The Best Defense is Attack: Repairing Semantics in Textual Adversarial Examples

Anonymous ACL submission

Abstract

Recent studies have revealed the vulnerability of pre-trained language models to adversarial attacks. Adversarial defense techniques have been proposed to reconstruct adversarial examples within feature or text spaces. However, these methods struggle to effectively repair the semantics in adversarial examples, resulting in unsatisfactory defense performance. To repair the semantics in adversarial examples, we introduce a novel approach named **Rea**ctive **P**erturbat**i**on **D**efocusing (RAPID), which employs an adversarial detector to identify the fake labels of adversarial examples and leverages adversarial attackers to repair the semantics in adversarial examples. Our extensive experimental results, conducted on four public datasets, demonstrate the consistent effectiveness of RAPID in various adversarial attack scenarios. For easy evaluation, we provide a click-to-run demo of RAPID at `https://tinyurl.com/22ercuf8`.

1 Introduction

Pre-trained language models (PLMs) have achieved state-of-the-art (SOTA) performance across a variety of natural language processing tasks (Wang et al., 2019a,b). However, PLMs are reported to be highly vulnerable to adversarial examples, a.k.a., *adversaries*, created by subtly altering selected words in natural examples, a.k.a. *clean* or *benign examples* (Li et al., 2019; Garg and Ramakrishnan, 2020; Li et al., 2020; Jin et al., 2020; Li et al., 2021; Boucher et al., 2022). While the significance of textual adversarial robustness has been broadly recognized within the deep learning community (Alzantot et al., 2018; Ren et al., 2019; Zang et al., 2020; Zhang et al., 2021; Jin et al., 2020; Li et al., 2021; Wang et al., 2022a; Xu et al., 2023), efforts to enhance adversarial robustness remain very limited, especially compared to other deep learning fields like computer vision (CV) (Rony et al., 2019; Gowal et al., 2021; Wang et al., 2023; Xu et al., 2023). Current works on textual adversarial robustness can be classified into three categories: *adversarial defense*, *adversarial training* (Liu et al., 2020a,b; Ivgi and Berant, 2021; Dong et al., 2021b,a), and *adversary reconstruction* (Zhou et al., 2019; Jones et al., 2020; Bao et al., 2021; Keller et al., 2021; Mozes et al., 2021; Li et al., 2022; Shen et al., 2023). Since both adversarial training and reconstruction are resource-intensive, there has been growing interest in adversarial defense. Nevertheless, the current adversarial defense techniques have two bottlenecks.

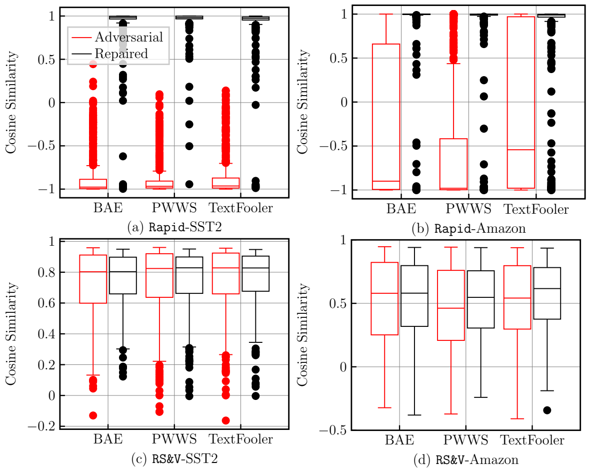

Figure 1: Box plots of the cosine similarity between the *adversary–natural example pairs* (marked in red) and the *repaired adversary–natural example pairs* obtained by RAPID versus RS&V. The cosine similarity is evaluated based on the features extracted by the victim models of RAPID and RS&V, respectively. It is observed that the victim model cannot discern the semantic differences between the adversaries and the repaired adversaries produced by RS&V, whereas RAPID can precisely differentiate between adversaries and natural examples. Conversely, when using RAPID, the repaired adversaries regain their semantic alignment with the natural examples.

- Current works can hardly identify the semantic discrepancies between natural and adversarial examples[1]. We show an example of RS&V (Wang et al., 2022c) in Figure 1, it is clear that RS&V fails to discern the semantic differences between adversarial and repaired examples. This is attributed to the augmentation method used in RS&V that is not only untargeted but also does not effectively identify or neutralize adversaries.

- Given the time-intensive nature of the defense process, adversarial defense is also notorious for its computational inefficiency (Mozes et al., 2021; Wang et al., 2022c). This can be partially attributed to their inability to *pre-detect* adversaries and to indiscriminately process all input texts. This not only wastes computational budget on unnecessary defense actions regarding natural examples but also leads to an unwarranted defensive stance towards natural examples, which may further compromise performance.

Bearing the above two challenges in mind, we propose a simple yet effective textual adversary defense paradigm, named reactive perturbation defocusing (RAPID), which has the following two distinctive features.

- To address the first bottleneck, we propose a novel concept of perturbation defocusing (Section 2.2.2). The basic idea is to leverage adversarial attackers to *re-inject* some perturbations into the *pre-detected* adversaries to distract the victim model from malicious perturbations, and to *repair* these adversaries based on the inherent robustness of the victim models. Further, the accuracy of adversarial defense is augmented by a pseudo-semantic similarity filtering strategy (Section 2.2.3).
- To overcome the second bottleneck, RAPID trains an *in-victim-model* adversarial detector, without introducing additional cost (Section 2.1), to proactively concentrate the defense efforts on the examples *pre-detected* as adversaries. In particular, this adversarial detector is jointly trained with the victim model in a multi-task way and is capable of recognizing adversaries generated by different attackers. This helps not only minimize collateral impacts on natural examples (Xu et al., 2022) but also reduces the waste of computational budget upon defending against natural examples.

Figure 2 provides a pedagogical example of the working mechanism of RAPID in the context of sentiment analysis. There are four key takeaways from our empirical study.

- RAPID achieves up to 99.9% repair accuracy upon pre-detected adversaries, significantly surpassing text/feature-level reconstruction and voting-based methods (Table 2).
- RAPID reduces nearly 50% computational cost for adversarial defense compared against adversarial attack (Table 15).
- RAPID is robust in recognizing and defending against a wide range of unknown adversarial attacks (Table 5), such as CLARE (Li et al., 2021) and large language models like `ChatGPT-3.5` (OpenAI, 2023).
- We develop a user-friendly API[2] as a benchmarking platform for different adversarial attackers under the defense of RAPID.

2 Proposed Method

Our proposed RAPID framework comprises two phases. `Phase #1` trains a joint model that not only performs the standard text classification task but is also capable of detecting adversaries. `Phase #2` is dedicated to implementing a pseudo-supervised adversary defense based on PD. It diverts the victim model's attention from malicious perturbations and rectifies the outputs without compromising performance on natural examples.

2.1 `Phase #1`: joint model training

The crux of `Phase #1` is the joint training of two models: one is the victim model as the standard text classifier, and the other is an in-victim-model adversarial detector, which is a binary classifier that pre-detects adversaries before the defense.

2.1.1 Multi-attack-based adversary sampling

To derive the data used for training the adversarial detector, we apply adversarial attack methods upon the victim model F_S to sample adversaries. To enable the adversarial detector to identify various unknown adversaries, we employ three widely used open-source adversarial attackers: BAE (Garg and Ramakrishnan, 2020), PWWS (Ren et al., 2019), and TEXTFOOLER (Jin et al., 2020). For each data instance $\langle \mathbf{x}, y \rangle \in \mathcal{D}$, the set of natural examples,

[1]In this work, we refer to the semantics in adversaries as the features encoded by PLM for simplicity.

[2]For the sake of anonymous requirement, we will release this tool upon the acceptance of this paper.

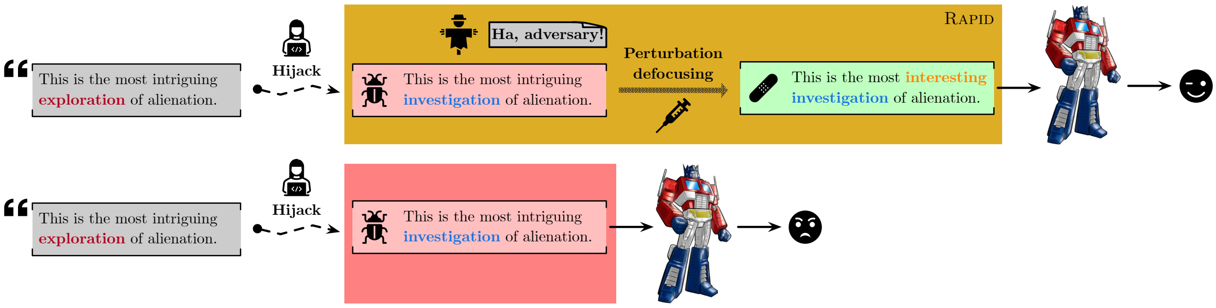

Figure 2: A pedagogical example of RAPID in sentiment analysis. The original word in this example is exploration. Perturbation defocusing repairs the adversary by injecting perturbations (interesting) to distract the objective model from the malicious perturbation (i.e., investigation). RAPID only implements defense on the pre-detected adversary.

we apply each of the adversarial attackers to sample three adversaries[3]:

$$\langle \tilde{\mathbf{x}}, \tilde{y} \rangle_i \leftarrow \mathcal{A}_i \left(F_S, \langle \mathbf{x}, y \rangle \right), \tag{1}$$

where $\mathcal{A}_i$, $i \in \{1, 2, 3\}$, represents BAE, PWWS, and TEXTFOOLER, respectively. $\langle \tilde{\mathbf{x}}, \tilde{y} \rangle_i$ is the adversary generated by $\mathcal{A}_i$. Note that we collect all adversaries, including both successful and failed ones, to constitute the adversarial dataset $\tilde{\mathcal{D}}$. Finally, we compose a hybrid dataset as shown in the left part of Figure 3. $\overline{\mathcal{D}} := \mathcal{D} \bigcup \tilde{\mathcal{D}}$ for the joint model training.

