Improving Textual Adversarial Attacks using Metric-Guided Rewrite and Rollback

Anonymous ACL submission

Abstract

Adversarial examples are helpful for analyzing and improving the robustness of text classifiers. Generating high-quality adversarial examples is a challenging task as it requires the generation of adversarial sentences that are fluent, semantically similar to the original ones and should lead to misclassification. Existing methods prioritize misclassification by maximizing each perturbation's effectiveness at misleading a text classifier; thus, the generated adversarial examples fall short in terms of fluency and similarity. In this paper, we define a critique score that synthesizes the fluency, similarity, and misclassification metrics. We propose a rewrite and rollback (R&R) framework guided by the optimization of this score to improve the adversarial attack. R&R generates high-quality adversarial examples by allowing exploration of perturbations without immediate impact on the misclassification, and yet optimizing critique score for better fluency and similarity. We evaluate our method on 5 representative datasets and 3 classifier architectures. Our method outperforms current state-of-the-art in attack success rate by +16.2%, +12.8%, and +14.0% on the classifiers respectively. All code and results will be publicly available.

1 Introduction

Recently, adversarial attacks in text classification have received a great deal of attention. Adversarial attacks are defined as subtle perturbations in the input text such that a classifier misclassifies it. They can serve as a tool to analyze and improve the robustness of text classifiers, thus being more and more important because security-critical classifiers are being widely deployed (Wu et al., 2019; Torabi Asr and Taboada, 2019; Zhou et al., 2019).

Existing attack methods either adopt a synonym substitution approach (Jin et al., 2020; Zang et al., 2020) or use a pre-trained language model to propose substitutions for better fluency and naturalness (Li et al., 2020; Garg and Ramakrishnan, 2020; Li et al., 2021). They follow a similar framework: first, construct some candidate perturbations, and then, use the perturbations that most effectively mislead the classifier to modify the sentence. This process is repeated multiple times until an adversarial example is found. This framework prioritizes misclassification by picking perturbations that most effectively mislead the classifier. Despite the success in changing the classifier prediction, it has two main disadvantages. First, it is prone to modify words that are critical to the sentence's meaning, or introduce low-frequency words to mislead the classifier, causing the similarity and fluency to decrease. Second, some perturbations do not have immediate impacts on misclassification, but can trigger it when combined with other perturbations, while this framework cannot find adversarial examples with these perturbations.

To overcome these problems, the attack method needs to consider jointly fluency, similarity, and misclassification, while also efficiently exploring various perturbations that do not show direct impacts on the latter. We define a critique score that synthesizes fluency, similarity and misclassification metrics. Then, we present our design for a Rewrite and Rollback framework (R&R) which optimizes this score to generate better adversarial examples. In the rewrite stage, we explore multi-word substitutions proposed by a pre-trained language model. We accept or reject a substitution according to the critique score. We can generate a high-quality adversarial example after multiple iterations of rewrite. Rewrite may introduce changes that do not contribute to misclassification and may also reduce similarity and fluency. Therefore, we periodically apply the rollback operation to reduce the number of modifications without changing the misclassification result. Figure 1 illustrates the process using an example.

Our contributions are summarized as follows:

- We synthesize similarity, fluency, and misclassi-

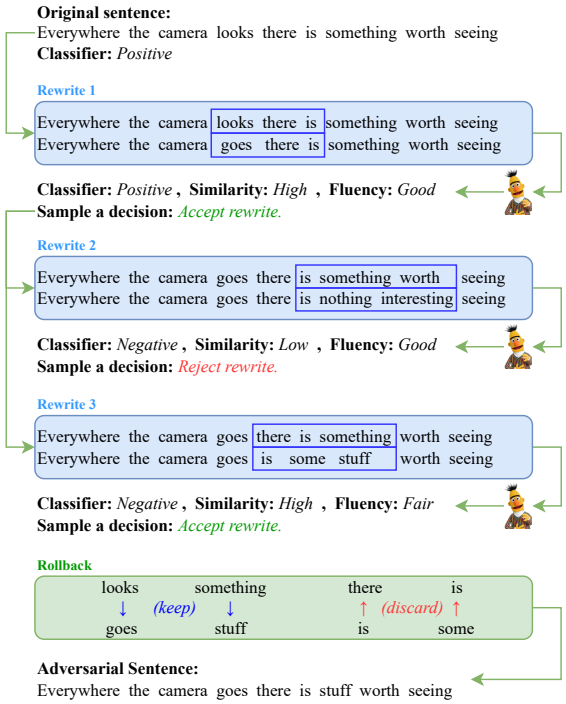

Figure 1: R&R generates adversarial examples by rewrite and rollback. The rewrite step explores possible perturbations stochastically and is guided by similarity metric and fluency metric to ensure better quality of the example. The rollback operation further improves the similarity.

fication metrics into a single optimization objective called critique score;

- We propose R&R to optimize the critique score, whereas traditional methods only optimize misclassification;
- We carry out extensive experimentation on 5 representative datasets and 3 classifier architectures to show the wide range of applications and generalizability of our framework;
- Both automatic evaluations and human evaluations show R&R outperforms existing methods by large margin;
- We provide the code and datasets for reproducible experiments.

2 Problem Formulation

The adversarial attack consists of modifying a sentence such that it keeps its meaning and correctness but gets a label different from the original one when using the same classifier. Specifically, let $\mathbf{x} = x_1, \ldots, x_l$ be a sentence of length l, y be its classification label, and $f(\mathbf{x})$ be a text classifier that predicts a probability distribution over classes. The objective of an attack method $\mathcal{A}(\mathbf{x}, f)$ is to

$$\text{construct } \mathbf{u} = u_1, \ldots, u_{l'} \text{ satisfying 3 conditions:}$$

$$\begin{cases} \mathbf{u} \text{ is misclassified, i.e., } f(\mathbf{u}) \neq y, \\ \text{Human considers } \mathbf{u} \text{ as a fluent sentence,} \\ \text{Human considers } \mathbf{u} \text{ to be semantically similar to } \mathbf{x}. \end{cases}$$

