PlugAT: A Plug and Play Module to Defend against Textual Adversarial Attack

Rui Zheng[1*], **Rong Bao**[1,2*], **Qin Liu**[3], **Tao Gui**[4†],
Qi Zhang[1,5†], **Xuanjing Huang**[1], **Rui Xie**[6], **Wei Wu**[5]
[1] School of Computer Science, Fudan University [2] Ant Group
[3] Viterbi School of Engineering, University of Southern California
[4] Institute of Modern Languages and Linguistics, Fudan University, Shanghai, China
[5] Shanghai Collaborative Innovation Center of Intelligent Visual Computing
[6] Meituan Inc., Beijing, China
{rzheng20,tgui,qz,xjhuang}@fudan.edu.cn
rbao22@m.fudan.edu.cn qliu4174@usc.edu

Abstract

Adversarial training, which minimizes the loss of adversarially perturbed examples, has received considerable attention. However, these methods require modifying all model parameters and optimizing the model from scratch, which is parameter inefficient and unfriendly to the already deployed models. As an alternative, we propose a pluggable defense module PlugAT, to provide robust predictions by adding a few trainable parameters to the model inputs while keeping the original model frozen. To reduce the potential side effects of using defense modules, we further propose a novel forgetting restricted adversarial training, which filters out bad adversarial examples that impair the performance of original ones. The PlugAT-equipped BERT model substantially improves robustness over several strong baselines on various text classification tasks, whilst training only 9.1% parameters. We observe that defense modules trained under the same model architecture have domain adaptation ability between similar text classification datasets.

1 Introduction

Deep neural networks have achieved great success in many fields, but they can be easily fooled by adversarial examples crafted by imperceptible perturbations on their normal counterparts (Goodfellow et al., 2015). Recent studies have shown that this phenomenon is widespread in NLP tasks (Jia and Liang, 2017; Liang et al., 2018; Wallace et al., 2019). In response to adversarial attackers, various adversarial defense methods are proposed to improve model robustness while maintaining high accuracy on both clean and adversarial examples (Dinan et al., 2019; Wang et al., 2021; Zheng et al., 2022; Liu et al., 2022).

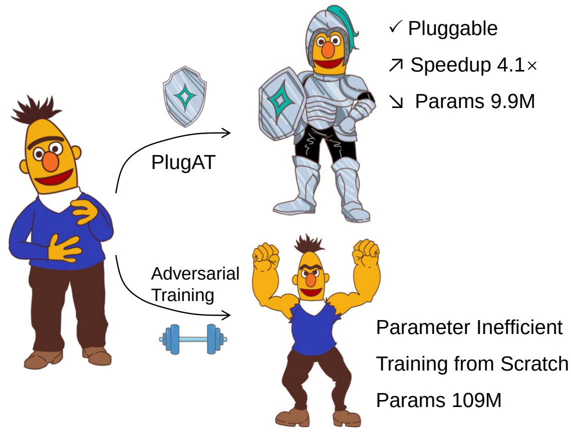

Figure 1: Comparison of adversarial training and our proposed PlugAT. Adversarial training requires training the model from scratch and modifying all model parameters, which is time-consuming and unfriendly for the deployed model. The proposed defense module improve the robustnesss of models in an extensible and efficient manner.

Among them, adversarial training is generally regarded as one of the strongest defense methods (Madry et al., 2018).

A major drawback of adversarial learning based methods is their high computational cost, as they require multi-step gradient descent to generate adversarial examples (Andriushchenko and Flammarion, 2020). The total time consumption of adversarial training is much more than that of standard training, and therefore some recent works have attempted to reduce the computational burden of adversarial training. FastAT (Wong et al., 2020) uses weaker and cheaper adversarial examples to replace strong ones and demonstrates that extremely weak adversarial training is capable of learning robust models. YOPO (Zhang et al., 2019) limits the number of forward and backward propagation without hurting the performance of the

* Equal contribution.
† Corresponding authors.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2873–2882
October 12–17, 2022.

trained model. FreeAT (Shafahi et al., 2019) and FreeLB (Zhu et al., 2020) leverage "free" strategies to generate diversified adversarial examples at a negligible additional cost compared to standard training.

These efficient approaches mainly focus on reducing the cost of generating adversarial examples, which comes at the cost of training the model from scratch with the parameters entirely modified. The prohibitively huge training demand poses a severe challenge for the deployment of practical robust NLP systems. We seek to mitigate this problem by learning external modules to achieve robust results. This way, we only need to train and load a small number of robustness-specific parameters without retraining the entire model, greatly improving the ease of use.