2.1.2 Joint model training objectives

To conduct the joint model training of both the victim model and the adversarial detector, we propose an aggregated loss function as follows:

$$\mathcal{L} := \mathcal{L}_{\mathrm{c}} + \mathcal{L}_{\mathrm{d}} + \mathcal{L}_{\mathrm{a}} + \lambda ||\boldsymbol{\theta}||_2^2, \tag{2}$$

where λ is the ℓ_2 regularization parameter, and $\boldsymbol{\theta}$ represents the parameters of the underlying PLM. $\mathcal{L}_{\mathrm{c}}$, $\mathcal{L}_{\mathrm{d}}$, and $\mathcal{L}_{\mathrm{a}}$ denotes the loss for training a standard classifier, an adversarial detector, and adversarial training, respectively.

- *Standard classification loss $\mathcal{L}_{\mathrm{c}}$*: Here we use the cross-entropy loss widely used for text classification:

$$\mathcal{L}_{\mathrm{c}} := -\sum_{i=1}^{C} \left[p_i \log \left(\tilde{p}_i \right) + q_i \log \left(\tilde{q}_i \right) \right], \tag{3}$$

  where C is the number of classes. p and $\tilde{p}$ respectively indicate the true and predicted probability distributions of the standard classification label, while q and $\hat{q}$ represent any incorrect standard classification label and its likelihood, respectively. Note that the labels of the adversaries within $\overline{\mathcal{D}}$ are set to a dummy value $\varnothing$ in this loss. By doing so, we can make sure that $\mathcal{L}_{\mathrm{c}}$ focuses on the natural examples.
- *Adversarial detection loss $\mathcal{L}_{\mathrm{d}}$*: It only calculates the binary cross-entropy for both natural examples and adversaries within $\overline{\mathcal{D}}$, where the labels are either 0 or 1 in practice. Note that $\mathcal{L}_{\mathrm{d}}$ is used to train the adversarial detector as a binary classifier that determines whether the input example is an adversary or not.
- *Adversarial training loss $\mathcal{L}_{\mathrm{a}}$*: In practice, the calculation of $\mathcal{L}_{\mathrm{a}}$ is the same as $\mathcal{L}_{\mathrm{c}}$. To improve the robustness of adversaries, $\mathcal{L}_{\mathrm{a}}$ only calculates the loss for the adversaries by setting the labels of natural examples within $\overline{\mathcal{D}}$ as a dummy $\varnothing$. By doing so, we can prevent this adversarial training loss from negatively impacting the performance on pure natural examples, which have been reported to be notorious in recent studies (Dong et al., 2021a,b).

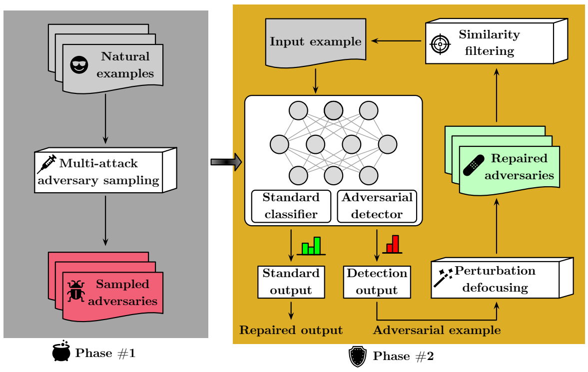

Figure 3: The overall architecture and workflow of RAPID.

All in all, each instance $\langle \overline{\mathbf{x}}, \overline{\mathbf{y}} \rangle \in \overline{\mathcal{D}}$ is augmented with three different labels to accommodate these three training losses, where $\overline{\mathbf{y}} := (\overline{y}_1, \overline{y}_2, \overline{y}_3)^\top$.

2.2 Phase #2: reactive adversarial defense

To address the efficiency and semantic challenges discussed in Section 1, the reactive adversarial de-

[3]The formulation of word-level adversarial attack is available in Appendix B.

fense consists of the following three steps.

2.2.1 Adversarial defense detection

Our preliminary experiments suggested that PLMs like BERT and DeBERTa are sensitive to semantic shifts caused by adversarial attacks. Thereby, different from the current adversarial defense methods, which often indiscriminately run defense upon all input examples, we will first apply the joint model F_J trained in the `Phase #1` to determine whether the input $\hat{\mathbf{x}}$ is adversarial or not using the following prediction:

$$(\hat{y}_1, \hat{y}_2, \hat{y}_3) \leftarrow F_J(\hat{\mathbf{x}}), \tag{4}$$

where $\hat{y}_1$, $\hat{y}_2$, and $\hat{y}_3$ are predicted labels according to the three training losses in equation (2), respectively. Thereafter, only the inputs identified as adversaries (i.e., those with $\hat{y}_2 = 1$) are used for the follow-up perturbation defocusing.

2.2.2 Perturbation defocusing

The basic idea of this perturbation defocusing is to inject *safe* perturbations into the adversary $\hat{\mathbf{x}}$ identified by the adversarial defense detection in Section 2.2.1. The process is shown in `Phase #2` inFigure 3. In practice, we apply an adversarial attacker to *attack* $\hat{\mathbf{x}}$ to obtain a *repaired example*:

$$\langle \tilde{\mathbf{x}}^{\mathrm{r}}, \tilde{y}_1^{\mathrm{r}} \rangle \leftarrow \hat{\mathcal{A}}_{PD} \left(F_J, \langle \hat{\mathbf{x}}, \hat{y}_1 \rangle \right), \tag{5}$$

where $\hat{y}_1$ is the predicted label of $\hat{\mathbf{x}}$, and $\hat{\mathcal{A}}_{PD}$ is an adversarial attacker[4]. Note that the above perturbation is considered safe because it does not alter the semantics of $\hat{\mathbf{x}}$. By this means, we divert the standard classifier's focus away from the malicious perturbations, allowing the standard classifier to concentrate on the adversary's original semantics. In essence, the repaired examples can be correctly classified based on the model inherent robustness.

2.2.3 Pseudo-semantic similarity filtering

To prevent repaired adversaries from being misclassified, we propose a feature-level pseudo-semantic similarity filtering strategy to mitigate semantic bias. Specifically, for each $\hat{\mathbf{x}}$, we generate a set of repaired examples $\mathcal{S} := \{\tilde{\mathbf{x}}_i^{\mathrm{r}}\}_{i=1}^k$. Then, we encode these repaired examples using F_J to extract their semantic features. Thereafter, for each repaired example within $\mathcal{S}$, we calculate its similarity score as:

$$s_i = \frac{\sum_{j=1, j\neq i}^{k} \mathrm{sim}(\mathcal{H}_i, \mathcal{H}_j)}{k}, \tag{6}$$

where $\mathcal{H}_i$ and $\mathcal{H}_j$ are the hidden states of $\tilde{\mathbf{x}}_i^{\mathrm{r}}$ and $\tilde{\mathbf{x}}_j^{\mathrm{r}}$ encoded by F_J, and $\mathrm{sim}(*, *)$ evaluates the cosine similarity. For the sake of efficiency, we set $k = 3$ in this paper. After the defense, the label of the repaired $\hat{\mathbf{x}}$ is assigned as the predicted label of the repaired example within $\mathcal{S}$ having the largest similarity score.

Remark 1. *Generally speaking, the basic idea of perturbation defocusing is to search and inject limited specific perturbations to change the fake prediction to disable the malicious perturbations, and we use adversarial attackers to search specific perturbations in perturbation defocusing.*

Remark 2. *Note that the perturbation defocusing in* Rapid *is decoupled with the adversarial detector, and its performance is agnostic to the adversarial attackers used for this adversary sampling. The empirical results in Table 5 demonstrate that the adversarial detector can adapt to unknown attack methods, even when trained on a small set of adversaries.*

3 Experimental Settings

In this section, we introduce the experimental settings used in our experiments.

Table 1: The statistics of datasets used for evaluating Rapid. We use subsets from `Amazon`, `AGNews` and `Yahoo!` datasets to evaluate Rapid as the previous works due to high resource occupation.

Dataset	Categories	Number of Examples		
		Training	Valid	Testing
SST2	2	6,920	872	1,821
Amazon	2	7,000	1,000	2,000
AGNews	4	120,000	0	7,600
Yahoo!	10	1,400,000	0	60,000

Victim models: while any PLM can be used in a plug-in manner in Rapid, this paper considers BERT (Devlin et al., 2019) and DeBERTa (He et al., 2021), two widely used PLMs based on the transformer structure[5], as both the victim classifier and the joint model. Their corresponding hyperparameter settings are in Appendix C.2.

Datasets: we consider three widely used text classification datasets[6], including `SST2` (Socher et al., 2013), `Amazon` (Zhang et al., 2015), and `AGNews` (Zhang et al., 2015) whose key statistics are outlined in Table 1. `SST2` and `Amazon` are binary sentiment classification datasets. `AGNews` and

[4] We choose PWWS because it is cost-effective, and it can be replaced by any (or an ensemble of) adversarial attackers.

[5] `https://github.com/huggingface/transformers`

[6] We have released the detailed source codes and processed datasets in the supplementary materials.

Yahoo! is a multi-categorical news classification dataset containing 4 and 10 categories, respectively.

