Fooling the Textual Fooler via Randomizing Latent Representations

Duy C. Hoang[1], **Quang H. Nguyen**[1], **Saurav Manchanda**[2*],
MinLong Peng[3], **Kok-Seng Wong**[1], **Khoa D. Doan**[1]

[1]College of Engineering and Computer Science, VinUniversity, Vietnam
[2]Amazon, USA
[3]Cognitive Computing Lab, Baidu Research, China

{duy.hc, quang.nh}@vinuni.edu.vn, sauravm.kgp@gmail.com,
pengminlong@baidu.com, {wong.ks, khoa.dd}@vinuni.edu.vn

Abstract

Despite outstanding performance in a variety of Natural Language Processing (NLP) tasks, recent studies have revealed that NLP models are vulnerable to adversarial attacks that slightly perturb the input to cause the models to misbehave. Several attacks can even compromise the model without requiring access to the model architecture or model parameters (i.e., a black-box setting), and thus are detrimental to existing NLP applications. To perform these attacks, the adversary queries the victim model many times to determine the most important parts in an input text and transform. In this work, we propose a lightweight and attack-agnostic defense whose main goal is to perplex the process of generating an adversarial example in these query-based black-box attacks; that is to fool the textual fooler. This defense, named AdvFooler, works by randomizing the latent representation of the input at inference time. Different from existing defenses, AdvFooler does not necessitate additional computational overhead during training nor does it rely on assumptions about the potential adversarial perturbation set while having a negligible impact on the model's accuracy. Our theoretical and empirical analyses highlight the significance of robustness resulting from confusing the adversary via randomizing the latent space, as well as the impact of randomization on clean accuracy. Finally, we empirically demonstrate near state-of-the-art robustness of AdvFooler against representative adversarial attacks on two benchmark datasets. Code for AdvFooler is available at: https://github.com/mail-research/AdvFooler-text-defender

1 Introduction

In the last decade, deep neural networks have achieved impressive performance in the Natural Language Processing (NLP) domain. Several deep NLP models have reached state-of-the-art results in several NLP tasks (Devlin et al., 2019) using models such as Recurrent Neural Networks, Transformers, and Pretrained Language Models (PrLMs). However, several works (Iyyer et al., 2018) also reveal that deep NLP models can be fooled by the creation of adversarial examples (Jin et al., 2020). Adversarial examples are synthetically perturbed inputs that are optimized to increase the errors between the predictions of the model and the true labels while being imperceptible to human evaluators (Jin et al., 2020; Iyyer et al., 2018). These developments have sparked concerns about the security and robustness of deep neural networks deployed in NLP applications.

To generate adversarial examples, adversarial attack methods manipulate different aspects of the input sentence, from introducing character errors such as typos or visually similar characters (Gao et al., 2018a; Eger et al., 2019) and replacing words without significantly changing the original semantic in the perturbed inputs (Jin et al., 2020; Li et al., 2020) to recreating sentences with similar meanings using paraphrasing (Iyyer et al., 2018; Qi et al., 2021). Among these methods, black-box adversarial attacks (Alzantot et al., 2018) are some of the most widely studied and effective adversarial approaches. These methods query the victim model multiple times to find the corresponding adversarial perturbations of important words of an input text.

In response to the adversarial threat and to enhance the robustness of NLP models, numerous defenses against textual adversarial attacks have been developed (Zhu et al., 2020; Si et al., 2021; Wang et al., 2021; Ye et al., 2020; Shi et al., 2020a; Xu et al., 2020). The most popular approaches are adversarial training (Zhu et al., 2020; Dong et al., 2021; Madry et al., 2018), which augments the training data with adversarial examples using an additional optimization step, and randomized

---

*The work was done prior to joining Amazon.

smoothing (Ye et al., 2020; Zeng et al., 2023), which replaces the model with its stochastic ensemble based on random perturbations of the input; both of these approaches require training modifications and additional non-trivial training computation (e.g., for adversarial training). Due to the discrete nature of texts, to create adversarial inputs, these defenses also substitute input words with adversarial words sampled from predefined perturbation sets constructed based on the potential attacks instead of gradient-based optimization. This substitution process also aims to preserve the original input semantics. Nonetheless, assuming knowledge of the potential attacks is often unrealistic and impractical.

In this paper, we propose a *lightweight*, *attack-agnostic* defensive method that can increase the robustness of NLP models against textual adversarial attacks. Our defense, called **AdvFooler**, *randomizes the latent representation* of the input *at test time* to *fool the adversary throughout their attack*, which typically involves iteratively sampling of discrete perturbations to generate an adversarial sample. Being a test-time defense with negligible computational overhead, AdvFooler also does not incur any training-time computation. Furthermore, as AdvFooler operates within the latent space of the model, it remains agnostic of the perturbation sets associated with potential attacks. While there are some randomization defenses in NLP (Ye et al., 2020; Zeng et al., 2023), they rely on randomized smoothing with necessary modifications to the model's training to reduce the variance in the model's outputs caused by randomizing the input or embedding space. The advantages of AdvFooler over the other defenses are shown in Table 1. Our contributions can be summarized as follows:

- We propose a lightweight, attack-agnostic, and pluggable defensive method that hinders the attacker's ability to optimize for adversarial perturbations leading to adversarial examples. Consequently, the attack success rate is significantly decreased, making the model more robust against adversarial attacks.

- We provide important theoretical and empirical analyses showing the impact of randomizing latent representations on perplexing the attack process of the adversary.

- We extensively evaluate AdvFooler through experiments on various benchmark datasets and rep-

Table 1: Characteristics of all defenses. $\times$ or $\checkmark$ indicates the method lacks or has the specific characteristic, respectively. AdvFooler satisfies all the characteristics by adding lightweight randomization to the latent space.

Method	no training required	randomized	trivial inference overhead	no additional network	pluggable
ASCC	$\times$	$\times$	$\checkmark$	$\checkmark$	$\times$
InfoBERT	$\times$	$\times$	$\checkmark$	$\checkmark$	$\times$
FreeLB	$\times$	$\times$	$\checkmark$	$\checkmark$	$\times$
TMD	$\times$	$\times$	$\times$	$\times$	$\times$
SAFER	$\times$	$\checkmark$	$\times$	$\checkmark$	$\times$
RanMASK	$\times$	$\checkmark$	$\times$	$\checkmark$	$\times$
AdvFooler	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$

resentative attacks. The results demonstrate that AdvFooler is a competitive defense, compared to the existing representative textual adversarial defenses, while being under more constraints, including few modeling assumptions, being pluggable, and incurring negligible additional computational overhead.

2 Background

2.1 Adversarial attacks in NLP

Adversarial attacks in NLP can be divided into two categories, depending on what information the attacker has on the victim model. These categories are white-box attacks (Ebrahimi et al., 2018; Cheng et al., 2020) - those that require access to the model's architecture and its parameters - and black-box attacks (Li et al., 2020; Jin et al., 2020; Li et al., 2018) - those that rely only querying the model for its output. Since the model user rarely shares the underlying model's architecture of their NLP service, let alone their model's parameters, white-box attacks have extremely limited practicality, compared to black-box attacks. In the paper, we will focus on the black-box, query-based attack setting, as it is a more realistic scenario in practice and is commonly studied in similar works (Zeng et al., 2023; Nguyen Minh and Luu, 2022; Zhang et al., 2022). Note that, there are also black-box, transfer-based attacks, which, however, still require knowledge of the model architecture for the attack to be effective (Yuan et al., 2021; Li et al., 2021a).

Black-box Query-based Attack. Given the text classification task, a model $f : \mathcal{R}^d \rightarrow \mathcal{R}^{|\mathcal{C}|}$ maps an input $x \in \mathcal{R}^d$ to a logit vector of dimension $|\mathcal{C}|$, where $\mathcal{C}$ is the set of label. The goal of textual adversarial attacks is to search for adversarial examples on which the model makes incorrect predictions. Specifically, given an input sentence x, a corresponding adversarial example x' can be

crafted to satisfy the following objective:

$$\arg\max_c f(x_{adv}) \neq \arg\max_c f(x)$$
$$\text{s.t.} \quad d(x_{adv} - x) \leq \sigma \quad (1)$$

where $d(x_{adv} - x)$ is the perceptual distance between x and x_{adv}, and σ is the maximum acceptable distance. For example, d can measure the number of perturbed words between the original and adversarial input (Gao et al., 2018b). Another popular distance is the semantic dissimilarity between a pair of texts (Iyyer et al., 2018; Li et al., 2020), defined as the cosine distance between the extracted embedding vectors of the inputs from Universal Sentence Encoder (USE) (Cer et al., 2018).

Solving the objective in Eq. (1) is equivalent to searching for perceptually valid perturbations that flip the prediction of the victim model. However, this is a challenging task due to the considerably large search space for textual data. Specifically, the works in (Mrkšić et al., 2016; Li et al., 2018) demonstrate that perturbing random words in the sentence fails to effectively fool the model with a reasonable number of queries. To accelerate the search procedure, existing attacks locate important parts of the sentence, e.g., word (Li et al., 2018; Jin et al., 2020; Li et al., 2020) or group of words (Chen et al., 2021; Lei et al., 2022), to perturb.