In this work, we propose PlugAT, a plug-and-play module for transformer-based pre-trained language models (PLMs) to defend against textual adversarial attacks. The defense module consists of layerwise trainable parameters that are prepended to the input sequence of each PLM layer. By optimizing the defense module with adversarial training, the model is guided to response with robust outputs without updating its parameters. To alleviate the possible damage caused by training adversarial examples on the performance of original examples, we propose a novel forgetting restricted adversarial training, which filter out "aggressive" adversarial examples. PlugAT-equipped BERT has promising robust performance on text classification tasks while updating only 9.1% robustness-specific parameters, reducing GPU time by about half compared to state-of-the-art efficient adversarial training methods. In addition, we prove that defense module has the domain adaptation capability and can work when transferred to similar tasks.Our codes are publicly available at *Github*[1]. The main contributions of our work are summarized as follows:

- A plug-and-play module PlugAT is proposed to improve robustness of the deployed models in an extensible and efficient manner.

- We propose the forgetting constrained adversarial training to mitigate performance degradation of the original examples caused by training "aggressive" adversarial examples.

- We verify the effectiveness of the defense module under three adversarial attacks and enrich more potential applications of adversarial training.

[1]https://github.com/ruizheng20/PlugAT

2 Related Work

2.1 Adversarial Attack

For the purpose of exploring robustness, adversarial attack has been extensively studied for continuous data of images (Goodfellow et al., 2015; Madry et al., 2018) as well as discrete data of texts (Li et al., 2018; Ren et al., 2019; Li et al., 2020), with the latter aspect being more challenging than the former. Textual attacks typically generate explicit adversarial examples by swapping the components of sentences into their counterparts, be it high in similarity semantically (Ren et al., 2019) or in terms of embedding (Li et al., 2020). TextFooler (Jin et al., 2020) and TextBugger (Li et al., 2018) leverages genetic algorithms to search for word-level substitution that is semantically similar and grammatically correct. To improve the success rate of this kind of attack, Li *et al.* (2020) repeat the process of searching and substituting until a successful attack. In this work, we demonstrate the effectiveness of our proposed method on the mentioned adversarial attacks.

2.2 Adversarial Training

To improve models' robustness against attacks, adversarial defence has also attracted increasing interests, the one of most powerful methods is adversarial training. These methods generally incorporate a min-max optimization between the adversarial perturbations and the models by limiting embedding-level perturbations to Frobenius normalization balls, such as PGD (Madry et al., 2018) and FreeLB (Zhu et al., 2020), or to token-level normalization balls as implemented in TAVAT (Li and Qiu, 2020). Since training a model from scratch is time-consuming and effort-taking, there has been a series of recent efforts to try to improve the efficiency of adversarial training. Fast adversarial training (Wong et al., 2020) uses weaker and cheaper adversarial examples to replace strong ones and achieves comparable performance. Free adversarial training (Shafahi et al., 2019) recycles the gradient information computed when updating model parameters to generate adversarial examples for free. These two methods eliminate the overhead cost of generating adversarial examples. Zhang *et* *al.* (2019) restrict most of the forward and back propagation within the first layer of the network during adversary updates limits, which reduces the total number of propagation and largely saves GPU time. A more practical question is that adversarial training needs to optimize all model parameters, which is inefficient and unfriendly for the deployed model in industrial applications.

2.3 Lightweight method

Researchers have long been studying how to efficiently transfer pre-trained models to downstream tasks (Ye et al., 2021; Ma et al., 2022). Adapter-tuning (Houlsby et al., 2019) inserts small task-specific trainable modules between the layers of pre-trained language models, and only trains these adapter modules while keeping the original network frozen. Similarly , Side-tuning (Zhang et al., 2020) trains a lightweight "side" network that can be fused with a pre-trained network through summation. Being even more lightweight, Prefix-Tuning (Li and Liang, 2021) maintains comparable performance while keeping model parameters frozen and tuning only 0.1% of the parameters to optimize a small continuous task-specific vector. These methods have been proven to reduce catastrophic forgetting in downstream tasks, and because the parameters are optimized in a limited space, they are more robust to adversarial attacks (Han et al., 2021). We take inspiration from these works and study how to use trainable modules to conduct adversarial training in an efficient and extensible method.

3 Methodology

In this section, we first detail the adversarial training on PLMs and then introduce our proposed plug-and-play defense module, PlugAT, which helps PLMs to get rid of optimizing all model parameters from scratch when performing adversarial training. Finally, we design a novel adversarial training method based on gradient alignment constraints and avoid performance degradation on clean examples.

