ON ROBUST PREFIX-TUNING FOR TEXT CLASSIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Recently, increasing interest has been focused on prefix-tuning, an efficient finetuning method for large-scale pretrained language models that only needs to maintain the embedding parameters of prefix tokens for each downstream task. Despite being lightweight and modular, prefix-tuning is still lacking in robustness, as a variety of adversarial attack approaches have been proposed to fool a text classifier. However, most of the currently developed textual defense techniques necessitate auxiliary model update and storage, which will inevitably hamper the modularity and low storage of prefix-tuning. In this work, we propose a robust prefix-tuning framework that preserves the efficiency and modularity of prefix-tuning. The core idea of our framework is leveraging the layerwise activations of the language model by correctly classified training data as the standard for additional prefix finetuning. During the test phase, an extra batch-level prefix is tuned for each batch and added to the original prefix for robustness enhancement. Extensive experiments on three text classification benchmarks show that our framework substantially improves robustness over several strong baselines against five textual attacks of different types while maintains comparable accuracy on clean texts. We also interpret our robust prefix-tuning framework from the optimal control perspective and pose several directions for future research.

1 INTRODUCTION

Large-scale pretrained language models (LMs) (Peters et al., 2018; Devlin et al., 2019; Radford et al., 2019; Liu et al., 2019; Yang et al., 2019; Raffel et al., 2020; Lewis et al., 2020; Brown et al., 2020; Xue et al., 2021) have proven effective for downstream NLP tasks. While finetuning a pretrained model for a specific task has been the common practice, it comes at the cost of maintaining a full copy of the LM with the parameters entirely modified. The prohibitively huge memory demand poses a severe challenge for the deployment of practical NLP systems, which motivates the development of *low-storage adaptation* methods (Houlsby et al., 2019; Li & Liang, 2021).

Recently, increasing interest has been focused on prompt-based tuning approaches for pretrained language models (Wallace et al., 2019; Puri & Catanzaro, 2019; Shin et al., 2020; Jiang et al., 2020b; Zhong et al., 2021; Gao et al., 2021; Hu et al., 2021; Liu et al., 2021). By prepending several elaborately-selected tokens to the given input sequences, the LM is triggered to respond with appropriate outputs without updating its parameters. Prefix-tuning (Li & Liang, 2021) introduces the idea of replacing the discrete prompt tokens at the input with the virtual ones at the start of each layer in the LM. By optimizing the layerwise continuous prefix embedding instead of selecting candidates in the vocabulary list, the expressive ability of prompts is further enhanced with a rather small amount of parameters to be updated. As a result, prefix-tuning requires near $1000\times$ fewer task-specific parameters than finetuning the entire pretrained model (Bommasani et al., 2021).

Despite being lightweight and modular, prefix-tuning is still lacking in robustness. In the NLP community, a variety of techniques of generating adversarial examples have been proposed to attack a text classifier by perturbing inputs (Zhang et al., 2020). Conventional attack techniques include character-level (Eger et al., 2019; He et al., 2021), word-level (Alzantot et al., 2018; Ren et al., 2019; Garg & Ramakrishnan, 2020), sentence-level modification (Iyyer et al., 2018; Ribeiro et al., 2018; Xu et al., 2021), or a mixture of them (Ebrahimi et al., 2018; Li et al., 2019). Instead of perturbing each input sentence separately, recently, universal adversarial triggers (UAT) (Wallace

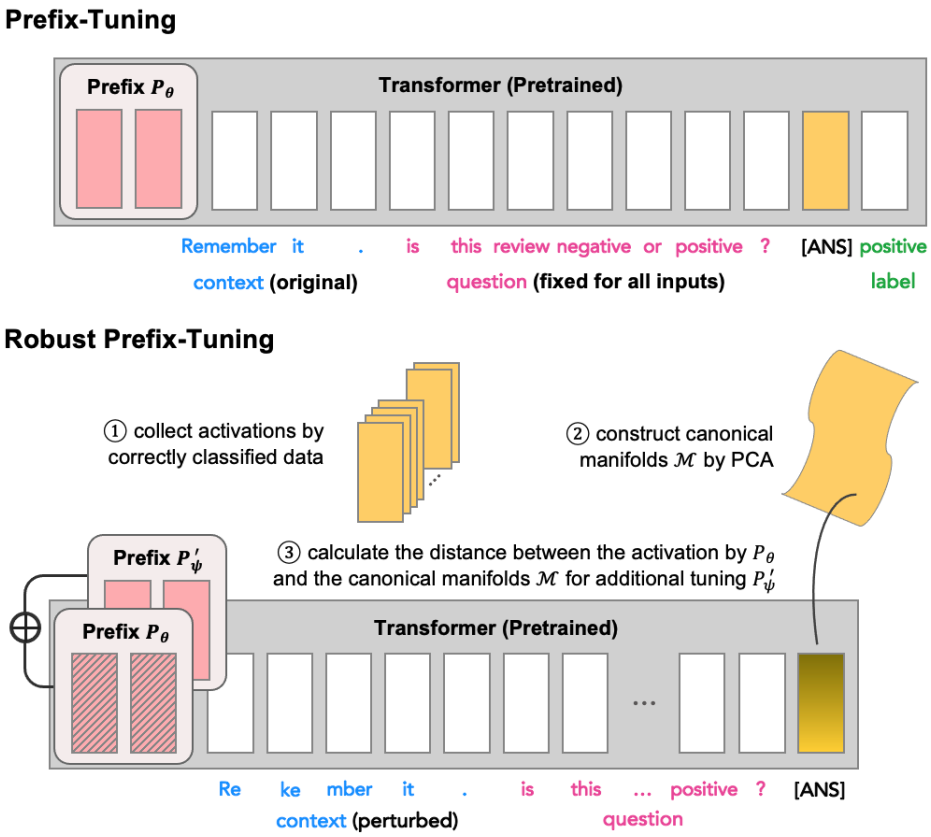


Figure 1: Overview of prefix-tuning as well as our robust prefix-tuning framework for text classification. We frame the samples into a SQuAD-like scheme consisting context, question, and label and optimize the original prefix P_θ. For the robust prefix-tuning framework, we fix the obtained prefix P_θ and tune an additional prefix P'_ψ for each test batch. The additional tuning follows the three steps indicated in the figure, which aims to lead the summed prefix to steer correct activations at the position of the $[\mathrm{ANS}]$ token with those activated by correctly classified training data as the standard.

et al., 2019) becomes powerful by prepending the same adversarial tokens to all test inputs. UAT prompts the model to generate malicious outputs, which shares the same spirit with the prompt-based tuning approaches. It remains a mystery whether prefix-tuning, a variant of prompt-based tuning techniques, can defend against UAT as well as other different kinds of attacking techniques.

In defense of adversarial attacks, different types of defense techniques are developed, including model functional improvement (Li & Sethy, 2019; Jones et al., 2020), certification (Jia et al., 2019; Huang et al., 2019; Shi et al., 2020; Xu et al., 2020; Ye et al., 2020), adversary detection (Pruthi et al., 2019; Zhou et al., 2019), and adversarial training (Miyato et al., 2017; 2018; Zhu et al., 2020; Jiang et al., 2020a; Liu et al., 2020; Wang et al., 2021; Dong et al., 2021; Zhou et al., 2021). While these approaches have enhanced model robustness, difficulties emerge when fitted to prefix-tuning. Most of the techniques require modification to the architecture and the parameters of the LM or additional maintenance of adversary detectors. Directly applying such techniques necessitates auxiliary model update and storage, which will inevitably hamper the modularity of prefix-tuning. Moreover, The excessively long time for adversarial training is also a hindrance to the efficient use of prefix-tuning. We ask the following question: *Can we improve the robustness of prefix-tuning while preserving its efficiency and modularity, as well as remaining the pretrained models unmodified?*

In this work, we propose a robust prefix-tuning framework for text classification. The main idea of our framework is to add an extra batch-level prefix tuned for each batch to the original prefix embedding during test time for robustness enhancement. We first record the layerwise activations in the LM at the position of generating label prediction with correctly classified training data. We project the collected activation matrices of each layer onto low-level canonical manifolds as the charac-

terization of "correct" model behavior. In this way, the correctness of any layerwise activations at the position of prediction generation can be estimated by projecting to the canonical manifolds and measuring the distance between them. For each test batch during inference, the added extra prefix is tuned on the fly with the original prefix fixed to minimize the calculated distance. Triggered by the summed prefix, the LM is prone to generating correct label predictions. We conduct extensive experiments on three text classification benchmarks and show that the proposed framework substantially improves model robustness against five strong textual attack approaches including input perturbation attack of different levels as well as the UAT attack. To the best of our knowledge, we are the first to propose the defense approach for prefix-tuning while remaining its lightweightness and modularity. Moreover, we provide interpretation of our robust prefix-tuning framework from the optimal control perspective and pose several directions for future research.

2 Prefix-Tuning for Text Classification

Prefix-tuning is a lightweight alternative to finetuning when using large-scale pretrained language models to solve downstream NLP tasks. The intuition of prefix-tuning follows prompt-based methods that a proper context prepended to input sentences triggers the desired response of the LM without changing the large amount of LM parameters. Instead of instantiating the prepended context with discrete tokens, prefix-tuning uses trainable prefix embeddings as a replacement, which is also known as *soft prompts*. The continuous prefix embeddings enable continuous optimization, and are prepended to all Transformer layers to improve its expressiveness. Following the notation of Li & Liang (2021), the activation at the i-th position of the j-th layer in a T-layer autoregressive Transformer LM is denoted as $h_i^{(j)}$. $h_i = [h_i^{(0)}; \cdots; h_i^{(T-1)}]$ represents the stacked activations:

$$h_i = \begin{cases} P_\theta[i,:], & \text{if } i \in \mathrm{P_{idx}}, \\ \mathrm{LM}_\phi(z_i, h_{<i}), & \text{otherwise.} \end{cases} \tag{1}$$

where $\mathrm{P_{idx}}$ is the sequence of prefix indices and z_i is the i-th token in the input sequence. The activations of the first $|\mathrm{P_{idx}}|$ positions are directly calculated by P_θ. All of the activations at the following positions depend on the prefix as the autoregressive Transformer LM follows the left-to-right calculation process. To stabilize the optimization, the prefix embedding matrix P_θ is reparameterized as $P_\theta[i,:] = \mathrm{MLP}_\theta(\hat{P}_\theta[i,:])$ by a feedforward network MLP_θ with a smaller matrix $\hat{P}_\theta$.

While prefix-tuning is proposed for conditional generation tasks, in this work, we use prefix-tuning for text classification. As shown in Figure 1, following the protocol of decaNLP (McCann et al., 2018), we frame the samples in classification tasks into a SQuAD-like scheme consisting $\mathrm{context}$, $\mathrm{question}$ and label: the $\mathrm{context}$ and the label part refer to the text sequence to be classified and the ground-truth label, while the $\mathrm{question}$ part is a prescribed task description sentence fixed for all samples. We denote $x = [\mathrm{context}, \mathrm{question}, [\mathrm{ANS}]]$, where $[\mathrm{ANS}]$ is a special token that separates $\mathrm{question}$ and label. We let $y = [\mathrm{label}]$ and $|y| = 1$ as the label is one token. At the position that $[\mathrm{ANS}]$ is inputted, the LM generates prediction of the next label token, and we denote this position as the output position o. While o can be different for different input x's, in this paper, we omit the relation $o = o(x)$ for simplicity. Prefix-tuning aims to steer the LM to maximize the probability of the label. We use all samples in the training set $\mathcal{D}_{tr}$ to optimize the prefix $P_\theta[i,:]$. The objective is

$$\min_\theta \mathbb{E}_{(x,y)\sim\mathcal{D}_{tr}} \mathcal{L}(y|x;\theta) = \max_\theta \mathbb{E}_{(x,y)\sim\mathcal{D}_{tr}} \log \left[W(h_o^{(T)}) \right]_y, \tag{2}$$

where W in the LM transforms the top-layer output $h_o^{(T)}$ to a probability vector over the vocabulary.