Among textual adversarial attacks, adversarial word substitution is one of the most widely studied approaches. These attacks create an adversarial example of an input text x by first identifying the most important words in x and then replacing those words with synonyms from their corresponding synonym sets to cause the victim model to misbehave while aiming to preserve the original semantic meaning. The synonym sets can be created using word embedding models (Mrkšić et al., 2016; Pennington et al., 2014) or using the predictions of a PrLMs given the input sentence (Li et al., 2020). One important feature of these attacks is that both of the aforementioned steps require querying the victim model many times to determine which words are important or which synonyms maximize the model's prediction errors. In the first step, the adversary queries the model to calculate the importance score of each word in x and select the most potential locations (with the highest importance scores) for later perturbations. To find the substituting synonyms that maximize the model's prediction error, the attacks can either perform a greedy search (Li et al., 2020; Jin et al.,

2020; Li et al., 2018) or combinatorial optimization algorithm (Zang et al., 2020; Alzantot et al., 2018). Chen et al. (2021) extend this principle to multi-granularity by determining important constituents in the syntactic tree that maximizes the decrease in confidence score when being paraphrased. Similarly, this step involves a sequence of queries on the victim model. By introducing randomization to the latent representations of queries, our defensive mechanism disrupts the adversary's ability to estimate important words (or paraphrases), making their synonym-substituting process significantly harder during the attack.

2.2 Adversarial defense methods

To defend against textual adversarial attacks, several defenses (Wang et al., 2022) have been developed. The goal of an adversarial defense is to improve the robustness of the victim model; that is to achieve good performance on both clean and adversarial examples. Existing adversarial defenses can be divided into two categories: certified (Ye et al., 2020; Zeng et al., 2023; Jia et al., 2019) and empirical (Zhu et al., 2020; Wang et al., 2021; Dong et al., 2021; Li and Qiu, 2020; Zhou et al., 2021; Miyato et al., 2017).

First introduced in Goodfellow et al. (2014), adversarial training is the most often studied empirical defense method. It provides additional regularization to the model and improves the model's robustness when training it with adversarial samples. Several subsequent works study adversarial training for NLP tasks with adversarial examples created on the input space (Ren et al., 2019; Li et al., 2020; Jin et al., 2020). Some adversarial training works improve the model's robustness by introducing adversarial perturbations in the embedding space (Zhu et al., 2020), or by incorporating inductive bias to prevent the model from learning spurious correlations (Wang et al., 2021; Madry et al., 2018), as well as methods that increase diversity of ensembled models, leading to globally better robustness (Pang et al., 2019). These methods effectively improve the model's robustness without significant compromise to its clean accuracy.

In contrast to the empirical methods, certified defenses can provably guarantee model robustness even under sophisticated attackers. One popular certified-defense approach is randomized smoothing (Ye et al., 2020; Zeng et al., 2023), which constructs a set of stochastic ensembles from the input

and leverages their statistical properties to provably guarantee robustness. Differential Privacy (DP) defenses such as (Lecuyer et al., 2019) are similar to SAFER and RanMASK, achieving certified robustness through the use of DP framework. Note that RanMASK and DP-defenses incur high training overhead. In contrast, Interval Bound Propagation (IBP) methods (Shi et al., 2020b; Jia et al., 2019; Huang et al., 2019) utilizes axis-aligned bounds to confine adversarial examples. Despite their effectiveness and ability to certify robustness, both approaches rely on access to the synonym sets of potential attacks. Randomized-smoothing methods also modify the training phase to reduce the variance of the outputs of the ensemble. Due to the ensemble, they incur significant computational overhead in the inference phase. In contrast, our defense is lightweight and attack-agnostic, and can better control the variance of the outputs by randomizing the latent representations, instead of via randomization in the input space. A summary of the characteristics of AdvFooler and the existing defenses is shown in Table 1.

3 Methodology

3.1 Randomized latent-space defense against adversarial word substitution

---
Algorithm 1 AdvFooler

Input: a model f, an input sentence x, scale ν.
Output: output logit l.
1:   $z_0 = Emb(x)$
2: **for** layer l in $\{1...L\}$ of model f **do**
3:     $\epsilon \sim \mathcal{N}(0, \nu I)$
4:     $z_{i+1} = h_l(z_i + \epsilon)$
5: **end for**
6: **return:** z_L

Since the adversary relies on querying the victim model multiple times to find adversarial examples, receiving inconsistent feedback from the model can make it significantly harder for the adversary to find the optimal adversarial perturbations. To fool such an adversary, we introduce stochasticity to the model by randomizing the latent representation of an input x. Formally, for h_l as the $l-$th layer of the model, we sample an independent noise vector ϵ, which is added to $h_l(x)$ as input to the next layer of the model. Without loss of generality, ϵ is sampled from a Gaussian distribution $\mathcal{N}(0, \Sigma)$ with Σ as a diagonal covariance matrix, or $\mathcal{N}(0, \nu I), \nu \in \mathcal{R}$. The detailed algorithm is presented in Algorithm 1.

Note that, instead of randomizing the latent representation, the defender can directly randomize the textual input; however, this approach makes it significantly harder to control the corresponding change in the prediction, or to maintain an acceptable level of the model's performance when applying the defense.

Let $f_{\text{AdvFooler}}$ be the proposed randomized model corresponding to the original f. When the variance of injected noise is low, we can assume that small noise slightly changes the output logits but does not shift the prediction of the model.

3.2 Effect of randomizing latent representations on adversarial attacks

In textual adversarial attacks, the adversary first determines important parts in a sentence $x = \{w_0, \ldots, w_l\}$ to reduce the search space by using the importance score, which is defined as the difference in the confidence:

$$I_{\tilde{x}} = f_y(x) - f_y(\tilde{x}), \quad (2)$$

where f_y is the logit returned by the model f w.r.t ground-truth label y and $\tilde{x}$ is the perturbed sentence. For example, Li et al. (2018) and Jin et al. (2020) remove a word w_i in the sentence x, while Li et al. (2020) replace w_i by the token $MASK$ to obtain $\tilde{x} = \{w_0, \ldots, w_{i-1}, MASK, w_{i+1}, \ldots, w_l\}$. On the other hand, phrase-level attacks transform x by paraphrasing constituents extracted from the syntactic tree (Chen et al., 2021) or masking them out (Lei et al., 2022). The adversary then selects perturbations with the highest scores. Intuitively, this process emulates estimating the gradient of the importance score in the discrete space.

When introducing randomness to the latent space, the model output is changed. Such changes can mislead the attacker to select ineffective perturbations. For example, the attacker may instead select "less important" words (i.e., those that have lower original importance scores before randomization) to perturb, thus reducing the attack's success probability (Mrkšić et al., 2016; Li et al., 2018).

Theorem 3.1. *If a random vector $v \sim \mathcal{N}(0, \nu I)$ where ν is small is added to the hidden layer h of the model f which can be decomposed into $f = g \circ h$, the new important score $I_{\tilde{x}}^{new}$ is a random variable that follows Gaussian distribution $\mathcal{N}(I_{\tilde{x}}, \nu(\|\nabla_{h(x)} g_y(h(x))\|_2 + \|\nabla_{h(\tilde{x})} g_y(h(\tilde{x}))\|_2))$.*

Theorem 3.1 (proof is in the Supplementary) states that randomly perturbing the hidden presentation leads to a randomized important score. When ν is high, the randomized importance scores have high variance, which can cause the attack to select the wrong important words or phrases.

3.3 Effect of randomizing latent representations on clean accuracy

If randomization induces substantial fluctuation in the model's output, there is a possibility that the model might predict a wrong label, resulting in a decrease in clean accuracy. Randomized defenses such as SAFER (Ye et al., 2020) and RanMASK (Zeng et al., 2023) introduce randomness to the model in the input space (i.e., randomizing the input tokens). While they are also effective against textual adversarial attacks, the input space of NLP models is discrete. Perturbing discrete inputs can lead to significant changes in the model's outputs (a fact that we will empirically prove in the next section), which results in a different model's prediction. To mitigate this problem, existing randomized input defenses rely on randomized smoothing: train the model with perturbed input and apply ensemble at inference to reduce the variance in the model's output. Nevertheless, these approaches are computationally expensive and require access to the training phase. Conversely, by randomizing the latent representations, AdvFooler induces a significantly smaller variance in the model's output. Furthermore, AdvFooler gives the defender flexibility to select the suitable induced variance without requiring access to the training process. Specifically, given a pretrained classification model and a small clean test set, the defender can select the noise scale ν at which the variance causes the clean accuracy drops by a chosen percentage (e.g., in our experiments, it is 1%).

3.4 Empirical analysis

In this section, we empirically study the effect on the model's performance and attack process of various randomized defenses, including AdvFooler.

