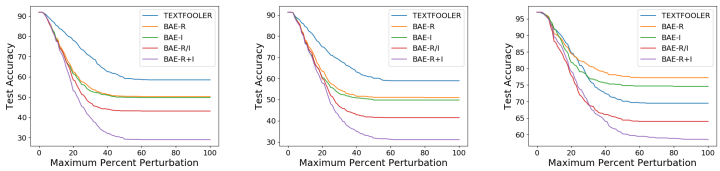

(a) Word-LSTM (b) Word-CNN (c) BERT

Figure 6: MR




(a) Word-LSTM (b) Word-CNN (c) BERT

Figure 7: MPQA




(a) Word-LSTM (b) Word-CNN (c) BERT

Figure 8: Subj