With continuous optimization on training samples, prefix-tuning is expected to steer the LM to generate correct label predictions for test data. With the large-scale LM parameters fixed, the obtained task-specific prefix is lightweight and modular. However, prefix-tuning is still vulnerable against text attack. With the context part perturbed by text attack techniques, the LM can be fooled to generate erroneous label prediction at the output position. Figure 1 shows an example of perturbation: by modifying a single character *m* in the word *remember* with *k*, the prediction of the LM is shifted from positive to negative. Therefore, it remains under exploration how to robustify prefix-tuning without hampering its modularity or introducing additional large model update and storage.

3 Robust Prefix-Tuning

We propose a robust prefix-tuning framework for text classification. Our intuition follows prefix-tuning that proper prefix embedding prepended to inputs can steer a LM with correct response. When

the inputs are adversarially perturbed, the LM activations at the output position fails to be steered in the correct way by the original prefix $P_\theta[i,:]$. Inspired by Khoury & Hadfield-Menell (2018) that the perturbed data often deviates from the low-dimensional data manifold, our robust prefix-tuning framework uses the layerwise activations by correctly classified training data to construct canonical manifolds $\mathcal{M}$. When provided with perturbed inputs during inference, we add an extra prefix $P'_\psi[i,:]$ tuned for each test batch to $P_\theta[i,:]$ that aims to rectify the erroneous activations at the output position so that they stay close to the canonical manifolds. In this way, we expect the summed prefix to steer the LM with correct label generation against input perturbations. Shown in Figure 1, our robust prefix-tuning framework consists of three steps. The first step is collecting correct LM activations at the output position o triggered by $P_\theta[i,:]$. We denote S_C as the set of correctly classified training examples. For the j-th layer, the collected activation matrix $H_C^{(j)}$ stacks the j-th layer LM activation at the output position o with input of all $c \in S_C$:

$$H_C^{(j)} = \left[h_{o,c}^{(j)}\right] \in \mathbb{R}^{|S_C| \times d}. \tag{3}$$

The d represents the dimension of the LM hidden state. In practice, we always have $|S_C| >> d$.

The second step is constructing canonical manifolds. We project the collected j-th layer activation matrix $H_C^{(j)}$ onto a low-level manifold $\mathcal{M}^{(j)}$ as the characterization of the correct j-th layer behavior. We use PCA (Pearson, 1901) to get the projection $Q^{(j)}$ onto the canonical manifold of the j-th layer:

$$X_C^{(j)} = U^{(j)} \Sigma^{(j)} {V^{(j)}}^{\mathrm{T}}, \tag{4}$$

$$Q^{(j)} = {V_p^{(j)}}^{\mathrm{T}} V_p^{(j)}, \tag{5}$$

where $V_p^{(j)}$ consists of rows of singular vectors with the p-largest singular values and $Q^{(j)} \in \mathbb{R}^{d \times d}$.

The third step is tuning $P'_\psi[i,:]$ to robustify prefix-tuning during inference. Here the vector $P'_\psi[i,:]$ is *not* reparameterized by MLP. With the additional prefix $P'_\psi[i,:]$, the token-wise activations become

$$h_i = \begin{cases} P_\theta[i,:] + P'_\psi[i,:], & \text{if } i \in \mathrm{P}_{\mathrm{idx}}, \\ \mathrm{LM}_\phi(z_i, h_{<i}), & \text{otherwise.} \end{cases} \tag{6}$$

For the j-th layer at the output position, the LM activation matrix triggered by $P_\theta[i,:] + P'_\psi[i,:]$ with the potentially perturbed test input batch S_{te} is stacked as

$$H_{te}^{(j)} = \left[h_{o,t}^{(j)}\right] \in \mathbb{R}^{|S_{te}| \times d} \tag{7}$$

for all $t \in S_{te}$. We use the distance from $H_T^{(j)}$ to the j-th canonical manifold $\mathcal{M}^{(j)}$ as the loss for the additional tuning of $P'_\psi[i,:]$ for each batch:

$$\min_\psi \sum_{j=0}^{N-1} L\left(\psi^{(j)}, Q^{(j)}\right) = \sum_{j=0}^{N-1} \left\| \left(H_{te}^{(j)} - H_{te}^{(j)} \mathbf{1}\mathbf{1}^{\mathrm{T}}\right)\left(I - Q^{(j)}\right) \right\|_2. \tag{8}$$

After tuning $P'_\psi[i,:]$, the activated $H_{te}^{(j)}$ is closer to the canonical manifold $\mathcal{M}^{(j)}$. As the manifold characterizes the correct behavior of the j-th layer activation, by regulating the layerwise activations at the output position, the summed prefix $P_\theta[i,:] + P'_\psi[i,:]$ is prone to steering the LM to generate correct label predictions. Our framework is also applicable to other soft prompt-based tuning methods (Qin & Eisner, 2021; Hambardzumyan et al., 2021; Lester et al., 2021; Cho et al., 2021; Tsimpoukelli et al., 2021) by recording the activations of correctly classified training data at the output position, constructing the canonical manifolds for the soft prompts, and tuning additional soft prompts for robustness during test phase. In this work, we conduct experiments on prefix-tuning.

4 Experiments

4.1 Setup

We consider three text classification benchmarks in our experiments: binary Stanford Sentiment Treebank (SST-2) (Socher et al., 2013), AG's News (Zhang et al., 2015), and Stanford Natural

Table 1: Results of different baselines with our framework applied. Our framework substantially improves robustness of both standard and adversarial prefix-tuning against all types of attacks.

Benchmark	Method	Clean	PWWS	VIPER	SCPN	BUG	UAT
SST-2	std. prefix-tuning	92.48	16.64	1.92	31.58	8.84	5.05
	+ our framework	**92.59**	**50.36**	**44.65**	**58.54**	**46.68**	**85.72**
	adv. prefix-tuning	93.57	30.64	7.25	35.42	25.04	4.88
	+ our framework	**93.79**	**57.55**	**43.60**	**60.68**	**57.17**	**91.87**
AG's News	std. prefix-tuning	86.42	43.00	24.55	43.20	43.22	52.20
	+ our framework	**86.50**	**53.91**	**24.93**	**48.38**	**51.79**	**76.47**
	adv. prefix-tuning	89.46	50.91	26.30	45.25	47.11	59.74
	+ our framework	**90.26**	**56.45**	**31.25**	**48.03**	**54.66**	**87.26**
SNLI	std. prefix-tuning	72.48	25.51	29.69	42.46	32.88	30.20
	+ our framework	**72.74**	**34.68**	**33.64**	**43.37**	**36.59**	**71.03**
	adv. prefix-tuning	77.22	27.43	28.58	46.83	34.94	35.76
	+ our framework	**77.65**	**33.88**	**33.98**	**46.92**	**38.21**	**76.15**

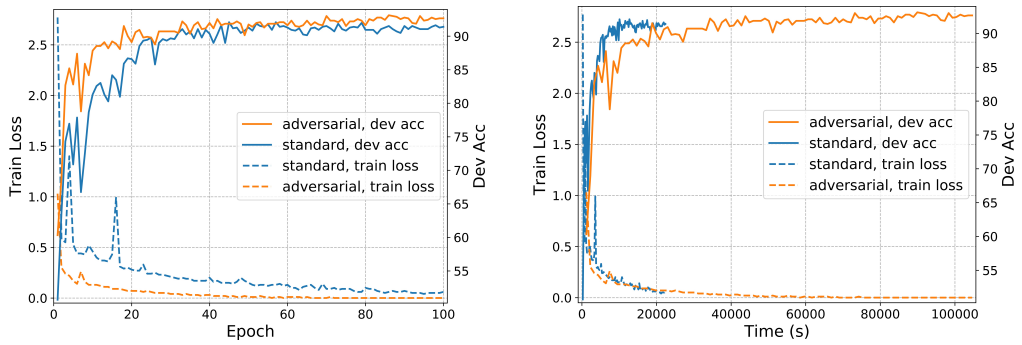
Language Inference (SNLI) (Bowman et al., 2015). We evaluate our robust prefix-tuning with five text attacks: PWWS (Ren et al., 2019), VIPER (Eger et al., 2019), SCPN (Iyyer et al., 2018), TextBugger ("BUG" for short) (Li et al., 2019), and UAT (Wallace et al., 2019). We use the GPT2-medium (with $T = 24$ layers in total) as the large-scale language model and set prefix length $= 10$ for all prefix-tuning experiments. We train 100 epochs for SST-2 and 25 epochs for AG's News and SNLI. We set $N = 3$ and record the bottom N-layer activations of the LM at the output position for additional finetuning. Other experiment configuration and details can be found in Appendix A.

4.2 Results on Adversarial Training

We first apply adversarial training to prefix-tuning as our baselines named adversarial prefix-tuning. Following Miyato et al. (2017), we inject perturbation restricted within the ℓ_2 ball of the flattened word embedding of original sentences during training to adversarially optimize the prefix $P_\theta[i, :]$. We have also attempted to use other types of adversarial training, (i.e., adding the KL-divergence term for regularization (Miyato et al., 2018; Zhang et al., 2019)) but obtained poor accuracy, suggesting the difficulty of optimizing the small amount of prefix embedding parameters. The experimental details can be found in Appendix B.1. Our robust prefix-tuning framework is applicable to the adversarially-trained prefix embedding, as it keeps the acquired prefix $P_\theta[i, :]$ fixed and tunes extra $P'_\psi[i, :]$ for each test batch. The experimental results are listed in Table 1. According to Table 1, our approach significantly improves robustness of prefix-tuning over all baselines against each type of text attacks. Compared with the standard prefix-tuning baseline, the adversarial baselines achieve better robustness and clean accuracy. In fact, fewer training epochs are needed to achieve certain clean accuracy when applying adversarial training to prefix-tuning. Figure 2-(a) illustrates the loss on the training set and the clean accuracy on the validation set for each epoch during standard and adversarial prefix-tuning. From Figure 2-(a), it is clear that adversarial prefix-tuning outperforms standard prefix-tuning in both the convergence rate and generalization.

However, the adversarial prefix-tuning approaches suffer from the catastrophic drawback of taking far greater training time than the standard prefix-tuning baseline. Figure 2-(b) changes the horizontal axis in Figure 2-(a) from epoch to clock time. According to Figure 2-(b), given the same time budget, standard prefix-tuning finishes training for 100 epochs while adversarial prefix-tuning only manages to train for 20 epochs. The training losses of the two methods at the time are also roughly the same (0.06). Earlystopping adversarial prefix-tuning at 20 epochs achieves slightly better robustness results than 100 epochs, but the clean accuracy becomes lower than standard prefix-tuning (results can be found in Appendix B.2). To conclude, the adversarial prefix-tuning approaches make regularization effect on prefix-tuning but lack competitiveness in practice.