where l' is the length of the adversarial sentence. However, this formulation requiring human evaluation is intractable for large-scale data. Therefore, we approximate the sentence fluency with the perplexity of the sentence. It is defined as

$$\text{ppl}(\mathbf{x}) = \exp\Big[-\tfrac{1}{l}\textstyle\sum_{i=1}^{l} \log p(x_i|x_1 \ldots x_{i-1})\Big],$$

where $p(x_i|x_1 \ldots x_{i-1})$ is measured by a language model. Low perplexity means the sentence is predictable by the language model, which usually indicates the sentence is fluent. Sentence similarity can be quantified as $\cos\big(H(\mathbf{x}), H(\mathbf{u})\big)$, where $H(\cdot)$ is a pre-trained sentence encoder that encodes the meaning of a sentence into a vector. Thus, finding the adversarial sentence $\mathbf{u}$ is formulated as a multi-objective optimization problem as follows:

$$\begin{aligned} \text{Construct } \mathbf{u} = u_1, \ldots, u_{l'} &\text{ to minimize } \text{ppl}(\mathbf{u}) \\ &\text{and maximize } \cos\big(H(\mathbf{x}), H(\mathbf{u})\big) \\ &\text{subject to } f(\mathbf{u}) \neq y. \end{aligned}$$

In this paper, we use a fine-tuned BERT-base model to measure perplexity and use USE to measure sentence similarity. Ultimately, fluency and similarity need to be verified by humans. We discuss human verification in Section 4.3.

Threat Model. We assume the attacker can query the classifier for the prediction (i.e., the probability distribution over all classes). But they do not have knowledge on architecture of the classifier nor query for the gradient. They can also access some unlabeled text in the domain of the classifier.

3 Metric-Guided Rewrite and Rollback

In this section, we first give an overview of the R&R framework. Then, we introduce the rewrite and rollback components respectively. Finally, we give a summary of pre-trained models used in the framework.

3.1 Overview

R&R solves the multi-objective optimization problem by synthesizing the fluency, similarity and

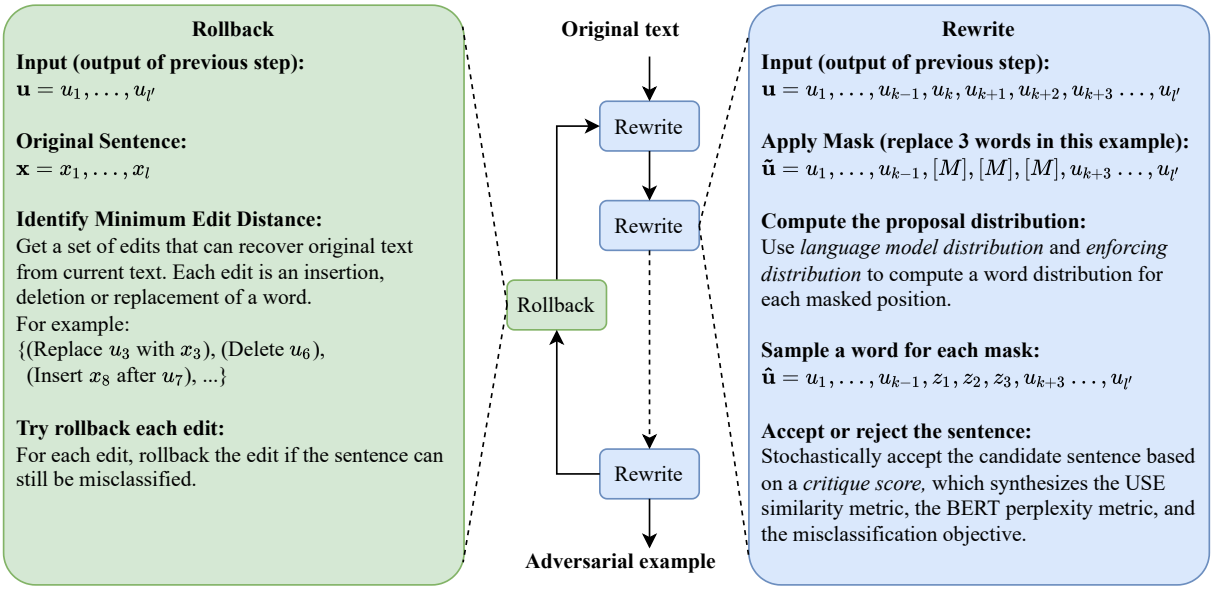

Figure 2: R&R Framework.

misclassification objectives into a single *critique score*, and maximizing this score. Therefore, our framework can construct adversarial sentences with lower perplexity and higher similarity. R&R contains the rewrite and rollback steps.

In the rewrite step, we randomly mask several consecutive words, and compute a *proposal distribution*, which is a distribution over the vocabulary on each masked position. We construct a multi-word substitution for the masked positions according to the distribution, then compute the critique score. If the score increases, we accept the substitution. If the score decreases, we accept it with a probability depending on the degree of decrease. The rewrite step contains randomness to encourage exploration of different modifications, while the critique score will guide the rewritten sentence to a high-quality adversarial example. After several steps of rewriting, we apply a rollback operation on the sentences that have already been misclassified to reduce the number of changes introduced in the rewriting. In the rollback step, we identify a minimum set of edits required to change the current sentence back to the original sentence. We rollback an edit if it does not affect the misclassification.

We implement the framework in batch so that it simultaneously perturbs multiple copies of the input text in different ways. The loop ends when half of the sentences in the batch are misclassified to make the algorithm more efficient. Figure 2 shows the R&R framework.

3.2 Rewrite

In each rewrite, we mask then substitute a span of words. It is composed of the following steps.

Apply mask in the sentence. First, we randomly pick m consecutive words in the sentence, and replace them with t mask, where t can be m, $m-1$, or $m+1$ meaning *replace*, *delete*, and *insert* operation respectively. Compared with CLARE (Li et al., 2021) which masks one word at a time (i.e., $m = 1$), masking multiple words can make it easier to modify common phrases. We use $\tilde{\mathbf{u}}$ to denote the masked sentence.