Empirical effect on model's performance. We compute the differences in cross-entropy losses between those of the randomized model and those of the base model for each sample in the AGNEWS test dataset, which expresses how the predictions vary, under different randomization approaches. Figure 1 shows the loss changes in SAFER, RanMASK, and AdvFooler. As we can observe, in SAFER and RanMASK, perturbing the input to-
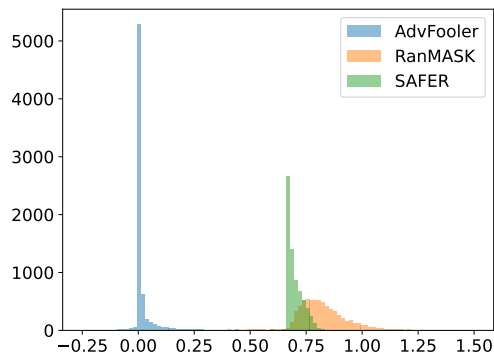

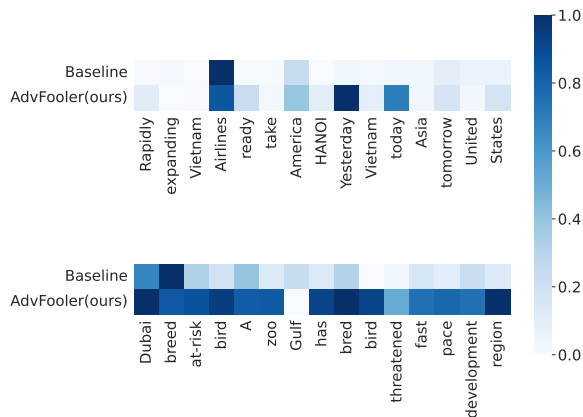Figure 1: Loss changes when randomizing input (RanMASK/SAFER) and latent space (AdvFooler).

Figure 2: Illustration of each word's important score when calculated with and without AdvFooler.

kens induces a higher variance in the loss compared to AdvFooler. Consequently, both of these methods lead to significant drops in the model's performance without adversarial training and ensemble.

Empirical effect on fooling the attacks. To demonstrate that AdvFooler can fool the adversary into selecting non-important words, we calculate the important score of each word for multiple samples from the AGNEWS dataset. The important words before and after applying AdvFooler are presented in Figure 2. As we can observe, Adv-Fooler causes the attack to select different important words, many of which are originally unimportant. Specifically, in the first sentence, the token "Airlines" is the most important. After adding random noise to the latent representation, AdvFooler changes the important score for each token enough to mislead the adversary into thinking a different token ("Yesterday" token) is the most important token. A similar phenomenon can be seen in the second sentence, where the important scores change significantly, and unimportant tokens become important ones after randomization. Note that, these experiments use the random noise scale ν that in-

Table 2: The robustness performance of AdvFooler and other defenses on AGNEWS. The best and second-best performances are **bolded** and underlined, respectively.

Models	Clean% (Drop%)	TextBugger			TextFooler			BERT-Attack		
		AuA% (ASR%)	#Query		AuA% (ASR%)	#Query		AuA% (ASR%)	#Query	
BERT-base (baseline)	95.14	35.20 (62.63)	351.92		36.80 (60.93)	317.27		46.50 (50.64)	337.84	
+ASCC	90.07 (-4.13)	39.30 (55.94)	284.58		41.20 (53.81)	262.07		45.90 (48.54)	263.59	
+FreeLB	95.07 ($+0.87$)	47.30 (49.89)	394.19		49.90 (47.14)	356.69		52.00 (44.92)	370.82	
+InfoBERT	95.01 ($+0.81$)	45.20 (52.07)	373.83		47.50 (49.63)	336.77		<u>56.00 (40.62)</u>	366.40	
+TMD	94.36 ($+0.16$)	47.80 (49.04)	**833.43**		<u>51.60 (44.81)</u>	**744.87**		52.60 (43.74)	**766.72**	
+RanMASK	90.14 (-4.06)	**52.50 (41.60)**	582.21		**54.60 (38.93)**	511.98		**61.10 (31.73)**	595.57	
+SAFER	94.42 ($+0.22$)	43.30 (53.74)	370.77		46.70 (50.05)	333.17		51.40 (45.03)	357.47	
+AdvFooler (ours)	93.67 (-0.53)	<u>50.10 (45.90)</u>	<u>819.44</u>		50.10 (45.90)	<u>701.05</u>		53.40 (42.89)	<u>752.61</u>	
RoBERTa-base (baseline)	95.05	38.40 (58.93)	371.20		41.00 (56.15)	335.88		47.30 (49.41)	372.99	
+ASCC	92.61 (-2.44)	49.50 (46.37)	372.91		51.00 (44.75)	330.69		55.70 (39.65)	374.58	
+FreeLB	95.03 (-0.02)	50.50 (46.22)	448.97		52.00 (44.62)	402.14		<u>58.20 (38.02)</u>	442.97	
+InfoBERT	94.97 (-0.08)	47.00 (50.11)	394.56		48.90 (48.09)	357.44		55.00 (41.61)	397.35	
+TMD	93.70 (-1.35)	**52.80 (43.59)**	**878.22**		**55.20 (41.09)**	**766.74**		57.90 (38.14)	**805.19**	
+RanMASK	90.41 (-4.64)	<u>52.30 (41.50)</u>	575.94		54.20 (39.37)	516.98		**60.00 (32.43)**	594.93	
+SAFER	94.67 (-0.38)	47.70 (49.15)	377.67		50.30 (46.43)	337.81		53.20 (43.22)	357.56	
+AdvFooler (ours)	94.21 (-0.84)	51.70 (44.88)	<u>804.99</u>		<u>53.30 (42.81)</u>	<u>717.31</u>		57.80 (38.38)	<u>767.47</u>	

duces at most 1% clean performance drop.

4 Experiments

4.1 Experimental setup

Evaluation Metrics. We evaluate the defenses using four metrics: Clean accuracy (**Clean%**), Accuracy under attack (**AuA%**), Attack success rate (**ASR%**), Average number of queries (**#Query**). Clean% measures the defense's impact on the clean performance. Both AuA% and ASR% measure the model's robustness under attack. #Query is the averaged number of queries used by the adversary to find adversarial examples; the higher #Query is, the more difficult the attack is.

Datasets and Model Architectures. We evaluate the defenses on two benchmark datasets: AG-News Corpus (**AGNEWS**) (Zhang et al., 2015) and Internet Movie Database (**IMDB**) (Maas et al., 2011). To assess the generality of AdvFooler on model architectures, we evaluate it on two state-of-the-art, pre-trained language models: BERT$_{base}$ (Devlin et al., 2019) and RoBERTa$_{base}$ (Liu et al., 2020). More details are provided in the Appendix.

Adversarial Attacks and Parameters. We choose the following widely used and state-of-the-art adversarial attacks to evaluate our defense: TextFooler (Jin et al., 2020), TextBugger (Li et al., 2018), and BERT-Attack (Li et al., 2020). Both TextFooler and BERT-Attack rely on the important scores to identify important words and replace them with the corresponding synonyms; TextFooler uses word embeddings from (Mrkšić et al., 2016) for synonym generations, while BERT-Attack uses pretrained BERT to generate the synonyms. In addition to replacing words, TextBugger also introduces

word augmentation - perturbing the characters in words.

Baseline Defenses. We compare AdvFooler against various types of defense methods. For empirical defenses, we select **Adversarial Sparse Convex Combination** (ASCC) (Dong et al., 2021), **InfoBERT** (Wang et al., 2021), **FreeLB** (Zhu et al., 2020), and **Textual Manifold Defense** (TMD) (Nguyen Minh and Luu, 2022). ASCC and FreeLB improve the model's robustness by employing adversarial training; FreeLB introduces adversarial perturbations to word embeddings, while ASCC generates adversarial examples from a convex hull spanned by word substitution. InfoBERT incorporates information theory to refine both local and global features. Textual Manifold Defense reduces the effects of the adversarial examples by projecting the embeddings of input sentences to an approximated embedding manifold. For certified defenses, we evaluate on **RanMASK** (Zeng et al., 2023) and **SAFER** (Ye et al., 2020), which construct a set of randomly perturbed inputs using token masking and synonym substitution, respectively. Then, they leverage the statistical properties of the predicted output to achieve better model robustness.

Implementation Details. For a fair evaluation, we follow the implementation guidelines for evaluating adversarial attacks and defenses in Li et al. (2021b): (1) the maximum percentage of modified words ρ_{max} for AGNEWS and IMDB are 0.3 and 0.1, respectively, (2) the maximum number of candidate replacement words K_{max} is set to 50, (3) the maximum percentage of modified words ρ_{max} for AGNEWS and IMDB must be 0.3 and

Table 3: The robustness performance of AdvFooler and other defenses on IMDB. The best and second-best performances are **bolded** and <u>underlined</u>, respectively.