In contrast with adversarial prefix-tuning, our approach is more practical as the duration of the inference-phase on-the-fly optimization of the additional prefix $P'_\psi[i, :]$ is negligible compared with



(a) (b)

Figure 2: Comparison between standard and adversarial prefix-tuning for 100 epochs with respect to (a) epoch and (b) clock time. While adversarial prefix-tuning gains strengths in epoch-wise convergence rate and generalization, it takes far greater training time than standard prefix-tuning.

training-phase adversarial prefix-tuning. From Table 1, the robustness of the standard prefix-tuning applied with our approach has surpassed that of the adversarial prefix-tuning without our approach. The adversarial prefix-tuning accompanied with our method achieves the strongest robustness against most of the text attacks.

4.3 Results on Adversarial Data Augmentation

Table 2: Results of PWWS adversarial data augmentation baselines as well as our methods on the SST-2 benchmark. Our methods consistently improve robustness over all baselines.

Method	Clean	PWWS	VIPER	SCPN	BUG	UAT
std. prefix-tuning	92.48	16.64	1.92	31.58	8.84	5.05
+ our framework	**92.59**	**50.36**	**44.65**	**58.54**	**46.68**	**85.72**
20-epoch adv. aug.	**87.15**	48.11	42.12	48.11	47.01	10.43
+ our framework	86.93	**55.57**	**51.62**	**57.33**	**52.00**	**70.07**
50-epoch adv. aug.	**86.82**	55.08	35.04	44.65	49.37	3.13
+ our framework	86.66	**58.26**	**48.05**	**58.98**	**51.73**	**83.47**
100-epoch adv. aug.	**91.21**	56.62	23.01	38.55	45.14	7.91
+ our framework	90.39	**62.55**	**46.95**	**59.25**	**54.75**	**86.38**

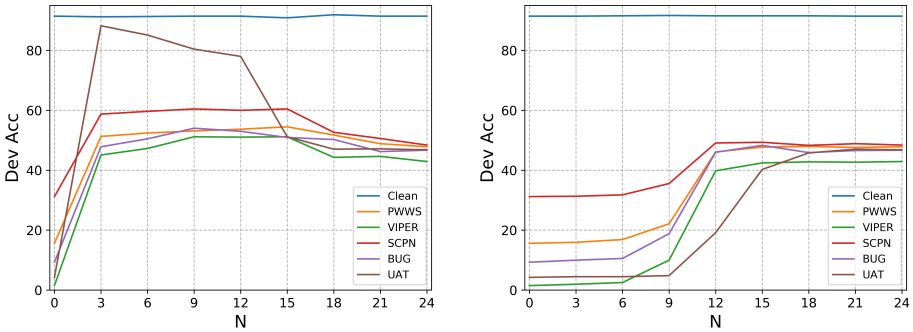
In this section, we conduct adversarial data augmentation during the training phase of standard prefix-tuning as our baseline methods. During training, we use a specified text attack method to perturb each batch and augment the perturbed data to the training set. Due to the computational inefficiency of discrete optimization for text attacks, the training of adversarial data augmentation is even far slower than that of adversarial prefix-tuning introduced in Section 4.2. We set milestones for the training of adversarial data augmentation at 20, 50, and 100 epochs for comparison. When applying our framework, we calculate the canonical manifolds $\mathcal{M}$ using the correctly classified samples from both the clean training set and the perturbed training set by the specified text attack.

Considering the training time budget, we select SST-2 as the benchmark. We use PWWS as the attack method to generate perturbed inputs with results listed in Table 2. On one hand, the clean accuracy of the PWWS adversarial data augmentation method is affected at the 20-epoch and the 50-epoch milestones. This might be attributed to the consideration of perturbed inputs during training. The robustness results, on the other hand, are improved compared with standard prefix-tuning. The robustness against PWWS obtains steady improvement thanks to the adversarial data augmentation generated by the attack method of the same type. However, for other types of attack, the robustness improvement is decreasing with the training process, suggesting the possibility that overfitting to the specified type of adversarial data can be harmful to robustness against other types of attacks.

With our robust prefix-tuning framework applied, all milestones of the adversarial data augmentation baseline obtain substantial robustness improvement. While we calculate the layerwise canonical manifolds with the correctly classified samples from both the clean training set and the PWWS-perturbed training set, it can be seen that the robustness against PWWS attack is still enhanced by several percent, while the clean accuracy is slightly affected. This shows that when calculating the layerwise manifolds, the used samples from the two types of training data, though are all correctly classified, trigger the LM with activations of different properties at the output position. It is left for future work about how to construct the dataset to calculate the canonical manifolds to achieve the best performance on both accuracy and robustness. We have also used VIPER and SCPN as the attack methods for adversarial data generation and draw similar conclusions to the PWWS adversarial data generation experiments, results of which can be found in Appendix C.

4.4 Effect of Tuning Different Layers

In this section, we study the effect of tuning our robust prefixes with different layers. While we tuned the robust prefixes of the bottom $N = 3$ layers for all previously reported experiments, different results are obtained when tuning the robust prefixes of alternative layers. We experiment on the SST-2 development set with tuning the bottom N-layer robust prefixes as well as the top ones with N enumerating from 0 to 24 with the step size $= 3$. $N = 0$ represents the original standard prefix-tuning method, and $N = 24$ means tuning all layers. The results are shown in Figure 3. According to the results, tuning the bottom N layers achieves better robustness improvement than



(a) Tuning bottom N layers (b) Tuning top N layers

Figure 3: Results on the SST-2 development set when applying robust prefixes of different layers. Tuning bottom N layers outperforms tuning top N layers in terms of robustness improvement.

tuning the top ones. This can be attributed to layerwise bottom-up error accumulation of the output position activations triggered with perturbed inputs. For the in-sentence text attacks, tuning the bottom $N \leq 15$ layers achieves comparable robustness improvement and slightly outperforms the larger N's. For the UAT attack, setting the bottom $N = 3$ is significantly better than the choices of larger N's. One possible reason for the exceptional performance is that the collected low-layer activations capture richer positional information (Voita et al., 2019), which helps to defend against UAT as UAT shifts right all inputs with the same trigger. We proceed to study how our method steers the LM with different N's for a better understanding of its effect on robustness improvement.

5 How does Robust Prefix-Tuning Steer the LM?

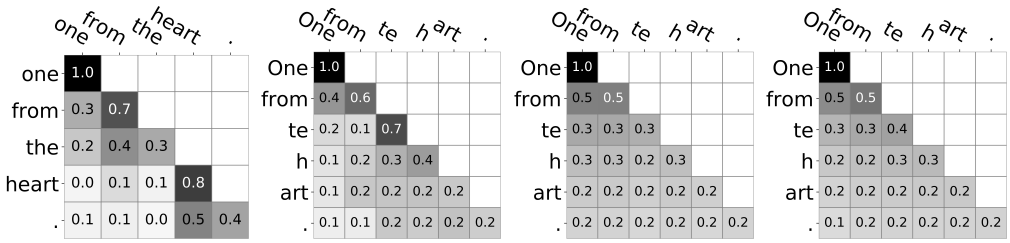
As the robust prefix-tuning framework with the proper choice of N substantially improves the robustness of prefix-tuning, it deserves to be explored how the robust prefix-tuning steers the LM compared with the original prefix. In this section, we investigate the attention weights of the top layer in the LM to study the behavior of our framework with the standard prefix-tuning baseline. We first visualize the attention maps averaged with all heads triggered by the original prefix with the original and the perturbed inputs to realize how the classifier is fooled, and then the ones triggered by our robust prefix with the same perturbed inputs to observe how our robust prefix-tuning updates model activations. We ignore the attention weights of the prefix tokens and renormalize the rest of the tokens in visualization. We find that the behavior of our robust prefix-tuning differs between the

in-sentence attacks and UAT. For the in-sentence attacks, its behavior can be summarized as *averaging the attention*. For UAT, with the proper setting of N, it functions as *ignoring the distraction*.

5.1 In-Sentence Attacks: Averaging the Attention

Table 3: An example original input from the development set of SST-2 and its perturbed version by the BUG attack. It can be seen that the prediction triggered by the original prefix-tuning is flipped.

Type	Input (the context part)	Predict	Label
Original	one from the heart .	positive	positive
Perturbed	One from te hart .	negative	positive



(a) Predict:positive (b) Predict:negative (c) Predict:positive (d) Predict:positive

Figure 4: Visualized attention weight maps for the case study with the BUG attack. (a): the original input by the original prefix-tuning; (b): the BUG-perturbed input by the original prefix-tuning; (c) and (d): the BUG-perturbed input by our robust prefix-tuning with (c) $N = 24$ and (d) $N = 3$. Our robust prefix-tuning (with both $N = 24$ and $N = 3$) shows the behavior of *averaging the attention* and corrects the prediction of the perturbed input by the original prefix-tuning.

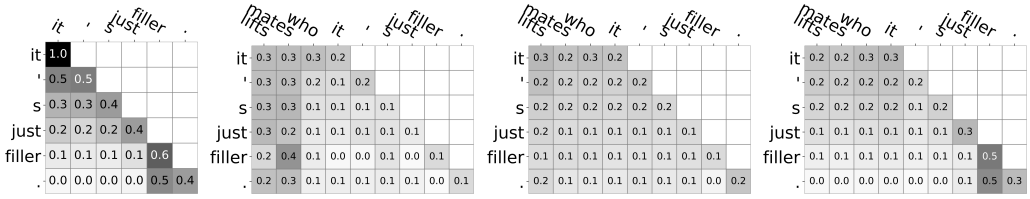
We use the adversarial inputs generated by the BUG attack on SST-2 development set for our following in-sentence case studies. Similar behavior is observed as well by other in-sentence attacks, and more case studies can be found in Appendix D.1. Table 3 shows an example of input perturbed by the BUG attack, and Figure 4 shows the attention map visualization. Shown in Figure 4-(a), the original prefix-tuning manages to steer the LM to assign higher attention weights to the important token "heart". Because of character omission, the attention weights over the perturbed input by the original prefix-tuning are perplexed (Figure 4-(b)). Our robust prefix-tuning shows the behavior of *averaging the attention* in this case. In Figure 4-(c) and Figure 4-(d), when generating each token, our robust prefix-tuning steers the LM to assign averaged attention weights over its previous tokens. The effect of averaged attention is similar to a continuous bag-of-words model, which might help the LM to consider global information and avoid concentrating on the specific perturbed tokens. As a result, our robust prefix-tuning corrects the erroneous prediction by the original prefix-tuning.

5.2 Universal Adversarial Triggers: Ignoring the Distraction

Table 4: The example input from the development set of SST-2 and its attacked version by UAT. With the adversarial triggers "lifts mates who", the prediction by the original prefix-tuning is shifted.