Compute proposal distribution. Then, we compute proposal distribution for t masks in the sentence. This distribution assigns a high probability to words that can construct a fluent and legitimate paraphrase. Let $z_1, \ldots z_t$ be the words to be placed at the masked positions. The distribution is

$$p_{\text{proposal}}(z_i|\tilde{\mathbf{u}}, \mathbf{x}) \propto p_{\text{lm}}(z_i|\tilde{\mathbf{u}}) \times p_{\text{enforce}}(z_i|\tilde{\mathbf{u}}, \mathbf{x})$$

where p_{lm} is a *language model distribution* that ensures the sentence will be fluent and meaningful, and p_{enforce} is the *enforcing distribution*, which improves the semantic similarity. p_{lm} is computed by sending $\tilde{\mathbf{u}}$ into BERT and taking the predicted word distribution on masked positions. Depending on the position, the word distributions for t masks are different. The enforcing distribution is measured by word embeddings. We use the sum of word embeddings $R(\mathbf{u}) = \sum_{u_k} E(u_k)$ as a sentence embedding, where $E(\cdot)$ is the counter-fitted

word embedding (Mrkšic et al., 2016). Then we define the enforcing distribution as

$$p_{\text{enforce}}(z_i|\tilde{\mathbf{u}}, \mathbf{x}) \propto \exp\big[w_{\text{enforce}} \times (\cos(R(\mathbf{x}) - R(\tilde{\mathbf{u}}), E(z_i)) - 1)\big].$$

w_{enforce} is a hyper-parameter. If the embedding of a word $E(z)$ perfectly aligns with the sentence representation difference $R(\mathbf{x}) - R(\tilde{\mathbf{u}})$, it gets the largest probability. The enforcing distribution aims at making the candidate modification more similar to the original sentence. Note that enforcing distribution is identical on all t masks.

Sample a candidate sentence. We sample a candidate word for each masked position by $z_i \sim p_{\text{proposal}}(z_i|\tilde{\mathbf{u}}, \mathbf{x})$. We do not consider the effect of sampling one word on other masked positions (i.e., we do not recompute proposal distribution for the remaining masks after sampling a word) because language model distribution already considers the position of the mask and assigns a different distribution for each mask, meanwhile recomputing is inefficient. We use $\hat{\mathbf{u}}$ to denote the candidate sentence.

Critique score and decision function. We decide whether to accept the candidate sentence using a decision function. The decision function computes a heuristic critique score

$$C(\mathbf{u}) = \big(w_{\text{ppl}} \min(1 - \text{ppl}(\mathbf{u})/\text{ppl}(\mathbf{x}), 0) \quad (1)$$
$$+ w_{\text{sim}} \min(\cos\big(H(\mathbf{u}), H(\mathbf{x})\big) - \phi_{\text{sim}}, 0) \quad (2)$$
$$+ w_{\text{clf}} \min(\max_{y' \neq y} f(\mathbf{u})_{y'} - f(\mathbf{u})_y, 0)\big) \quad (3)$$

Eq. 1 penalizes sentences with high perplexity, where $\text{ppl}(\mathbf{x})$ is perplexity measured by a BERT model. Eq. 2 penalizes sentences with sentences with cosine similarity lower than ϕ_{sim}, where $H(\cdot)$ is the sentence representation by USE. Eq. 3 penalizes sentences that cannot be misclassified where $f(\mathbf{u})_y$ means the log probability of class y predicted by the classifier. Let $\alpha = \exp[C(\hat{\mathbf{u}}) - C(\mathbf{u})]$. If $\alpha > 1$, the decision function accepts $\hat{\mathbf{u}}$; otherwise it accepts $\hat{\mathbf{u}}$ with probability α. The critique score is a straightforward way to convert the multi-objective optimization problem into a single objective. Although it introduces several hyper-parameters, R&R is no more complicated than conventional methods, which also require hyper-parameter setting.

3.3 Rollback

In the rollback step, we eliminate modifications that do not correct the misclassification. It contains the following steps.

Find a minimum set of simple edits. We first find a set of simple edits that change the current rewritten sentence back to the original sentence. Simple edits mean the insertion, deletion or replacement of a single word, which is different from the modification in the rewrite step.

Rollback edits. For each edit, if reverting it does not correct the misclassification, then we revert the edit. For convenience, we scan each word in the sentence from right to left, and try to rollback each edit. Note that rollback may introduce grammar errors, but they can be fixed in future rewrite steps.

3.4 Vocabulary Adaptation

The computation of p_{propose} is challenging because of the inconsistent vocabulary. The BERT language model used in $p_{\text{lm}}(\cdot)$ uses a 30k-word-piece vocabulary. It contains common words and affixes. Rare words will be handled as multiple affixes. For example "hyperparameter" does not appear in the vocabulary, so it is handled as "hyper" and "##parameter". The counter fitted word embeddings in $p_{\text{enforce}}(\cdot)$ works on a 65k-word vocabulary. Since the BERT model is more complicated, we keep it as it is and transfer word embeddings to BERT vocabulary. We train the word-piece embeddings as follows. Let $\mathbf{w} = \{w_1, \ldots, w_L\}$ be a plain text corpus tokenized by words. Let $T(w)$ be word-piece tokenization of a word. Let $E(w)$ be the original word embeddings and $E'(x)$ be the transferred embeddings on word-piece. We train the word-piece embeddings E' by minimizing the absolute error $\sum_{w \in \mathbf{w}} ||E(w) - \sum_{x \in T(w)} E'(x)||_1$. We initialize E' by copying the embedding on words shared by two vocabularies and set other embeddings to 0. We optimize the absolute error using stochastic gradient descent. In each step, we sample 5000 words from $\mathbf{w}$, then update E' accordingly.

3.5 Summary of pre-trained models in R&R

In R&R, we employ several pre-trained models. Choices are made according to the different characteristics of these pre-trained models.

BERT for masked word prediction and perplexity: Because BERT is originally trained for masked

word prediction, it can predict the word distribution given context from both sides. Thus, BERT is preferable for generating p_{lm}. Estimating the perplexity for a sentence requires BERT to run in decoder mode and be fine-tuned. Perplexity can also be measured by other language models such as GPT2 (Radford et al., 2019). We use BERT mainly for the consistent vocabulary with p_{lm}.

