Better Robustness by More Coverage: Adversarial Training with Mixup Augmentation for Robust Fine-tuning

Chenglei Si[1*], **Zhengyan Zhang**[2,3,4*], **Fanchao Qi**[2,3,4], **Zhiyuan Liu**[2,3,4†], **Yasheng Wang**[5], **Qun Liu**[5], **Maosong Sun**[2,3,4]
[1]University of Maryland, College Park, MD, USA
[2]Department of Computer Science and Technology, Tsinghua University, Beijing, China
[3]Institute for Artificial Intelligence, Tsinghua University, Beijing, China
[4]State Key Lab on Intelligent Technology and Systems, Tsinghua University, Beijing, China
[5]Huawei Noah's Ark Lab
clsi@terpmail.umd.edu, zy-z19@mails.tsinghua.edu.cn

Abstract

Pre-trained language models (PLMs) fail miserably on adversarial attacks. To improve the robustness, adversarial data augmentation (ADA) has been widely adopted, which attempts to cover more search space of adversarial attacks by adding the adversarial examples during training. However, the number of adversarial examples added by ADA is extremely insufficient due to the enormously large search space. In this work, we propose a simple and effective method to cover much larger proportion of the attack search space, called Adversarial Data Augmentation with Mixup (MixADA). Specifically, MixADA linearly interpolates the representations of pairs of training examples to form new virtual samples, which are more abundant and diverse than the discrete adversarial examples used in conventional ADA. Moreover, to evaluate the robustness of different models fairly, we adopt a challenging setup, which dynamically generates new adversarial examples for each model. In the text classification experiments of BERT and RoBERTa, MixADA achieves significant robustness gains under two strong adversarial attacks and alleviates the performance degradation of ADA on the original data. Our source codes will be released to support further explorations.

1 Introduction

Pre-trained language models (PLMs) such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2020) have established state-of-the-art results on various NLP tasks and the pre-train-then-fine-tuning paradigm has become the status quo. However, recent works have shown the flaws of these PLMs in terms of robustness (Jin et al., 2020; Zang et al., 2020; Si et al., 2020; Li et al., 2020; Garg and Ramakrishnan, 2020; Tan et al., 2020; Wang et al., 2020a), where PLMs fine-tuned on various downstream datasets are easily fooled by different types of adversarial attacks, including different levels of access to model parameters (black-box and white-box attacks) and different granularity of perturbation level (sentence-level, word-level and character-level attacks).

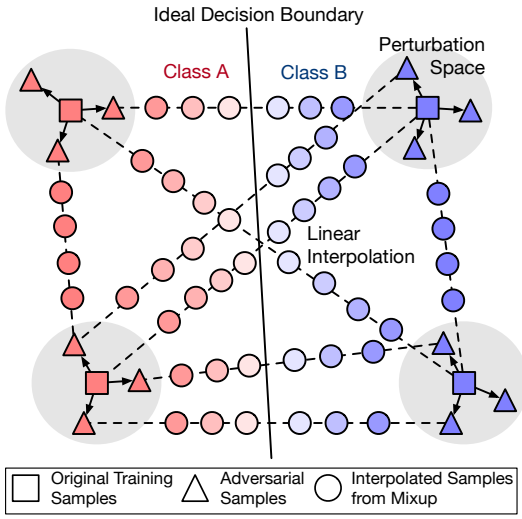

Figure 1: Illustration of MixADA. Part of the interpolated samples are plotted. We interpolate the representations of each pair of training samples including original samples and adversarial samples. Blue and red represent two different classes. The solid line represents the resultant decision boundary. MixADA helps achieve a more robust decision boundary.

Previous works explore to improve PLMs' robustness from both the pre-training (Zhu et al., 2020) and fine-tuning process (Zang et al., 2020; Jin et al., 2020). Compared to the pre-training process, enhancing the fine-tuning process is less time-consuming and can better adapt to downstream tasks. In this work, we focus on developing a general fine-tuning framework based on adversarial data augmentation (ADA) to improve model robustness against any type of adversarial attacks.

* indicates equal contribution
† Corresponding author e-mail: liuzy@tsinghua.edu.cn

ADA is widely used in the training process to enhance the model robustness (Alzantot et al., 2018; Ren et al., 2019; Zhang et al., 2019; Jin et al., 2020; Li et al., 2020; Tan et al., 2020; Yin et al., 2020; Zheng et al., 2020; Zou et al., 2020; Wang et al., 2020c). Since the training data is limited and the searching space of adversarial algorithms is very large, it is intuitive to augment adversarial examples. For example, the search space of word-substitution attacks consists of all combinations of the synonym replacement candidates, which is exponentially large (Zang et al., 2020).