Models	Clean% (Drop%)	TextBugger		TextFooler		BERT-Attack	
		AuA% (ASR%)	#Query	AuA% (ASR%)	#Query	AuA% (ASR%)	#Query
BERT-base (baseline)	92.14	9.20 (90.08)	500.47	11.90 (87.16)	439.15	8.90 (90.40)	366.52
+ASCC	88.48 (−3.66)	13.00 (85.68)	597.07	16.90 (81.20)	529.55	7.70 (91.52)	416.50
+FreeLB	92.33 (+0.19)	25.80 (71.74)	776.31	28.90 (68.35)	670.20	21.80 (76.12)	549.85
+InfoBERT	91.71 (−0.43)	22.50 (75.52)	719.98	25.30 (72.47)	645.03	20.90 (77.26)	510.46
+TMD	92.14 (−0.00)	40.40 (56.47)	<u>3251.52</u>	45.10 (51.35)	<u>2735.58</u>	36.20 (60.95)	<u>2464.83</u>
+RanMASK	92.61 (+0.47)	31.00 (66.38)	2740.39	35.80 (61.13)	2392.46	33.10 (64.33)	2463.24
+SAFER	92.12 (−0.02)	**46.10 (50.00)**	1455.25	**50.50 (45.64)**	1262.35	**43.10 (52.74)**	1133.61
+AdvFooler (ours)	91.90 (−0.24)	<u>42.40 (53.41)</u>	**3261.41**	<u>49.10 (47.32)</u>	**2759.37**	<u>40.70 (55.76)</u>	**2645.36**
RoBERTa-base (baseline)	93.23	6.90 (92.63)	517.58	11.40 (87.79)	456.35	8.20 (91.29)	439.79
+ASCC	92.62 (−0.61)	14.70 (84.31)	770.35	20.20 (77.92)	606.14	15.20 (83.57)	548.34
+FreeLB	94.20 (+0.97)	25.40 (72.95)	887.31	29.80 (68.30)	726.84	23.10 (75.19)	647.98
+InfoBERT	94.18 (+0.95)	20.90 (78.14)	684.64	27.60 (70.61)	583.97	15.50 (83.53)	518.17
+TMD	93.22 (−0.01)	**66.10 (29.30)**	**4799.56**	**66.40 (28.83)**	**3477.32**	**56.00 (40.36)**	**3330.08**
+RanMASK	94.33 (+1.10)	49.40 (47.45)	3611.03	54.10 (43.29)	2951.45	44.20 (53.67)	2706.83
+SAFER	93.75 (+0.52)	**62.30 (34.21)**	2059.37	63.60 (32.27)	1874.79	<u>54.10 (41.77)</u>	1415.56
+AdvFooler (ours)	92.69 (−0.54)	62.20 (33.55)	<u>4205.77</u>	<u>63.70 (32.02)</u>	<u>3255.35</u>	49.40 (47.00)	<u>3141.88</u>

0.1, respectively, and (4) the maximum number of queries to the victim model is $Q_{max} = K_{max} * L$, where L is the length of, or the number of tokens in, the input sentence. Clean% is calculated on the entire test set from each dataset, while the robustness metrics AuA%, ASR%, and #Query are computed on 1,000 randomly chosen samples from the test set. All attacks are based on TextAttack implementation (Morris et al., 2020).

4.2 Defense performance

We report the defense performance on AGNEWS in Table 2. As we can observe in this table, on the base model BERT, AdvFooler outperforms all adversarial training techniques (ASCC, FreeLB, and InfoBERT) and the randomized smoothing method, SAFER. AdvFooler is consistently in the top-3 robust defenses; the robustness of AdvFooler and TMD are within 1-2% of each other, while they are generally both lower than the robustness of another randomized-smoothing defense, RanMASK. However, the clean accuracy in RanMASK drops significantly, approximately 4%, while the clean accuracy in AdvFooler is guaranteed to be within 1% of the base model. Note that, all of the evaluated textual defenses require access to the training data. On the base model RoBERTa, AdvFooler similarly achieves a top-3 performance.

For IMDB's results in Table 3, we can observe similar robustness results. TMD's performance is observed to be consistently better with RoBERTa-base. AdvFooler is effective against the attacks and achieves comparable results to the best defenses (SAFER and TMD).

In summary, AdvFooler achieves competitive robustness to the state-of-the-art defenses; however, AdvFooler has significantly lower training and inference overhead compared to TMD, RandMASK, and SAFER and does not have access to the training data.

Difficulty of Adversarial Attacks against AdvFooler. For all the considered adversarial attacks, the number of queries to locate important words (the first step) of an input text is the same with or without the defenses. Thus, the higher number of queries in an experiment is a result of a more difficult synonym-replacement phase. This also explains considerably smaller average numbers of queries for the baseline (without any defense) in Tables 2 and 3, compared to the protected models.

We can observe, in Tables 2 and 3, that the average numbers of queries for the adversarial attacks to be successful against AdvFooler are significantly higher than those in all the defenses except TMD in a few cases. The results demonstrate the effectiveness of fooling the attacks in AdvFooler. As explained in Section 3, randomizing the important scores just enough makes the adversary select a different set of important words, which leads to a more challenging synonym-replacement process with a significantly higher number of queries.

AdvFooler with other attacks. In the Supplementary, we provide additional experiments evaluating AdvFooler against other types of attacks, including black-box hard-label (Appendix 4.7) and white-box attacks (Appendix C.5), even though they do not align with the threat model studied in this work and several related works (Nguyen Minh and Luu,
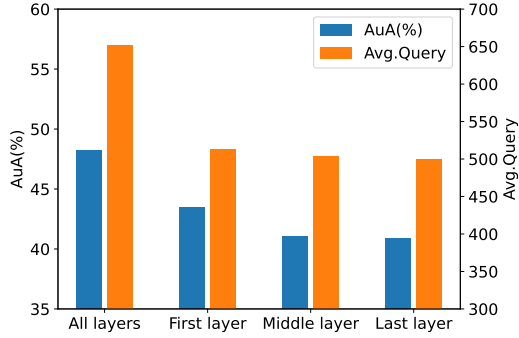
Figure 3: The robustness of AdvFooler when randomizing different layers of the model on AGNEWS.

2022; Zhang et al., 2022; Zeng et al., 2023; Li et al., 2023) to AdvFooler. The results show that AdvFooler is still effective in defending against these attacks, further highlighting the versatility of AdvFooler.

4.3 Robustness from randomizing different latent spaces

In this section, we study the model's robustness when randomizing different layers of the model. For comparison, Specifically, we randomize only the first, middle, and last layers and all layers of the model. The model's robustness against TextFooler is reported in Figure 3. As we can observe, randomizing the early layer of the model is more effective than randomizing the later layers of the model. More importantly, adding noise to all the layers leads to a significantly higher accuracy under attack and a higher average number of queries.

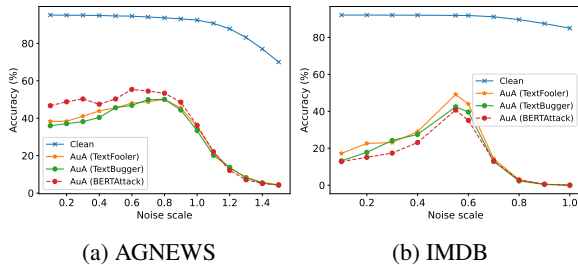### 4.4 Trade-off between clean accuracy and robustness

(a) AGNEWS   (b) IMDB

Figure 4: Clean Accuracy and Accuracy under Attack (AuA) when using different noise scales ν.

As discussed in Section 3, our approach allows the defender to flexibly trade-off between the clean accuracy (or variance in the output logits) and the model's robustness, a feature that is challenging to perform in other defenses. In AdvFooler, this is

accomplished via tuning the noise scale ν. A large ν value leads to both a large variance of the model's output, which can lead to a lower clean accuracy, and a high chance of fooling the adversarial attacks.

We report the model's clean accuracy and robustness when varying the value of ν in Figure 4. As we can observe, increasing ν generally increases the model's robustness against all the evaluated attacks while the clean accuracy only slightly changes. However, when ν reaches a certain value, the clean accuracy begins to decrease, at which point the robustness of the model also decreases. The plot also shows that the noise scale when both accuracy under attack and clean accuracy start to decrease would also be the noise scale that makes the model more robust to all types of attack. This suggests that, in practice, the defender selects as large ν value as possible for a certain small drop of clean accuracy. For example, in our experiments, we select the ν value at which the clean accuracy drops by at most 1% using the test set.

4.5 AdvFooler with adversarially trained models

Table 4: Clean Accuracy and Accuracy under Attack of TextBugger (TB), TextFooler (TF), and BERT-Attack (BA) for FreeLB- and InfoBERT-trained models, with and without using AdvFooler .

Dataset	Models	Clean (%)	AuA(%)		
			TB	TF	BA
AGNEWS	FreeLB	95.07	47.3	49.9	52.0
	FreeLB+AdvFooler	94.61	62.5	63.2	63.2
	InfoBERT	95.01	45.2	47.5	56.0
	InfoBERT+AdvFooler	94.34	64.3	64.8	67.1
IMDB	FreeLB	92.33	25.8	28.9	21.8
	FreeLB+AdvFooler	92.22	49.6	50.3	40.7
	InfoBERT	91.71	22.5	25.3	20.9
	InfoBERT+AdvFooler	91.46	46.6	51.7	42.1

As discussed previously, AdvFooler can be used to improve the robustness of any existing pretrained model; i.e., AdvFooler is a pluggable defense. In fact, AdvFooler can also be combined with other types of defenses to provide another layer of protection and further improve the model's robustness. In this section, we study the effectiveness of AdvFooler in conjunction with adversarial training methods, including FreeLB and InfoBERT.