Adversarial attackers: our experiments employ three open-source attackers provided by TEXTATTACK[7] (Morris et al., 2020). Their functionalities are outlined as follows, while their working mechanisms are in Appendix C.1.

a) *Adversary sampling*. BAE, PWWS and TEXTFOOLER are used to sample adversaries for training the adversarial detector (Section 2.1). Since they represent different types of attacks, we can train a detector that recognizes a variety of adversarial attacks.

b) *Adversary repair*. We employ PWWS as the attacker $\hat{\mathcal{A}}_{PD}$ in the perturbation defocusing (Section 2.2). Compared to BAE, our preliminary experiments demonstrate that PWWS rarely changes the natural examples' semantics, and it is more computationally efficient than TEXTFOOLER.

c) *Generalizability evaluation*. We use IGA (Wang et al., 2021a), DEEPWORDBUG (Gao et al., 2018), PSO (Zang et al., 2020) and CLARE to evaulate RAPID's generalization capability.

Evaluation metrics: we use the following five fine-grained metrics[8] for text classification to evaluate the adversarial defense performance.

- *Nature accuracy* (NTA): it evaluates the victim's performance on the target dataset that only contains natural examples.
- *Attack accuracy* (ATA): It evaluates the victim's performance under adversarial attacks.
- *Detection accuracy* (DTA): It measures the defender's adversaries detection performance.
- *Defense accuracy* (DFA): It evaluates the defender's performance of adversaries repair.
- *Repaired accuracy* (RPA): It evaluates the victim's performance on the attacked dataset after being repaired.

Note that we evaluate the adversarial detection and defense performance on the entire testing set, while current works (Xu et al., 2022; Yang et al., 2022; Dong et al., 2021a,b) only evaluated a small amount of data extracted from the testing set.

Baseline methods: RAPID is compared against the following six adversarial defense baselines.

- DISP (Zhou et al., 2019): It is an embedding feature reconstruction method and uses a perturbation discriminator to evaluate the probability that a token is perturbed and provides a set of potential perturbations.
- FGWS (Mozes et al., 2021): It uses frequency-guided word substitutions to exploit the frequency properties of adversarial word substitutions to detect adversaries.
- RS&V (Wang et al., 2022c): It is a text reconstruction method based on the randomized substitution-to-vote strategy. RS&V accumulates the logits of massive samples generated by randomly substituting the words in the adversaries with synonyms.
- RSMI (Moon et al., 2023): RSMI is a two-stage framework that combines randomized smoothing (RS) with masked inference (MI) to improve the adversarial robustness of NLP systems. This is not a technical analogy because adversarial detection support is not included in this research.

Note that the rationale for choosing the above baselines is their open-source nature, while we can hardly reproduce the experimental results of other methods like TEXTSHIELD (Shen et al., 2023).

3.1 Ablation Experiments

Due to the page limitation, **we include the following ablation experiments in Appendix D:**

- Performance of RAPID against LLM-based adversarial attack
- Performance of RAPID with different adversarial attackers in perturbation defocusing
- Performance of RAPID without adversarial training Objective
- Performance of RAPID without multitask training objective
- Performance Comparison between RAPID and adversarial training baseline
- Efficiency evaluation of RAPID
- Impact of k in RAPID
- Adversarial example evaluation

4 Experimental Results

4.1 Adversary detection performance

Results shown in Table 2 demonstrate the effectiveness of the adversarial detector in RAPID. Compared to the previous adversary detection-based defense (Mozes et al., 2021; Wang et al., 2022c; Shen et al., 2023), the in-victim-model adversarial detector identifies the adversaries with no extra cost. On the other hand, our evaluation confirms a very low

[7] https://github.com/QData/TextAttack

[8] The mathematical definitions of these evaluation metrics can be found in Appendix C.3.

Table 2: The adversarial detection and defense performance of RAPID on four public datasets. The victim model is BERT and the results in **bold** font indicate the best performance. We report the average accuracy of five random runs. For fair comparisons, all the baseline experiments are re-implemented based on the latest adversarial attackers from the Textattack library to avoid biases. "TF" indicates TEXTFOOLER and "-" means the results are not available because of a lack of adversarial detection support.

DEFENDER	ATTACKER	AGNews(4-category)					Yahoo!(10-category)					SST2 (2-category)					Amazon(2-category)				
		NTA	ATA	DTA	DFA	RPA	NTA	ATA	DTA	DTA	RPA	NTA	ATA	DTA	DFA	RPA	NTA	ATA	DTA	DFA	RPA
DISP	PWWS		32.09	55.49	57.82	68.23		5.70	61.67	54.95	50.24		23.44	38.93	34.46	35.33		15.56	41.90	45.92	59.80
	TF	94.13	50.50	53.78	56.18	70.16	75.63	13.60	50.73	57.48	53.18	91.24	16.21	37.80	34.37	37.16	93.67	21.77	43.10	47.15	60.56
	BAE		74.80	45.26	45.75	81.39		27.50	54.82	53.75	50.90		35.21	36.59	37.51	42.22		44.00	40.28	42.74	61.85
FGWS	PWWS		32.09	65.24	68.35	71.78		5.70	65.83	61.46	53.28		23.44	40.28	40.38	39.20		15.56	44.47	56.89	60.29
	TF	94.25	50.50	68.88	70.71	73.40	76.24	13.60	68.57	65.17	54.53	91.34	16.21	42.79	41.05	41.53	94.26	21.77	45.75	58.74	61.51
	BAE		74.80	44.29	47.95	83.57		27.50	58.63	56.33	52.94		35.21	43.83	48.37	44.90		44.00	42.26	43.04	64.63
RS&V	PWWS		32.09	83.67	84.96	83.80		5.70	65.01	65.22	57.22		23.44	36.90	37.10	38.54		15.56	29.60	45.30	46.17
	TF	94.14	50.50	82.44	83.45	82.53	76.39	13.60	74.21	74.54	58.10	91.55	16.21	39.70	38.40	39.70	94.32	21.77	40.70	42.30	55.70
	BAE		74.80	46.98	48.67	86.90		27.50	37.41	37.88	62.27		35.21	19.84	20.92	43.65		44.00	38.59	39.01	65.03
RSMI	PWWS		32.09	–	–	76.10		5.70	–	–	62.75		23.44	–	–	65.96		15.56	–	–	70.48
	TF	93.71	50.50	–	–	63.20	76.45	50.50	–	–	63.73	91.55	16.21	–	–	61.67	94.11	21.77	–	–	72.62
	BAE		74.80	–	–	86.10		27.50	–	–	65.22		35.21	–	–	67.42		44.00	–	–	75.21
RAPID	PWWS		32.09	**90.11**	**95.88**	**92.36**		5.70	**87.33**	**92.47**	**69.40**		23.44	**94.03**	**98.62**	**89.85**		15.56	**97.33**	**99.99**	**94.42**
	TF	94.30	50.50	**90.29**	**96.76**	**92.14**	76.45	13.60	**87.49**	**93.54**	**70.50**	91.70	16.21	**94.03**	**99.86**	**89.72**	94.24	21.77	**93.85**	**99.99**	**93.96**
	BAE		74.80	**57.55**	**96.25**	**93.64**		27.50	**82.46**	**96.30**	**73.06**		35.21	**78.99**	**99.28**	**89.77**		44.00	**80.55**	**99.99**	**93.89**

false positive rate ($\approx 2\%$) of adversary detection on natural examples, resulting in a very slight performance degradation on natural examples. Further, the adaptability of RAPID to previously unseen attack methods is evidenced in Table 5, highlighting the versatility of our adversarial detector. It excels at identifying adversaries by detecting disruptions introduced by malicious attackers, such as grammar errors and word misuse. Note that detection performance on the AGNews dataset is lower due to the absence of news data in the BERT training corpus, as discussed in Table 8 of He et al. (2021).

4.2 Adversary defense performance

As for the adversary defense, RAPID outperforms existing methods across all datasets, as outlined in Table 2. When we focus on correctly identified adversaries, RAPID can effectively repair up to 92% to 99% of them, even on the challenging 10-category Yahoo datasets. Our research also sheds light on the limitations of unsupervised text-level and feature-level reconstruction methods, as reported in studies such as Zhou et al. (2019); Mozes et al. (2021); Wang et al. (2022c). These methods struggle to rectify the deep semantics in adversaries, rendering them inefficient and inferior. Additionally, we find that previous methods are not robust when defending against adversaries in short texts, as evidenced by their failure on the SST2 and Amazon datasets. In summary, RAPID employs adversarial attackers to repair adversaries' deep semantics and minimize edits in the text space, resulting in satisfactory adversarial defense. We emphasize the importance of dedicated deep semantics repair in the context of adversarial defense against unsupervised features and text space reconstruction.

Table 3: The performance of RAPID **without pseudo-similarity filtering**, where colored numbers indicate performance declines. The metrics not unaffected by the pseudo-similarity filtering are omitted.

DATASET	ATTACKER	DTA	RPA
AGNews	PWWS	94.19(−1.69 ↓)	90.80(−1.56 ↓)
	TF	94.26(−2.50 ↓)	91.35(−0.79 ↓)
	BAE	92.98(−3.27 ↓)	91.44(−2.20 ↓)
Yahoo!	PWWS	88.04(−4.43 ↓)	65.38(−4.02 ↓)
	TF	91.28(−2.26 ↓)	67.48(−3.02 ↓)
	BAE	92.48(−3.84 ↓)	71.35(−1.71 ↓)
SST2	PWWS	98.12(−0.50 ↓)	87.80(−2.05 ↓)
	TF	98.03(−1.83 ↓)	88.40(−1.32 ↓)
	BAE	95.87(−3.41 ↓)	87.52(−2.25 ↓)
Amazon	PWWS	99.99(0.00)	94.40(−0.02 ↓)
	TF	98.92(−1.07 ↓)	93.31(−0.65 ↓)
	BAE	98.53(−1.41 ↓)	93.62(−0.27 ↓)

4.3 Ablation for similarity filtering

The pseudo-semantic similarity filtering (Section 2.2.3) exclusively affects the defense process, so we have omitted the unaffected metrics, such as the detection accuracy in Table 2. From the results shown in Table 3, we find that the adversarial defense performance of RAPID without this filtering strategy is notably inferior ($\approx 1\%$) in most cases. Further, the degradation in defense performance is more pronounced in the case of the AGNews and Yahoo! datasets compared to the SST2 and Amazon datasets. This discrepancy is attributed to the larger vocabularies and longer text lengths in the AGNews and Yahoo! datasets, resulting in diversified repaired examples in terms of similarity.