Although ADA has achieved promising results on improving the model robustness, there still are two challenges when utilizing ADA. Firstly, the number of adversarial examples is still insufficient considering the large attack space and the generation of adversarial examples is time-consuming. It is important to further increase the space coverage. Secondly, ADA usually causes performance degradation (Ren et al., 2019) because the distribution of adversarial examples is different from that of the original dataset. Hence, users need to sacrifice model performance for better robustness when using ADA.

To solve these two challenges, we propose to incorporate mixup (Zhang et al., 2018) to adversarial data augmentation (MixADA), where we linearly interpolate the representations and labels of pairs of training examples to create different virtual training samples during fine-tuning as shown in Figure 1. The generation of virtual samples is much easier and can effectively increase the coverage of the attack space. Meanwhile, compared to the adversarial examples, the virtual training samples are closer to the distribution of the original data, which alleviates the performance degradation brought by ADA.

To better evaluate the model robustness, we survey the literature on adversarial attack and defense in NLP and observe that two different evaluation schemes (*static evaluation* and *dynamic evaluation*) have been adopted while these two schemes are fundamentally different and could lead to conflicting conclusions. We explicitly differentiate these two evaluation schemes and adopt both of them in the experiments to show that *static evaluation* is flawed, thereby encouraging future works to adopt *dynamic evaluation*, which is the fairer evaluation scheme.

To summarise, our contributions in this work are two-fold:

- We compare the two existing schemes of robustness evaluation, which we refer to as *static evaluation* and *dynamic evaluation*. We perform experiments to show that *static evaluation* is not a reliable measure of model robustness.
- We propose to MixADA to improve model robustness. We find that MixADA achieves significant improvement in model robustness on PLMs without significant performance degradation on the clean datasets.

2 Related Work

Defense Methods in NLP. To handle various adversarial attacks, many defense methods have been proposed, which can be divided into two types, attack-oriented methods and general methods.

Attack-oriented methods are targeted at specific types of attacks mainly including character-level attacks and word-substitution attacks. For character-level attacks, specifically-designed model modules (Pruthi et al., 2019; Jones et al., 2020) and pre-training tasks (Ma et al., 2020) have been proposed and have been shown to work well in handling adversarial typos. For word-substitution attacks, several other works (Huang et al., 2019; Jia et al., 2019; Zhou et al., 2020; Mozes et al., 2020) have proposed new defense methods with varying levels of improvement. However, they are not generally applicable to other types of attacks, largely limiting their usefulness.

On the other hand, adversarial training (AT) (Goodfellow et al., 2015) is a prevalent general defense method, which has been adapted in two different forms. The first type of AT directly adds gradient-based perturbations to the word embeddings during training (Miyato et al., 2017; Yasunaga et al., 2018; Zhu et al., 2020; Li and Qiu, 2021). Instead of adding continuous gradient perturbations, another type of AT directly adds discrete text adversarial examples into training (Jia and Liang, 2017; Wang and Bansal, 2018; Michel et al., 2019). In this paper, we refer to such discrete AT methods as adversarial data augmentation to avoid confusion with the gradient-based AT methods. It has been shown that gradient-based AT methods (Li and Qiu, 2021) are not very effective in defending against adversarial attacks because unlike attacks in computer vision, textual adversar-

ial attacks are discrete and do not directly apply gradient-based perturbations, thus leaving a discrepancy between gradient-based AT and the actual adversarial attacks. Although ADA achieves some promising results in NLP, it is not as good as those attack-oriented methods. Considering the generality, we focus on improving ADA here.

Mixup for NLP. Mixup (Zhang et al., 2018) is a form of data augmentation originated in the computer vision domain that encourages the model to behave linearly in-between training examples via linear interpolation. Such linear behaviour reduces the amount of undesirable oscillations when predicting outside the training examples, leading to better generalization. Recently, several works have attempted to apply mixup on language representations to improve performance on downstream NLP tasks (Sun et al., 2020; Chen et al., 2020; Cheng et al., 2020), on knowledge distillation (Liang et al., 2020) and on compositional generalization benchmarks (Guo et al., 2020). Our proposed MixADA tackles the important aspect of model robustness, which is an serious weakness of current PLM models.