Word embeddings and USE for similarity. Word embeddings is more efficient as it only computes the sum of vectors and cosine similarity. In enforcing distribution, we need to replace the selected position with all possible z's and measure the similarity, so we use word embeddings for efficiency. In the critique score, only the proposal sentence needs to be measured, so we can afford more computation time of USE.

4 Experiments

We conducted experiments on a wide range of datasets and multiple victim classifiers to show the efficacy of R&R. We first evaluate the quality of adversarial examples using automatic metrics. Then, we conducted human evaluation to show the necessity to generate highly similar and fluent adversarial examples. Finally, we conduct an ablation study to analyze each component of our method, and discuss defense against the attack.

4.1 Experimental Setup

Datasets. We use 3 conventional text classification datasets, including topic classification, sentiment classification, and question type classification. We also use 2 security-critical datasets, including hate speech detection and fake news detection. Dataset details are given in Table 1.

Name	#C	Len	Description
AG	4	43	News topic classification.
MR	2	32	Moview review dataset by Pang and Lee (2005).
TREC	6	8	Question type classification by Li and Roth (2002).
HATE	2	23	Hate speech detection dataset by Kurita et al. (2020).
FAKE	2	30	Fake news detection dataset by Yang et al. (2017). We use the first sentence of the news for classification.

Table 1: Dataset details. #C means number of classes. Len is the average number of words in a sentence.

Victim Classifiers. For each dataset, we use the full training set to train three victim classifiers: (1) bert-base classifier (Devlin et al., 2019); (2) RoBERTa-large classifier (Liu et al., 2019), and (3) FastText classifier (Joulin et al., 2017).

	AG	MR	TREC	HATE	FAKE
BERT-base	92.8	88.2	97.8	94.0	81.2
RoBERTa-large	92.7	91.6	97.3	95.0	75.5
FastText	89.2	79.5	85.8	91.5	72.4
Log Perplexity	3.38	5.27	3.91	3.56	4.92

Table 2: Accuracy of 3 classifers and sentence log perplexity on the clean test set.

Baselines. We compare our method against two strong baselines: TextFooler (Jin et al., 2020) and CLARE (Li et al., 2021).

Hyperparameters. In R&R, we use the BERT-base language model for p_{lm}. For each dataset, we fine-tune the BERT language model using 5k batches on the training set[1] with batch size 32 and learning rate 0.0001, so it is adapted to the dataset. We set the enforcing distribution hyper-parameters $w_{\text{enforce}} = 5$. The decision function hyper-parameters $w_{\text{ppl}} = 5$, $w_{\text{sim}} = 20$, $\phi_{\text{sim}} = 0.95$, $w_{\text{clf}} = 2$. To generate each paraphrase, we set maximum rewrite iterations to be 200, and replace a 3-word span in each iteration. We implement R&R in a 50-sentence batch and apply early-stop when half of the batch are misclassified. We apply rollback operation every 10 steps of rewrite. Then, we return the adversarial example with the best critique score.

Hardware and Efficiency. We conduct experiments on Nvidia RTX Titan GPUs. One attack on a BERT-base classifier using R&R takes 15.8 seconds on average. CLARE takes 14.4 seconds on average. TextFooler is the most efficient algorithm which takes 0.45 seconds.

Automatic Metrics. We evaluate the efficacy of the attack method using 3 automatic metrics:

- *Similarity (↑)*: We use Universal Sentence Encoder to encode the original and adversarial sentence, then use the cosine distance of two vectors to measure the similarity. We set a similarity threshold at 0.95, so the similarity of a legitimate adversarial example should be greater than 0.95.
- *Log Perplexity (↓)* shows the fluency of adversarial sentences.
- *Attack success rate (ASR) (↑)* shows the ratio of

[1] We use the plain text to fine-tune the language model, and do not use the label. In the threat model, we assume the attacker can access plain text data from a similar domain.

		AG			MR			TREC			HATE			FAKE		
	Attack	**ASR**	**Sim**	**PPL**	**ASR**	**Sim**	**PPL**	**ASR**	**Sim**	**PPL**	**ASR**	**Sim**	**PPL**	**ASR**	**Sim**	**PPL**
BERT	TextFooler	16.8	**0.98**	4.00	26.0	0.97	5.92	1.8	**0.97**	5.30	30.6	0.97	**3.53**	29.9	**0.98**	5.44
	CLARE	28.8	0.97	**3.60**	48.4	0.97	5.70	2.5	0.96	5.58	31.0	0.97	3.99	48.9	**0.98**	**5.02**
	R&R (Ours)	**54.1**	**0.98**	3.64	**63.4**	**0.98**	**5.36**	**10.8**	**0.97**	**5.29**	**55.3**	**0.98**	4.06	**57.0**	**0.98**	5.05
RoBERTa	TextFooler	15.6	**0.98**	5.21	18.0	**0.97**	6.06	0.4	0.96	7.09	24.0	**0.98**	4.20	26.6	**0.98**	5.45
	CLARE	23.3	0.97	5.24	45.9	**0.97**	5.67	2.5	**0.97**	6.53	35.7	0.97	4.37	46.0	**0.98**	**5.20**
	R&R (Ours)	**41.2**	**0.98**	**3.73**	**48.5**	**0.97**	**5.53**	**12.5**	**0.97**	**5.17**	**55.7**	0.97	**4.07**	**59.6**	**0.98**	5.25
FastText	TextFooler	25.8	**0.98**	4.16	33.1	**0.98**	5.85	6.5	**0.98**	5.04	21.7	**0.98**	**3.44**	35.3	**0.98**	5.46
	CLARE	28.9	0.97	3.91	41.5	0.97	5.79	8.5	0.97	6.06	35.6	0.97	4.24	76.0	**0.98**	5.15
	R&R (Ours)	**37.8**	**0.98**	**3.84**	**48.9**	**0.98**	**5.48**	**44.1**	**0.98**	**4.68**	**53.3**	**0.98**	4.03	**76.4**	**0.98**	**5.10**

Table 3: Automatic evaluation results. "Sim" and "PPL" represent similarity measured by USE and the log perplexity measured by BERT respectively.