3.1 Preliminaries

For a classification dataset $\mathcal{D} = \{\mathcal{X}, \mathcal{Y}\}$, where $\mathcal{X}$ is the set of input examples and $\mathcal{Y}$ is the set of label classes, we have a Transformer-based PLM with parameters θ, such as BERT to learn a mapping function $f : \mathcal{X} \rightarrow \mathcal{Y}$. Given an input instance $X \in \mathcal{X}$ that consists of several tokens $X = \{x_1, \ldots, x_{|X|}\} \in \mathcal{X}$, $Y \in \mathcal{Y}$ is the label, the PLM first converts it into the embeddings:

$$Z_0 = \mathrm{Embed}(X) \in \mathbb{R}^{|X| \times d}, \tag{1}$$

where d is hidden size and $\mathrm{Embed}(\cdot)$ denotes the input embedding layer, then the hidden states are encoded by l-th Transformer layer:

$$Z_l = \mathrm{Trans}_l(Z_{l-1}) \in \mathbb{R}^{|X| \times d}, \tag{2}$$

where $\mathrm{Trans}_l(\cdot)$ is l-th Transformer layer. Finally, the hidden states Z_L in last layer L is used to decode Y.

Adversarial Training. Adversarial training is a method for hardening classifiers against adversarial attacks and involves training the network on adversarially perturbed inputs. The adversarial training solves a min-max optimization problem as follows:

$$\min_{\theta} \mathbb{E}_{(X,Y)\sim\mathcal{D}} \max_{\|Z_0' - Z_0\|_F \leq \epsilon} \mathcal{L}(f(X', \theta), Y), \tag{3}$$

where X' is an adversarial example of X and ϵ is the maximum perturbation bound. The *inner maximization* problem is to find an adversarial example within the ϵ-ball centered at X in the embedding space ($\|Z_0' - Z_0\|_F \leq \epsilon$) that maximizes the classification loss. PGD applies the K-step stochastic gradient descent to search for the perturbation δ (Madry et al., 2018):

$$\delta_{k+1} = \prod_{\|\delta\| \leq \epsilon} \left(\delta_k + \eta \frac{g(\delta_k)}{\|g(\delta_k)\|} \right), \tag{4}$$

where $g(\delta_k) = \nabla_\delta \mathcal{L}(f(X + \delta_k, \theta), Y)$, δ_k is the perturbation in k-th step and $\prod_{\|\delta\| \leq \epsilon}(\cdot)$ projects the perturbation back onto the Frobenius normalization ball. Then robust training optimizes the network on adversarially perturbed input $X' = X + \delta_K$. And the objective of the *outer minimization* problem is to optimize the model parameters so that the loss on the adversarial examples is minimized.

In traditional adversarial training, all PLM parameters θ need to participate in the optimization starting from pre-training weights. It leads to the following limitations in practice: 1) it is not cost effective to retrain a large-scale deployed model for additional robustness; 2) it requires more training iterations (typically 3 times or more) compared to fine-tuning, so optimizing all model parameters is time-consuming and parametrically inefficient.

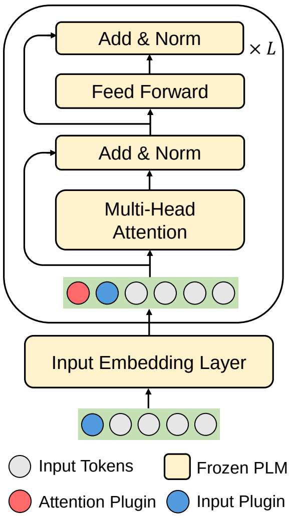

Figure 2: Architecture of the defense module and its integration with the Transformer-based model. **Input plugin** can be regarded as "virtual inputs", which is added before the original input tokens. **Attention plugin** is inserted into all multi-head self-attention layers. Then we conduct adversarial training on defense modules to improve robustness.

3.2 PlugAT

We propose a plug-and-play adversarial defense module as an efficient alternative for the deployed model without modifying its structure and parameters. Inspired by recent prompt-based approaches, we design the defense module $\mathcal{P}_\phi$ with parameters ϕ, which is a series of tunable vectors that are added before inputs of all Transformer layers to guide the model to make correct predictions in the face of adversarial examples. Thus, the inputs of the first Transformer layer (i.e., outputs of the input embedding layer) is:

$$Z_1^* = \mathrm{Embed}(P_0; Z_0) \in \mathbb{R}^{(N+|X|)\times d}, \quad (5)$$

where $P_0 \in \mathbb{R}^{N\times d}$ denotes the part of $\mathcal{P}_\phi$ in the input embedding layer, N is the length of P_0, and $\mathcal{P}_\phi = \{P_0, P_1, \ldots, P_{L-1}\}$. Then the modified hidden states Z_l^* output by l-th layer is:

$$Z_l^* = \mathrm{Trans}_l(P_{l-1}; Z_{l-1}^*[i > N]) \in \mathbb{R}^{(N+|X|)\times d}, \quad (6)$$

where $Z_{l-1}^*[i > N]$ is the hidden states at positions larger than N, which is the hidden states of original input tokens. However, previous work found these prefix parameters are difficult to optimize, which confirms similar observations in our experimental section. To tackle this problem, the structure of $\mathcal{P}_\phi$ consists of two parts: input plugin $\mathcal{I}_\phi$ and attention plugin $\mathcal{A}_\phi = \{A_0, \ldots, A_{L-1}\}$. They are parameters that affect the model behaviour globally and locally, respectively. The architecture of PlugAT is illustrated in Figure 2.