3 MixADA

In this section, we introduce the details of ADA and our proposed MixADA.

3.1 Adversarial Data Augmentation

Given a victim model f_v and the original training instances $\mathbf{X} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{n}$, we employ an attacker to construct label-preserving adversarial training instances $\mathbf{X}_{adv} = \{(\mathbf{x}'_i, \mathbf{y}_i)\}_{i=1}^{n}$ such that: instances originally correctly classified ($\mathbf{y}_i = f_v(\mathbf{x}_i)$) are now classified wrongly ($\mathbf{y}' = f_v(\mathbf{x}'), \mathbf{y}' \neq \mathbf{y}$). We re-train the model on the augmented training data $\mathbf{X}_{ADA} = \mathbf{X} \cup \mathbf{X}_{adv}$. Note that $\mathbf{X}_{adv}$ is generated at one shot and will not be changed during training.

3.2 Mixed Adversarial Data Augmentation

The conventional ADA approach is very limited in the sense that it only incorporates a small set of discrete adversarial examples into training. In order to better robustify the model against the enormous number of possible attacks, we should cover more of the attack space during training. Driven by these motivations, we propose to adapt mixup into ADA for more robust fine-tuning of PLMs.

Mixup linearly interpolates the representations and labels of pairs of training examples to create different virtual training samples, which is formulated by

$$\begin{aligned} \hat{\mathbf{x}} &= \lambda \mathbf{x}_i + (1-\lambda)\mathbf{x}_j, \\ \hat{\mathbf{y}} &= \lambda \mathbf{y}_i + (1-\lambda)\mathbf{y}_j, \end{aligned} \tag{1}$$

where $(\mathbf{x}_i, \mathbf{y}_i)$ and $(\mathbf{x}_j, \mathbf{y}_j)$ are two labeled examples, and $\lambda \in [0, 1]$ usually comes from a beta distribution: $\lambda \sim Beta(\alpha, \alpha)$, where α is a hyperparameter. In the text domain, we cannot directly mix the discrete word tokens. Instead, we can either mix the word embedding vectors or the hidden representations of the model (e.g., the Transformer encoder). Meanwhile, we directly mix the labels, which are usually represented as one-hot vectors.

MixADA incorporates mixup into ADA to construct virtual examples to increase coverage of the attack space. Specifically, we take the two following steps:

(1) Given a victim model f_v, we use a white-box attacker to construct adversarial training examples $\mathbf{X}_{adv}$ on the full or subset of the original training data $\mathbf{X}_{ori}$. And, we combine the two to get the augmented training data: $\mathbf{X} = \mathbf{X}_{ori} \cup \mathbf{X}_{adv}$.

(2) We train the new model f on $\mathbf{X}$. We minimize the sum of the standard training loss and the mixup loss

$$L = \sum_{i=1}^{n} L_{CE}(f(\mathbf{x}_i), \mathbf{y}_i) + \sum_{i=1}^{m} L_{KL}(f(\hat{\mathbf{x}}_m), \hat{\mathbf{y}}_m), \tag{2}$$

where $(\mathbf{x}_i, \mathbf{y}_i)$ is the actual example from $\mathbf{X}$ and $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ is the virtual example obtained by applying mixup on the random pair of training data sampled from $\mathbf{X}$. We use cross-entropy to compute loss on the original examples $(\mathbf{x}_i, \mathbf{y}_i)$ and use KL-divergence as loss on the virtual examples $(\hat{\mathbf{x}}_i, \hat{\mathbf{y}}_i)$.

Notably, our approach allows the mixing of different types of data: mixup between original examples, mixup between original example with adversarial example, and mixup between adversarial examples. We do not constrain the mixup to be between pairs of original examples and adversarial examples only because the possible interpolation space between the original example and its corresponding adversarial examples is smaller than the space between two unrelated examples.

4 Evaluation

In this section, we first introduce two different evaluation setups adopted for the model robustness eval-

uation. Then we introduce the automatic attackers used for evaluation in this work.

4.1 Robustness Evaluation

By surveying existing literature on adversarial attack and defense, we find that there are two different ways of robustness evaluation under adversarial attack. In this work, we explicitly differentiate them as *static evaluation* and *dynamic evaluation*:

- **Static evaluation**: In this setting, there are usually two types of models, the original victim model and the robust models with ADA because ADA require a victim model to generate adversarial examples. Then, the original victim model is also used to generate the adversarial test set based on the test set. The adversarial test set will be fixed for the evaluation of all the other new models. This is the evaluation method adopted in (Tan et al., 2020; Yin et al., 2020; Zou et al., 2020; Ren et al., 2019; Wang et al., 2020b; Zheng et al., 2020; Wang et al., 2020c, *inter alia.*).