Table 4 shows the defense performance before and after applying AdvFooler on adversarially trained models using FreeLB and InfoBERT. We can observe that AdvFooler significantly improves the robustness of the adversarially trained models, with up to 20% improved accuracy under attack

in some experiments. While AdvFooler can also be combined with other defensive mechanisms, we leave these to future works.

4.6 Inference time analysis

Table 5: Inference time comparison with other defenses. Tested with BERT on the test set from the IMDB dataset.

Defense	Runtime (s)	% Increase
Baseline	75	
ASCC	90	(+20.0%)
FreeLB	75	(+0.0%)
InfoBERT	75	(+0.0%)
TMD	100	(+33.3%)
RanMask	1752	(+2236.0%)
SAFER	1772	(+2262.6%)
AdvFooler	77	(+2.6%)

In this experiment, we study the computational overhead that each defense introduces at inference time. We use the IMDB test set and record the inference runtime of the BERT model under different defenses. The results are shown in Table 5. From the result, we can see that AdvFooler introduces very little computational overhead compared to most other defenses. Randomized-smoothing methods, such as RanMask and SAFER, incur significant overhead to the inference phase. TMD also adds non-trivial inference overhead (33.3%); as acknowledged by the authors, the main bottleneck of TMD is its high computational overhead of the on-manifold projection (Nguyen Minh and Luu, 2022). Only adversarial training methods, such as ASCC and FreeLB, do not introduce additional overhead during inference time. Most importantly, AdvFooler is the only method that has consistently high robustness and a negligible inference overhead (2.6%).

4.7 AdvFooler's performance on hard-label attacks.

In some cases, the adversary can choose to ignore the output logits and only rely on the hard-label output of the model, as seen in Maheshwary et al. (2021). The so-called hard-label attacks first initialize a sample that flips the prediction, move it toward the original input until it passes through the decision boundary, then select the sample at the previous step as adversarial example. However, since AdvFooler is a randomized model, it might sometimes return labels that are different from the model without noise for the attack queries. Therefore, it can fool the attack to go a bit further, and the sample found by the attack turns out to be non-adversarial. In this section, we study the performance of AdvFooler against this hard-label attack. In this experiment, we ran these attacks against the base model and AdvFooler. The attack may accidentally flag a chosen perturbation as an adversarial example although this sample may be mispredicted due to the added noise in our defense; in other words, the selected sample is not a true adversarial example since, under a different random noise, the model prediction may still be correct. To avoid such a situation and fairly evaluate the performance of AdvFooler , for each "supposedly adversarial example" generated by the attack, we feed it to the model 5 times and record the model's predictions; the model's prediction used for performance evaluation is the prediction occurred most. As we can observe from Table 6, hard-label attack struggles to find consistent Adversarial Examples the bigger the noise being inserted into the model.

5 Conclusion

In this work, we proposed a lightweight, attack-agnostic defense, AdvFooler, that can improve the robustness of NLP models against textual adversarial attacks. Different from existing defenses, AdvFooler does not incur any training overhead nor relies on assumptions of the potential attacks. The main idea of AdvFooler is to mislead word-level perturbation-based adversary into selecting unimportant words by randomizing the latent space of the model, resulting in a significantly more challenging synonym-replacement process. We then provide both theoretical and empirical analyses to explain how AdvFooler fools the adversary, as well as its effect on the model's accuracy. Our extensive experiments validate that AdvFooler improves the robustness of NLP models against multiple state-of-the-art textual adversarial attacks on two datasets. Finally, being a pluggable defense, AdvFooler can also be combined with existing defenses, such as adversarial training, to further protect the NLP models against these threats.

Table 6: Hard-label results against base models with and without AdvFooler under different noise scale.

	Results [AuA%(ASR%)]
BERT-base	58.3% (38.11%)
AdvFooler (0.7 scale)	78.8% (15.35%)
AdvFooler (0.9 scale)	83.3% (9.45%)
AdvFooler (1.1 scale)	85.7% (5.72%)

Limitations

Although AdvFooler is a lightweight, pluggable, and effective defense, there are some limitations that can be improved. Our analysis and empirical evaluation focus on black-box query-based attacks. We do not include black-box attacks that utilize the transferability from a surrogate to a target model, even though our threat model does not make any assumptions about the network architecture or the training dataset. We also do not consider an adaptive attacker, who makes multiple queries to estimate the importance of each word or phrase (i.e., $I_{\tilde{x}}$). The expected value of $I_{\tilde{x}}$ calculated by this adaptive attacker against AdvFooler (i.e., w.r.t model $f_{AdvFooler}$) will be similar to that of the original importance score without the defense (i.e, w.r.t model f). Evaluating AdvFooler against this adaptive attack in conjunction with existing word-level perturbation-based attacks could be an interesting study. Similarly, besides adversarially-trained approaches, it would also be interesting to study the robustness of the combination of AdvFooler and other types of defenses. Finally, for different model architectures, randomizing different latent layers could yield different effects on the robustness. This would be interesting to boost the effectiveness of AdvFooler for specific models. We leave these to future works.

Ethics Statement

The rapid integration of NLP models in various domains and applications has brought significant transformations to our daily lives. Unfortunately, most NLP models face significant vulnerability to textual adversarial attacks, undermining confidence in their deployment and usage. Among the existing textual adversarial attacks, word-level perturbation-based attacks pose a severe threat to the model users since these attacks are the most effective and do not require access to the model architectures or their trained parameters.

To address the risks of these attacks, our work proposes a lightweight, attack-agnostic defense for existing NLP models. Our detailed theoretical and empirical analyses show the defense's comparable performance to the existing state-of-the-art defenses across a wide range of NLP models, textual adversarial attacks, and benchmark datasets. However, our defense does not necessitate any additional computational overhead at both training and inference time and can be used with any existing pretrained models. In summary, the proposed defense can improve the adversarial robustness of existing NLP models against word-level perturbation-based attacks, thereby bolstering user confidence in their utilization for real-world applications.

References

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

Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. 2018. Synthesizing robust adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 284–293. PMLR.

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 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, Brussels, Belgium. Association for Computational Linguistics.

Yangyi Chen, Jin Su, and Wei Wei. 2021. Multi-granularity textual adversarial attack with behavior cloning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4511–4526, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Minhao Cheng, Jinfeng Yi, Pin-Yu Chen, Huan Zhang, and Cho-Jui Hsieh. 2020. Seq2sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):3601–3608.

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

Xinshuai Dong, Anh Tuan Luu, Rongrong Ji, and Hong Liu. 2021. Towards robustness against natural language word substitutions. In *International Conference on Learning Representations*.

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 (Volume 2: Short Papers)*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

Steffen Eger, Gözde Gül Şahin, Andreas Rücklé, Ji-Ung Lee, Claudia Schulz, Mohsen Mesgar, Krishnkant Swarnkar, Edwin Simpson, and Iryna Gurevych. 2019. Text processing like humans do: Visually attacking and shielding NLP systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1634–1647, Minneapolis, Minnesota. Association for Computational Linguistics.

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

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

SongYang Gao, Shihan Dou, Yan Liu, Xiao Wang, Qi Zhang, Zhongyu Wei, Jin Ma, and Ying Shan. 2023. DSRM: Boost textual adversarial training with distribution shift risk minimization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12177–12189, Toronto, Canada. Association for Computational Linguistics.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572.

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 *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4083–4093, Hong Kong, China. Association for Computational Linguistics.

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)*, pages 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4129–4142, Hong Kong, China. Association for Computational Linguistics.

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. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8018–8025.

Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. 2019. Certified robustness to adversarial examples with differential privacy. In *2019 IEEE symposium on security and privacy (SP)*, pages 656–672. IEEE.

Yibin Lei, Yu Cao, Dianqi Li, Tianyi Zhou, Meng Fang, and Mykola Pechenizkiy. 2022. Phrase-level textual adversarial attack with label preservation. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1095–1112, Seattle, United States. Association for Computational Linguistics.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. Textbugger: Generating adversarial text against real-world applications. *ArXiv*, abs/1812.05271.

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

Linyang Li and Xipeng Qiu. 2020. Textat: Adversarial training for natural language understanding with token-level perturbation. *CoRR*, abs/2004.14543.

Linyang Li, Demin Song, and Xipeng Qiu. 2023. Text adversarial purification as defense against adversarial

attacks. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 338–350, Toronto, Canada. Association for Computational Linguistics.

Xinzhe Li, Ming Liu, Xingjun Ma, and Longxiang Gao. 2021a. Exploring the vulnerability of natural language processing models via universal adversarial texts. In *Proceedings of the The 19th Annual Workshop of the Australasian Language Technology Association*, pages 138–148, Online. Australasian Language Technology Association.