Input Plugin. Input plugin can be regarded as "virtual input tokens" that is added before original input embeddings, and its effects will propagate upward to all Transformer layers. Thus, the augmented input embeddings in (5) can be expanded as:

$$Z_0^* = \mathrm{Embed}(\mathcal{I}_\phi; X) \in \mathbb{R}^{(N_I+|X|)\times d}, \quad (7)$$

where $\mathcal{I}_\phi$ is the "virtual input tokens" of length N_I. We optimize the parameters of the $\mathcal{I}_\phi$ only in the embedding layer, while their hidden states in the upper layers are automatically encoded by PLMs. This means that $\mathcal{I}_\phi$ acts as a part of the original inputs, providing global semantic information to the model, which makes the optimization of $\mathcal{P}_\phi$ more stable.

Attention Plugin. Attention plugin is composed of feature vectors inserted into all multi-head self-attention layers. The hidden states of l-th Transformer layer is encoded as:

$$Z_l^* = \mathrm{Trans}_l(A_{l-1}; Z_{l-1}^*[i > N_A]) \in \mathbb{R}^{(N+|X|)\times d}, \quad (8)$$

where $N = N_I + N_A$ and N_A is the length of attention plugin. The parameter of $\mathcal{A}_\phi$ are independent at each layer, which avoids long-range dependencies and introduces more trainable parameters. The optimization of A_l is sensitive to the learning rate and initialization, thus A_l is reparameterized by a two-layer perceptron as shown below:

$$A_l = W_2(\tanh(W_1 A_l' + b_1) + b_2), \quad (9)$$

where $W_1 \in \mathbb{R}^{d'\times d}$, $W_2 \in \mathbb{R}^{N_A\times d'}$, $b_1 \in \mathbb{R}^{d'}$, $b_2 \in \mathbb{R}^{N_A\times d}$ and $A_l' \in \mathbb{R}^d$ are trainable parameters.

3.3 Forgetting of Clean Examples

Deep networks do not perform well in the sequential continual learning setting because they forget the past learned tasks after learning new ones. Therefore, it is necessary to overcome the problem of forgetting clean samples caused by using PlugAT in the deployed model. Our goal is to understand the effect of training adversarial examples $\{X_1', \ldots, X_{|\mathcal{D}|}'\}$ on their clean counterparts $\{X_1, \ldots, X_{|\mathcal{D}|}\}$. We follow a similar analysis procedure used in the influence function to implement it (Koh and Liang, 2017). First, the parameter changes induced from the training of adversarial example can be calculated by reweighting (X', Y) with some λ: $\hat{\phi}_{\lambda,X'} \triangleq \arg\min_{\phi} \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}\mathcal{L}(X'_i,\phi) + \lambda\mathcal{L}(X',\phi)$ where $\mathcal{L}(X'_i,\phi)$ is a shorthand of $\mathcal{L}(f(X'_i,\phi),Y_i)$. Atkinson et al. (1983) show that the influence of reweighting X' on the parameters $\hat{\phi}$ is

$$\frac{d\hat{\phi}_{\lambda,X'}}{d\lambda}\Big|_{\lambda=0} = -H_{\hat{\phi}}^{-1}\nabla_{\phi}\mathcal{L}(X',\phi), \tag{10}$$

where $H_{\hat{\phi}} \triangleq \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}\nabla^2_{\phi}\mathcal{L}(X'_i,\hat{\phi})$ is the positive definite Hessian matrix. Second, we apply the chain rule to measure the influence of reweighting X' on the loss of X:

$$\begin{aligned}\mathcal{F}(X',X) &\triangleq \frac{d\mathcal{L}(X'_i,\hat{\phi}_{\lambda,X'})}{d\lambda}\Big|_{\lambda=0}\\ &= \nabla_{\phi}\mathcal{L}(X,\hat{\phi})^T\frac{d\hat{\phi}_{\lambda,X'}}{d\lambda}\Big|_{\lambda=0}\\ &= -\nabla_{\phi}\mathcal{L}(X,\hat{\phi})^T H_{\hat{\phi}}^{-1}\nabla_{\phi}\mathcal{L}(X',\hat{\phi}).\end{aligned} \tag{11}$$

$\mathcal{F}(X',X) > 0$ means the training process of the adversarial example X' will impair the performance of the clean one X. Pezeshkpour *et al.* (2021) show that Hessian information is unnecessary and $\mathcal{F}(X',X)$ can be approximated as $\mathcal{F}(X',X) \approx -\nabla_{\phi}\mathcal{L}(X,\hat{\phi})^T\mathcal{L}(X',\hat{\phi})$. For computational reasons, we use the subset of parameters corresponding to the last linear layer to compute $\mathcal{F}(X',X)$.