- **Dynamic evaluation**: The adversarial algorithms attack every new model on the test set to form their corresponding adversarial test sets. In other words, attackers re-generate a new adversarial test set to target every model being evaluated. This is the evaluation method adopted in (Zhang et al., 2019; Huang et al., 2019; Jia et al., 2019; Li et al., 2020; Zang et al., 2020; Li and Qiu, 2021, *inter alia.*)

We observe that many of the above authors did not explicitly specify the mode of evaluation in their descriptions[1], leading to confusion and even misleading or conflicting conclusions. Thus, we explicitly differentiate the two modes of evaluation and provide a comparison in our experiments.

4.2 Evaluation Benchmarks

We use PWWS (Ren et al., 2019) and TextFooler (Jin et al., 2020) as our attack methods, which have been shown to effectively attack state-of-the-art NLP models including PLMs like BERT. Both attackers have access to model predictions but not gradients, and iteratively search for word synonym substitutes that flip model predictions without drastically changing the original semantic meanings or the original gold labels. Besides word-substitution based method, there are some other attack algorithms such as character-level or sentence-level attacks, while these kinds of attacks currently cannot cause serious robustness problems on PLMs.

[1]We have to email some of the authors to clarify the evaluation method being adopted.

5 Experiments

We evaluate our methods on two sentiment analysis datasets: SST-2 (Socher et al., 2013) and IMDB (Maas et al., 2011), where both datasets are binary classification tasks. For SST-2, we attack the entire test set (1821 samples) for robustness evaluation and report the robustness accuracy under attacks. For IMDB, we find that it is prohibitively slow to attack the whole test set (25k samples) and hence we use part of the test set as released in Gardner et al. for faster evaluation, which consists of 488 test instances. This also allows easy reproduction and fair comparison for future works. We experiment with both BERT-base (Devlin et al., 2019) and RoBERTa-base (Liu et al., 2019) as the victim model.

When performing mixup, we follow (Chen et al., 2020) to mix hidden representations of the upper layers of the Transformer encoder. In other words, the vectors used for mixup are hidden representations of the input examples at layer i of the Transformer encoder, where i is randomly sampled from $\{7, 9, 12\}$, which was found to be empirically effective.

Furthermore, we explore two different ways of obtaining the hidden representations of input examples from PLMs like BERT: (1) We use the vector of the [CLS] token at the ith-layer of BERT as the hidden representation for mixing. We name this approach **SMix**. (2) We perform mixup on every token's vector representation at the ith-layer. We name this approach **TMix**, which is the approach taken by Chen et al..

5.1 Static Evaluation Can Be Misleading

As a probing to compare static and dynamic adversarial evaluation, we attack the fine-tuned victim model (BERT_v, RoBERTa_v) and use the generated adversarial test set as the hold-out test set for *static evaluation*. We then change the random seeds and re-fine-tune the same models on the same data (BERT_{r1}, BERT_{r2}, RoBERTa_{r1}, RoBERTa_{r2}). We evaluate all these models using both dynamic and static evaluation, the results are shown in Table 1.

We find that by simply changing the random

	SST-2					IMDB				
	Original	PWWS-d	PWWS-s	TF-d	TF-s	Original	PWWS-d	PWWS-s	TF-d	TF-s
BERT_v	92.04	19.17	19.17	5.66	5.66	97.34	23.36	23.36	3.48	3.48
BERT_{r1}	91.10	18.73	44.98	3.46	45.36	96.72	25.61	69.88	1.64	76.64
BERT_{r2}	90.94	20.26	45.63	2.97	45.80	97.13	30.12	65.78	2.46	76.23
RoBERTa_v	94.45	25.48	25.48	3.29	3.29	97.75	15.98	15.98	1.84	1.84
RoBERTa_{r1}	94.29	31.03	50.47	5.82	41.63	97.54	27.46	65.57	3.48	77.46
RoBERTa_{r2}	93.85	32.13	50.69	9.34	40.91	97.34	17.21	73.36	2.87	76.64

Table 1: Comparison between dynamic and static evaluation. PWWS-d, PWWS-s, TF-d, TF-s represent PWWS dynamic, PWWS static, TextFooler dynamic, TextFooler static, respectively. Numbers in the table represent accuracy. BERT_v and RoBERTa_v are the victim model for generating static evaluation examples. BERT_{r1}, BERT_{r2}, RoBERTa_{r1}, and RoBERTa_{r2} are the fine-tuned models with new random seeds.