	AG			MR			TREC			HATE			FAKE	
	S.	**F.**	**M.**	**S.**	**F.**	**M.**	**S.**	**F.**	**M.**	**S.**	**F.**	**M.**	**S.**	**F.**
TextFooler	3.93	3.58	0.90	3.3	3.49	**0.92**	3.25	2.88	0.88	3.76	3.61	0.46	3.58	3.58
CLARE	3.75	3.65	0.93	2.44	3.33	0.74	3.00	3.00	0.75	**3.89**	**4.41**	**0.81**	3.67	3.65
R&R (Ours)	**4.12**	**3.87**	**0.99**	**3.48**	**3.61**	0.85	**3.59**	**3.14**	**0.89**	3.59	3.94	0.76	**3.81**	**3.87**

Table 4: Human evaluation. "S.", "F." and "M." represents the similarity, fluency and label match annotated by human.

correctly classified text that can be successfully attacked.

Human Metrics: Automatic metrics are not always reliable. We use Mechanical Turk to verify the similarity, fluency, and whether the label of the text is preserved with respect to human evaluation.

Table 6 in Appendix shows some examples. We find R&R makes natural modifications to the sentence and preserves the semantic meanings.

4.2 Automatic Evaluation

Table 3 shows the ASR, similarity and perplexity metrics on the three victim classifiers.

Since we already apply a rigorous 0.95 threshold on similarity to ensure the adversarial examples are similar to the original sentences, the similarity metrics do not show significant differences. R&R achieves the best ASR on all datasets and across all classifiers. The average improvement compared with CLARE baseline is +16.2%, +12.8%, +14.0% on BERT-base, RoBERTa-large and FastText classifiers respectively. It means with the same similarity threshold, R&R is capable of finding more adversarial examples, i.e. for some text, R&R can find adversarial examples with similarity higher than 0.95 while baseline methods cannot. Thus it shows R&R can find adversarial examples with high similarity.

On AG, MR, TREC and FAKE datasets, R&R outperforms baseline methods on 9 cases. Baseline methods outperforms by a tiny margin on the other 3 cases. It shows R&R keeps sentence fluency as good as baseline methods. It does not sacrifice sentence fluency for higher ASR. The only failure case is on the HATE dataset, where Textfooler outperforms R&R in perplexity. Thus we investigate the average log perplexity of corresponding original sentences for each method. We find it is 3.24 for TextFooler and 3.94 for R&R. So TextFooler achieves low perplexity because it succeeds on original sentences with low perplexity while fails on those with higher perplexity.

We further measure ASR with various similarity and perplexity thresholds. On Figure 3, we set different thresholds and show the corresponding ASR. We observe that the curves of R&R are above the baseline curves in most cases, showing that R&R outperforms baselines on most threshold settings. It means R&R can achieve a higher ASR with the same similarity or perplexity threshold.

4.3 Human Evaluation

We use Mechanical Turk to evaluate the following metrics.

Sentence similarity (↑): Turkers are shown pairs of original and adversarial sentences, and are asked whether the two sentences have the same semantic meaning. They annotate the sentence in a 5-likert,

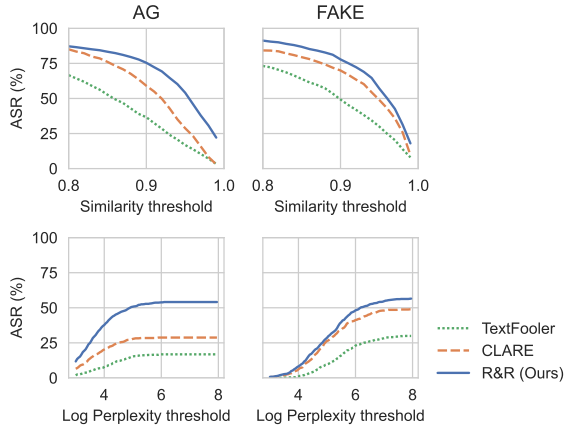

Figure 3: Attack success rate with respect to different similarity and perplexity constraints on BERT classifier. When evaluating different similarity thresholds, we do not set thresholds on perplexity. When evaluating perplexity thresholds, we fix the similarity threshold to 0.95. See Figure 8 in Appendix for other datasets and classifiers.

where 1 means strongly disagree, 2 means disagree, 3 means not sure, 4 means agree, and 5 means strongly agree.

Sentence fluency (↑): Turkers are shown a random shuffle of adversarial sentences, and are asked to rate the fluency in a 5-likert, where 1 describes a bad sentence, 3 describes a meaningful sentence with a few grammar errors, and 5 describes a perfect sentence.

Label match (↑): Turkers are shown a random shuffle of adversarial sentences and are asked whether it belongs to the class of the original sentence. They are asked to rate 0 as disagree, 0.5 as not sure, and 1 as agree.

We sample 100 adversarial sentences from each method, and each task is annotated by 2 Turkers. We do not annotate label matches on the FAKE dataset because identifying fake news is too challenging for Turkers. We require the location of the Turkers to be in United States, and their Hit Approval Rate to be greater than 95%. The screenshots of the annotation task is shown on Figure 7 in Appendix.

Table 4 shows the human evaluation results. R&R outperforms baselines on similarity and fluency on 4 datasets. It shows that by optimizing the critique score, R&R improves the similarity and fluency of adversarial sentences. Our method fails on the HATE dataset despite good automatic metrics. We hypothesize that this dataset collected from Twitter is more noisy than the others, causing the malfunction of automatic similarity and fluency metrics.

4.4 Ablation Study

We conduct ablation study on AG and FAKE datasets to understand the contribution of stochastic decision function, and periodic rollback. We also show the effect of multiple-word masking in Appendix A.

Decision Function In the Rewrite stage, we use a stochastic decision function based on the critique score. One alternative can be a deterministic greedy decision function, which accepts a rewrite only if the rewrite increases the critique score. Figure 4 shows the ASR with respect to different similarity thresholds. We find that the stochastic decision function outperforms the greedy one. We interpret the phenomenon as the greedy decision function gets stuck in local maxima, whereas the stochastic one can overcome this issue by accepting a slightly worse rewrite.

