Supplementary Material of
How Should Pre-Trained Language Models Be Fine-tuned Towards Adversarial Robustness?

Anonymous Author(s)
Affiliation
Address
email

A Appendix

A.1 The Proof of Lemma 1

The loss \( \mathcal{L}_{info} \) is the categorical cross-entropy loss of identifying \( t_i \) among \( \{t_j\}_{j=1}^N \), given \( s_i \) and \( y \). Thus, the optimal \( E_{f(s,i)} \) that minimizes \( \mathcal{L}_{info} \) is proportional to \( \frac{p(t_i|s_i,y)}{p(t_i|y)} \) (refer to [6] for more details). We then insert \( E_{f(s,i)} \) into \( \mathcal{L}_{info} \) and get what follows:

\[
\mathcal{L}_{info} = \mathbb{E}_{y \sim p_d(y)} \left[ \mathbb{E}_{(x_i,y) \sim \mathcal{N}_{\mathcal{N}}(1)} \left[ \frac{1}{N} \sum_{i=1}^{N} \log N - \frac{p(t_i|s_i,y)}{p(t_i|y)} \sum_{j=1}^{N} \frac{p(t_j|s_j,y)}{p(t_j|y)} \right] \right] \tag{12}
\]

(13)

\[
= \mathbb{E}_{y \sim p_d(y)} \left[ \mathbb{E}_{(x_i,y) \sim \mathcal{N}_{\mathcal{N}}(1)} \left[ \frac{1}{N} \sum_{i=1}^{N} \log N - \frac{p(t_i|s_i,y)}{p(t_i|y)} \sum_{j=1}^{N} \frac{p(t_j|s_i,y)}{p(t_j|y)} \right] \right] \tag{13}
\]

(14)

\[
= \mathbb{E}_{y \sim p_d(y)} \left[ \mathbb{E}_{(x_i,y) \sim \mathcal{N}_{\mathcal{N}}(1)} \left[ \frac{1}{N} \sum_{i=1}^{N} \log \left( \frac{1}{N} + \frac{N-1}{N} \mathbb{E}_{x \sim p_d(x|y)} \left[ \frac{p(t_j|s_j,y)}{p(t_j|y)} \frac{p(t_i|y)}{p(t_i|s_i,y)} \right] \right) \right] \right] \tag{15}
\]

(16)

\[
= \mathbb{E}_{y \sim p_d(y)} \left[ \mathbb{E}_{(x_i,y) \sim \mathcal{N}_{\mathcal{N}}(1)} \left[ \frac{1}{N} \sum_{i=1}^{N} \log \left( \frac{1}{N} + \frac{N-1}{N} \frac{p(t_i|y)}{p(t_i|s_i,y)} \right) \right] \right] \tag{17}
\]

(18)

\[
= \mathbb{E}_{y \sim p_d(y)} \left[ -\log \frac{p(t_i|s_i,y)}{p(t_i|y)} \right] \tag{19}
\]

6 Eq. [16] to Eq. [17] is by Jensen’s inequality. As such, \( -\mathcal{L}_{info} \) is a lower bound on \( I(S;T \mid Y) \) and a larger \( N \) makes the bound tighter.

A.2 Implementation Details

Textual sequence processing. For consistent word numbers per input under word substitution attacks, we separate word-level tokens by space and punctuations, and then follow the original tokenizer of BERT/RoBERTa to tokenize the input sequence. The byte-level RoBERTa tokenizer is further

Table 5: Vanilla Accuracy(%) of different finetuning methods on IMDB.

<table>
<thead>
<tr>
<th>Method</th>
<th>Model</th>
<th>Vanilla Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adv-Base</td>
<td>BERT</td>
<td>74.6</td>
</tr>
<tr>
<td>Adv-PTWD</td>
<td>BERT</td>
<td>76.6</td>
</tr>
<tr>
<td>Adv-Mixout</td>
<td>BERT</td>
<td>77.8</td>
</tr>
<tr>
<td>RIFT</td>
<td>BERT</td>
<td><strong>78.3</strong></td>
</tr>
</tbody>
</table>

(a) Accuracy (%) based on BERT-base-uncased.

