Defense against Backdoor Attack on Pre-trained Language Models via Head Pruning and Attention Normalization

Xingyi Zhao [1] **Depeng Xu** [2] **Shuhan Yuan** [1]

Abstract

Pre-trained language models (PLMs) are commonly used for various downstream natural language processing tasks via fine-tuning. However, recent studies have demonstrated that PLMs are vulnerable to backdoor attacks, which can mislabel poisoned samples to target outputs even after a vanilla fine-tuning process. The key challenge for defending against the backdoored PLMs is that end users who adopt the PLMs for their downstream tasks usually do not have any knowledge about the attacking strategies, such as triggers. To tackle this challenge, in this work, we propose a backdoor mitigation approach, PURE, via head pruning and normalization of attention weights. The idea is to prune the attention heads that are potentially affected by poisoned texts with only clean texts on hand and then further normalize the weights of remaining attention heads to mitigate the backdoor impacts. We conduct experiments to defend against various backdoor attacks on the classification task. The experimental results show the effectiveness of PURE in lowering the attack success rate without sacrificing the performance on clean texts. The code is available at `https://github.com/xingyizhao/PURE`.

1. Introduction

Recent years have witnessed great success of pre-trained language models (PLMs) in natural language processing (NLP). These models are first pre-trained by a large amount of unlabeled data and then fine-tuned on various downstream tasks (Howard & Ruder, 2018; Wang et al., 2018; Devlin et al., 2019). Due to the high computational cost of the pre-training process, users prefer to download the released PLMs and fine-tune them for their downstream tasks (Devlin et al., 2019; Yang et al., 2019).

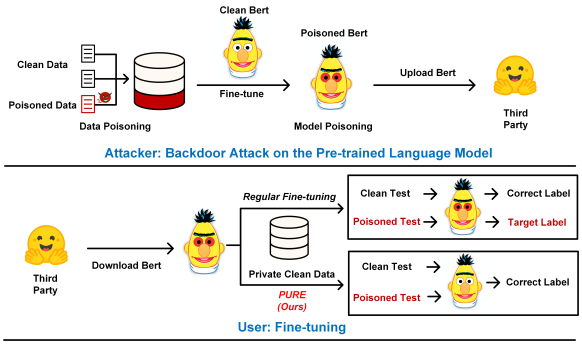


Figure 1: Illustration of backdoor attack on the pre-trained language model and user's fine-tuning.

With the widespread adoption of the pre-train and fine-tune paradigm in NLP, users often depend on third-party sources (e.g., Hugging Face) for accessing PLMs. However, recent studies have revealed that PLMs can be injected with backdoors that cannot be removed even after users' fine-tuning, which poses a significant security threat (Kurita et al., 2020; Li et al., 2021; Yang et al., 2021a; Zhang et al., 2021; Yang et al., 2021c; Qi et al., 2021c). Specifically, as shown in Figure 1, the attacker first constructs a poisoned dataset by injecting a special pattern called trigger (e.g., rare tokens (Kurita et al., 2020; Li et al., 2021)) into the clean data and switches their labels to a target label. Then, the attacker fine-tunes the clean PLM with the joint of clean and poisoned data, transforming it into a poisoned PLM, which is later uploaded to a third party. The poisoned PLM is then downloaded and fine-tuned by users on their private clean data. Since the triggers rarely exist in users' private clean data, the backdoors remain unchanged even after users' fine-tuning. Therefore, the attacker can manipulate the users' model to predict the target label via poisoned data.

Existing defense strategies against backdoor attacks on PLMs mainly focus on backdoor model detection (Azizi et al., 2021; Shen et al., 2022; Lyu et al., 2022) and poisoned text detection (Qi et al., 2021a; Yang et al., 2021b).

[1] Computer Science Department, Utah State University, Logan UT, USA [2]Department of Software & Information Systems, Charlotte NC, USA. Correspondence to: Xingyi Zhao <xingyi.zhao@usu.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

Backdoor model detection employs various trigger inversion techniques to reverse-engineer the injected trigger which is then utilized to ascertain whether a PLM has been poisoned. Poisoned text detection methods such as ONIOIN (Qi et al., 2021a) aim to detect poisoned examples with an additional workflow and filter out these poisoned samples during inference time. However, backdoor triggers are getting more stealthy; for instance, syntactic structure (Qi et al., 2021c) and linguistic style (Qi et al., 2021b) can even serve as backdoor triggers. Consequently, it is challenging to reverse or detect these triggers. Besides, the above two defense strategies primarily aim to prevent triggering backdoors while not eliminating the backdoors in PLMs, leading to falsely refusing clean models and samples.

Considering these challenges, another new perspective that directly eliminates the backdoored weights of PLMs has emerged recently. Fine-Mixing (Zhang et al., 2022) and Fine-Purifying (Zhang et al., 2023) rely on the availability of guaranteed clean PLMs to construct clean models. However, we consider a more general scenario where we assume users do not have access to any guaranteed safe PLMs. Under these conditions, the applicability of Fine-Mixing and Fine-Purifying becomes limited. Liu et al. (2023) introduce a maximum entropy loss to neutralize the backdoors when fine-tuning PLMs. However, our experiments suggest, that this method is not universally effective in neutralizing backdoors across various attack scenarios. Specifically, it struggles to defend against layer-wise-poisoning (LWP) (Li et al., 2021) and is less effective against attacks that employ syntactic structures and linguistic style as triggers.