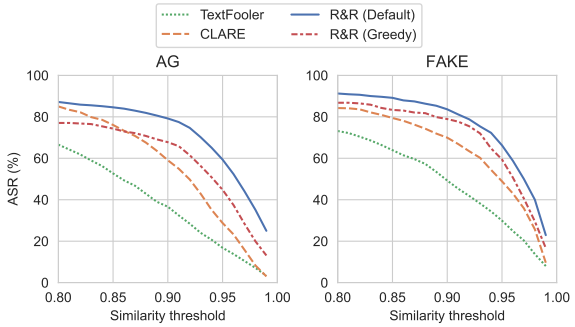

Figure 4: The ASR of R&R using different decision settings. "Greedy" means using a greedy decision function, which accepts a rewrite only if it has a higher critique score.

Rollback We apply rollback periodically during the attack. We compare it with two alternatives: (1) no rollback (NRB) which only uses rewrite to construct the adversarial sentences, and (2) single rollback (SRB) which applies rollback once on the NRB results. Figure 5 shows the result. We find that rollback has a significant impact. NRB performs the worst. Without rollback, it is difficult to get high cosine similarity when many words in the sentence have been changed. Single rollback increases the number of overlapped words, which usually increases the similarity measurement. By periodically applying the rollback, the rollbacked sentence can be further rewritten to improve the

similarity and fluency metrics, thus yielding to the best performance.

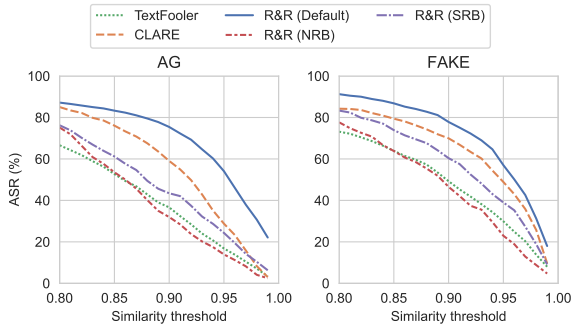

Figure 5: The ASR of R&R using different rollback settings. "NRB" means no rollback operation and "SRB" means single rollback.

4.5 Defense

We further explore the defense against this attack:

- Adversarial attack methods sometimes introduce outlier words to trigger misclassification. Therefore we follow Qi et al. (2020) and apply a perplexity-based filtering to eliminate outlier words in sentences. We generate adversarial sentences on vanilla classifier, then apply the filtering.
- SHIELD (Le et al., 2022) is a recently proposed algorithm that modifies the last layer of a neural network to defend against adversarial attack. We apply this method to classifiers and attack the robust classifier.

	AG		FAKE	
	+Filter	**+SHIELD**	**+Filter**	**+SHIELD**
TextFooler	6.2	8.2	13.8	16.7
CLARE	5.6	18.2	19.0	51.1
R&R (ours)	**22.3**	**30.6**	**23.1**	**59.4**

Table 5: The ASR of attack methods when applying the perplexity-based filtering (Filter) and the SHIELD defense on the BERT classifier.

Table 5 shows the ASR of attack methods with the a defense applied. We show that existing defense methods cannot effectively defend against R&R. It still outperforms existing methods in ASR by large margin.

5 Related Work

Several recent works proposed word-level adversarial attacks on text classifiers. This type of attack misleads the classifier's predictions by perturbing the words in the input sentence. TextFooler (Jin et al., 2020) shows the adversarial vulnerability of the state-of-the-art text classifiers. It uses heuristics to replace words with synonyms to mislead the classifier effectively. It relies on several pre-trained models, such as word embeddings (Mrkšic et al., 2016), part-of-speech tagger, and Universal Sentence Encoder (Cer et al., 2018) to perturb the sentence without changing its meaning. However, simple synonym substitution without considering the context results in unnatural sentences. Several works (Garg and Ramakrishnan, 2020; Li et al., 2020, 2021) address this issue by using masked language models such as BERT (Devlin et al., 2019) to propose more natural word substitutions. Our method also belongs to this category. But R&R does not maximize the efficacy of each perturbation, instead it allows exploring combinations of perturbations to generate adversarial examples with high similarity with the original sentence. Besides word-level attacks, there are also character-level attacks which introduce typos to trigger misclassification (Papernot et al., 2016; Liang et al., 2017; Samanta and Mehta, 2018), and sentence-level attacks which attack a classifier by altering the sentence structure (Iyyer et al., 2018). Zhang et al. (2020) gives a comprehensive survey on such attack methods. Other work on robustness to adversarial attacks in NLP includes robustness of the machine translation models (Cheng et al., 2019), robustness in domain adaptation (Oren et al., 2019), adversarial examples generated by reinforcement learning (Wong, 2017; Vijayaraghavan and Roy, 2019), and certified robustness (Jia et al., 2019).

6 Conclusion

In this paper, we formulate the textual adversarial attack as a multi-objective optimization problem. We use a critique score to synthesize the similarity, fluency, and misclassification objectives, and propose R&R that optimizes the critique score to generate high-quality adversarial examples. We conduct extensive experiments. Both automatic and human evaluation show that the proposed method succeeds in optimizing the automatic similarity and fluency metrics to generate adversarial examples of higher quality than previous methods.

Ethical Considerations

In this paper, we propose R&R to generate adversarial sentences. Like all other adversarial attack methods, this method could be abused by malicious users to attack NLP systems and obtain illegitimate benefits. However, it is still necessary for the research community to develop methods to exploit all vulnerabilities of a classifier based on which more robust classifiers can be developed.

References

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 Proceedings of the Conference on Empirical Methods in Natural Language Processing: System Demonstrations.

Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust neural machine translation with doubly adversarial inputs. In Proceedings of the Annual Meeting of the Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics.

Siddhant Garg and Goutham Ramakrishnan. 2020. Bae: Bert-based adversarial examples for text classification. In Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers).