4.4 Adaptive adversarial defesne

Table 4: Defense performance (RPA) of the original attacks (OA) and adaptive attacks (AA), respectively.

	PWWS		TF		BAE	
	OA	AA	OA	AA	OA	AA
AGNews	92.36	85.82	92.14	89.13	93.64	86.61
Yahoo!	69.30	65.41	80.50	87.58	73.06	70.17
SST2	89.85	83.51	89.72	83.68	89.77	86.92
Amazon	99.42	93.43	93.96	88.75	93.89	88.51

Although the defense against adaptive adversarial attacks in CV has been well established, the adaptive attacks in natural language processing are still under early investigation because of the non-differentiable nature of text space. To the best of our knowledge, there are no open-source or click-to-run adversarial defenses for adaptive adversarial attacks. We adopt a simple evaluation experiment of defense against EOT (Athalye et al., 2018), an adaptive attack from CV, where the implementation details of this experiment are available in Wang et al. (2022b). We show the results in Table 4.

Overall, while adaptive attacks typically resulted in reduced effectiveness compared to original attacks, there were instances (e.g., TF on Yahoo!) where adaptive strategies either maintained or slightly improved performance. This indicates variability in how different attack methods and datasets interact with adaptive defensive strategies.

4.5 Further research questions

We discuss more findings about RAPID by answering the following research questions (RQs).

RQ1: How is the generalization ability of RAPID to unknown attackers?

Methods: To assess the generalization ability of the in-victim-model adversarial detector in RAPID, we have conducted experiments among various state-of-the-art adversarial attackers: PSO, IGA, DEEPWORDBUG, and CLARE, which were not included in the training of the adversarial detector in RAPID. Note that better adversarial detection and defense performance against unknown adversarial attackers indicates a superior generalizability of RAPID.

Results: The results are listed in Table 5. In terms of adversarial defense, RAPID is capable of repairing a substantial number of adversaries generated by various unknown attack methods (up to 87.68% and 94.65% on the SST2 and Amazon datasets, respectively). However, RAPID experiences a decline in performance in identifying and defending

Table 5: Performance of RAPID for adversarial detection and defense **against unknown adversarial attacks**.

DATASET	ATTACKER	ATA	DTA	DFA	RPA
AGNews	PSO	14.83	68.46	67.82	90.39
	IGA	26.87	76.74	74.59	92.33
	DEEPWORDBUG	45.53	72.73	87.23	89.33
	CLARE	8.46	62.78	61.54	64.78
Yahoo!	PSO	6.28	80.26	76.89	87.82
	IGA	14.75	82.69	81.02	54.55
	DEEPWORDBUG	51.34	72.73	87.10	62.27
	CLARE	3.56	64.85	62.40	52.47
SST2	PSO	7.95	87.50	87.50	82.61
	IGA	18.39	89.33	98.67	87.68
	DEEPWORDBUG	30.67	95.44	83.59	81.90
	CLARE	2.59	62.50	59.37	65.30
Amazon	PSO	5.76	90.48	90.48	91.55
	IGA	14.91	92.31	92.31	94.65
	DEEPWORDBUG	43.43	87.04	85.19	86.87
	CLARE	3.25	60.44	59.37	62.94

against adversaries when facing the challenging CLARE attack. This performance degradation is likely attributed to their ineffective adversarial detection, which could potentially be improved by training CLARE-based adversaries for adversarial detection within RAPID. In summary, RAPID has demonstrated robust generalization ability, effectively detecting and repairing a wide array of adversaries generated by unknown attackers.

RQ2: Does perturbation defocusing really repair adversaries?

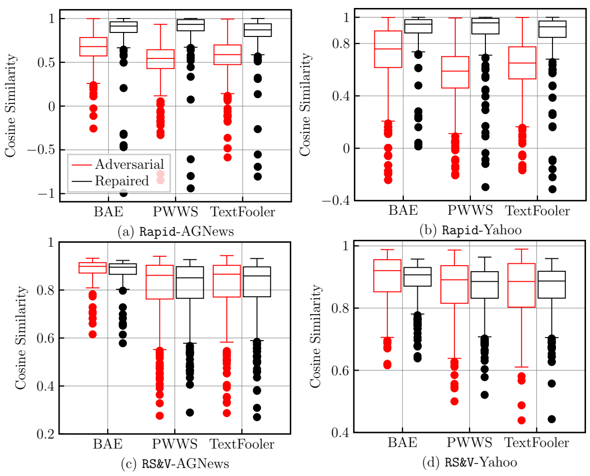

Figure 4: Box plots of semantic cosine similarity score distributions on multi-categorial datasets. Similar to Figure 1, RAPID is more competent to repair semantics according to the feature similarity score distributions.

Methods: To address this RQ, we investigate the discrepancy between adversaries and their repaired counterparts in the feature space. Specifically, we employ three attackers (i.e., BAE, PWWS, TEXTFOOLER) to generate adversaries and their corresponding repaired examples. Using the victim model, we encode these examples into the feature space and evaluate the cosine similarity between adversary-natural example pairs and repaired

Table 6: The performance of RAPID on four public datasets **based on the victim model DEBERTA**. The numbers in red color indicate performance declines compared to the BERT-based RAPID.

DATASET	ATTACKER	NTA	ATA	DTA	DFA	RPA
AGNews	PWWS	96.69	62.77	96.47	98.47	93.12
	TF		39.85	91.41	95.90 ↓	93.69
	BAE		81.64	90.20	97.92	93.40 ↓
Yahoo!	PWWS	78.63	15.70	88.91	92.64	70.47
	TF		6.19	89.32	92.60	69.96 ↓
	BAE		47.50	90.25	93.74 ↓	72.12 ↓
SST2	PWWS	95.01	37.14	95.21	98.42	94.15
	TF		22.59	93.06 ↓	99.08	94.58
	BAE		38.84	80.82	98.59	94.16
Amazon	PWWS	95.51	22.72	97.62	99.99	94.55
	TF		23.95	94.91	99.99	94.84
	BAE		56.65	82.71	99.99	94.50

adversary-natural example pairs. The larger cosine similarity scores indicate better performance in repairing the deep semantics in the adversaries.

Results: The semantic similarity score distributions (e.g., the median similarity scores of repaired examples are always larger than the adversaries) from Figure 1 and Figure 4 reveal a notable global similarity between the natural examples and repaired examples by RAPID, which means RAPID does repair the deep semantics of the adversaries. Conversely, the similarity scores of the repaired examples obtained using defenders (without adversarial detection) are indistinguishable from the adversarial examples across all datasets. In conclusion, our observations show the ability of RAPID to effectively repair the deep semantics of adversaries.

RQ3: How does the inherent robustness of the victim model affect RAPID?

Methods: We assessed the impact of the inherent robustness of the victim model, focusing on DEBERTA, a cutting-edge PLM utilized across various tasks. Specifically, we trained a victim model based on DEBERTA, replicating the experimental setup and evaluating the performance variation of RAPID based on this DEBERTA victim model.

Results: As in Table 6, the DEBERTA-based victim model demonstrates superior accuracy under adversarial attacks, indicating higher inherent robustness in DEBERTA compared to the victim model built on BERT. In particular, DEBERTA-based RAPID excels in identifying adversaries across all classification datasets, especially on the binary datasets. In short, the performance in adversarial detection and defense follows a similar upward trajectory to the capacity of the base model.

5 Related Works

Prior research on adversarial defense can be classified into three categories: adversarial training-based methods (Miyato et al., 2017; Zhu et al., 2020; Ivgi and Berant, 2021); context reconstruction-based methods (Pruthi et al., 2019; Liu et al., 2020b; Mozes et al., 2021; Keller et al., 2021; Chen et al., 2021; Xu et al., 2022; Li et al., 2022; Swenor and Kalita, 2022); and feature reconstruction-based methods(Zhou et al., 2019; Jones et al., 2020; Wang et al., 2021a). Some studies (Wang et al., 2021b) also investigated hybrid defense methods. As for the adversarial training-based methods, they are notorious for the performance degradation of natural examples. They can improve the robustness of PLMs by fine-tuning, yet increasing the cost of model training caused by catastrophic forgetting (Dong et al., 2021b). Text reconstruction-based methods, such as word substitution (Mozes et al., 2021; Bao et al., 2021) and translation-based reconstruction, may fail to identify semantically repaired adversaries or introduce new malicious perturbations (Swenor and Kalita, 2022). Feature reconstruction methods, on the other hand, may struggle to repair typo attacks (Liu et al., 2020a; Tan et al., 2020; Jones et al., 2020), sentence-level attacks (Zhao et al., 2018; Cheng et al., 2019), and other unknown attacks. There are some works towards the adversarial detection and defense joint task (Zhou et al., 2019; Mozes et al., 2021; Wang et al., 2022c). However, these adversarial detection methods may be ineffective for unknown adversarial attackers and can hardly alleviate resource waste in adversarial defense. Another similar work to RAPID is `Textshield` (Shen et al., 2023), which aims to defend against word-level adversarial attacks by detecting adversarial sentences based on a saliency-based detector and fixing the adversarial examples using a corrector. Overall, our study focuses on maintaining the semantics by introducing minimal safe perturbations into adversaries, thus alleviating the semantic shifting problem in all reconstruction-based works.

6 Conclusion

We propose RAPID to repair semantics in adversarial examples. RAPID shows an outstanding adversarial defense performance (up to $\approx 99\%$ of identified adversarial examples). It is believed that RAPID has the potential to significantly shift the landscape of textual adversarial defense.