By combining the above constraints with adversarial training in (3), we propose the forgetting restricted adversarial training (FRAT) for our defense module:

$$\begin{aligned}&\min_{\phi}\mathbb{E}_{(X,Y)\sim\mathcal{D}}\max_{\|Z'_0-Z_0\|_F\le\epsilon}\mathcal{L}(f(X',\phi),Y),\\ &\text{s.t. } \mathcal{F}(X',X)\le 0.\end{aligned} \tag{12}$$

We use only the parameters of proposed defense module to optimize the above objectives, while keeping the original model frozen.

4 Experimental Setup

We first introduce the experimental setup, including datasets, baselines and adversarial attackers involved in our experiments.

4.1 Datasets

We compare the proposed method with baselines on two widely applied benchmark corpora: IMDB dataset (Maas et al., 2011) and SST-2 dataset (Socher et al., 2013). Collected from online movie reviews, IMDB contains long samples created by users, while SST samples are shorter (with an average length of 19 words) and show a higher diversity.

Method: PlugAT

Input: Training dataset $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^{M}$, perturbation bound ϵ, initialization of perturbation σ, ascent step size η, ascent steps K, a deployed model f_θ, learning rate τ, defense module $\mathcal{P}_\phi$

Output: defense module parameters ϕ

```
Initialize D_φ
for epoch= 1, ..., N_ep do
  for minibatch B ⊂ P do
    δ_0 ← (1/N_δ) Uniform(−σ, σ)
    for k = 0, ..., K − 1 do
      g_k ← ∇_δ L(f_θ(X + δ, φ), Y)
      δ_k ← ∏_{||δ||_F ≤ ε}(δ_k + η · g_k/||g_k||_F)
      if F(X + δ_k, X) > 0 do
        δ_k ← δ_{k−1}
        break
      end if
    end for
    φ ← φ − τ∇_φ L(f_θ(X + δ_k, φ), Y)
  end for
end for
return: φ
```

4.2 Baseline Methods

We select four adversarial defence methods as baselines for a thorough comparison, including three methods of adversarial training, one from information-theoretic perspective, and one of adversarial data augmentation.

PGD (Madry et al., 2018): A gradient-based adversarial training method that uses multiple projected gradient ascent steps to find the adversarial perturbations, which is then leveraged to update the model parameters.

FreeLB (Zhu et al., 2020): A similar gradient-based method with PGD while accumulating the gradients and then back propagate once, resulting in a faster process of searching for more effective perturbations.

TAVAT (Li and Qiu, 2020): An token-aware adversarial training method that crafts fine-grained perturbations by constraining them into a token-level normalization ball.

InfoBERT (Wang et al., 2021): A learning framework for robust fine-tuning from an information-theoretic perspective, where two mutual-information-based regularizers are used for model training.

4.3 Adversarial Attack Methods

Three widely accepted attack methods are used to verify the ability of our proposed method against baselines.

TextBugger (Li et al., 2018): An adversarial attack method that is applicable in both white-box and black-box scenarios. Important words or sentences (in white- or black-box scenario respectively) are first discovered, and then an optimal perturbation is chosen from the generated perturbations, or, for the black-box situation, a scoring function is leveraged to find important words to manipulate.

BERT-Attack (Li et al., 2020): A method using BERT to generate adversarial text, and thus the generated adversarial examples are fluent and semantically preserved.

TextFooler (Jin et al., 2020): A comprehensive attack paradigm that creates adversarial examples that maintain human prediction consistency, semantic similarity, and language fluency. The important words for the target model are first identified and are then replaced with words, which are semantically similar and grammatically correct, until the prediction is altered.

4.4 Evaluation Metrics

The evaluation metrics adopted in our experimental analyses are listed as follows. For a robust defense method, higher accuracy under attack as well as query times is expected.

Clean Accuracy (Clean%): The accuracy on the clean test dataset. **Accuracy under Attack (Aua%)**: The model's prediction accuracy facing specific adversarial attacks. **Number of Queries (#Query)** The average number of times the attacker queries the model, which means the more average query number is, the harder the defense model is to be compromised. **Trainable Params**: The number of parameters to be optimized. **Speed UP**: The training speedups re reported against PGD and we evaluate all the adversarial defense methods on an NVIDIA RTX3090 GPU.