	SST-2				IMDB			
	PWWS		TextFooler		PWWS		TextFooler	
	Original	Adversarial	Original	Adversarial	Original	Adversarial	Original	Adversarial
BERT	91.27	14.83 *(20.88%)*	91.27	2.97 *(16.21%)*	97.75	24.18 *(24.10%)*	97.75	1.64 *(10.18%)*
+ADA	90.12	27.18 *(24.46%)*	90.50	9.01 *(18.32%)*	96.93	25.82 *(34.53%)*	96.93	3.07 *(11.81%)*
+TMix	91.82	21.20 *(19.36%)*	91.82	3.51 *(16.39%)*	97.13	43.24 *(32.51%)*	97.13	0.00 *(12.06%)*
+TMixADA	91.54	**38.82** *(23.73%)*	91.93	13.23 *(19.66%)*	97.34	51.02 *(36.76%)*	96.72	4.51 *(17.23%)*
+SMix	91.82	22.52 *(20.47%)*	91.82	4.61 *(16.76%)*	97.13	31.97 *(23.74%)*	97.13	2.66 *(12.39%)*
+SMixADA	91.10	31.52 *(24.11%)*	92.15	**17.35** *(18.64%)*	96.72	**60.86** *(27.79%)*	96.72	**17.42** *(13.85%)*
RoBERTa	94.62	28.39 *(23.06%)*	94.62	5.44 *(18.51%)*	97.54	28.07 *(37.48%)*	97.54	6.35 *(12.61%)*
+ADA	94.07	25.26 *(27.07%)*	92.75	9.67 *(19.71%)*	97.54	24.80 *(49.36%)*	96.93	12.50 *(14.39%)*
+TMix	94.18	30.04 *(23.19%)*	94.18	11.04 *(17.69%)*	97.54	44.06 *(39.33%)*	97.54	21.11 *(14.01%)*
+TMixADA	93.90	36.74 *(26.02%)*	93.03	13.78 *(20.15%)*	98.57	50.41 *(59.68%)*	97.13	**51.84** *(16.62%)*
+SMix	93.96	31.52 *(22.86%)*	93.96	8.29 *(17.80%)*	97.34	41.39 *(34.90%)*	97.34	22.34 *(11.96%)*
+SMixADA	93.96	**41.85** *(27.17%)*	93.47	**16.80** *(21.88%)*	97.54	**55.12** *(45.30%)*	97.54	49.18 *(15.52%)*

Table 2: Accuracy of the various models under PWWS and TextFooler attacks. Best performance for BERT-based models and RoBERTa-based models under each attack is **boldfaced**, the second best performance is underlined. *Numbers* in brackets indicate average word modification rate of each attack.

seeds, models achieve significant improvement under *static evaluation*. However, when we re-generate the adversarial test set for each model, their performances under *dynamic evaluation* stay consistently poor. We hypothesize that although the models differ only in the initial weights of the classification layer and the order of training examples, the resultant decision boundary can be substantially different. Consequently, the adversarial examples found by the attacker target specifically at the victim model and do not generalize well across different models. This observation reveals that **models performing well under static evaluation are not necessarily more robust**, as they can still be successfully attacked by a white-box attacker. We believe that using static evaluation for comparison could potentially lead to misleading conclusions about model robustness, and dynamic evaluation is the more reliable way for comparison. We adopt dynamic evaluation for the rest of the experiments in this paper and encourage future works to do so as well for fair comparisons.

5.2 Mixup Improves Robustness

For adversarial data augmentation (ADA), we generate adversarial training examples on the training set $\mathbf{X}_{orig}$ of SST-2, with a fine-tuned BERT and RoBERTa model as the victim model. For both ADA and MixADA, we add the corresponding adversarial examples of PWWS and TextFooler into training. For comparison, we also experiment with mixup alone without adding the adversarial examples. In such case, the model would only interpolate pairs of original training examples. We perform a greedy hyper-parameter search for the ratio of adversarial training samples and mixup parameter α as described in Section 5.3.2. We also report average word modification rates, which indicate the percentage of words being replaced for attacking. Higher word modification rates indicate that the model is harder to attack and hence needs more words to be replaced.