Type	Input (the context part)	Predict	Label
Original	it 's just filler .	negative	negative
Perturbed	lifts mates who it 's just filler .	positive	negative

We use the Universal Adversarial Triggers (UAT) to attack SST-2 development set, and the exploited triggers are "lifts mates who". Table 4 shows an example of input attacked by the UAT. Figure 5 shows the attention map visualization. Shown in Figure 5-(a), the essential token "filler" is assigned with higher attention weights steered by the original prefix-tuning. However, according to Figure 5-(b), the adversarial triggers attract major attention weights. For our robust prefix-tuning framework, when tuning $N = 24$ layers, the behavior of *averaging the attention* can be observed. The robust prefix tuned in this way, however, fails to fully exploit its potential. In contrast, When tuning $N = 3$



(a) Predict:negative (b) Predict:positive (c) Predict:positive (d) Predict:negative

Figure 5: Visualized attention weights for the case study with the UAT attack. (a): the original input by the original prefix-tuning; (b): the UAT-attacked input by the original prefix-tuning; (c) and (d): the UAT-attacked input by our robust prefix-tuning with (c) $N = 24$ and (d) $N = 3$. For $N = 24$, the behavior of robust prefix-tuning remains *averaging the attention*, but for $N = 3$, robust prefix-tuning functions as *ignoring the distraction*, thus improves the robustness of the original prefix.

layers, our robust prefix-tuning manages to steer the LM to assign low attention weights over the adversarial trigger and high attention weights over the essential token "filler". As a result, the tuned robust prefix functions in the manner of *ignoring the distraction*, which substantially improves the robustness of the original prefix. Due to page limit, we attach more visualization in Appendix D.2.

6 Interpretation from the Optimal Control Perspective

In this section, we provide an interpretation from the optimal control perspective for prefix-tuning as well as our robust prefix-tuning framework. Prefix-tuning seeks $P_\theta[i,:]$ with forward propagation of the j-th layer of the T-layer LM at the output position o as

$$h_o^{(j+1)} = \mathrm{LM}_\Phi^{(j)}\left(h_o^{(j)} \,\middle|\, h_{<o}^{(j)}\right) \tag{9}$$

with $h_i^{(j)} = P_{\theta^{(j)}}[i,:]$ for all $j = 0$ to $T-1$, $i \in \mathrm{P_{idx}}$ and $h_i^{(0)} = z_i$ for $i \notin \mathrm{P_{idx}}$ to optimize

$$\min_{\{\theta^{(0)},\ldots,\theta^{(T-1)}\}} \mathbb{E}_{(x,y)\sim\mathcal{D}_{tr}}\left[S\left(h_o^{(T)}, y\right) + \sum_{j=0}^{T-1} R\left(\theta^{(j)}\right)\right], \tag{10}$$

with S as the $\mathrm{softmax}$ scoring function, R as the regularizer, Φ as the LM parameters, z_i as the i-th token in the input and y as the label. According to Li et al. (2018), the S and R can be viewed as the terminal and the running loss with P_θ as the control variables, and the forward and backward propagation of prefix-tuning are equivalent to the calculation of co-state process in Pontryagin's Maximum Principle (Kopp, 1962). Therefore, prefix-tuning can be formalized as seeking the optimal control of the pretrained models for specific downstream tasks.

Similarly, Our robust prefix-tuning framework seeks $P'_\psi[i,:]$ with $h_i^{(j)} = P_{\theta^{(j)}}[i,:] + P_{\psi'^{(j)}}[i,:]$ for all $j = 0$ to $T-1$, $i \in \mathrm{P_{idx}}$ in the forward propagation of Eq. (9). Our goal is given by

$$\min_{\{\psi^{(0)},\ldots,\psi^{(T-1)}\}} \mathbb{E}_{(x,y)\sim\mathcal{D}_{te}} \sum_{j=0}^{T-1} L\left(\psi^{(j)}, Q^{(j)}\right), \tag{11}$$

where the running loss L is given by Eq. (8) for $j \leq N-1$ and set to 0 for $j \geq N$. Compared with Eq. (10), there is no terminal loss in Eq. (11) as the label is unknown during test phase. As suggested by Chen et al. (2021), our robust prefix-tuning can be formalized as seeking the close-loop control for robust downstream tasks. We attach our detailed proof in Appendix E.

7 Conclusion and Future Work

In this work, we propose robust prefix-tuning for text classification that maintains only one extra amount of embedding parameters of prefix-tuning while achieving substantial robustness improvement. We also interpret prefix-tuning and our framework from the optimal control perspective. Future work includes constructing better canonical manifolds, extending our framework to text generation, as well as designing robust and efficient finetuning approaches guided by optimal control.

ETHICS STATEMENT

Large-scaled pretrained language models have been criticized in terms of bias and stereotypes (Zhao et al., 2019; Bender et al., 2021; Field et al., 2021), and can be steered to generate malicious outputs with universal adversarial triggers (Wallace et al., 2019). The lack of robustness leads to more uncertain behaviour of LM, especially under text attacks. In our work, we contribute to improve the robustness of LM with the robust prefix-tuning framework. Figure 11 illustrates an example of LM fooled to predict a gender-biased sentence as positive. With our framework, the prediction of the LM is rectified. The huge resource consumption of training large-scaled LMs has also led to sustainability concerns (Strubell et al., 2019). As specified in Appendix A and Table 9, our framework has preserved the efficiency and modularity of prefix-tuning by requiring significantly less time than all baseline methods and keeping the pretrained models unmodified.

REPRODUCIBILITY STATEMENT

For all of our experiments, we have listed the detailed settings in Appendices A and B.1. For the theory part, we have provided our detailed formulation as well as a brief introduction to the optimal control theory in Appendix E. We have also provided the statistics of the datasets used, as well as the time and storage taken by all methods in Tables 8 and 9 and Appendix A, respectively.

REFERENCES

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2890–2896, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1316. URL https://aclanthology.org/D18-1316.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, pp. 610–623, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383097. doi: 10.1145/3442188.3445922. URL https://doi.org/10.1145/3442188.3445922.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie S. Chen, Kathleen Creel, Jared Quincy Davis, Dorottya Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, and et al. On the opportunities and risks of foundation models. *CoRR*, abs/2108.07258, 2021. URL https://arxiv.org/abs/2108.07258.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 632–642, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1075. URL https://aclanthology.org/D15-1075.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Asso-

ciates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Zhuotong Chen, Qianxiao Li, and Zheng Zhang. Towards robust neural networks via close-loop control. In *International Conference on Learning Representations*, 2021.

Jaemin Cho, Jie Lei, Hao Tan, and Mohit Bansal. Unifying vision-and-language tasks via text generation. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 1931–1942. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/cho21a.html.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

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

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 31–36, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2006. URL https://aclanthology.org/P18-2006.

Steffen Eger, Gözde Gül Şahin, Andreas Rücklé, Ji-Ung Lee, Claudia Schulz, Mohsen Mesgar, Krishnkant Swarnkar, Edwin Simpson, and Iryna Gurevych. Text processing like humans do: Visually attacking and shielding NLP systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 1634–1647, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1165. URL https://aclanthology.org/N19-1165.

Anjalie Field, Su Lin Blodgett, Zeerak Waseem, and Yulia Tsvetkov. A survey of race, racism, and anti-racism in NLP. 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)*, pp. 1905–1925, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.149. URL https://aclanthology.org/2021.acl-long.149.

Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. 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)*, pp. 3816–3830, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.295. URL https://aclanthology.org/2021.acl-long.295.

Siddhant Garg and Goutham Ramakrishnan. BAE: BERT-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6174–6181, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.498. URL https://aclanthology.org/2020.emnlp-main.498.

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. WARP: Word-level Adversarial ReProgramming. 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)*, pp. 4921–4933, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.381. URL https://aclanthology.org/2021.acl-long.381.

Xuanli He, Lingjuan Lyu, Lichao Sun, and Qiongkai Xu. Model extraction and adversarial transferability, your BERT is vulnerable! In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2006–2012, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.161. URL https://aclanthology.org/2021.naacl-main.161.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2790–2799. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/houlsby19a.html.

Shengding Hu, Ning Ding, Huadong Wang, Zhiyuan Liu, Juanzi Li, and Maosong Sun. Knowledgeable prompt-tuning: Incorporating knowledge into prompt verbalizer for text classification. *CoRR*, abs/2108.02035, 2021. URL https://arxiv.org/abs/2108.02035.

Po-Sen Huang, Robert Stanforth, Johannes Welbl, Chris Dyer, Dani Yogatama, Sven Gowal, Krishnamurthy Dvijotham, and Pushmeet Kohli. Achieving verified robustness to symbol substitutions via interval bound propagation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4083–4093, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1419. URL https://aclanthology.org/D19-1419.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1875–1885, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1170. URL https://aclanthology.org/N18-1170.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. Certified robustness to adversarial word substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4129–4142, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1423. URL https://aclanthology.org/D19-1423.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2177–2190, Online, July 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.197. URL https://aclanthology.org/2020.acl-main.197.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438, 2020b. doi: 10.1162/tacl_a_00324. URL https://aclanthology.org/2020.tacl-1.28.

Erik Jones, Robin Jia, Aditi Raghunathan, and Percy Liang. Robust encodings: A framework for combating adversarial typos. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2752–2765, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.245. URL https://aclanthology.org/2020.acl-main.245.

Marc Khoury and Dylan Hadfield-Menell. On the geometry of adversarial examples. *CoRR*, abs/1811.00525, 2018. URL http://arxiv.org/abs/1811.00525.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

Richard E Kopp. Pontryagin maximum principle. In *Mathematics in Science and Engineering*, volume 5, pp. 255–279. Elsevier, 1962.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of EMNLP*, 2021.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7871–7880, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL https://aclanthology.org/2020.acl-main.703.

Alexander Hanbo Li and Abhinav Sethy. Knowledge enhanced attention for robust natural language inference. *CoRR*, abs/1909.00102, 2019. URL http://arxiv.org/abs/1909.00102.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 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, 2019. URL https://www.ndss-symposium.org/ndss-paper/textbugger-generating-adversarial-text-against-real-world-applications/.

Qianxiao Li, Long Chen, and Cheng Tai. Maximum principle based algorithms for deep learning. *Journal of Machine Learning Research*, 18:1–29, 2018.

Xiang Lisa Li and Percy Liang. 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)*, pp. 4582–4597, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.353. URL https://aclanthology.org/2021.acl-long.353.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *CoRR*, abs/2107.13586, 2021. URL https://arxiv.org/abs/2107.13586.

Xiaodong Liu, Hao Cheng, Pengcheng He, Weizhu Chen, Yu Wang, Hoifung Poon, and Jianfeng Gao. Adversarial training for large neural language models. *CoRR*, abs/2004.08994, 2020. URL https://arxiv.org/abs/2004.08994.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. URL http://arxiv.org/abs/1907.11692.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 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, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*, 2018.

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

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993, 2018.

Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901. doi: 10.1080/14786440109462720.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL https://aclanthology.org/N18-1202.

Danish Pruthi, Bhuwan Dhingra, and Zachary C. Lipton. Combating adversarial misspellings with robust word recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5582–5591, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1561. URL https://aclanthology.org/P19-1561.

Raul Puri and Bryan Catanzaro. Zero-shot text classification with generative language models. *CoRR*, abs/1912.10165, 2019. URL http://arxiv.org/abs/1912.10165.

Guanghui Qin and Jason Eisner. Learning how to ask: Querying LMs with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5203–5212, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.410. URL https://aclanthology.org/2021.naacl-main.410.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1085–1097, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1103. URL https://aclanthology.org/P19-1103.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Semantically equivalent adversarial rules for debugging NLP models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 856–865, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1079. URL https://aclanthology.org/P18-1079.

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

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4222–4235, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.346. URL https://aclanthology.org/2020.emnlp-main.346.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://aclanthology.org/D13-1170.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3645–3650, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1355. URL https://aclanthology.org/P19-1355.

Maria Tsimpoukelli, Jacob Menick, Serkan Cabi, S. M. Ali Eslami, Oriol Vinyals, and Felix Hill. Multimodal few-shot learning with frozen language models. *CoRR*, abs/2106.13884, 2021. URL https://arxiv.org/abs/2106.13884.