4.5 Implementation Details

Our implementation of PlugAT is mainly based on BERT and FreeLB, so most of the hyperparameter settings are based on these methods.[2] We use AdamW as our optimizers with the learning rate $1e^{-5}$, a batch size $\in \{16, 32\}$ and a linear learning rate decay schedule with a warm-up of 0.1. The dropout rate is set to 0.1 for all task-specific layers. For the defense module, $N_A = 35$, $N_i = 5$ and $d' = 512$. We set the adversarial perturbation step as 2, the perturbation step size as 0.03, the constrain bound of perturbations as 0.1, the initial magnitudes of perturbations as 0.05. The maximum number of epochs is set to 10. Since the results of the SST-2 test set need to be submitted to the GLUE evaluation server and cannot be used for adversarial attacks, the results of SST-2 are tested on the development set.

We implement three adversarial attack methods using TextAttack framework and follow the default parameter settings.[3] We adopt a "free" strategy similar to FreeLB (Zhu et al., 2020) to solve the inner maximization problem in (12) and obtain more adversarial examples. The clean accuracy ((**Clean**%)) is tested on the whole test/dev set. Other adversarial robustness evaluation metrics (e.g., **Aua**% and **#Query**) are evaluated on the dev set for SST-2 and 1000 randomly selected test samples for IMDB. We choose models trained during the last period to compare the robustness. All experiments are conducted using NVIDIA RTX3090 GPUs.

4.6 Main Results

Defense Effectiveness. As illustrated in Table 1, BERT equipped with our module with forgetting restricted adversarial training has seen an increase in robustness against attacks while maintaining clean accuracy. The adversarial training based approaches, e.g., FreeLB, achieve higher clean accuracy than the baseline and our proposed method, and their improvement in robustness is also very insignificant. Generally, PlugAT lifts BERT's accuracy under attack from 4.9% to 20.8% on SST-2 and 12.2% to 36.1% on IMDB when attacked by TextFooler, which outperforms all of the compared methods. The improvements indicate that our proposed defense module indeed helps BERT achieve greater robustness against attacks

[2] https://github.com/huggingface/transformers

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

Dataset	Methods	Trainable Params	Speed Up	Clean%	TextFooler		BERT-ATTACK		TextBugger	
					Aua%	#Query	Aua%	#Query	Aua%	#Query
SST-2	Fine-tune	109M	–	92.5	4.9	98.5	2.9	114.2	26.3	48.6
	PGD	109M	1×	92.9	16.5	122.3	11.8	156.8	43.4	56.1
	FreeLB	109M	2.1×	**93.1**	14.4	123.8	10.2	154.6	42.4	54.9
	TAVAT	109M	2.8×	93.1	13.5	117.8	9.9	147.9	38.5	49.7
	InfoBERT	109M	1.3×	92.1	18.3	121.4	14.4	162.3	40.3	51.2
	PlugAT	**9.9M**	**4.1×**	92.6	**20.8**	**126.5**	**15.6**	**162.4**	**41.8**	**56.3**
IMDB	Fine-tune	109M	–	94.2	12.2	1209.8	7.8	1572.2	25.8	783.2
	PGD	109M	1×	94.6	34.5	1616.0	28.6	2454.7	43.5	957.3
	FreeLB	109M	3.4×	94.7	27.2	1479.1	22.6	1954.7	36.0	907.3
	TAVAT	109M	4.1×	94.3	21.2	1417.7	17.6	1825.4	32.9	842.6
	InfoBERT	109M	1.2×	**95.2**	32.4	1572.2	26.0	2326.0	43.6	969.8
	PlugAT	**9.9M**	**6.1×**	93.9	**36.1**	**1624.9**	**32.8**	**2777.1**	**44.7**	**1045.3**

Table 1: Main results of the adversarial robustness evaluation on two text classification datasets. The defense module **PlugAT** achieves better robustness with fewer trainable parameters and less GPU time. The best performance is marked in bold.

compared with a vanilla BERT with the same kind of adversarial training while maintaining the clean accuracy to a large extent. Moreover, our proposed method helps BERT to undergo adversarial training by learning only 9.1% parameters, and the computational cost is also outstanding among the baseline methods.

Dataset		Methods	Clean%	TextFooler	
Source	Target			Aua%	#Query
SST-2	**IMDB**	Fine-tune	90.8	2.0	564.1
		FreeLB	89.2	22.5	874.2
		PGD	89.9	20.0	827.8
		PlugAT	**93.8**	**34.0**	**1536.0**
IMDB	**SST-2**	Fine-tune	86.7	3.2	87.7
		FreeLB	87.8	4.5	93.5
		PGD	88.5	5.4	100.8
		PlugAT	**91.4**	**10.3**	**113.1**

Table 2: Experimental results when the models are trained on the **source** dataset and then transferred to the **target** dataset for testing. BERT (fine-tuned on the **target** dataset) equipped with our module (trained on the **source** dataset) improves robustness to a large extent and with minor damage to the clean accuracy.