Limitations

One limitation of the proposed method is that it tends to introduce new perturbations into the adversaries, which may lead to semantic shifts. This may be unsafe for some tasks, e.g., machine translation. Furthermore, the method requires a large amount of computational resources to generate the adversaries during the training phase, which may be a limitation in some scenarios. Finally, the method has not been tested on a wide range of NLP tasks and domains, and further evaluations on other tasks and domains are necessary to fully assess its capabilities.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani B. Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *EMNLP'18: Proc. of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896. Association for Computational Linguistics.

Anish Athalye, Nicholas Carlini, and David A. Wagner. 2018. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 274–283. PMLR.

Rongzhou Bao, Jiayi Wang, and Hai Zhao. 2021. Defending pre-trained language models from adversarial word substitution without performance sacrifice. In *ACL-IJCNLP'21: Findings of the 2021 Conference of the Association for Computational Linguistics: ACL-IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL-IJCNLP 2021 of *Findings of ACL*, pages 3248–3258. Association for Computational Linguistics.

Nicholas Boucher, Ilia Shumailov, Ross Anderson, and Nicolas Papernot. 2022. Bad characters: Imperceptible NLP attacks. In *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*, pages 1987–2004. IEEE.

Guandan Chen, Kai Fan, Kaibo Zhang, Boxing Chen, and Zhongqiang Huang. 2021. Manifold adversarial augmentation for neural machine translation. In *ACL-IJCNLP'21: Findings of the 2021 Conference of the Association for Computational Linguistics*, pages 3184–3189. Association for Computational Linguistics.

Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust neural machine translation with doubly adversarial inputs. In *ACL'19: Proc. of the 57th Conference of the Association for Computational Linguistics*, pages 4324–4333. 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 *NAACL-HLT'19: Proc. of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186. Association for Computational Linguistics.

Xinshuai Dong, Anh Tuan Luu, Rongrong Ji, and Hong Liu. 2021a. Towards robustness against natural language word substitutions. In *ICLR'21: Proc. of the 9th International Conference on Learning Representations*. OpenReview.net.

Xinshuai Dong, Anh Tuan Luu, Min Lin, Shuicheng Yan, and Hanwang Zhang. 2021b. How should pre-trained language models be fine-tuned towards adversarial robustness? In *NeurIPS'21: Proc. of the 2021 Conference on Neural Information Processing Systems*, pages 4356–4369.

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, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 31–36. Association for Computational Linguistics.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *SP'18: Proc. of the 2018 IEEE Security and Privacy Workshops*, pages 50–56. IEEE Computer Society.

Siddhant Garg and Goutham Ramakrishnan. 2020. BAE: bert-based adversarial examples for text classification. In *EMNLP'20: Proc. of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 6174–6181. Association for Computational Linguistics.

Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, Florian Stimberg, Dan Andrei Calian, and Timothy A. Mann. 2021. Improving robustness using generated data. In *NeurIPS'21: Advances in Neural Information Processing Systems*, pages 4218–4233.

Chuan Guo, Alexandre Sablayrolles, Hervé Jégou, and Douwe Kiela. 2021. Gradient-based adversarial attacks against text transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5747–5757. Association for Computational Linguistics.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *CoRR*, abs/2111.09543.

Maor Ivgi and Jonathan Berant. 2021. Achieving model robustness through discrete adversarial training. In *EMNLP'21: Proc. of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1529–1544. 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. In *AAAI'20: Proc. of the 34th AAAI Conference on Artificial Intelligence*, pages 8018–8025. AAAI Press.

Erik Jones, Robin Jia, Aditi Raghunathan, and Percy Liang. 2020. Robust encodings: A framework for combating adversarial typos. In *ACL'20: Proc. of the 58th Annual Meeting of the Association for Computational Linguistics Conference*, pages 2752–2765. Association for Computational Linguistics.

Yannik Keller, Jan Mackensen, and Steffen Eger. 2021. Bert-defense: A probabilistic model based on BERT to combat cognitively inspired orthographic adversarial attacks. In *ACL-IJCNLP'21: Findings of the 2021 Conference of the Association for Computational Linguistics*, volume ACL-IJCNLP 2021 of *Findings of ACL*, pages 1616–1629. Association for Computational Linguistics.

Dianqi Li, Yizhe Zhang, Hao Peng, Liqun Chen, Chris Brockett, Ming-Ting Sun, and Bill Dolan. 2021. Contextualized perturbation for textual adversarial attack. In *NAACL-HLT'21: Proc. of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 5053–5069. Association for Computational Linguistics.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2019. Textbugger: Generating adversarial text against real-world applications. In *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*. The Internet Society.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: adversarial attack against BERT using BERT. In *EMNLP'20: Proc. of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 6193–6202. Association for Computational Linguistics.

Linyang Li, Demin Song, Jiehang Zeng, Ruotian Ma, and Xipeng Qiu. 2022. Rebuild and ensemble: Exploring defense against text adversaries. *CoRR*, abs/2203.14207.

Hui Liu, Yongzheng Zhang, Yipeng Wang, Zheng Lin, and Yige Chen. 2020a. Joint character-level word embedding and adversarial stability training to defend adversarial text. In *AAAI'20: Proc. of the 34th AAAI Conference on Artificial Intelligence*, pages 8384–8391. AAAI Press.

Kai Liu, Xin Liu, An Yang, Jing Liu, Jinsong Su, Sujian Li, and Qiaoqiao She. 2020b. A robust adversarial training approach to machine reading comprehension. In *AAAI'20: Proc. of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 8392–8400. AAAI Press.

Takeru Miyato, Andrew M. Dai, and Ian J. Goodfellow. 2017. Adversarial training methods for semi-supervised text classification. In *ICLR'17: Proc. of the 5th International Conference on Learning Representations*. OpenReview.net.

Han Cheol Moon, Shafiq R. 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), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 5145–5165. Association for Computational Linguistics.

John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 119–126. Association for Computational Linguistics.

Maximilian Mozes, Pontus Stenetorp, Bennett Kleinberg, and Lewis D. Griffin. 2021. Frequency-guided word substitutions for detecting textual adversarial examples. In *EACL'21: Proc. of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, pages 171–186. Association for Computational Linguistics.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Danish Pruthi, Bhuwan Dhingra, and Zachary C. Lipton. 2019. Combating adversarial misspellings with robust word recognition. In *ACL'19: Proc. of the 57th Conference of the Association for Computational Linguistics*, pages 5582–5591. 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 *ACL'19: Proc. of the 57th Conference of the Association for Computational Linguistics*, pages 1085–1097. Association for Computational Linguistics.

Jérôme Rony, Luiz G. Hafemann, Luiz S. Oliveira, Ismail Ben Ayed, Robert Sabourin, and Eric Granger. 2019. Decoupling direction and norm for efficient gradient-based L2 adversarial attacks and defenses. In *CVPR'19: IEEE Conference on Computer Vision and Pattern Recognition*, pages 4322–4330. Computer Vision Foundation / IEEE.

Lingfeng Shen, Ze Zhang, Haiyun Jiang, and Ying Chen. 2023. Textshield: Beyond successfully detecting adversarial sentences in text classification. In *ICLR'23: The Eleventh International Conference on Learning Representations*. OpenReview.net.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL.

Abigail Swenor and Jugal Kalita. 2022. Using random perturbations to mitigate adversarial attacks on sentiment analysis models. *CoRR*, abs/2202.05758.

Samson Tan, Shafiq R. Joty, Lav R. Varshney, and Min-Yen Kan. 2020. Mind your inflections! improving NLP for non-standard englishes with base-inflection encoding. In *EMNLP'20: Proc. of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 5647–5663. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. In *NeurIPS'19: Advances in Neural Information Processing Systems*, pages 3261–3275.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR'19: 7th International Conference on Learning Representations*. OpenReview.net.

Boxin Wang, Hengzhi Pei, Boyuan Pan, Qian Chen, Shuohang Wang, and Bo Li. 2020. T3: tree-autoencoder constrained adversarial text generation for targeted attack. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6134–6150. Association for Computational Linguistics.

Boxin Wang, Chejian Xu, Xiangyu Liu, Yu Cheng, and Bo Li. 2022a. Semattack: Natural textual attacks via different semantic spaces. In *Findings of the Association for Computational Linguistics: NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 176–205. Association for Computational Linguistics.

Jiayi Wang, Rongzhou Bao, Zhuosheng Zhang, and Hai Zhao. 2022b. Rethinking textual adversarial defense for pre-trained language models. *IEEE ACM Trans. Audio Speech Lang. Process.*, 30:2526–2540.

Xiaosen Wang, Jin Hao, Yichen Yang, and Kun He. 2021a. Natural language adversarial defense through synonym encoding. In *UAI'21: Proc. of the 37th Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pages 823–833. AUAI Press.

Xiaosen Wang, Yifeng Xiong, and Kun He. 2022c. Detecting textual adversarial examples through randomized substitution and vote. In *UAI*, volume 180 of *Proceedings of Machine Learning Research*, pages 2056–2065. PMLR.

Xiaosen Wang, Yichen Yang, Yihe Deng, and Kun He. 2021b. Adversarial training with fast gradient projection method against synonym substitution based text attacks. In *AAAI'21: Proc. of the 35th AAAI Conference on Artificial Intelligence*, pages 13997–14005. AAAI Press.

Zekai Wang, Tianyu Pang, Chao Du, Min Lin, Weiwei Liu, and Shuicheng Yan. 2023. Better diffusion models further improve adversarial training. *CoRR*, abs/2302.04638.

Jianhan Xu, Cenyuan Zhang, Xiaoqing Zheng, Linyang Li, Cho-Jui Hsieh, Kai-Wei Chang, and Xuanjing Huang. 2022. Towards adversarially robust text classifiers by learning to reweight clean examples. In *ACL'22: Findings of the 2022 Conference of the Association for Computational Linguistics*, pages 1694–1707. Association for Computational Linguistics.