Zongyi Li, Jianhan Xu, Jiehang Zeng, Linyang Li, Xiaoqing Zheng, Qi Zhang, Kai-Wei Chang, and Cho-Jui Hsieh. 2021b. Searching for an effective defender: Benchmarking defense against adversarial word substitution. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3137–3147, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Ro{bert}a: A robustly optimized {bert} pretraining approach.

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

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*.

Rishabh Maheshwary, Saket Maheshwary, and Vikram Pudi. 2021. Generating natural language attacks in a hard label black box setting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15):13525–13533.

Takeru Miyato, Andrew M. Dai, and Ian Goodfellow. 2017. Adversarial training methods for semi-supervised text classification. In *International Conference on Learning Representations*.

Han Cheol Moon, Shafiq Joty, Ruochen Zhao, Megh Thakkar, and Chi Xu. 2023. Randomized smoothing with masked inference for adversarially robust text classifications. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5145–5165, Toronto, Canada. Association for Computational Linguistics.

John 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. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126, Online. Association for Computational Linguistics.

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

Dang Nguyen Minh and Anh Tuan Luu. 2022. Textual manifold-based defense against natural language adversarial examples. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6612–6625, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. 2019. Improving adversarial robustness via promoting ensemble diversity. In *International Conference on Machine Learning*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Fanchao Qi, Yangyi Chen, Xurui Zhang, Mukai Li, Zhiyuan Liu, and Maosong Sun. 2021. Mind the style of text! adversarial and backdoor attacks based on text style transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4569–4580, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Zhouxing Shi, Huan Zhang, Kai-Wei Chang, Minlie Huang, and Cho-Jui Hsieh. 2020a. Robustness verification for transformers. In *International Conference on Learning Representations*.

Zhouxing Shi, Huan Zhang, Kai-Wei Chang, Minlie Huang, and Cho-Jui Hsieh. 2020b. Robustness verification for transformers. In *International Conference on Learning Representations*.

Chenglei Si, Zhengyan Zhang, Fanchao Qi, Zhiyuan Liu, Yasheng Wang, Qun Liu, and Maosong Sun. 2021. Better robustness by more coverage: Adversarial and mixup data augmentation for robust finetuning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1569–1576, Online. Association for Computational Linguistics.

Boxin Wang, Shuohang Wang, Yu Cheng, Zhe Gan, Ruoxi Jia, Bo Li, and Jingjing Liu. 2021. Info{bert}: Improving robustness of language models from an information theoretic perspective. In *International Conference on Learning Representations*.

Xuezhi Wang, Haohan Wang, and Diyi Yang. 2022. Measure and improve robustness in NLP models: A survey. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4569–4586, Seattle, United States. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Kaidi Xu, Zhouxing Shi, Huan Zhang, Yihan Wang, Kai-Wei Chang, Minlie Huang, Bhavya Kailkhura, Xue Lin, and Cho-Jui Hsieh. 2020. Automatic perturbation analysis for scalable certified robustness and beyond. In *Advances in Neural Information Processing Systems*, volume 33, pages 1129–1141. Curran Associates, Inc.

Mao Ye, Chengyue Gong, and Qiang Liu. 2020. SAFER: A structure-free approach for certified robustness to adversarial word substitutions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3465–3475, Online. Association for Computational Linguistics.

Liping Yuan, Xiaoqing Zheng, Yi Zhou, Cho-Jui Hsieh, and Kai-Wei Chang. 2021. On the transferability of adversarial attacks against neural text classifier. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1612–1625, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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*, pages 6066–6080, Online. Association for Computational Linguistics.

Jiehang Zeng, Jianhan Xu, Xiaoqing Zheng, and Xuanjing Huang. 2023. Certified robustness to text adversarial attacks by randomized [MASK]. *Computational Linguistics*, 49(2):395–427.

Cenyuan Zhang, Xiang Zhou, Yixin Wan, Xiaoqing Zheng, Kai-Wei Chang, and Cho-Jui Hsieh. 2022. Improving the adversarial robustness of NLP models by information bottleneck. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3588–3598, Dublin, Ireland. Association for Computational Linguistics.

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

Yi Zhou, Xiaoqing Zheng, Cho-Jui Hsieh, Kai-Wei Chang, and Xuanjing Huang. 2021. Defense against synonym substitution-based adversarial attacks via Dirichlet neighborhood ensemble. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5482–5492, Online. Association for Computational Linguistics.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2020. Freelb: Enhanced adversarial training for natural language understanding. In *International Conference on Learning Representations*.

A Proof of Section

Theorem 3.1. *If a random vector $v \sim \mathcal{N}(0, \nu I)$ where ν is small is added to the hidden layer h of the model f which can be decomposed into $f = g \circ h$, the new important score $I_{\tilde{x}}^{new}$ is a random variable that follows Gaussian distribution $\mathcal{N}(I_{\tilde{x}}, \nu(\|\nabla_{h(x)} g_y(h(x))\|_2 + \|\nabla_{h(\tilde{x})} g_y(h(\tilde{x}))\|_2))$.*

Proof. When v is small, we have the first-order approximation of the model with noise

$$
\begin{aligned}
f_{\text{AdvFooler}}(x) &= g(h(x) + v) \\
&\approx f(x) + \nabla_{h(x)} g(h(x))v.
\end{aligned}
$$

In this case, since each application samples a different noise vector, the new important score becomes

$$
\begin{aligned}
I_{\tilde{x}}^{\text{new}} &\approx f_y(x) + \nabla_{h(x)} g_y(h(x))v_1 - f_y(\tilde{x}) \\
&\quad - \nabla_{h(\tilde{x})} g_y(h(\tilde{x}))v_2 \\
&= I_{\tilde{x}} + v_3
\end{aligned}
$$

where

$$
\begin{aligned}
v_1, v_2 &\sim \mathcal{N}(0, \nu I), \\
v_3 &\sim \mathcal{N}(0, \nu(\|\nabla_{h(x)} g_y(h(x))\|_2 \\
&\quad + \|\nabla_{h(\tilde{x})} g_y(h(\tilde{x}))\|_2)).
\end{aligned}
$$

$\square$

B Additional Implementation Details

B.1 Dataset

We evaluate AdvFooler on two benchmark datasets: AG-News Corpus (**AGNEWS**) (Zhang et al., 2015) and Internet Movie Database (**IMDB**) (Maas et al., 2011). AGNEWS is a classification dataset of news articles created from the AG's corpus[*]. It contains 120,000 training samples and 7,600 test samples where a sample belongs to one of four classes: World, Sports, Business, and Sci/Tech. IMDB is a binary sentiment classification dataset of reviews extracted from the IMDB website. This dataset contains 50,000 samples and is split equally into the training and test sets. For IMDB, we use the original dataset [†] from Maas et al. (2011).

---

[*] http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html
[†] https://ai.stanford.edu/~amaas/data/sentiment/

For AGNEWS, we use the datasets from HuggingFace Datasets (Lhoest et al., 2021). Similar to (Nguyen Minh and Luu, 2022), we also set the model's max sequence lengths for IMDB and AGNEWS to be 256 and 128, respectively based on their average sample length.

B.2 Models

We conduct experiments on two state-of-the-art, pre-trained language models: BERT$_{base}$ (Devlin et al., 2019) and RoBERTa$_{base}$ (Liu et al., 2020). BERT is a Transformer-based language model that outperforms other models in various benchmarks at the time of its release and is widely used for text classification tasks. RoBERTa is a variant of BERT with an improved training process and performance.

B.3 Training Hyperparameters

For the NLP models, we employ the pretrained models from HuggingFace Transformer (Wolf et al., 2020) and finetune them for another ten epochs. We use grid search from 1e-5 to 1e-3 to find the optimal learning rate for each model on the respective dataset. The optimally fine-tuned models are used for the robustness evaluation.

C Additional Experiment

C.1 AdvFooler on varying model layers

In this experiment, we study the performance of AdvFooler when randomization is applied at various latent layers of the model. The accuracy and robustness results are shown in Tables 7 and 8. As we can observe from the tables, adding noise to the output of the attention layer yield better robustness in general. Furthermore, injecting noise into the [CLS] token is more effective than to all the hidden layers.

C.2 Randomization techniques comparison

To show how a lower loss variance makes a randomization method more effective against adversarial attacks, we ran both RanMASK and SAFER, two randomization methods that have a higher variance, through the same experiments on the base model without ensembling. As we can observe from the results in Table 9, both RanMASK and SAFER performed significantly lower than their corresponding baselines in the main experiment. This shows that

Table 7: Performance results when randomization is applied to different token positions and different locations of the transformer model, on the IMDB dataset. Here, randomization is applied to every self-attention layer.