Transferability. The proposed method is competitive even in challenging scenarios, with the extra robustness obtained by our defense module being able to generalize across datasets. To get a clear view of this phenomenon, we conduct transferability experiments on IMDB and SST-2 datasets, which is reported in Table 2. The results indicate that the robustness of BERT with PlugAT trained on one dataset can transfer to the other to a degree. To be more specific, BERT with the defense module trained on SST-2 dataset transfers its ability of defending to IMDB and vice versa, while still outperforming the BERT trained on each dataset in all of the three metrics. This suggests that the defense module shows strong potential in transferability, the resource of which is left for a more thorough investigation of future work. We believe that the reason for the above phenomenon may be that these two datasets are similar. Another possible explanation is that due to the transferability of adversarial examples, the adversarial examples generated by adversarial training may be similar on SST-2 and IMDB. These adversarial examples may make the model more robust on both SST-2 and IMDB.

4.7 Intrinsic Evaluation

Several variants that affect the performance of our method are compared, including the length of input or attention plugins. Besides, we conduct ablation experiments to study the contribution of each component in the proposed method.

Effect of Attention Plugin. A longer attention plugin introduces more trainable parameters, enabling stronger expressive power. The lengths of 5 to 45 with an interval of 5 are experimented. As reported in Figure 3, **Aua%** increases as the length increases up to a threshold (about 35) and then a slight performance drop occurs.

Effect of Input Plugin. We find that the input

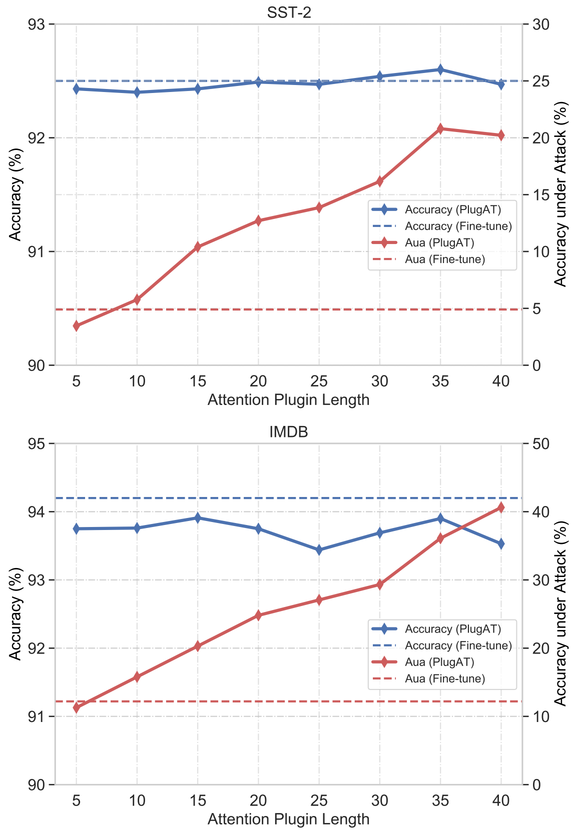

Figure 3: Length of attention plugins vs. performance on SST-2 and IMDB datasets. Performance increases up to a threshold 35 and then a slight performance drop occurs.

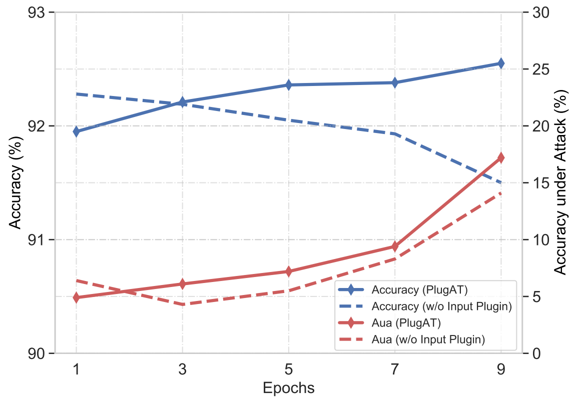

Figure 4: Full vs. without input plugin. The input plugin enables the training process to be more stable and reach a better performance.

Methods	Clean%	TextFooler	
		Aua%	#Query
PlugAT	92.6	20.8	126.5
w/o FRAT	91.5	21.5	128.2
w/o Input Plugin	92.1	16.7	114.6
w/o Attention Plugin	91.9	6.3	101.2

Table 3: Ablation study of PlugAT. We can observe that forgetting restricted adversarial training (FRAT) can effectively mitigate forgetting on clean data.

plugin makes the training process more stable and reduces the performance fluctuations. Figure 4 shows that generally, the existence of input plugin enables the deployed model to initialize better with a starting point at a higher accuracy both on clean data and under attack.