Yuancheng Xu, Yanchao Sun, Micah Goldblum, Tom Goldstein, and Furong Huang. 2023. Exploring and exploiting decision boundary dynamics for adversarial robustness. *CoRR*, abs/2302.03015.

Puyudi Yang, Jianbo Chen, Cho-Jui Hsieh, Jane-Ling Wang, and Michael I. Jordan. 2020. Greedy attack and gumbel attack: Generating adversarial examples for discrete data. *J. Mach. Learn. Res.*, 21:43:1–43:36.

Yichen Yang, Xiaosen Wang, and Kun He. 2022. Robust textual embedding against word-level adversarial attacks. In *UAI*, volume 180 of *Proceedings of Machine Learning Research*, pages 2214–2224. PMLR.

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 *ACL'20: Proc. of the 58th Annual Meeting of the Association for Computational Linguistics Conference*, pages 6066–6080. Association for Computational Linguistics.

Guoyang Zeng, Fanchao Qi, Qianrui Zhou, Tingji Zhang, Zixian Ma, Bairu Hou, Yuan Zang, Zhiyuan Liu, and Maosong Sun. 2021. Openattack: An open-source textual adversarial attack toolkit. In *Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL 2021 - System Demonstrations, Online, August 1-6, 2021*, pages 363–371. Association for Computational Linguistics.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657.

Xinze Zhang, Junzhe Zhang, Zhenhua Chen, and Kun He. 2021. Crafting adversarial examples for neural machine translation. In *ACL-IJCNLP'21: Proc. of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 1967–1977. Association for Computational Linguistics.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2018. Generating natural adversarial examples. In *ICLR'18: Proc. of the 6th International Conference on Learning Representations*. OpenReview.net.

Yichao Zhou, Jyun-Yu Jiang, Kai-Wei Chang, and Wei Wang. 2019. Learning to discriminate perturbations for blocking adversarial attacks in text classification. In *EMNLP-IJCNLP'19: Proc. of the Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 4903–4912. 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 *ICLR'20: Proc. of the 8th International Conference on Learning Representations*. OpenReview.net.

A Reproducibility

To encourage everyone interested in our work to implement RAPID, we have taken the following steps:

- We have created an online click-to-run demo alailable at `https://tinyurl.com/22ercuf8` for easy evaluation. Everyone can input adversarial examples and obtain the repaired examples immediately.
- We have released the detailed source codes and processed datasets that can be retrieved in the supplementary materials. This enables everyone to access the official implementation, aiding in understanding the paper and facilitating their own implementations.
- We will also release an online benchmark tool for evaluating the performance of adversarial attackers under the defense of RAPID. This step is essential for reducing evaluation variance across different codebases.

These efforts are aimed at promoting the reproducibility of our work and facilitating its implementation by the research community.

B Word-level adversarial attack

Our focus is on defending against word-level adversarial attacks. However, our method can be easily adapted to different types of adversarial attacks. Let $x = (x_1, x_2, \cdots, x_n)$ be a natural sentence, where x_i, $1 \leq i \leq n$, denotes a word. y is the ground truth label. Word-level attackers generally replace some original words with similar words (e.g., synonyms) to fool the objective model. For example, substituting x_i with $\hat{x}_i$ generates an adversary: $\hat{x} = (x_1, \cdots, \hat{x}_i, \cdots, x_n)$, where $\hat{x}_i$ is an alternative substitution for x_i. For an adversary $\hat{x}$, the objective model F predicts its label as follows:

$$\hat{y} = \operatorname{argmax} F\left(\cdot|\hat{x}\right), \tag{7}$$

where $\hat{y} \neq y$ if $\hat{x}$ is a successful adversary. To represent adversarial attacks to F using an adversarial attacker $\mathcal{A}$, we denote an adversarial attack as follows:

$$(\hat{x}, \hat{y}) \leftarrow \mathcal{A}(F, (x, y)), \tag{8}$$

where x and y denote the natural example and its true label. $\hat{x}$ and $\hat{y}$ are the perturbed adversary and label, respectively.

B.1 Investigation of textual adversarial attack

This section delves into an examination of textual adversarial attacks.

Traditional approaches, such as those noted by Li et al. (2019) and Ebrahimi et al. (2018), often involve character-level modifications to words (e.g., changing "good" to "go0d") to deceive models by altering their statistical patterns. In a different approach, knowledge-based perturbations, exemplified by the work of Zang et al. (2020), employ resources like HowNet to confine the search space, especially in terms of substituting words.

Recent research (Garg and Ramakrishnan, 2020; Li et al., 2020) has investigated using pre-trained models for generating context-aware perturbations (Li et al., 2021). Semantic-based methods, such as SemAttack (Wang et al., 2022a), typically use BERT embedding clusters to create sophisticated adversarial examples. This differs from prior heuristic methods that employed greedy algorithms (Yang et al., 2020; Jin et al., 2020) or genetic algorithms (Alzantot et al., 2018; Zang et al., 2020), as well as gradient-based techniques (Wang et al., 2020; Guo et al., 2021) that concentrated on syntactic limitations.

With the evolution of adversarial attack techniques, numerous tools such as TextAttack (Morris et al., 2020) and OpenAttack (Zeng et al., 2021) have been developed and made available in the open-source community.

These resources facilitate deep learning researchers to efficiently assess adversarial robustness with minimal coding. Therefore, our experiments in adversarial defense are conducted using the TextAttack framework, and we extend our gratitude to the authors and contributors of TextAttack for their significant efforts.

C Experiments implementation

C.1 Adversarial attackers in our experiments

We employ BAE, PWWS, and TEXTFOOLER to generate adversaries for training the adversarial detector. These attackers are chosen because they represent different types of attacks, allowing us to train a detector capable of recognizing a variety of adversarial attacks. This detector exhibits good generalization ability, which we confirm through experiments with other adversarial attackers such as IGA, DEEPWORDBUG, PSO, and CLARE. Including a larger number of adversarial attackers in the training process can further enhance the performance of the detector. We provide a brief introduction to these adversarial attackers:

a) **BAE** (Garg and Ramakrishnan, 2020) generates perturbations by replacing and inserting tagged words based on the candidate words generated by the masked language model (MLM). To identify the most important words in the text, BAE employs a word deletion-based importance evaluation method.
b) **PWWS** (Ren et al., 2019) is an adversarial attacker based on synonym replacement, which combines word significance and classification probability for word replacement.
c) **TEXTFOOLER** (Jin et al., 2020) considers additional constraints (such as prediction consistency, semantic similarity, and fluency) when generating adversaries. TEXTFOOLER uses a gradient-based word importance measure to locate and perturb important words.

C.2 Hyperparameter settings

We employ the following configurations for fine-tuning classifiers:

1. The learning rates for both BERT and DEBERTA are set to 2×10^{-5}.
2. The batch size is 16, and the maximum sequence modeling length is 128.
3. Dropouts are set to 0.1 for all models.
4. The loss functions of all objectives use cross-entropy.
5. The victim models and RAPID models are trained for 5 epochs.
6. The optimizer used for fine-tuning objective models is `AdamW`.

Please refer to our released code for more details.

C.3 Evaluation metrics

In this section, we introduce the adversarial defense metrics. First, we select a target dataset, referred to as $\mathcal{D}$, containing only natural examples. Our goal is to generate adversaries that can deceive a victim model F_J. We group the successful adversaries into a subset called $\mathcal{D}_{adv}$ and the remaining natural examples with no adversaries into another subset called $\mathcal{D}_{nat}$. We then combine these two subsets to form the attacked dataset, $\mathcal{D}_{att}$. We apply RAPID to $\mathcal{D}_{att}$ to obtain the repaired dataset, $\mathcal{D}_{rep}$. The evaluation metrics used in the experiments are described as follows:

$$\text{NTA} = \frac{TP_{\mathcal{D}} + TN_{\mathcal{D}}}{P_{\mathcal{D}} + N_{\mathcal{D}}}$$

$$\text{ATA} = \frac{TP_{\mathcal{D}_{att}} + TN_{\mathcal{D}_{att}}}{P_{\mathcal{D}_{att}} + N_{\mathcal{D}_{att}}}$$

$$\text{DTA} = \frac{TP^{*}_{\mathcal{D}_{adv}} + TN^{*}_{\mathcal{D}_{adv}}}{P^{*}_{\mathcal{D}_{adv}} + N^{*}_{\mathcal{D}_{adv}}}$$

$$\text{DFA} = \frac{TP_{\mathcal{D}_{adv}} + TN_{\mathcal{D}_{adv}}}{P_{\mathcal{D}_{adv}} + N_{\mathcal{D}_{adv}}}$$

$$\text{RPA} = \frac{TP_{\mathcal{D}_{rep}} + TN_{\mathcal{D}_{rep}}}{P_{\mathcal{D}_{rep}} + N_{\mathcal{D}_{rep}}}$$

where TP ,TN, P and N are the number of true positives and true negatives, positive and negative in standard classification, respectively. TP^{*}, TN^{*}, P^{*} and N^{*} indicate the case numbers in adversarial detection.

C.4 Experimental environment

The experiments are carried out on a computer running the Cent OS 7 operating system, equipped with an RTX 3090 GPU and a Core i-12900k processor. We use the PyTorch 1.12 library and a modified version of TextAttack, based on version 0.3.7.

D Ablation experiments

D.1 Performance of RAPID against LLM-based adversarial attack

Table 7: Defense performance of RAPID against adversarial attacks generated by `ChatGPT-3.5`.