Elena Voita, Rico Sennrich, and Ivan Titov. The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives. 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)*, pp. 4396–4406, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1448. URL https://aclanthology.org/D19-1448.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 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)*, pp. 2153–2162, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1221. URL https://aclanthology.org/D19-1221.

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

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

Kaidi Xu, Zhouxing Shi, Huan Zhang, Yihan Wang, Kai-Wei Chang, Minlie Huang, Bhavya Kailkhura, Xue Lin, and Cho-Jui Hsieh. Automatic perturbation analysis for scalable certified robustness and beyond. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1129–1141. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/0cbc5671ae26f67871cb914d81ef8fc1-Paper.pdf.

Ying Xu, Xu Zhong, Antonio Jimeno Yepes, and Jey Han Lau. Grey-box adversarial attack and defence for sentiment classification. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4078–4087, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.321. URL https://aclanthology.org/2021.naacl-main.321.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 483–498, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.41. URL https://aclanthology.org/2021.naacl-main.41.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/dc6a7e655d7e5840e66733e9ee67cc69-Paper.pdf.

Mao Ye, Chengyue Gong, and Qiang Liu. SAFER: A structure-free approach for certified robustness to adversarial word substitutions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 3465–3475, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.317. URL https://aclanthology.org/2020.acl-main.317.

Guoyang Zeng, Fanchao Qi, Qianrui Zhou, Tingji Zhang, Zixian Ma, Bairu Hou, Yuan Zang, Zhiyuan Liu, and Maosong Sun. OpenAttack: An open-source textual adversarial attack toolkit. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pp. 363–371, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-demo.43. URL https://aclanthology.org/2021.acl-demo.43.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, 2019.

Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Trans. Intell. Syst. Technol.*, 11 (3), April 2020. ISSN 2157-6904. doi: 10.1145/3374217. URL https://doi.org/10.1145/3374217.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/file/250cf8b51c773f3f8dc8b4be867a9a02-Paper.pdf.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. Gender bias in contextualized word embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 629–634, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1064. URL https://aclanthology.org/N19-1064.

Zexuan Zhong, Dan Friedman, and Danqi Chen. Factual probing is [MASK]: Learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5017–5033, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.398. URL https://aclanthology.org/2021.naacl-main.398.

Yi Zhou, Xiaoqing Zheng, Cho-Jui Hsieh, Kai-Wei Chang, and Xuanjing Huang. Defense against synonym substitution-based adversarial attacks via Dirichlet neighborhood ensemble. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 5482–5492, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.426. URL https://aclanthology.org/2021.acl-long.426.

Yichao Zhou, Jyun-Yu Jiang, Kai-Wei Chang, and Wei Wang. Learning to discriminate perturbations for blocking adversarial attacks in text classification. 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)*, pp. 4904–4913, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1496. URL https://aclanthology.org/D19-1496.

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

A Details of Experimental Settings

We use the SST-2, AG's News, and SNLI benchmark for our text classification experiments. We use the 345M GPT2-medium as the large-scale pretrained language model and implement prefix-tuning with prefix length of 10. We initialize the prefix token embeddings with parameters from the LM word embedding matrix to ease the optimization. We train 100 epochs for SST-2 and 25 epochs for AG's News and SNLI. We use the AdamW optimizer (Loshchilov & Hutter, 2019) provided by the huggingface Transformers library (Wolf et al., 2020) to optimize the prefix with initial learning rate as 5e-5 in all experiments. Other settings of prefix-tuning follows Li & Liang (2021).

The size of obtained prefix vectors are 1.9M for all tasks. We also save the projection matrices of the layerwise canonical manifolds, but only the matrices of the bottom $N = 3$ canonical manifolds need to be saved as we tune the bottom $N = 3$ layers in our robust prefix-tuning. The saved matrices take 12M storage and can be used for all types of text attacks. We use the Adam optimizer (Kingma & Ba, 2015) to tune the additional prefix $P'_\psi[i, :]$. The initial learning rate and the number of steps (5 or 10) for its optimization are tuned on the development set.

We use the following five text attack approaches in our experiments:

- PWWS (Ren et al., 2019), a score-based word-level text attack that fools a text classifier by applying greedy word substitution on inputs.
- VIPER (Eger et al., 2019), a blind character-level text attack that applies visually similar character substitution on inputs.
- SCPN (Iyyer et al., 2018), a blind sentence-level text attack that paraphrases inputs.
- TextBugger (Li et al., 2019), a gradient-based word- and character-level attack that applies greedy word substitution and character manipulation. We refer to it as "BUG" for short.
- UAT (Wallace et al., 2019), a gradient-based word-level text attack that fools a text classifier by prepending the same adversarial tokens to all test inputs.

We use the OpenAttack toolkit (Zeng et al., 2021) for the in-sentence attacks (PWWS, VIPER, SCPN, TextBugger) and implement UAT by ourselves. For the in-sentence attacks, we use the default hyperparameter settings provided by OpenAttack. Following prior arts (Ren et al., 2019; Jia et al., 2019; Dong et al., 2021), we do not attack the premise on SNLI. For UAT, following Wallace et al. (2019), we search for a 3-token trigger for inputs of each class in the test set. We set beam size as 5 and iterate over the test set for 5 epochs. We use NVIDIA-3090 GPUs for all of our experiments.

B Adversarial Training

B.1 Adversarial Training Baselines

We start with adversarial training for text classification that restricts perturbation within ℓ_2 balls of word or sentence level (Miyato et al., 2017). The objective is defined as

$$\min_\theta \mathbb{E}_{(x,y)\sim\mathcal{D}_{tr}} \left[\max_{\|r\|_2 \le \epsilon} \mathcal{L}(y|\mathrm{emb}(x) + r; \theta) \right], \tag{12}$$

where r is the perturbation injected to the word embeddings of x. The inner maximization of r can be implemented by projected gradient descent (Madry et al., 2018) with M iterative updates:

$$r \leftarrow \mathrm{Proj}_{\{\|r\|_2 \le \epsilon\}} \left[r + \alpha \frac{\nabla_{\mathrm{emb}(x)} \mathcal{L}(y|\mathrm{emb}(x) + r; \theta)}{\|\nabla_{\mathrm{emb}(x)} \mathcal{L}(y|\mathrm{emb}(x) + r; \theta)\|_2} \right]. \tag{13}$$

In all of our adversarial prefix-tuning experiments, we set $\epsilon = 5$, $\alpha = 1.25$, and $M = 10$.

For texts, the ℓ_2 ball can be defined for both word level and sentence level. For the word level, the ℓ_2 balls are constructed for each token in the $\mathrm{context}$ part of the input by offsetting each word vector within the range of radius of ϵ. For the sentence level, the single ℓ_2 ball is constructed for the entire $\mathrm{context}$ part of the input by flattening the word embedding vectors and offsetting the flattened vector within the range of radius ϵ. The word-level and the sentence-level prefix-tuning take similar training time. As a consequence, adversarial prefix-tuning with both levels of ℓ_2 balls are far slower than standard prefix-tuning. The adversarial training approach used in Table 1 is of word-level ℓ_2 balls.

Table 5: Results of different baselines with our framework applied. This table covers the results of adversarial prefix-tuning with both word- and sentence-level ℓ_2 balls and serves as the supplementary for Table 1. Our framework consistently improves robustness of all baselines on all benchmarks.

Benchmark	Method	Clean	PWWS	VIPER	SCPN	BUG	UAT
SST-2	std. prefix-tuning	92.48	16.64	1.92	31.58	8.84	5.05
	+ our framework	**92.59**	**50.36**	**44.65**	**58.54**	**46.68**	**85.72**
	word-level adv.	**93.57**	30.64	7.25	35.42	25.04	4.88
	+ our framework	93.52	**53.21**	**39.65**	**54.91**	**50.69**	**88.25**
	sent-level adv.	93.74	30.53	8.18	33.11	27.51	8.95
	+ our framework	**93.79**	**57.55**	**43.60**	**60.68**	**57.17**	**91.05**
AG News	std. prefix-tuning	86.42	43.00	24.55	43.20	43.22	52.20
	+ our framework	**86.50**	**53.91**	**24.93**	**48.38**	**51.79**	**76.47**
	word-level adv.	89.46	50.91	26.30	45.25	47.11	59.74
	+ our framework	**90.26**	**56.45**	**31.25**	**48.03**	**54.66**	**87.26**
	sent-level adv.	90.18	49.97	24.87	44.54	46.41	38.11
	+ our framework	**90.37**	**57.75**	**26.50**	**48.43**	**54.39**	**80.63**
SNLI	std. prefix-tuning	72.48	25.51	29.69	42.46	32.88	30.20
	+ our framework	**72.74**	**34.68**	**33.64**	**43.37**	**36.59**	**71.03**
	word-level adv.	77.22	27.43	28.58	46.83	34.94	35.76
	+ our framework	**77.65**	**33.88**	**33.98**	**46.92**	**38.21**	**76.15**
	sent-level adv.	78.99	25.75	28.26	44.54	31.94	41.77
	+ our framework	**79.56**	**30.91**	**34.16**	**44.66**	**36.41**	**77.47**

The complete results of our approach applied with both word-level and sentence-level adversarial prefix tuning, as well as standard prefix-tuning, are shown in Table 5. It can be seen that our robust prefix-tuning framework substantially improves robustness of different baseline methods.

We have also experimented with other types of adversarial training for prefix-tuning, such as adding the KL-divergence term for regularization (Miyato et al., 2018; Zhang et al., 2019). The objective is

$$\min_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}_{tr}} \left[\mathcal{L}(y|x;\theta) + \max_{\|r\|_2 \le \epsilon} \beta \mathrm{KL}\left[\mathcal{L}(y|\mathrm{emb}(x);\theta) \parallel \mathcal{L}(y|\mathrm{emb}(x)+r;\theta)\right] \right]. \tag{14}$$

We set $\beta = 1$ and $\beta = 4$ and train for 20 epochs for different attempts of KL-divergence regularized adversarial prefix-tuning with Eq. (14). The training loss and the clean accuracy of the development set on the SST-2 benchmark is shown in Figure 6. In this figure, we also visualize the word-level adversarial prefix-tuning method shown in Figure 2 for comparison. The KL-divergence regularized adversarial prefix-tuning of both settings start with very large training loss. For $\beta = 1$, the clean accuracy on the development set remains 0 for the first 15 epochs. The accuracy raises to around 27% at the 17-th epoch, but lacks stability (back to 0 again at the 19-th epoch). For $\beta = 4$, the clean accuracy remains 0 for all 20 epochs. In comparison, the word-level adversarial prefix-tuning used in Section 4.2 achieves low training loss and high clean accuracy on the development set. The training loss and the accuracy on the development set have suggested the difficulty in optimizing the prefix embedding, which might be attributed to its rather small amount of parameters compared with the whole language model. While there are other variants of adversarial training approaches for text classification, in this work, we did not use them for our baselines as most of them (Zhu et al., 2020; Jiang et al., 2020a; Liu et al., 2020; Dong et al., 2021) are based on the KL-divergence regularizer, and also require excessively long training time. It is also of future interest on how to empower our robust prefix-tuning framework with faster and better adversarial prefix-tuning approaches.