The results are shown in Table 2. We observe that: (1) Unlike what some previous works have shown, we find that ADA is generally ineffective

	PWWS	TextFooler
TMixADA	36.74	13.78
+iterative	28.45	6.26
SMixADA	41.85	16.80
+iterative	28.78	7.69

Table 3: Performance of MixADA under attacks in the one-shot approach and the iterative approach.

in improving model robustness. The discrepancy is mainly because that we adopt the more reliable *dynamic evaluation* rather than *static evaluation*. (2) Mixup alone (both TMix and SMix) can often improve model robustness. For example, for RoBERTa on IMDB, TMix and SMix improve the robust accuracy significantly under both PWWS and TextFooler attacks. This shows that mixup is effective on language data in improving the model's generalization ability on adversarial examples. (3) MixADA (both TMixADA and SMixADA) can achieve further robustness improvement as compared to ADA and mixup in all cases. For example, RoBERTa+SMixADA achieves significantly better robustness accuracy than RoBERTa+ADA and RoBERTa+TMix under both attacks. This proves that mixup and ADA can complement each other to better improve model robustness under adversarial attacks. (4) Compared to ADA, our MixADA method does not incur significant performance degradation on the original test sets while improving robustness accuracy. In some cases, for example, BERT+TMix and BERT+TMixADA even improve the model performance on the original test set. Such benefit is likely because that mixup creates virtual examples that are closer to the empirical data distribution. (5) We find that the models with MixADA also incur higher word modification rates under both attacks. For example, RoBERTa+TMixADA incurs 59.68% word modification rate under PWWS attack, while the RoBERTa baseline only needs 37.48% words to be replaced. This further demonstrates that our proposed method achieves better robustness.

5.3 Hyper-parameter Analysis

In this subsection, we perform further analysis to examine the effects of different hyper-parameters. There are two hyper-parameters involved in MixADA: the amount of adversarial data added for training, and the α parameter in the beta distribution of mixup coefficient. We also experiment with an alternative ADA strategy - iterative ADA.

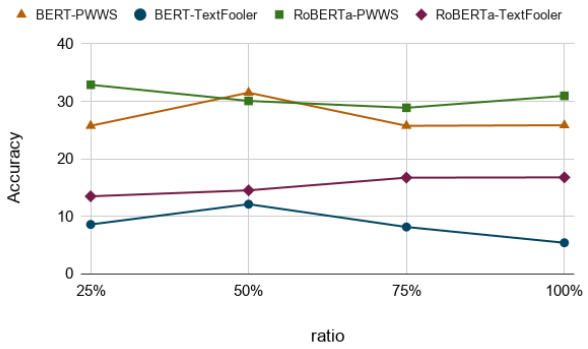

Figure 2: Performance under attacks on the SST-2 dataset with varying ratio of adversarial training samples.

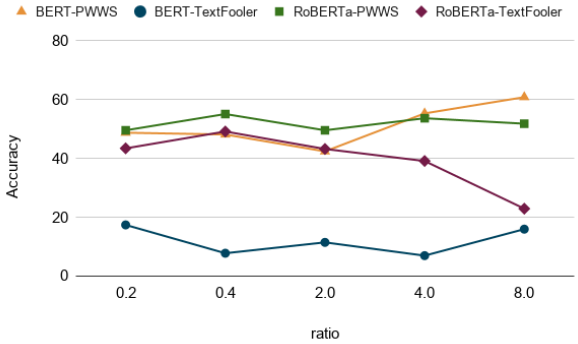

Figure 3: Performance under attacks on the IMDB dataset with varying α parameter for mixup.

5.3.1 Amount of Adversarial Training Data

We vary the ratio of the training dataset that we generate adversarial training samples on and add to the MixADA fine-tuning. We experiment with SMixADA with the hyper-parameter of mixup being fixed. On SST-2, we vary the ratio in $\{25\%, 50\%, 75\%, 100\%\}$. On IMDB, since the average sequence length is significantly longer and the adversarial example generation process becomes much slower, we experiment with a set of smaller ratios: $\{5\%, 10\%, 15\%, 20\%\}$. The results are plotted in in Figure 2. Interestingly, we find that higher ratio of adversarial training samples does not necessarily bring in additional robustness gains.