DATASET	ATTACKER	DFA	RPA
AGNews	CHATGPT	RS&V	59.0
		RAPID	**72.0**
Yahoo!	CHATGPT	RS&V	49.0
		RAPID	**61.0**
SST2	CHATGPT	RS&V	37.0
		RAPID	**74.0**
Amazon	CHATGPT	RS&V	58.0
		RAPID	82.0

Recent years have witnessed the superpower of large language models (LLMs) such as `ChatGPT` (OpenAI, 2023), which we hypothesize to have a stronger ability to generate adversaries. In this subsection, we evaluate the defense performance of RAPID against adversaries generated by `ChatGPT-3.5`. Specifically, for each dataset considered in our previous experiments, we use `ChatGPT`[9] to generate 100 adversaries and investigate the defense accuracy achieved by RAPID.

From the experimental results shown in Table 7, we find that RAPID consistently outperforms RS&V in terms of defense accuracy. Specifically, in the `SST2` dataset, RS&V records a defense accuracy of 37.0%, however, RAPID impressively repairs 74.0% of the attacks. Similar trends hold for the `Amazon` and `AGNews` datasets, where RAPID achieves defense accuracy of 82.0% and 72.0% respectively, in contrast to the 58.0% and 59.0% offered by RS&V. In conclusion, RAPID can defend against various unknown adversarial attacks which have a remarkable performance in contrast to existing adversarial defense approaches.

D.2 Performance of RAPID with different adversarial attackers in perturbation defocusing

In RAPID, PD can incorporate any adversarial attacker or even an ensemble of attackers, as the process doesn't require prior knowledge of the specific malicious perturbations. Regardless of which adversaries are deployed against RAPID, PWWS consistently seeks safe perturbations for the current adversarial examples. The abstract nature of PD is critical, allowing for adaptability and effectiveness against a broad spectrum of adversarial attacks, rendering it a versatile defense mechanism in our study.

In order to investigate the impact of $\hat{\mathcal{A}}_{PD}$ in `Phase #2`, we have implemented further experiments to demonstrate the adversarial defense performance of PD using different attackers, e.g., TEXTFOOLER and BAE. The results are shown in Table 8. According to the experimental results, it is observed that PWWS has a similar performance to TEXTFOOLER in PD, while BAE is slightly inferior to both PWWS and TEXTFOOLER. However, the variances are not significant among different attackers in PD, which means the performance of RAPID is not sensitive to the choice of $\hat{\mathcal{A}}_{PD}$, in contrast to the adversarial attack performance of the adversarial attacker.

D.3 Performance of RAPID without adversarial training objective

The rationale behind the adversarial training objective $\mathcal{L}_a$ in our study is founded on two key hypotheses.

a) **Enhancing Adversarial Detection:** We recognize an implicit link between the tasks of adversarial training and adversarial example detection. Our theory suggests that by incorporating an adversarial training objective, we can indirectly heighten the model's sensitivity to adversarial examples, leading to more accurate detection of such instances.

b) **Improving Model Robustness:** We posit that an adversarial training objective can bolster the model's robustness, thereby mitigating performance degradation when the model faces an attack. This approach is designed to strengthen the model against potential adversarial threats.

To validate these hypotheses, we conducted ablation experiments on the adversarial training objective. The experimental setup was aligned with that described in Table 2, and the results are outlined in Table 9.

These experimental findings reveal that omitting the adversarial training objective in RAPID consistently leads to a reduction in model robustness

[9] `ChatGPT3.5-0301`

Table 8: The adversarial detection and defense performance of RAPID based on different backends ($\hat{\mathcal{A}}_{PD}$). We report the average accuracy of five random runs. “TF” indicates TEXTFOOLER.

DEFENDER	ATTACKER	AGNews(4-category)					Yahoo!(10-category)					SST2 (2-category)					Amazon(2-category)				
		NTA	ATA	DTA	DFA	RPA	NTA	ATA	DTA	DTA	RPA	NTA	ATA	DTA	DFA	RPA	NTA	ATA	DTA	DFA	RPA
RAPID (PWWS)	PWWS		32.09	90.11	95.88	92.36		5.70	87.33	92.47	69.40		23.44	94.03	98.62	89.85		15.56	97.33	99.99	94.42
	TF	94.30	50.50	90.29	96.76	92.14	76.45	13.60	87.49	93.54	70.50	91.55	16.21	94.03	99.86	89.72	94.32	21.77	93.85	99.99	93.96
	BAE		74.80	57.55	96.25	93.64		27.50	82.46	96.30	73.06		35.21	78.99	99.28	89.77		44.00	80.55	99.99	93.89
RAPID (TF)	PWWS		32.09	83.67	94.07	92.27		5.70	65.01	83.25	65.33		23.44	36.90	98.90	90.67		15.56	29.60	99.99	94.33
	TF	94.30	50.50	82.44	96.46	92.67	76.45	13.60	74.21	92.96	71.00	91.55	16.21	39.70	99.98	90.73	94.32	21.77	40.70	99.99	94.33
	BAE		74.80	46.98	92.68	91.00		27.50	37.41	86.49	72.67		35.21	19.84	99.98	91.33		44.00	38.59	99.99	94.33
RAPID (BAE)	PWWS		32.09	83.67	93.22	92.08		5.70	65.01	81.15	64.00		23.44	36.90	93.92	87.67		15.56	29.60	99.54	94.00
	TF	94.30	50.50	82.44	95.96	92.33	76.45	13.60	74.21	87.79	67.33	91.55	16.21	39.70	96.55	89.00	94.32	21.77	40.70	99.61	93.64
	BAE		74.80	46.98	95.12	91.33		27.50	37.41	83.78	72.00		35.21	19.84	97.55	90.00		44.00	38.59	99.15	93.80

Table 9: The adversarial detection and defense performance of RAPID with (“w/”) and without (“w/o”) the adversarial training objective. We report the average accuracy of five random runs. “TF” indicates TEXTFOOLER.

DEFENDER	ATTACKER	AGNews(4-category)					Yahoo!(10-category)					SST2 (2-category)					Amazon(2-category)				
		NTA	ATA	DTA	DFA	RPA	NTA	ATA	DTA	DTA	RPA	NTA	ATA	DTA	DFA	RPA	NTA	ATA	DTA	DFA	RPA
RAPID (w/ $\mathcal{L}_a$)	PWWS		32.09	90.11	95.88	92.36		5.70	87.33	92.47	69.40		23.44	94.03	98.62	89.85		15.56	97.33	99.99	94.42
	TF	94.30	50.50	90.29	96.76	92.14	76.45	13.60	87.49	93.54	70.50	91.55	16.21	94.03	99.86	89.72	94.32	21.77	93.85	99.99	93.96
	BAE		74.80	57.55	96.25	93.64		27.50	82.46	96.30	73.06		35.21	78.99	99.28	89.77		44.00	80.55	99.99	93.89
RAPID (w/o $\mathcal{L}_a$)	PWWS		11.10	82.88	92.07	90.70		3.46	78.43	87.42	63.79		10.70	91.41	99.62	89.60		16.5	96.50	99.30	93.60
	TF	94.44	16.09	84.88	93.07	87.28	76.32	0.42	78.65	78.36	56.72	91.54	5.30	89.48	95.15	85.80	94.29	17.53	98.63	99.17	92.78
	BAE		67.93	83.17	91.49	91.15		45.10	71.89	75.47	64.56		25.70	57.01	95.64	87.10		45.54	92.67	99.48	93.31

across all datasets. This reduction can be as substantial as approximately 30%, adversely affecting the performance of the adversarial defense. Additionally, adversarial detection capabilities also diminish, with the most significant drop being around 20%. These results highlight the critical role of the adversarial training objective in RAPID, confirming its efficacy in enhancing both model robustness and adversarial example detection capabilities.

D.4 Performance of RAPID without multitask training objective

DATASET	MODEL	VICTIM-S	VICTIM-M
AGNews	BERT	94.30	93.90 (−0.40 ↓)
Yahoo!	BERT	76.45	76.61 (+0.16 ↑)
SST2	BERT	91.70	91.49 (−0.21 ↓)
Amazon	BERT	94.24	94.24 (—)

Table 10: Victim model’s accuracy (%) on clean dataset-based single-task and multitask training scenarios, i.e., **Victim-S** and **Victim-M** respectively. The experiments are based on the BERT model.

Before developing RAPID, we carefully considered the potential impact on classification performance due to multitask training objectives. This consideration was explored in our proof-of-concept experiments.

To delve deeper into this impact, we trained victim models as single-task models (i.e., no adversarial detection objective and adversarial training objective), instead of multitask training, and then collated detailed results for comparison with RAPID. In this experiment, we focused solely on evaluating performance using pure natural examples. The results of this comparison are outlined in Table 10. The symbols "↑" and "↓" accompanying the numbers indicate whether the performance is better or worse than that of the single-task model, respectively.

Based on these results, it is apparent that the inclusion of additional loss terms in multitask training objectives does impact the victim model’s performance on clean examples. However, this influence is not substantial across all datasets and shows only slight variations. This finding suggests that the impact of multitask training objectives is relatively minor when compared to traditional adversarial training methods.

D.5 Performance Comparison between RAPID and adversarial training baseline

We have conducted experiments to showcase the experimental results of the adversarial training baseline (AT). The victim model is BERT, and the experimental setup is the same as for RAPID, including the number of adversaries used for training. We only show the metric of repaired accuracy, as AT does not support detect-to-defense. The results (i.e., RPA (%)) are available in Table 11.