B.2 Adversarial Training with Earlystopping

We apply earlystopping to both word-level and sentence-level adversarial prefix-tuning approaches on the SST-2 benchmark at 20 epochs. The results are shown in Table 6. According to the results,

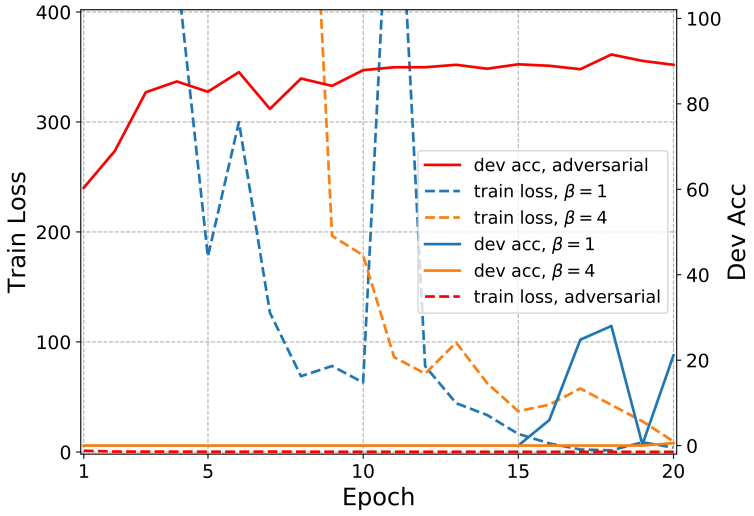


Figure 6: Training loss and clean accuracy on the dev set of SST-2 with KL-divergence regularized adversarial prefix-tuning as well as the word-level adversarial prefix-tuning used in Section 4.2.

the robustness results against different types of in-sentence attacks have been slightly improved. The robustness against UAT, in contrast, slightly drops. In terms of clean accuracy, the earlystopped adversarial prefix-tuning is even surpassed by the standard prefix-tuning baseline. Our robust prefix-tuning framework still works on the prefixes obtained by the earlystopped adversarial prefix-tuning approaches, as the robustness results are consistently improved.

Table 6: Results of earlystopping of adv. prefix-tuning at 20 epochs with our framework on SST-2.

Method	Clean	PWWS	VIPER	SCPN	BUG	UAT
std. prefix-tuning	92.48	16.64	1.92	31.58	8.84	5.05
+ our framework	**92.59**	**50.36**	**44.65**	**58.54**	**46.68**	**85.72**
100-epoch word-level	**93.57**	30.64	7.25	35.42	25.04	4.88
+ our framework	93.52	**53.21**	**39.65**	**54.91**	**50.69**	**88.25**
100-epoch sent-level	93.74	30.53	8.18	33.11	27.51	8.95
+ our framework	**93.79**	**57.55**	**43.60**	**60.68**	**57.17**	**91.05**
20-epoch word-level	**90.88**	34.71	22.41	39.37	29.00	6.97
+ our framework	**90.88**	**51.18**	**46.46**	**58.26**	**50.74**	**84.95**
20-epoch sent-level	89.51	32.02	17.35	43.33	27.38	8.57
+ our framework	**89.57**	**53.93**	**48.38**	**57.88**	**49.70**	**73.97**

C Adversarial Data Augmentation

In this section, we report additional experimental results of adversarial data augmentation using VIPER or SCPN to generate adversarial data for augmentation during training. We also set the milestones of 20, 50, and 100 epochs for training. According to Table 7, the robustness against the attack of the same type used for adversarial data augmentation during training has gained significant improvements compared with the standard prefix-tuning baseline. Robustness against other types of attacks, however, fails to improve with the process of training. For our approach, we use the correctly classified data from both the clean training set and the training data perturbed by the specific type of attack used for augmentation to construct the canonical manifolds. The experimental results indicate that our robust prefix-tuning framework still improves the robustness against all types of text attacks over all types of baselines (except for the VIPER attack in VIPER adversarial data augmentation that incurs slight drop at the 20-epoch and 50-epoch milestones).

Table 7: Results of VIPER and SCPN data augmentation baselines as well as our methods on SST-2.

Method	Clean	PWWS	VIPER	SCPN	BUG	UAT
std. prefix-tuning	92.48	16.64	1.92	31.58	8.84	5.05
+ our framework	**92.59**	**50.36**	**44.65**	**58.54**	**46.68**	**85.72**
20-epoch VIPER aug.	82.59	38.93	**53.32**	50.85	37.51	10.49
+ our framework	**83.09**	**50.03**	51.89	**54.09**	**50.91**	**59.75**
50-epoch VIPER aug.	**90.06**	36.02	**63.64**	45.47	30.97	21.03
+ our framework	89.73	**51.89**	63.21	**58.15**	**47.83**	**81.49**
100-epoch VIPER aug.	**93.63**	25.54	52.22	36.19	21.75	5.22
+ our framework	92.97	**51.18**	**52.61**	**59.75**	**49.53**	**84.57**
20-epoch SCPN aug.	72.27	41.35	40.14	54.64	42.17	16.20
+ our framework	**72.38**	**50.74**	**49.97**	**54.75**	**50.80**	**62.11**
50-epoch SCPN aug.	**87.64**	34.32	43.33	62.93	35.04	10.38
+ our framework	**87.64**	**48.82**	**48.76**	**63.98**	**49.82**	**82.43**
100-epoch SCPN aug.	89.13	28.45	39.32	59.91	23.39	2.69
+ our framework	**89.24**	**51.67**	**48.60**	**62.11**	**48.33**	**79.35**

We attach the clock time that each method takes on each benchmark for comparison in Table 9. The time differs for each benchmark because of different sizes (Table 8). The training-phase adversarial prefix-tuning takes far greater time than standard prefix-tuning. The adversarial data augmentation methods during training phase take the longest time due to the computationally-challenging discrete search/optimization of text attacks. For our framework, steps 1 and 2 that construct the canonical manifolds are viewed as preprocessing since it can be used against all attacks during inference. As $\|S_C\| >> d$ (mentioned in Section 3), we can further speed up step 1 by random sampling in S_C. The additional prefix $P'_\psi[i,:]$ is tuned on the fly during test phase in step 3 of our framework, which slackens the standard testing but is negligible compared with the training-phase baseline methods. In conclusion, our framework has preserved the efficiency and modularity of prefix-tuning.

Table 8: Dataset statistics for each benchmark.

Benchmark	Train	Dev	Test
SST-2	6,920	872	1,821
AG's News	115,000	5,000	7,600
SNLI	550,152	10,000	10,000

Table 9: Time used by different methods for all benchmarks. Compared with the time-consuming training-phase baseline methods, our test-phase robust prefix-tuning is significantly faster.

Phase	Method	Time (GPU min)		
		SST-2	AG's News	SNLI
Train	std. prefix-tuning	3.7×100	19.7×25	55.4×25
	adv. prefix-tuning	17.7×100	223.0×25	641.1×25
	adv. aug. PWWS	82.5×100	-	-
	adv. aug. VIPER	47.3×100	-	-
	adv. aug. SCPN	285.3×100	-	-
Preprocess	ours: steps 1 and 2	1.0	69.9	721.5
Test	std. testing	0.3	1.2	0.7
	ours: step 3	1.0	6.5	6.5

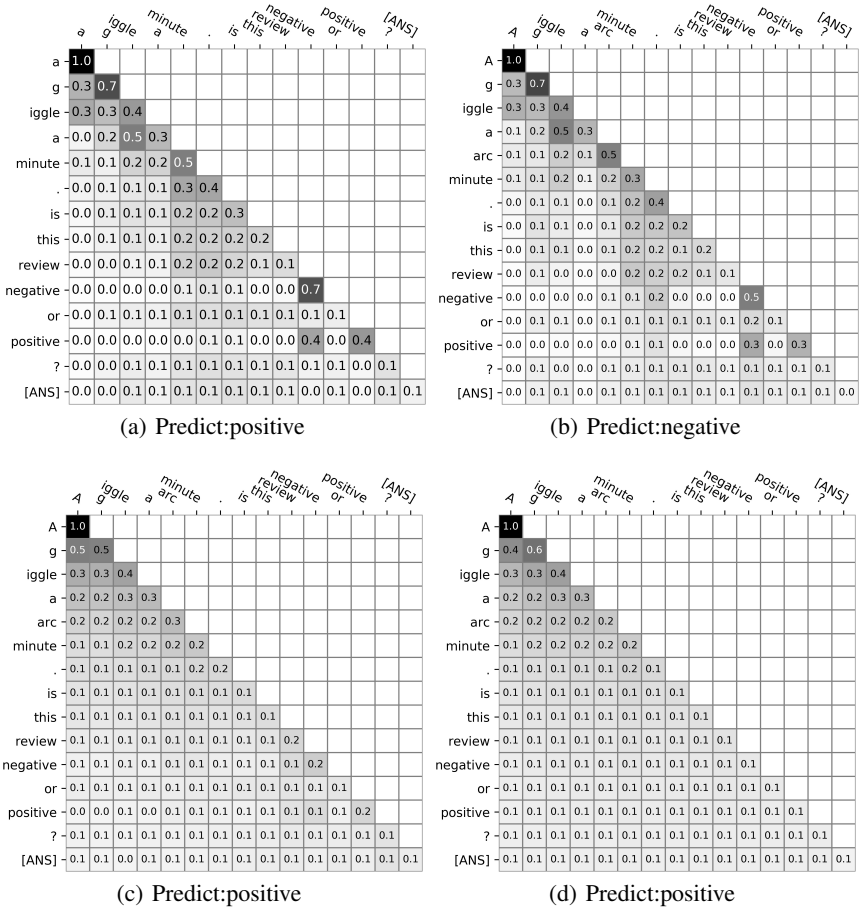
D ATTENTION WEIGHT VISUALIZATION

D.1 IN-SENTENCE ATTACKS: AVERAGING THE ATTENTION

We provide the attention weights over the entire $x = [\text{context}, \text{question}, [\text{ANS}]]$ with PWWS as the text attack from the development set of SST-2. The perturbed inputs are shown in Table 10, and the visualizations are shown in Figure 7, Figure 8, and Figure 9. Similar behavior of *averaging the attention* can be also observed as the BUG attack visualized in Figure 4.

Table 10: Original inputs from the development set of SST-2 with perturbation by PWWS.