5.3.2 Interpolation Coefficient in Mixup

We also analyse the hyper-parameter of mixup: the α parameter in the beta distribution, from which the interpolation coefficient is sampled. We fix the ratio of adversarial training data and vary α in the range of $\{0.2, 0.4, 2.0, 4.0, 8.0\}$. The results are plotted in Figure 3. We find that there is no

consistent pattern across different datasets on what is the optimal α. Hence, for our main experiments in Table 2, we perform a greedy hyper-parameter search: we first tune the ratio of adversarial training samples, then fix the ratio and tune the α parameter for mixup. A more exhaustive hyper-parameter search might bring additional performance gains but would also incur extra computation costs.

5.3.3 Iterative ADA

For our MixADA experiments in Table 2, we generate all adversarial training samples at one shot and mix them with the original examples before fine-tuning. An alternative is to generate a new batch of adversarial training samples dynamically with the current model at each epoch (Michel et al., 2019; Huang et al., 2019). We compare this iterative approach with our MixADA and use the same ratio of adversarial training samples and mixup parameter α. We evaluate RoBERTa on the SST-2 dataset. The results are in Table 3. We find that the iterative approach is far worse than our one-shot approach. We hypothesize that in the one-shot approach, we generate the adversarial examples on a fully-fine-tuned model while the iterative approach generates adversarial examples on the not-well-fine-tuned model in the first few epochs, and hence the adversarial examples generated in the iterative approach are not as challenging and useful as those in our one-shot approach.

6 Conclusion

In this work, we first compared two modes of evaluation: *static evaluation* and *dynamic evaluation*. We reveal that the commonly used *static evaluation* is flawed. Based on the observation that conventional adversarial data augmentation perform poorly under dynamic adversarial evaluation, we propose to use mixup to increase the coverage of the attack space. We show that our proposed MixADA method greatly improves PLMs' robustness under strong adversarial attacks. Our MixADA method is one of the first defense methods that achieved significant robustness gains on pre-trained language models. We believe that our work can establish the appropriate evaluate protocol and provide a competitive baseline for future works on improving the robustness of pre-trained language models.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, M. Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of EMNLP*.

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020. Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification. In *Proceedings of ACL*.

Yong Cheng, Lu Jiang, Wolfgang Macherey, and Jacob Eisenstein. 2020. Advaug: Robust adversarial augmentation for neural machine translation. In *Proceedings of ACL*.

J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*.

Matt Gardner, Yoav Artzi, Jonathan Berant, Ben Bogin, Sihao Chen, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hanna Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, S. Singh, N. A. Smith, Sanjay Subramanian, Eric Wallace, A. Zhang, and Ben Zhou. 2020. Evaluating models' local decision boundaries via contrast sets. In *Proceedings of EMNLP(findings)*.

Siddhant Garg and Goutham Ramakrishnan. 2020. Bae: Bert-based adversarial examples for text classification. In *Proceedings of EMNLP*.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szeged. 2015. Explaining and harnessing adversarial examples. In *Proceedings of ICLR*.

Demi Guo, Y. Kim, and Alexander M. Rush. 2020. Sequence-level mixed sample data augmentation. In *Proceedings of EMNLP*.

Po-Sen Huang, Robert Stanforth, Johannes Welbl, Chris Dyer, Dani Yogatama, Sven Gowal, Krishnamurthy Dvijotham, and Pushmeet Kohli. 2019. Achieving verified robustness to symbol substitutions via interval bound propagation. In *Proceedings of EMNLP*.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of EMNLP*.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. In *Proceedings of EMNLP-IJCNLP*.

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 *Proceedings of AAAI*.

E. Jones, Robin Jia, Aditi Raghunathan, and P. Liang. 2020. Robust encodings: A framework for combating adversarial typos. In *Proceedings of ACL*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *Proceedings of ICLR*.

L. Li, Ruotian Ma, Qipeng Guo, X. Xue, and Xipeng Qiu. 2020. Bert-attack: Adversarial attack against bert using bert. In *Proceedings of EMNLP*.

Linyang Li and Xipeng Qiu. 2021. Tavat: Token-aware virtual adversarial training for language understanding. In *Proceedings of AAAI*.

Kevin J Liang, Weituo Hao, Dinghan Shen, Yufan Zhou, W. Chen, Changyou Chen, and L. Carin. 2020. Mixkd: Towards efficient distillation of large-scale language models. *ArXiv*, abs/2011.00593.

Y. Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Wentao Ma, Yiming Cui, Chenglei Si, Ting Liu, S. Wang, and Guo ping Hu. 2020. Charbert: Character-aware pre-trained language model. In *Proceedings of COLING*.

Andrew L. Maas, Raymond E. Daly, P. T. Pham, D. Huang, A. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of ACL*.