For these experiments, we used BERT as the vic-

DATASET	ATTACKER	RAPID	AT
AGNews	PWWS	92.36	60.10
	TF	92.14	61.87
	BAE	93.64	63.62
Yahoo!	PWWS	69.40	40.21
	TF	70.50	38.75
	BAE	73.06	42.97
SST2	PWWS	89.85	32.46
	TF	89.72	31.23
	BAE	89.77	34.61
Amazon	PWWS	94.42	51.90
	TF	93.96	49.49
	BAE	93.89	49.75

Table 11: The repaired performance of RAPID and the adversarial training baseline. We report the average accuracy of five random runs. "TF" indicates TEXTFOOLER.

tim model and maintained the same experimental setup as for RAPID, including the number of adversaries used for training. It's important to note that we focus solely on the repaired accuracy metric, as AT does not facilitate detect-to-defense functionality. From these results, it becomes apparent that the traditional adversarial training baseline is less effective compared to RAPID, which utilizes perturbation defocusing. Specifically, the adversarial defense accuracy of AT is generally below 50%. This finding underscores the limitations of traditional adversarial training methods, particularly their high cost and reduced effectiveness against adapted adversarial attacks.

D.6 Impact of k in RAPID

During the rebuttal period, we conducted additional experiments to investigate the impact of k. From the results shown in the following table, we can see that choosing a larger k can help improve the performance, especially when $1 \leq k \leq 3$. However, we also notice that when $k > 3$, the performance improvement becomes marginal.

D.7 Adversarial example evaluation

The importance of evaluation for adversarial defense instances has been emphasized in recent works. Consequently, we have devised a simulation strategy for human evaluation using ChatGPT-3.5. We curated prompts to leverage ChatGPT-3.5 as the human judge to evaluate 100 adversarial defense examples (i.e., repaired examples) from both Rapid and RS&V. We instructed ChatGPT-3.5 to identify which instances appear unnatural and calculated the number of unnatural repaired examples. Here are the experimental results in Table 13, where

Table 12: Defense performance of RAPID under different setting of k.

Attack	k	AGNews	Yahoo!	SST2	Amazon
PWWS	1	90.80	65.38	87.80	94.40
PWWS	2	91.24	65.24	88.01	94.21
PWWS	3	92.36	69.40	89.85	94.42
PWWS	4	92.41	69.32	89.49	94.29
PWWS	5	92.30	69.39	89.27	94.46
TF	1	91.35	67.48	88.40	93.31
TF	2	91.34	67.20	88.59	93.88
TF	3	92.14	70.50	89.72	93.96
TF	4	92.05	70.44	89.50	93.89
TF	5	92.31	80.56	89.32	93.99
BAE	1	91.44	71.35	87.52	93.62
BAE	2	91.38	72.91	88.07	93.22
BAE	3	93.64	73.06	89.77	93.89
BAE	4	93.71	73.11	89.68	93.95
BAE	5	93.55	73.42	89.81	93.78

LOWER is BETTER: Overall, RAPID produced

Table 13: The results of simulated human evaluation of adversarial defense example evaluation.

Defender	Attacker	AGNews	Yahoo!	SST2	Amazon
RAPID	PWWS	76	68	78	75
RAPID	TF	75	71	81	74
RAPID	BAE	82	72	82	77
RS&V	PWWS	96	86	87	83
RS&V	TF	98	78	83	80
RS&V	BAE	99	81	92	88

the least unnatural repaired examples. The results indicate that the repaired examples are generally easy to identify due to textual modifications, e.g., incorrect synonym replacements. The aim of adversarial defense is to repair model outputs, which have been effectively secured by Rapid according to our extensive experiments.

D.8 Efficiency evaluation of RAPID

The main efficiency depends on multiple adversarial perturbations search. We have implemented two experiments to investigate the efficiency of RAPID. Please note that the time costs for adversarial attack and defense are dependent on specific software and hardware environments.

Time costs for multiple defenses. We have collected three small sub-datasets that contain different numbers of adversarial examples and natural examples, say 200:0, 100:100, and 0:200. We apply adversarial detection and defense to this dataset and calculate the time costs. The results (measurement:

Attacker	AGNews			Yahoo!			SST2			Amazon		
	200:0	100:100	0:200	200:0	100:100	0:200	200:0	100:100	0:200	200:0	100:100	0:200
PWWS		142.090	298.603		313.317	621.196		36.268	126.054		438.532	875.083
TF	1.188	146.654	293.542	1.157	314.926	642.206	1.092	51.303	137.795	1.138	329.075	665.052
BAE		141.434	260.231		352.186	876.606		52.626	138.325		349.256	655.264

Table 14: The efficiency of RAPID defending against different adversarial attacks with different portions of natural and adversarial instances. The measurement is second.

Defender	Attacker	AGNews			Yahoo!			SST2			Amazon		
		Clean	Attack	Defense	Clean	Attack	Defense	Clean	Attack	Defense	Clean	Attack	Defense
	PWWS		2.081	1.356		4.958	3.308		0.529	0.588		4.745	3.678
Rapid	TF	0.008	2.460	1.317	0.008	4.693	3.128	0.006	0.662	0.571	0.007	4.003	4.607
	BAE		2.464	1.295		5.194	4.053		0.669	0.594		4.350	4.403

Table 15: The execution efficiency of inferring clean examples, generating, and defending against adversarial examples.

second) are available in Table 14.

Time costs for single defense. We have also detailed the time costs per natural example, adversarial attack, and adversarial defense in PDṪhe experimental results can be found in Table 15.

According to the experimental results, PD is slightly faster than the adversarial attack in most cases. Intuitively, the perturbed semantics in a malicious adversarial example are generally not robust, as most of the deep semantics remain within the adversarial example. Therefore, RAPID is able to rectify the example with fewer perturbations needed to search.

E Deployment Demo

We have created an anonymous demonstration of RAPID, which is available on Huggingface Space[10]. To illustrate the usage of our method, we provide two examples in Figure 5. In this demonstration, users can either input a new phrase along with a label or randomly select an example from a supplied dataset, to perform an attack, adversarial detection, and adversarial repair.

[10] https://huggingface.co/spaces/anonymous8/RPD-Demo

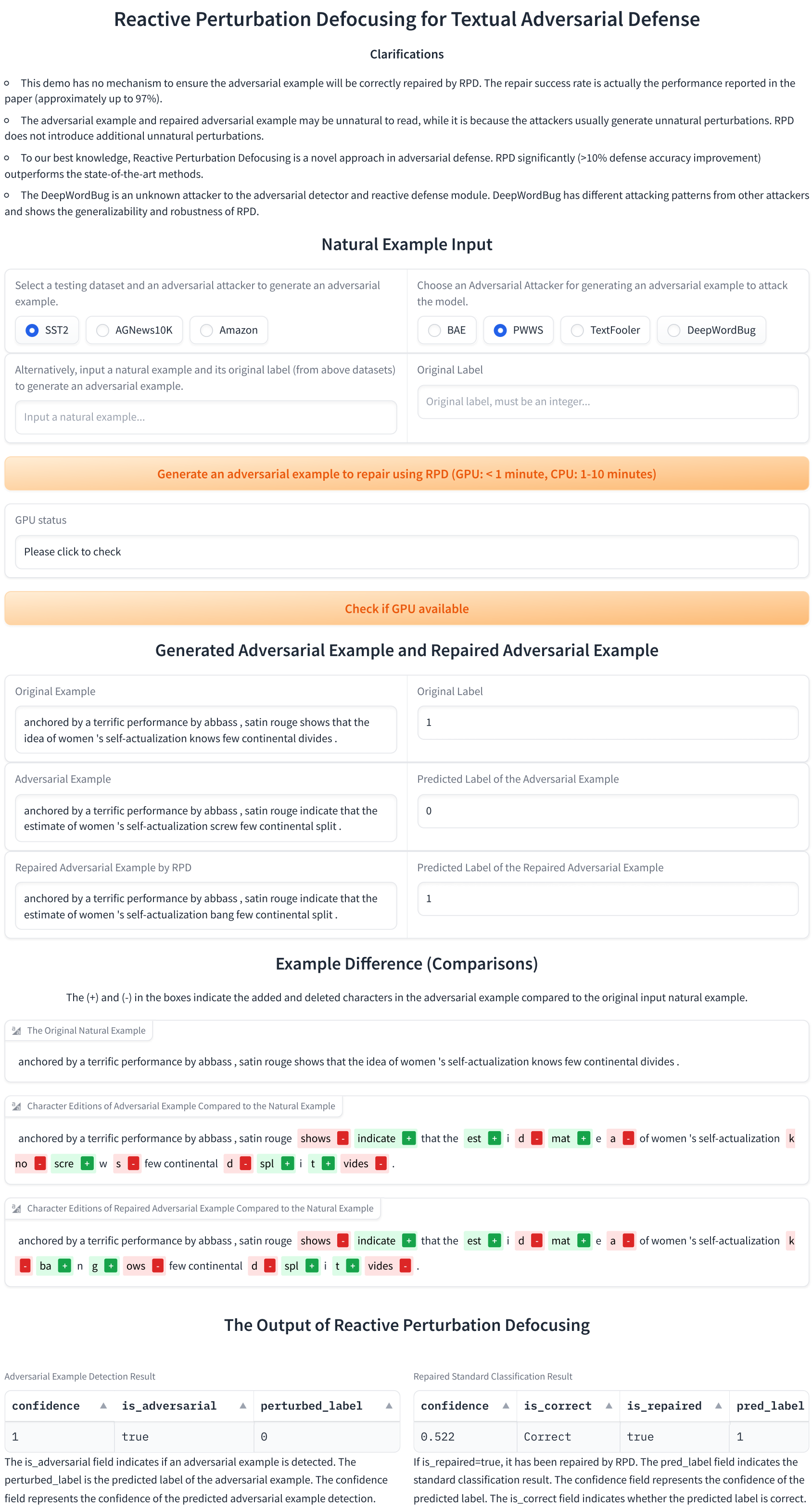

Figure 5: The demo examples of adversarial detection and defense built on RAPID for defending against multi-attacks. The comparisons between natural and repaired examples are available based on the "*difflib*" library. The "+" and "−" in the colored boxes indicate letters addition and deletion compared to the natural examples. It is observed that RAPID only injects only one perturbation to repair the adversarial example, i.e., changing "screw" to "bang" in the adversarial example.