<table>
<thead>
<tr>
<th>Method</th>
<th>Model</th>
<th>Vanilla Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adv-Base</td>
<td>RoBERTa</td>
<td>80.1</td>
</tr>
<tr>
<td>Adv-PTWD</td>
<td>RoBERTa</td>
<td>80.7</td>
</tr>
<tr>
<td>Adv-Mixout</td>
<td>RoBERTa</td>
<td>79.0</td>
</tr>
<tr>
<td>RIFT</td>
<td>RoBERTa</td>
<td><strong>84.2</strong></td>
</tr>
</tbody>
</table>

(b) Accuracy (%) based on RoBERTa-base.

Table 6: Vanilla Accuracy(%) of different finetuning methods on SNLI.

<table>
<thead>
<tr>
<th>Method</th>
<th>Model</th>
<th>Vanilla Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adv-Base</td>
<td>BERT</td>
<td>79.4</td>
</tr>
<tr>
<td>Adv-PTWD</td>
<td>BERT</td>
<td>78.4</td>
</tr>
<tr>
<td>Adv-Mixout</td>
<td>BERT</td>
<td>79.3</td>
</tr>
<tr>
<td>RIFT</td>
<td>BERT</td>
<td><strong>80.5</strong></td>
</tr>
</tbody>
</table>

(a) Accuracy (%) based on BERT-base-uncased.

<table>
<thead>
<tr>
<th>Method</th>
<th>Model</th>
<th>Vanilla Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adv-Base</td>
<td>RoBERTa</td>
<td>87.1</td>
</tr>
<tr>
<td>Adv-PTWD</td>
<td>RoBERTa</td>
<td>85.9</td>
</tr>
<tr>
<td>Adv-Mixout</td>
<td>RoBERTa</td>
<td>87.1</td>
</tr>
<tr>
<td>RIFT</td>
<td>RoBERTa</td>
<td><strong>87.9</strong></td>
</tr>
</tbody>
</table>

(b) Accuracy (%) based on RoBERTa-base.

modified to output one token per word to fit the setting of word substitution attacks. The maximum number of tokens including special tokens per input is set as 300 for IMDB, and 80 for SNLI.

Hyper-parameters and optimization details. We set $\alpha$ as 0.1 for IMDB and 0.7 for SNLI. $\tau$ is set as 0.2 for both IMDB and SNLI. Other hyper-parameters are set as the same among all compared methods for fair comparisons. For both standard fine-tuning and adversarial fine-tuning, we run for 20 epochs with batch size 32 for IMDB, and run for 20 epochs with batch size 120 for SNLI. Early stopping is used for all compared methods according to best robust accuracy. AdamW optimizer is employed with learning rate of 0.00002. We do not apply weight decay on an objective model, and set weight decay rate as 0.0002 for task-specific layers.

Model architectures. For both BERT and RoBERTa, the representation with respect to the sequence classification token of the last layer is employed as the output feature, which is later taken as the input of the task-specific layers for predictions. The task-specific layer is a MLP that has two linear layers with relu activation after the first layer and softmax after the second one.

A.3 Runtime Analysis

All models are trained using the Nvidia A100 GPU and our implementation is based on PyTorch. As for IMDB, it takes about 10 GPU hours to train a BERT or RoBERTa based model using RIFT. As for SNLI, it takes about 40 GPU hours to train a BERT or RoBERTa based model using RIFT.

A.4 Vanilla Accuracy

In addition to the main result in Sec. 4.3, we here show the vanilla accuracy of each methods in Tabs. 5 and 6 as a supplement. As we can see, RIFT surpasses all other adversarial fine-tuning method in terms of vanilla accuracy. It again validates that RIFT does help retain the generalizable information learned before.

A.5 Codes

Our codes are available at https://github.com/neurips5447/submission.

A.6 License of Used Assets

The assets and the corresponding licenses are as follows. IMDB dataset [5]: Non-Commercial Licensing. SNLI dataset [2]: Creative Commons Attribution-ShareAlike 4.0 International License. Genetic attack [1]: MIT License. PWWS attack [7]: MIT License. Certified robustness [4]: MIT License. ASCC-defense [3]: MIT License.
References


[5] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In ACL, 2011.