PrLMs	Models	Clean%	TextBugger			BERT-Attack			TextFooler		
			AuA% (ASR%)	#Query		AuA% (ASR%)	#Query		AuA% (ASR%)	#Query	
BERT-base	Input embeddings.	90.78	23.40 (74.26)	2089.57		19.50 (78.55)	1543.31		27.80 (69.45)	1782.04	
	Attention output, all tokens	90.82	31.50 (65.12)	2612.29		23.00 (74.59)	1893.96		30.20 (66.33)	1967.33	
	Attention output, [CLS] token	91.9	42.40 (53.41)	3261.41		40.70 (55.76)	2645.36		49.10 (47.32)	2759.37	
	Attention input, all tokens	91.05	24.80 (72.69)	2259.91		18.70 (79.38)	1707.90		28.70 (68.08)	1886.60	
	Attention input, [CLS] token	91.76	22.40 (75.89)	2153.99		21.70 (76.28)	1759.99		25.60 (71.90)	1781.55	
RoBERTa-base	Input embeddings.	92.02	26.80 (70.52)	2345.55		15.90 (82.29)	1478.49		29.80 (66.52)	2010.82	
	Attention output, all tokens	91.41	30.20 (67.17)	2408.47		21.50 (76.61)	1756.51		36.40 (60.56)	2066.70	
	Attention output, [CLS] token	92.69	60.30 (35.58)	4349.38		50.20 (45.55)	3274.13		64.70 (30.20)	3360.81	
	Attention input, all tokens	92.97	23.60 (74.46)	2086.32		15.10 (83.99)	1426.83		28.70 (69.14)	1747.67	
	Attention input, [CLS] token	92.64	27.80 (69.72)	2504.18		18.80 (79.76)	1553.59		33.30 (64.54)	1927.80	

Table 8: Performance results when randomization is applied to different token positions and different locations of the transformer model, on AGNEWS dataset. Here, randomization is applied to every self-attention layer.

PrLMs	Models	Clean%	TextBugger			TextFooler			BERT-Attack		
			AuA% (ASR%)	#Query		AuA% (ASR%)	#Query		AuA% (ASR%)	#Query	
BERT-base	Input embeddings.	94.57	44.80 (52.03)	711.12		45.50 (51.49)	638.84		51.10 (45.23)	681.39	
	Attention output, all tokens	94.59	45.60 (50.97)	738.90		48.60 (48.35)	652.88		54.90 (41.35)	727.93	
	Attention output, [CLS] token	93.67	50.10 (45.90)	819.44		50.10 (45.90)	701.05		53.40 (42.89)	752.61	
	Attention input, all tokens	94.47	44.80 (52.29)	745.68		49.00 (47.71)	675.86		53.10 (43.09)	731.95	
	Attention input, [CLS] token	93.43	34.70 (62.41)	670.61		38.90 (57.90)	610.20		39.40 (57.54)	640.75	
RoBERTa-base	Input embeddings.	94.72	48.90 (47.70)	752.97		48.50 (48.07)	645.89		54.50 (41.90)	707.38	
	Attention output, all tokens	94.21	51.70 (44.88)	804.99		53.30 (42.81)	717.31		57.80 (38.38)	767.47	
	Attention output, [CLS] token	94.63	49.50 (47.17)	823.36		50.20 (46.31)	718.93		59.60 (36.32)	813.69	
	Attention input, all tokens	93.01	40.80 (55.75)	726.04		39.30 (57.00)	618.76		46.10 (50.43)	672.77	
	Attention input, [CLS] token	94.31	40.70 (56.42)	721.77		41.90 (55.14)	624.96		45.60 (50.81)	681.12	

these methods cannot be used in a similar way to AdvFooler(i.e., without ensembling during inference), since their high variance makes them more susceptible to attacks, and can only be used in tandem with ensembling and adversarial training with data augmentation.

C.3 AdvFooler against Expectation over Transformation (EoT)

To combat against randomization defense, Athalye et al. (2018) proposed **Expectation over Transformation (EoT)**, a general framework to construct adversarial examples that remain adversarial over a chosen transformation distribution. EoT works by generating an adversarial example from a transformation distribution instead of a single example, which could negate the effects of AdvFooler. However, to capture the distribution of the transformation being made by AdvFooler's noise, the adversary needs to query the model multiple times for each query, thus increasing the query budget. To evaluate AdvFooleragainst EoT-based attacks, we ran our method against the modified TextBugger and TextFooler with the EoT framework. As we can observe in Table 10, the two EoT-based attacks achieve higher Accuracy Under Attack against AdvFooler compared to the original setting. Although EoT can help negate the effects of AdvFooler's randomness on adversarial input, the number of

queries needed to find an adversarial example exceeds the budget before it can generate an adversarial example. To re-confirm this explanation, we increased the query budget by a factor of 10. As expected, EoT was able to reduce the Accuracy Under Attack of AdvFooler. Nevertheless, this experiment shows the limited practicality and effectiveness of EoT against AdvFooler.

C.4 AdvFooler's effect on model's output error

Table 11: Expected Calibration Errors of the model's prediction, with and without AdvFooler.

	AdvFooler	Base
AGNEWS	2.3±0.2%	3.9±0.0%
IMDB	6.9±0.2%	6.3±0.0%

An immediate question is whether the score returned by AdvFooler is still reliable; otherwise we can just output an arbitrary fixed value for every input, which totally negates any score-based attacks. To calibrate the output of AdvFooler, we calculate the Expected Calibration Error (ECE) of the BERT-base model with and without the use of AdvFooler. We run the experiment five times and report the average ECE score and its standard deviation for comparison. From the results in Table

Table 9: Performance results for RanMASK and SAFER on the baseline BERT model without ensemble.

Dataset	Models	Attack methods		
		TextFooler	TextBugger	BERTAttack
AGNEWS	RanMASK.	0%	0%	0%
	SAFER	38.6%	37.5%	43.8%
IMDB	RanMASK.	19.5%	15.6%	19.3%
	SAFER	28.2%	21.4%	18.5%

Table 10: Accuracy Under Attack of AdvFooler against TextBugger and TextFooler on two datasets using EoT.

Dataset	Model	EOT results for advfooler	
		TextBugger	TextFooler
AGNEWS	AdvFooler	50.1%	50.1%
	EOT (same query budget)	72.2%	70.2%
	EOT (x10 query budget)	40.5%	41.0%
IMDB	AdvFooler	42.4%	40.70%
	EOT (same query budget)	59.5%	59.7%
	EOT (x10 query budget)	38.6%	41.2%

11, we can see that, the ECE differences between with and without using AdvFooler is low in both datasets, with the margin of error when using Adv-Fooler only at 0.2%. This shows that even though AdvFooler can affect the output logits of the model, the changes will only deviate slightly from the original prediction.

C.5 AdvFooler performances against the transferred, white-box attack

In the main paper, we focus on defending in the black-box setting, as it is a more realistic scenario in practice. Nevertheless, we provide an empirical study of the effectiveness of AdvFooler against a type of attack where the adversary has access to either the base model or the randomized model to generate the adversarial example of an input. We call this type of attack **transferred, white-box attacks**. Note that, this attack is not truly a white-box attack, where the fact that the adversary has access to the base model, but there still exists a mismatch between the attacker's possessed model and the defender's possessed model due to the added randomization. Even with the randomized model, the base white-box approach used in this study is not specifically designed to make good use of the noise distribution of the randomized model, thus, it is not truly a white-box attack case.

Most white-box attacks focus on increasing the loss of the model through gradient computation, choosing the perturbation in the sentence that increases loss the most. We employ **HotFlip** (Ebrahimi et al., 2018) as the studied white-box

attack. The experimental setup is similar to that in the main experiments, with the number of words of a sentence that the attacker can perturb increased to 10% and 30% (instead of only two words in the original paper) for IMDB and AGNEWS, respectively. We also give HotFlip access to the model's original parameters, which they can use to create adversarial examples and bypass defense mechanics provided by the defense methods (in the case of RanMASK, TMD, and AdvFooler). To make sure the adversarial examples created using the process can trick the model using the defense, we also ran the adversarial examples through the defense method 5 times and take the class that being classified to the most. As we can observe in Table 13, AdvFooler is effective against HotFlip in both datasets, increasing the AuA of the baseline model by over 20%. We hypothesize that introducing randomness into the model can turn it into a smoothed classifier, consequently diminishing the adversarial impact.

Furthermore, we also consider the case where the adversary ignores the noise from the defense and only attacks the original model. We compute the AuA/ASR of AdvFoolerin this case and compare it to RanMASK and TMD. Table 14 shows that AdvFooler still achieves high robustness compared to TMD.

C.6 Comparison to Other Defenses With Similar Accuracy

We conduct experiments on tuning the masking rate of RanMASK in AGNEWS such that the clean accuracy is similar to that of AdvFooler. We de-

Table 12: The performance of RanMASK on AGNEWS with different mask rates.