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. In Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? natural language attack on text classification and entailment. In Proceedings of the AAAI Conference on Artificial Intelligence.

Armand Joulin, Édouard Grave, Piotr Bojanowski, and Tomáš Mikolov. 2017. Bag of tricks for efficient text classification. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, pages 427–431.

Keita Kurita, Paul Michel, and Graham Neubig. 2020. Weight poisoning attacks on pretrained models. In ACL.

Thai Le, Noseong Park, and Dongwon Lee. 2022. Shield: Defending textual neural networks against multiple black-box adversarial attacks with stochastic multi-expert patcher. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 6661–6674.

Dianqi Li, Yizhe Zhang, Hao Peng, Liqun Chen, Chris Brockett, Ming-Ting Sun, and William B Dolan. 2021. Contextualized perturbation for textual adversarial attack. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. Bert-attack: Adversarial attack against bert using bert. In Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing.

Xin Li and Dan Roth. 2002. Learning question classifiers. In COLING 2002: The 19th International Conference on Computational Linguistics.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2017. Deep text classification can be fooled. In Proceedings of the International Joint Conferences on Artificial Intelligence.

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

Nikola Mrkšic, Diarmuid OSéaghdha, Blaise Thomson, Milica Gašic, 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 NAACL-HLT.

Yonatan Oren, Shiori Sagawa, Tatsunori B Hashimoto, and Percy Liang. 2019. Distributionally robust language modeling. In Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In Proceedings of the Annual Meeting of the Association for Computational Linguistics.

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

Fanchao Qi, Yangyi Chen, Mukai Li, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2020. Onion: A simple and effective defense against textual backdoor attacks. arXiv preprint arXiv:2011.10369.

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

Suranjana Samanta and Sameep Mehta. 2018. Generating adversarial text samples. In Proceedings of the European Conference on Information Retrieval.

Fatemeh Torabi Asr and Maite Taboada. 2019. Big data and quality data for fake news and misinformation detection. Big Data & Society, 6(1):2053951719843310.

Prashanth Vijayaraghavan and Deb Roy. 2019. Generating black-box adversarial examples for text classifiers using a deep reinforced model. In Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases.

Catherine Wong. 2017. Dancin seq2seq: Fooling text classifiers with adversarial text example generation. arXiv preprint arXiv:1712.05419.

Liang Wu, Fred Morstatter, Kathleen M Carley, and Huan Liu. 2019. Misinformation in social media: definition, manipulation, and detection. ACM SIGKDD Explorations Newsletter.

Fan Yang, Arjun Mukherjee, and Eduard Dragut. 2017. Satirical news detection and analysis using attention mechanism and linguistic features. In EMNLP.

Yuan Zang, Fanchao Qi, Chenghao Yang, Zhiyuan Liu, Meng Zhang, Qun Liu, and Maosong Sun. 2020. Word-level textual adversarial attacking as combinatorial optimization. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.

Wei Emma Zhang, Quan Z Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial attacks on deep-learning models in natural language processing: A survey. ACM Transactions on Intelligent Systems and Technology.

Zhixuan Zhou, Huankang Guan, Meghana Moorthy Bhat, and Justin Hsu. 2019. Fake news detection via NLP is vulnerable to adversarial attacks. In Proc. of the 11th Int. Conf. on Agents and Artificial Intelligence, ICAART'19, pages 794–800.

A Ablation Study: Multiple-Word Masking

In the Rewrite stage, we mask a span of multiple words in each iteration. Intuitively, when using a smaller span size, the masked words are easier to predict. The proposal distribution will assign high probability to the original words at masked positions. Therefore, the candidate sentences are likely to be identical to the original sentence, thus limiting the number of perturbations explored. When the span is large, predicting words becomes more difficult. Thus, we can sample different candidate sentences. But it is more likely to construct dissimilar or influential sentences. We vary the span size from 1, 2, 3, to 4 and show the results on Figure 6. We find that using span size 3 yields the best performance over most similarity thresholds.

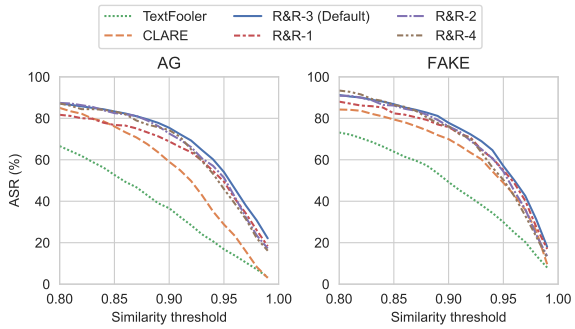

Figure 6: The ASR of R&R using different masking span sizes. R&R-1 to R&R-4 represent the span size of 1 to 4 respectively. We use span size 3 by default.

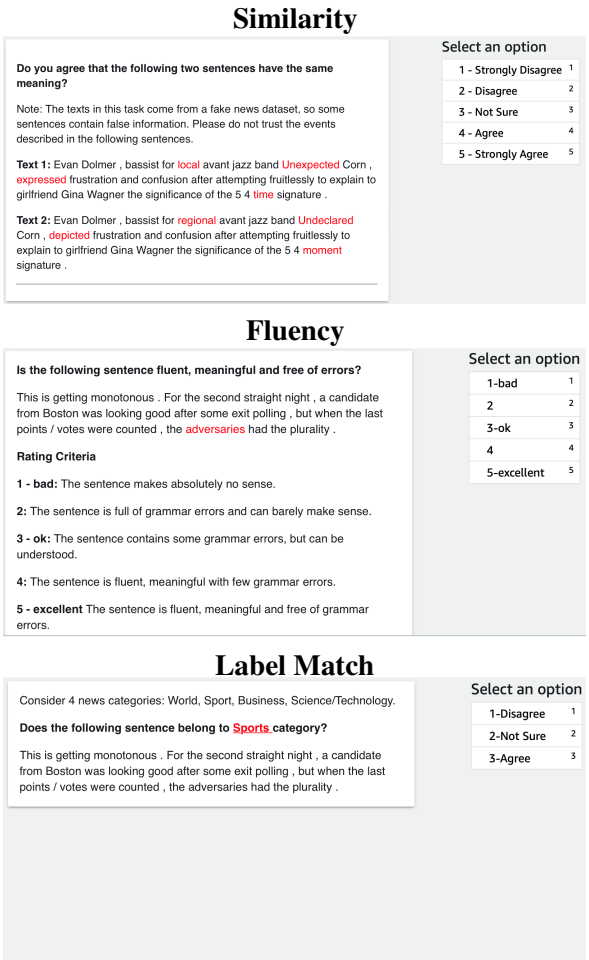

Figure 7: The screenshots of MTurk tasks.