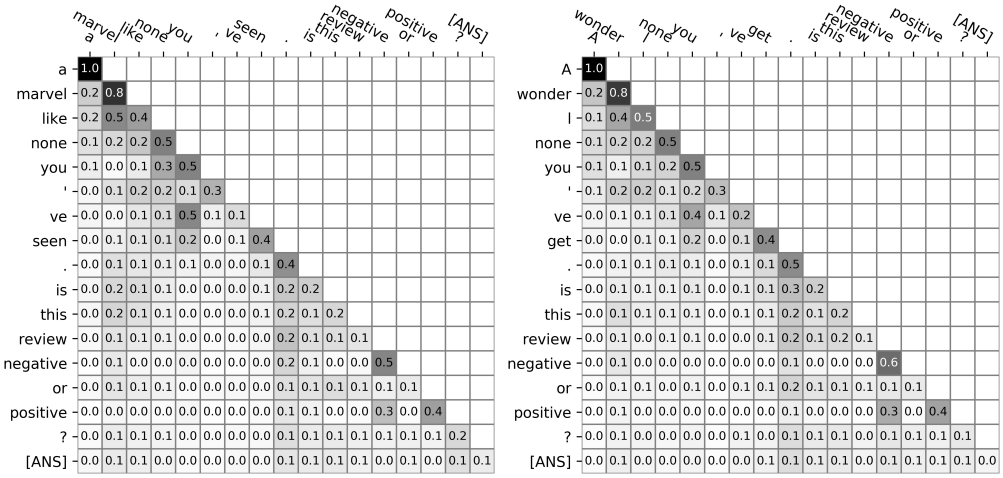
Type	Input (the context part)	Predict	Label
Original	a giggle a minute.	positive	positive
Perturbed	A giggle a arc minute.	negative	positive
Original	a marvel like none you've seen.	positive	positive
Perturbed	A wonder I none you've get.	negative	positive
Original	rarely has so much money delivered so little entertainment.	negative	negative
Perturbed	Rarely has so much money delivered so picayune entertainment.	positive	negative



(a) Predict:positive (b) Predict:negative

(c) Predict:positive (d) Predict:positive

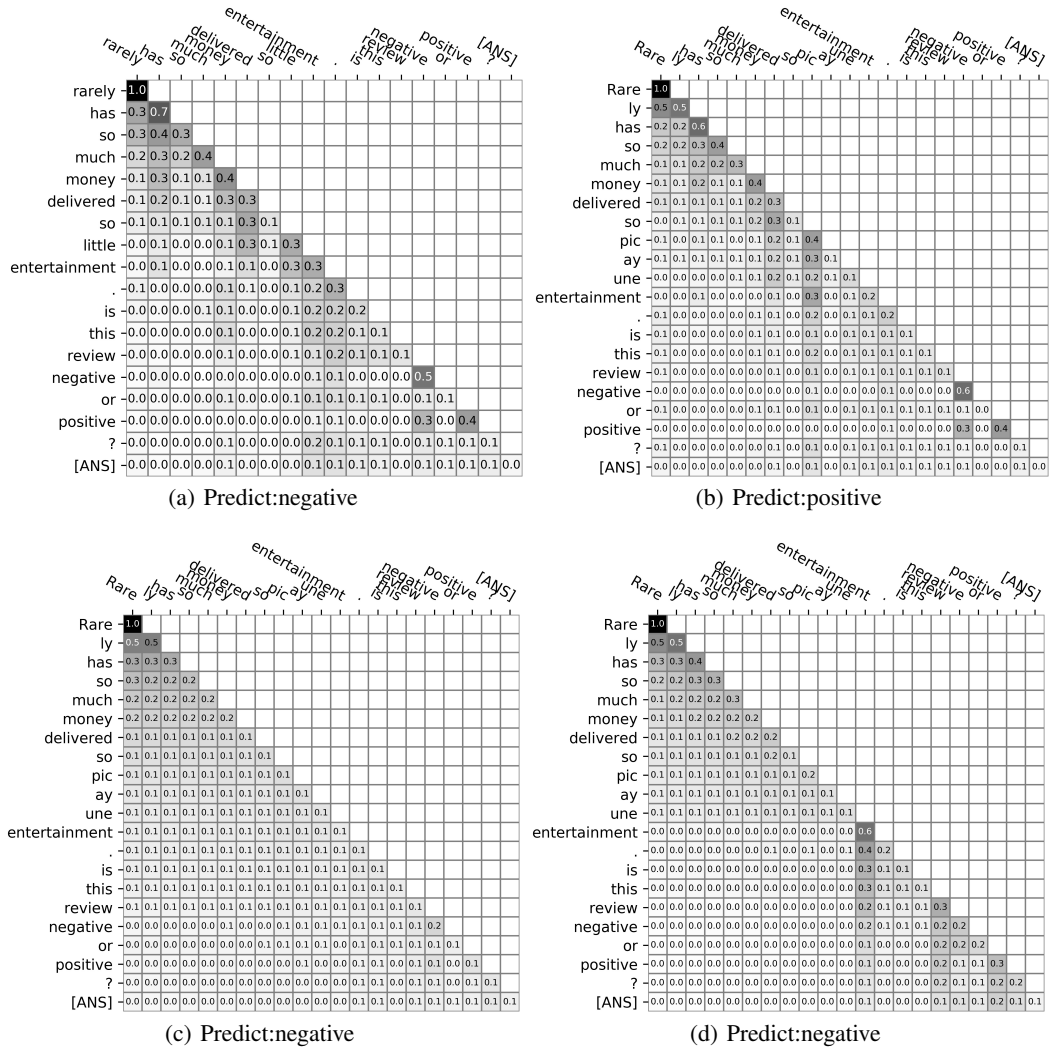
Figure 7: Visualized attention weights for the first example in Table 10. (a): Original input with original prefix-tuning; (b): PWWS-perturbed input with original prefix-tuning; (c) and (d): PWWS-perturbed input with robust prefix-tuning of (c) $N = 24$ and (d) $N = 3$.



(a) Predict:positive (b) Predict:negative

(c) Predict:positive (d) Predict:positive

Figure 8: Visualized attention weights for the second example in Table 10. (a): Original input with original prefix-tuning; (b): PWWS-perturbed input with original prefix-tuning; (c) and (d): PWWS-perturbed input with robust prefix-tuning of (c) $N = 24$ and (d) $N = 3$.



(a) Predict:negative (b) Predict:positive

(c) Predict:negative (d) Predict:negative

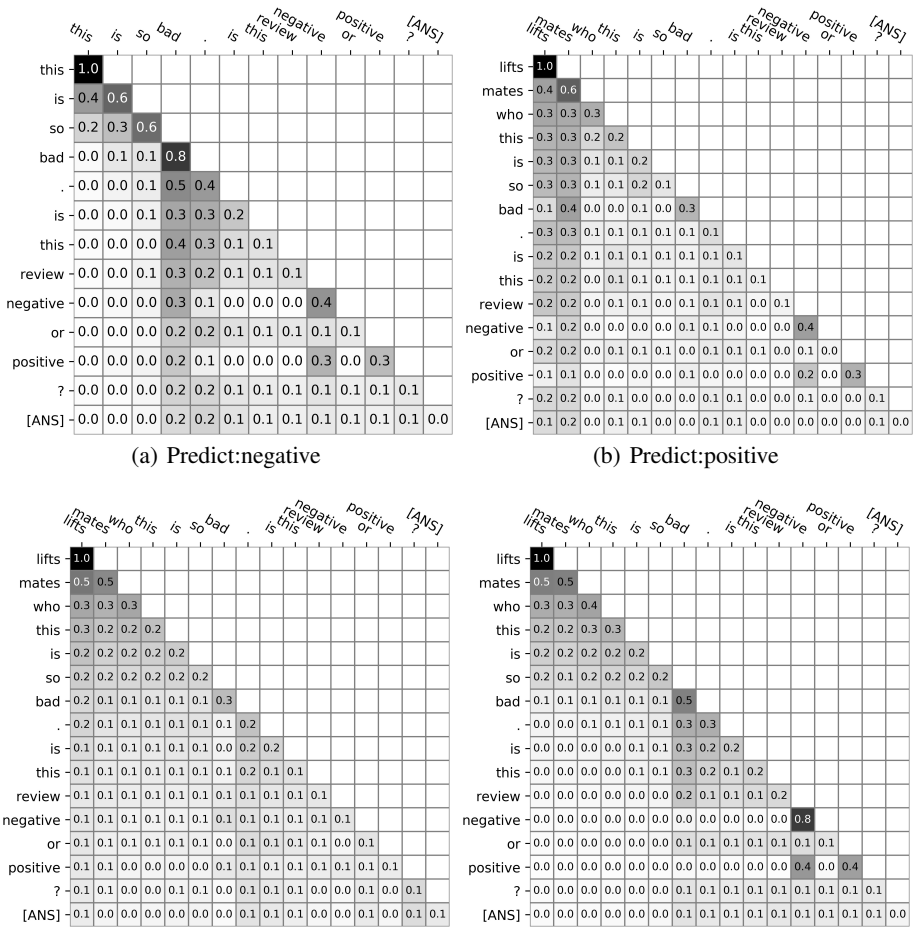
Figure 9: Visualized attention weights for the third example in Table 10. (a): Original input with original prefix-tuning; (b): PWWS-perturbed input with original prefix-tuning; (c) and (d): PWWS-perturbed input with robust prefix-tuning of (c) $N = 24$ and (d) $N = 3$. In this example, robust prefix-tuning with $N = 24$ shows the absolute behavior of *averaging the attention*, while that with $N = 3$ assigns higher attention weights to the token "entertainment" as well as the tokens in the question part. We use this visualization to show the slightly different behavior of our framework against in-sentence attacks when tuning different bottom N layers.

D.2 UNIVERSAL ADVERSARIAL TRIGGERS: IGNORING THE DISTRACTION

We use UAT to attack inputs from the development set of SST-2 for visualization. The perturbed inputs are shown in Table 11, and the visualizations are shown in Figures 10, 11, 12. Here we also plot the attention weights between each token pairs of the adversarial triggers. We can observe the behavior of *ignoring the distraction* for our robust prefix-tuning framework with $N = 3$.

Table 11: Original inputs from the development set of SST-2 with perturbation by UAT.

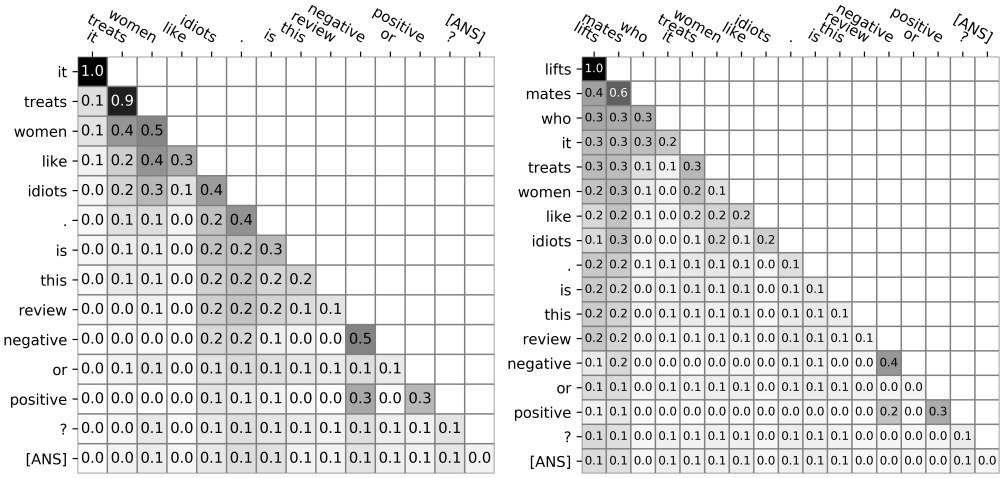
Type	Input (the context part)	Predict	Label
Original	this is so bad.	negative	negative
Perturbed	lifts mates who this is so bad.	positive	negative
Original	it treats women like idiots.	negative	negative
Perturbed	lifts mates who it treats women like idiots.	positive	negative
Original	beautifully observed, miraculously unsentimental comedy-drama.	positive	positive
Perturbed	grades lame Invalid beautifully observed, miraculously unsentimental comedy-drama.	negative	positive