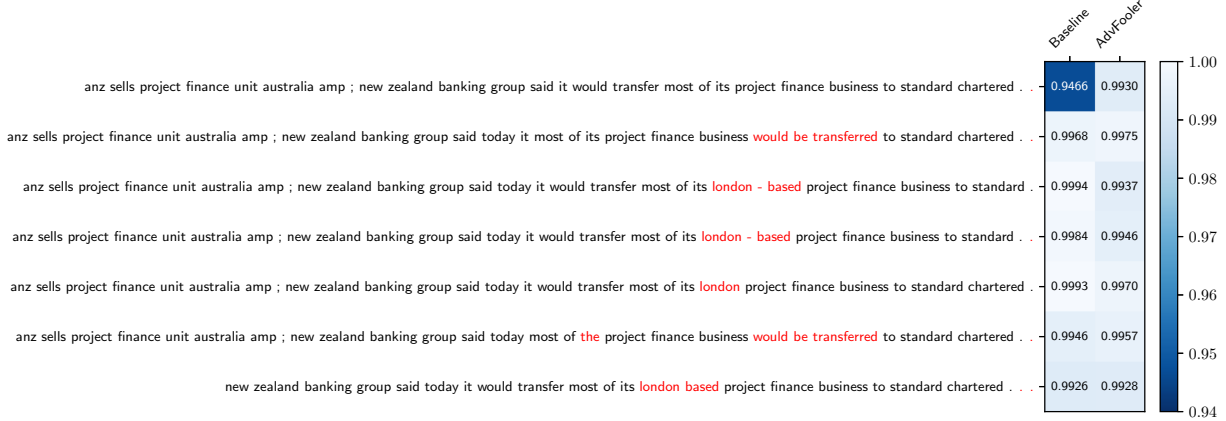| Models	Mask Rate	Clean Accuracy (%)	TextBugger			TextFooler			BERTAttack		
			AuA(%) (ASR(%)↓)	Avg. Query↓		AuA(%) (ASR(%)↓)	Avg. Query↓		AuA(%) (ASR(%)↓)	Avg. Query↓	
BERT	0.9	90.14%	52.50% (41.60%)	582.21		54.60% (38.93%)	511.98		61.10% (31.73%)	595.57	
BERT	0.3	93.50%	22.50% (75.88%)	360.12		24.10% (74.09%)	325.63		42.90% (54.12%)	412.99	
RoBERTa	0.9	90.41%	52.30% (41.50%)	575.94		54.20% (39.37%)	516.98		60.00% (32.43%)	594.93	
RoBERTa	0.3	93.30%	34.00% (63.56%)	396.72		35.7% (61.74%)	359.52		45.60% (50.97%)	429.72	

Figure 5: Illustration of each paraphrase's score when calculated with and without AdvFooler. In this case, MAYA selects the first paraphrase when attacking the baseline model, while selecting the last paraphrase when attacking the model protected by AdvFooler.

Table 13: Accuracy under Attack and Attack Success Rate of each method against HotFlip.

	AGNEWS	IMDB
BERT-Base	37.7% (59.98%)	59.3% (35.82%)
FreeLB	51.1% (45.87%)	62.3% (31.08%)
InfoBERT	43.3% (54.08%)	50.3% (44.66%)
AdvFooler	60.8% (34.19%)	84.4% (8.16%)

Table 14: Accuracy under Attack and Attack Success Rate of AdvFooler and other defense against HotFlip when bypassing defense.

	AGNEWS	IMDB
AdvFooler	48.3% (48.50%)	77.3% (16.61%)
RanMASK	85.3% (3.90%)	84.1% (9.8%)
TMD	44.2% (52.72%)	63.3% (29.9%)

Table 15: The robustness performance of AdvFooler and other defenses against MAYA on AGNEWs

	Results [AuA%(ASR%)]	#Query
BERT-Base	4.8% (94.8%)	172.96
FREELB	7.4% (92.2%)	178.25
InfoBERT	9.5% (89.9%)	183.63
AdvFooler	21.0% (77.5%)	199.38
SAFER	17.0% (82.0%)	188.8
RanMASK	19.8% (78.0%)	201.35
TMD	28.0% (70.2%)	167.57

curacy of the model, from 90% to 93% in accuracy. On the other hand, the accuracy under attack has decreased tremendously, reducing to half of the original AuA. This shows that, while RanMASK was able to be robust against adversarial attacks, this method also requires larger trade-offs in accuracy compared to other methods. The result also shows that RanMASK requires fine-tuning of the masking rate to achieve good performances under attack, as the effective masking rate was chosen from their certified robustness experiments.

C.7 AdvFooler's performance against multi-granular attacks.

In the main experiments, we focused on attacks that perturbed the text input on word level, i.e., focus on finding important words and replacing them

crease the random masking rate from 90% of the words in a text example to 30%, which would increase RanMASK's accuracy, as shown in their paper (Zeng et al., 2023). We also retrain a Ran-MASK with a random masking rate of 30%, to prevent differences in masking rate between training and inference.

Table 12 shows that when decreasing the masking rate in RanMASK does increase the clean ac-

Table 16: Accuracy under Attack and Attack Success Rate of DSRM and RSMI against the main experiment settings.

Dataset	Model	Accuracy	TextBugger	TextFooler	BERTAttack
AGNEWS	RSMI	92.75%	61.60% (33.62%)	63.50% (31.43%)	63.30% (32.01%)
	DSRM	94.78%	36.50% (61.21%)	38.00% (59.62%)	45.90% (51.22%)
	AdvFooler (Ours)	93.67%	50.10% (45.90%)	50.10% (45.90%)	53.40% (42.89%)
IMDB	RSMI	91.17%	48.40% (46.16%)	51.40 (42.95%)	42.20% (53.11%)
	DSRM	91.06%	25.70% (72.07%)	26.0% (71.30%)	26.20% (71.02%)
	AdvFooler (Ours)	91.9%	42.40% (53.41%)	49.10% (47.32%)	40.70% (55.76%)

with synonyms, leading the model to make wrong outputs. However, there are attacks that focus on perturbing a sequence of words, paraphrasing the phrases and sentences of the input. To test the effectiveness of our defense against these types of attacks, we use **MAYA** - **M**ulti-gr**A**nularit**Y** **A**ttack(Chen et al., 2021) as the attack that perturbs both words and sequences of words in the input text. This attack breaks the sentences into constituencies (phrases, phonemes), then paraphrases them to generate the perturbations. MAYA selects the best perturbation for the next round (if the perturbed sentence is not yet an adversarial example) by querying the model to calculate the confidence scores of these perturbations. The defense performance is reported in Table 15.

As can be observed, AdvFooler is effective in defending against perturbations in different granularities, achieving second-best accuracy under attack (AuA%) and number of queries (#Query). Since MAYA uses confidence scores to select the next perturbation, AdvFooler's randomization strategy can mislead MAYA's calculation, causing it to select the incorrect perturbations, similar to the case of word-perturbation attacks.

To empirically confirm this phenomenon, we show the effects of AdvFooler on the scores used by MAYA to select the paraphrase perturbation in Figure 5. From the illustrations, we can observe that AdvFooler causes the scores to slightly change from the baseline model, sufficient to rearrange the ranking of the perturbations. More specifically, as MAYA selects the perturbation with the lowest score, AdvFooler affects MAYA's selection from the first paraphrase in the Baseline case to the last paraphrase in the protected model with AdvFooler; this misleads the attacker to select a suboptimal perturbation selection. Furthermore, the slight change in scores caused by AdvFooler makes it more difficult to discern which paraphrases are more adversarial, as can be seen from the scores between the baseline and the AdvFooler columns, we can see

that the first paraphrase is the most adversarial in the baseline column, compare to AdvFooler, where the most adversarial paraphrase is harder to differentiate, making the process of selecting paraphrase less trivial for MAYA.

C.8 Comparison against recent defense methods.

In the main experiment, we have compared AdvFooler against several widely known and benchmarked defense methods to show the effectiveness of our method. To further investigate this point, we ran additional experiments against more recent defenses and compared their results to AdvFooler. We selected RSMI (Moon et al., 2023) and DSRM (Gao et al., 2023) for comparison as they are recent state-of-the-art methods and they have reported good results. DSRM increases the model's robustness through adversarial training by perturbing the input data's probability distribution. RSMI, on the other hand, makes the model more robust via randomized smoothing through ensemble and Gradient-Guided Masking. We ran these two methods against the main experiment setup and compared their results to ours. Table 16 shows that DSRM can increase the model's robustness, having higher AuA than InfoBERT and FreeLB in IMDB. However, the overall performance of DSRM is lower compared to AdvFooler. RSMI, on the other hand, was able to perform well compared to most methods, in both AGNEWs and IMDB datasets

Although the performance of RSMI is high, our method still holds many advantages. Firstly, RSMI's ensemble process adds a substantial inference time compared to AdvFooler (and also DSRM). As shown in Table 17, the inference time of RSMI is almost 12 times more than both AdvFooler and DSRM. Secondly, both DSRM and RSMI require access to and modification of training, while AdvFooler is pluggable, allowing it to

Table 17: Inference time comparison with RSMI and DSRM. Tested with BERT on the test set from the IMDB dataset.

Defense	Runtime (s)
RSMI	1015
DSRM	85
AdvFooler (Ours)	87

be used without modifying the existing parameters of the model. This is important, as adversarial training methods like DSRM could increase the training time substantially compared to normal fine-tuning(Gao et al., 2023). These characteristics of AdvFooler make it an attractive defense for a base model without needing further training, or for users to combine AdvFooler with other defenses to boost the performance of these defenses.

D   Usage of AI assistant in this paper.

We only used AI at the sentence level (e.g., fixing grammar, finding synonyms for words...).