Ablation Study. The results of ablation experiments are demonstrated in Table 3. The results show that the model's clean accuracy decreases when forgetting restricted adversarial training is removed, indicating that forgetting restricted adversarial training can effectively mitigate forgetting on clean data. As we showed before, the global information provided by the input plugin makes the training more stable and slightly improves the performance. The attention plugin, on the other hand, provides more local information, making the model more expressive.

5 Conclusion

In this paper, we focus on improving adversarial training in the NLP field. We propose PlugAT, a plug-and-play module, as an effective solution in terms of lifting model's robustness while preserving its clean accuracy as well as cutting down on the overall computational cost. It is discovered that despite learning $10\times$ fewer parameters than conventional adversarial training, the proposed defense module also shows the potential in transferability. Empirical evaluations on two widely used benchmark datasets demonstrate that training models with the defense module costs half the GPU time while outperforming the existing adversarial training methods. Extensive experiments demonstrate that our proposed module can significantly improve the model robustness and transfer the performance across different datasets effectively. Besides, we hope our work could inspire more research on improving adversarial training in NLP.

Acknowledgements

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by National Natural Science Foundation of China (No. 62076069, 61976056), Shanghai

Municipal Science and Technology Major Project (No.2021SHZDZX0103). This research was supported by Meituan and Ant Group through Ant Research Intern Program.

References

Maksym Andriushchenko and Nicolas Flammarion. 2020. Understanding and improving fast adversarial training. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Anthony C. Atkinson, R. Dennis Cook, and Sanford Weisberg. 1983. Residuals and influence in regression. *Biometrics*, 39:818.

Emily Dinan, Samuel Humeau, Bharath Chintagunta, and Jason Weston. 2019. Build it break it fix it for dialogue safety: Robustness from adversarial human attack. 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 4537–4546, Hong Kong, China. Association for Computational Linguistics.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Wenjuan Han, Bo Pang, and Ying Nian Wu. 2021. Robust transfer learning with pretrained language models through adapters. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 854–861, Online. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. 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 *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8018–8025. AAAI Press.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1885–1894. PMLR.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. Textbugger: Generating adversarial text against real-world applications. *ArXiv preprint*, 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. Tavat: Token-aware virtual adversarial training for language understanding. *ArXiv preprint*, abs/2004.14543.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. 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 4582–4597, Online. Association for Computational Linguistics.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2018. Deep text classification can be fooled. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4208–4215. ijcai.org.

Qin Liu, Rui Zheng, Bao Rong, Jingyi Liu, ZhiHua Liu, Zhanzhan Cheng, Liang Qiao, Tao Gui, Qi Zhang, and Xuanjing Huang. 2022. Flooding-X: Improving BERT's resistance to adversarial attacks via loss-restricted fine-tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5634–5644, Dublin, Ireland. Association for Computational Linguistics.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Linyang Li, Qi Zhang, and Xuanjing Huang. 2022. Template-free prompt tuning for few-shot NER. In *Proceedings of the 2022 Conference of the North American*

Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 5721–5732, Seattle, United States. Association for Computational Linguistics.

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 *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Pouya Pezeshkpour, Sarthak Jain, Byron Wallace, and Sameer Singh. 2021. An empirical comparison of instance attribution methods for NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 967–975, Online. 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.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John P. Dickerson, Christoph Studer, Larry S. Davis, Gavin Taylor, and Tom Goldstein. 2019. Adversarial training for free! In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3353–3364.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew 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*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. 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 2153–2162, Hong Kong, China. Association for Computational Linguistics.

Boxin Wang, Shuohang Wang, Yu Cheng, Zhe Gan, Ruoxi Jia, Bo Li, and Jingjing Liu. 2021. Infobert: Improving robustness of language models from an information theoretic perspective. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Eric Wong, Leslie Rice, and J. Zico Kolter. 2020. Fast is better than free: Revisiting adversarial training. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Jiacheng Ye, Tao Gui, Yichao Luo, Yige Xu, and Qi Zhang. 2021. One2Set: Generating diverse keyphrases as a set. 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 4598–4608, Online. Association for Computational Linguistics.

Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. 2019. You only propagate once: Accelerating adversarial training via maximal principle. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 227–238.

Jeffrey O Zhang, Alexander Sax, Amir Zamir, Leonidas Guibas, and Jitendra Malik. 2020. Side-tuning: A baseline for network adaptation via additive side networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 698–714. Springer.

Rui Zheng, Bao Rong, Yuhao Zhou, Di Liang, Sirui Wang, Wei Wu, Tao Gui, Qi Zhang, and Xuanjing Huang. 2022. Robust lottery tickets for pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2211–2224, Dublin, Ireland. 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 *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.