Paul Michel, Xian Li, Graham Neubig, and J. Pino. 2019. On evaluation of adversarial perturbations for sequence-to-sequence models. In *Proceedings of NAACL-HLT*.

Takeru Miyato, Andrew M. Dai, and Ian J. Goodfellow. 2017. Adversarial training methods for semi-supervised text classification. In *Proceedings of ICLR*.

Maximilian Mozes, Pontus Stenetorp, Bennett Kleinberg, and Lewis D. Griffin. 2020. Frequency-guided word substitutions for detecting textual adversarial examples. *ArXiv*, abs/2004.05887.

Danish Pruthi, Bhuwan Dhingra, and Zachary C. Lipton. 2019. Combating adversarial misspellings with robust word recognition. In *Proceedings of ACL*.

Shuhuai Ren, Yihe Deng, Kun He, and W. Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of ACL*.

Chenglei Si, Ziqing Yang, Yiming Cui, Wentao Ma, Ting Liu, and S. Wang. 2020. Benchmarking robustness of machine reading comprehension models. *ArXiv*, abs/2004.14004.

R. Socher, Alex Perelygin, J. Wu, Jason Chuang, Christopher D. Manning, A. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP*.

Lichao Sun, Congying Xia, Wenpeng Yin, Ting-Ting Liang, Philip S. Yu, and Lifang He. 2020. Mixup-transfomer: Dynamic data augmentation for nlp tasks. In *Proceedings of COLING*.

Samson Tan, Shafiq Joty, Min-Yen Kan, and Richard Socher. 2020. It's morphin' time! Combating linguistic discrimination with inflectional perturbations. In *Proceedings of ACL*.

Boxin Wang, Hengzhi Pei, Boyuan Pan, Qian Chen, Shuohang Wang, and Bo Li. 2020a. T3: Tree-autoencoder constrained adversarial text generation for targeted attack. In *Proceedings of EMNLP*.

Boxin Wang, Shuohang Wang, Y. Cheng, Zhe Gan, R. Jia, Bo Li, and Jing jing Liu. 2020b. Infobert: Improving robustness of language models from an information theoretic perspective. *ArXiv*, abs/2010.02329.

Tianlu Wang, Xuezhi Wang, Y. Qin, Ben Packer, Kang Li, J. Chen, Alex Beutel, and Ed Huai hsin Chi. 2020c. Cat-gen: Improving robustness in nlp models via controlled adversarial text generation. In *Proceedings of EMNLP*.

Yicheng Wang and Mohit Bansal. 2018. Robust machine comprehension models via adversarial training. In *Proceedings of NAACL*.

Michihiro Yasunaga, Jungo Kasai, and Dragomir R. Radev. 2018. Robust multilingual part-of-speech tagging via adversarial training. In *Proceedings of NAACL-HLT*.

Fan Yin, Quanyu Long, Tao Meng, and Kai-Wei Chang. 2020. On the robustness of language encoders against grammatical errors. In *Proceedings of ACL*.

Yuan Zang, Chenghao Yang, Fanchao Qi, Z. Liu, Meng Zhang, Qun Liu, and Maosong Sun. 2020. Word-level textual adversarial attacking as combinatorial optimization. In *Proceedings of ACL*.

Hongyi Zhang, M. Cissé, Yann Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *Proceedings of ICLR*.

Huangzhao Zhang, Hao Zhou, N. Miao, and Lei Li. 2019. Generating fluent adversarial examples for natural languages. In *Proceedings of ACL*.

Xiaoqing Zheng, Jiehang Zeng, Y. Zhou, C. Hsieh, Minhao Cheng, and X. Huang. 2020. Evaluating and enhancing the robustness of neural network-based dependency parsing models with adversarial examples. In *Proceedings of ACL*.

Y. Zhou, Xiaoqing Zheng, C. Hsieh, Kai-Wei Chang, and X. Huang. 2020. Defense against adversarial attacks in nlp via dirichlet neighborhood ensemble. *ArXiv*, abs/2006.11627.

C. Zhu, Y. Cheng, Zhe Gan, S. Sun, T. Goldstein, and Jing jing Liu. 2020. Freelb: Enhanced adversarial training for natural language understanding. In *Proceedings of ICLR*.

Wei Zou, Shujian Huang, John Xie, Xin-Yu Dai, and Jiajun Chen. 2020. A reinforced generation of adversarial samples for neural machine translation. In *Proceedings of ACL*.