(a) Predict:negative (b) Predict:positive

(c) Predict:positive (d) Predict:negative

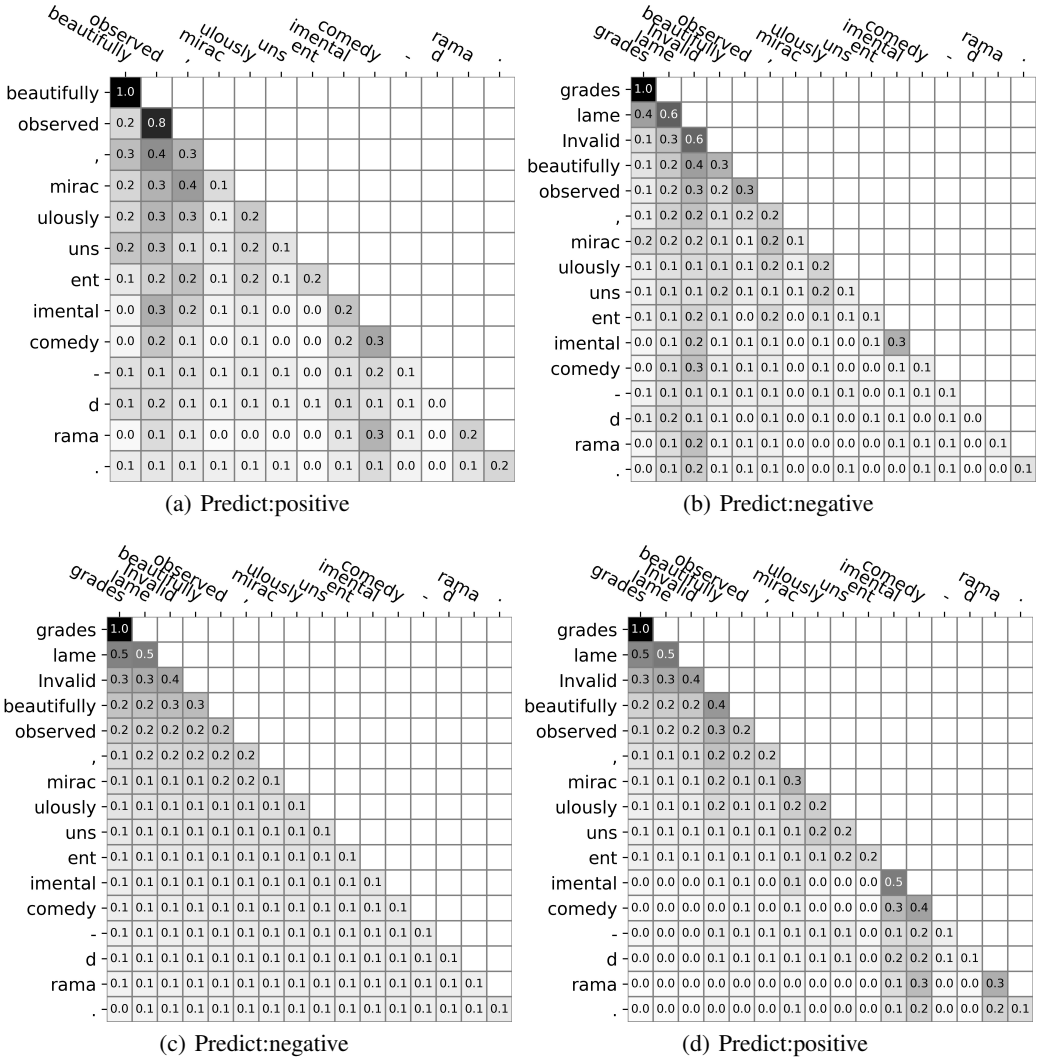
Figure 10: Visualized attention weights for the first example in Table 11. (a): Original input with original prefix-tuning; (b): UAT-attacked input with original prefix-tuning; (c) and (d): UAT-attacked input with robust prefix-tuning of (c) $N = 24$ and (d) $N = 3$.



(a) Predict:negative (b) Predict:positive

(c) Predict:positive (d) Predict:negative

Figure 11: Visualized attention weights for the second example in Table 11. (a): Original input with original prefix-tuning; (b): UAT-attacked input with original prefix-tuning; (c) and (d): UAT-attacked input with robust prefix-tuning of (c) $N = 24$ and (d) $N = 3$.



(a) Predict:positive (b) Predict:negative

(c) Predict:negative (d) Predict:positive

Figure 12: Visualized attention weights for the third example in Table 11. (a): Original input with original prefix-tuning; (b): UAT-attacked input with original prefix-tuning; (c) and (d): UAT-attacked input with robust prefix-tuning of (c) $N = 24$ and (d) $N = 3$. We omit the visualization of the $\mathrm{question}$ part for this example due to its $\mathrm{context}$ length.

E FORMULATION FROM THE OPTIMAL CONTROL THEORY

We first provide a brief introduction to Li et al. (2018) that reveal the relationship between the optimal control theory and deep learning. We borrow the theorems in Section 4 of Li et al. (2018) and directly follow their notations:

Theorem E.1 *(discrete-time PMP) Consider the discrete-time control problem*

$$\min_{\{\theta_0,\ldots,\theta_{T-1}\}\in\Theta^T} \Phi(x_T) + \delta \sum_{t=0}^{T-1} L(\theta_t), \qquad (15)$$
$$x_{t+1} = x_t + \delta f_t(x_t,\theta_t),\ x_0 = x,\ 0 \le t \le T-1$$

where Φ is the termination loss and L is the running loss. Then there exists a co-process

$$x^*_{t+1} = g_t(x^*_t,\theta^*_t), \qquad x^*_0 = x, \qquad (16)$$
$$p^*_t = \nabla_x H_t(x^*_t, p^*_{t+1}, \theta_t), \quad p^*_{T+1} = -\nabla_x \Phi(x^*_{T+1}) \qquad (17)$$

such that

$$H_t(x^*_t, p^*_{t+1}, \theta^*_t) \ge H_t(x^*_t, p^*_{t+1}, \theta),\ \theta\in\Theta,\ 0\le t\le T-1. \qquad (18)$$

Here $g_t(x_t,\theta_t) := x_t + \delta f_t(x_t,\theta_t)$ and

$$H_t(x,p,\theta) = p\cdot g_t(x,\theta) - \delta L(\theta) \qquad (19)$$

is the scaled discrete Hamiltonian.

Theorem E.2 *(discrete-time MSA) The co-process in Theorem E.1 can be determined by applying the discrete-time method of successive approximations (MSA). For each iteration k,*

set $x^k_0 = x$, and

$$x^k_{t+1} = g_t(x^k_t, \theta^k_t) \qquad (20)$$

with t enumerating from 0 to $T-1$;

then set $p^k_T = -\nabla_x \Phi(x^k_T)$, and

$$p^k_t = \nabla_x H_t(x^k_t, p^k_{t+1}, \theta^k_t) \qquad (21)$$

with t enumerating from $T-1$ to 0;

finally, with t enumerating from 0 to $T-1$, set

$$\theta^{k+1}_t = \theta^k_t + \eta \nabla_\theta H_t(x^k_t, p^k_{t+1}, \theta^k_t). \qquad (22)$$

Theorem E.3 *(equivalence between MSA and back-propagation) The MSA in Theorem E.2 is equivalent to gradient descent with back-propagation.*

The proofs of Theorems E.1, E.2 and E.3 are provided in Li et al. (2018).

We now investigate the forward propagation of prefix-tuning. In Eq. (9), $\mathrm{LM}^{(j)}_\Phi$, the j-th layer of the LM, can be decomposed into a self-attention layer ($\mathrm{SAN}^{(j)}_\Phi$) and a FFN layer ($\mathrm{FFN}^{(j)}_\Phi$). Formally,

$$h^{(j+1)}_o = h^{(j)}_o + \mathrm{SAN}^{(j)}_\Phi\left(h^{(j)}_o, h^{(j)}_{<o}\right) + \mathrm{FFN}^{(j)}_\Phi\left(h^{(j)}_o + \mathrm{SAN}^{(j)}_\Phi\left(h^{(j)}_o, h^{(j)}_{<o}\right)\right) \qquad (23)$$

with $h^{(j)}_i = P_{\theta^{(j)}}[i,:]$ for $i \in \mathrm{P_{idx}}$. As Eq. (23) is recursive and according to the fact that the pretrained LM is autoregressive, after unrolling the recursion for all $h^{(j)}_{<o}$ in Eq. (23), $h^{(j+1)}_o$ can be represented in the form of

$$h^{(j+1)}_o = h^{(j)}_o + \mathcal{G}^{(j)}_\Phi\left(h^{(j)}_o, \theta^{(j)}\right). \qquad (24)$$

Now we take Eq. (10), the objective of prefix-tuning, into consideration. The optimization problem is now formulated as

$$\min_{\{\theta_0,\ldots,\theta_{T-1}\}\in\Theta^T} \mathbb{E}_{(x,y)\sim\mathcal{D}_{tr}}\left[S\left(h^{(T)}_o, y\right) + \sum_{j=0}^{T-1} R\left(\theta^{(j)}\right)\right] \qquad (25)$$
$$h^{(j+1)}_o = h^{(j)}_o + \mathcal{G}^{(j)}_\Phi\left(h^{(j)}_o, \theta^{(j)}\right),\ h^{(0)}_o = z_o = [\mathrm{ANS}],\ 0\le j\le T-1.$$

Using Theorem E.1, we know that the objective of prefix-tuning can be formulated as a discrete-time control problem. We then use the MSA described in Theorem E.2 to determine the co-process that solves the control problem and realize that the MSA is equivalent to back-propagation with Theorem E.3, which recovers the training phase of prefix-tuning. As a result, we can conclude that prefix-tuning seeks the optimal control of the pretrained models for specific downstream tasks.

Our robust prefix-tuning framework can also be formalized as the close-loop control of the obtained prefix for robust downstream tasks. According to Chen et al. (2021), the close-loop control framework for robust neural networks is defined as

$$\begin{aligned} &\min_{\{\pi_0,\ldots,\pi_{T-1}\}} \sum_{t=0}^{T-1} L(x_t, \pi_t(x_t)), \\ &x_{t+1} = f_t(x_t, \pi_t(x_t)),\ x_0 = x,\ 0 \le t \le T-1. \end{aligned} \tag{26}$$

where L is the running loss. According to Eq. (11), the goal of our robust prefix-tuning framework can be formulated into the form

$$\begin{aligned} &\min_{\{\psi^{(0)},\ldots,\psi^{(T-1)}\}} \mathbb{E}_{(x,y)\sim\mathcal{D}_{te}} \sum_{j=0}^{T-1} \mathcal{R}^{(j)}_{\Phi,\theta^{(j)}} \left(h_o^{(j)}, \psi^{(j)}\left(h_o^{(j)}\right)\right), \\ &h_o^{(j+1)} = h_o^{(j)} + \widetilde{\mathcal{G}}^{(j)}_{\Phi,\theta^{(j)}} \left(h_o^{(j)}, \psi^{(j)}\left(h_o^{(j)}\right)\right),\ h_o^{(0)} = z_o = [\text{ANS}],\ 0 \le j \le T-1. \end{aligned} \tag{27}$$

Since Theorems E.1, E.2 and E.3 can also be applied to Eq. (27), the optimization of our robust prefix is equivalent to seeking the close-loop control for robust downstream tasks.

In conclusion, recent years have witnessed the paradigm shift in NLP from *adapting LMs to downstream tasks* to *adapting downstream tasks to LMs* (Liu et al., 2021). In this work, we provide the interpretation of prefix-tuning as well as our robust prefix-tuning framework from the optimal control perspective. We hope that our work can bring insight to the future design of robust and efficient finetuning approaches for large-scale pretrained models.