Original (prediction: Technology): GERMANTOWN , Md . A Maryland - based private lab that analyzes criminal - case DNA evidence has fired an analyst for allegedly falsifying test data . **Adversarial (prediction: Business):** GERMANTOWN , Md . A Maryland - based bio testing company that analyzes criminal - case DNA evidence has fired an analyst for allegedly falsifying test data .
Original (prediction: Sport): LeBron James scored 25 points , Jeff McInnis added a season - high 24 and the Cleveland Cavaliers won their sixth straight , 100 - 84 over the Charlotte Bobcats on Saturday night . **Adversarial (prediction: World):** LeBron James scored 25 points , Jeff McInnis added a season - high 24 and the Cleveland Cavaliers won their sixth straight , 100 - 84 Saturday over the visiting Charlotte Bobcats on Saturday night ..
Original (prediction: Negative): don ' t be fooled by the impressive cast list - eye see you is pure junk . **Adversarial (prediction: Positive):** don ' t be fooled by this impressive cast list - eye see you is pure junk .
Original (prediction: Ask for description): What is die - casting ? **Adversarial (prediction: Ask for entity):** What is the technique of die - casting ?
Original (prediction: Toxic) go back under your rock u irrelevant party puppet **Adversarial (prediction: Harmless)** go back under the rock u irrelevant party puppet

Table 6: A few adversarial examples generated by R&R with the perturbation in red

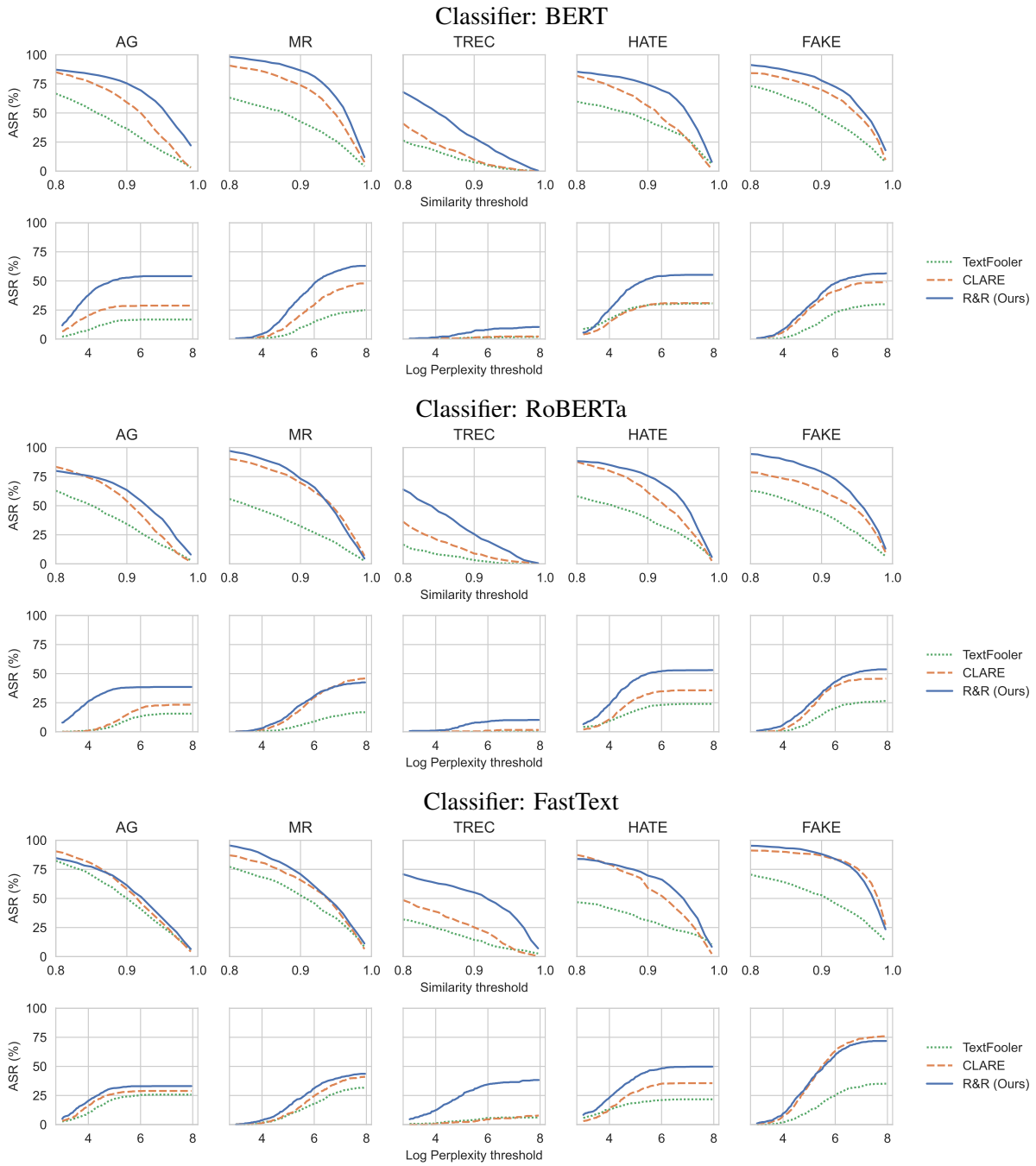

Figure 8: Attack success rate with respect to different similarity and perplexity constraints. When evaluating different similarity thresholds, we do not set thresholds on perplexity. When evaluating perplexity thresholds, we fix the similarity threshold to 0.95.