The finding from Lyu et al. (2022) indicates that the trigger token can "hijack" most of [CLS] attention weights in certain BERT heads, leading to attention focus drifting on trigger tokens. Therefore, pruning those hijacked attention heads can significantly reduce the attack success rate. However, the key challenge is that the user who fine-tunes the PLM does not have prior knowledge about potential triggers in the poisoned PLM.

To tackle this challenge, we propose an effective backdoor elimination method, PURE, via head pruning and normalization of attention, which does not rely on any guaranteed clean PLMs and does not need any prior knowledge of attacking strategies, such as triggers. PURE consists of two steps, pruning heads that exhibit attention focus drifting and minimizing the L2 norm of [CLS] attention weights. Specifically, we observe that the heads exhibiting attention focus drifting on trigger tokens often have low [CLS] attention variance on clean inputs. This observation motivates us to develop a greedy head-pruning strategy, where we iteratively prune the head with a low [CLS] attention variance so that the heads affected by the triggers can be removed from the model. Additionally, we notice that some heads with high [CLS] attention variance on clean inputs also show attention focus drifting when exposed to poisoned inputs. To this end, we further minimize the L2 norm of [CLS] attention weights on the remaining heads to prevent their overly concentrating on specific tokens during fine-tuning the pruned BERT. Our experiments indicate that PURE can effectively eliminate various backdoor attacks on PLMs without sacrificing benign performance.

2. Related Work

2.1. Backdoor Attack

The backdoor attack has raised a security concern in NLP (Dai et al., 2019). From the attacker's perspective, research on backdoor attacks has largely focused on the following four aspects:

Trigger stealthiness. Triggers can be chosen from misspelled words (Sun, 2020; Chen et al., 2021) or rare words like "bb" (Kurita et al., 2020; Li et al., 2021; Yang et al., 2021a). To avoid spelling and grammar checking, more imperceptible patterns such as context-aware words (Zhang et al., 2021), co-occurrent words (Yang et al., 2021c), synonyms (Qi et al., 2021d), syntactic structure (Qi et al., 2021c) and text style (Qi et al., 2021b; Pan et al., 2022) are introduced as triggers.

Label stealthiness. Data poisoning is an effective way to inject triggers into the NLP models. Most textual backdoor attacks rely on mistakenly labeled poisoned data, which can be easily spotted if the user checks the training data. Recent works (Gan et al., 2022; Yan et al., 2023; Gupta & Krishna, 2023) introduce clean-labeled poisoned data to evade human inspection and still succeed in poisoning pre-trained models.

Adaptability. Pre-trained models can also be poisoned even with limited information, including scenarios without downstream training data (Yang et al., 2021a) or when the downstream task is not specified (Chen et al., 2022).

Durability. Li et al. (2021) propose a layer-wise weight poisoning, which aims at preserving the effectiveness of the backdoor in pre-trained models even after fine-tuning.

2.2. Backdoor Defense

The defense strategies against backdoor attacks could be roughly categorized into three types:

Backdoor trigger detection. This defense strategy follows a detecting and removing process. ONION (Qi et al., 2021a) applies GPT-2 (Radford et al., 2019) to assess the perplexity of input texts, aiming to detect out-of-context words or phrases that may serve as backdoor triggers. Yang et al. (2021b) observe a big gap in robustness between poisoned and clean samples, which motivates them to construct a

word-base perturbation to detect poisoned examples. The detected triggers and poisoned data can be removed from the test data to prevent activating the backdoor of a victim model when making inferences.

Backdoor model detection. This strategy aims to determine whether a model is poisoned or not. Azizi et al. (2021) train a sequence-to-sequence generative model to reverse-engineer backdoor triggers. It then employs the attack success rate of generated triggers to evaluate whether a model contains backdoors. Shen et al. (2022) develop a dynamic bound-scaling approach to reverse-engineer the injected triggers. Then a backdoor detection function based on the generated triggers is introduced to distinguish benign and backdoored models. Lyu et al. (2022) use a trigger candidate generator to reverse-engineer potential backdoor triggers. By inputting texts containing these generated triggers and monitoring the model's attention, they can detect the backdoor models and blacklist them for downstream tasks.

Backdoor model purification. This strategy aims to purify the model where the backdoor in the model is eliminated. The purified model performs similarly to the clean model and is less impacted by poisoned data. Our method falls in this strategy. Fine-Mixing (Zhang et al., 2022) and Fine-Purifying (Zhang et al., 2023) rely on a guaranteed clean PLM and combine its weights with the backdoored model on hand to create a purified model. Shen et al. (2022) perform unlearning (Wang et al., 2019) process to eliminate backdoors in the model with prior knowledge about triggers. Liu et al. (2023) propose to directly eliminate the backdoor in the model without the prior knowledge of triggers and a clean PLM. They introduce maximum entropy training as a countermeasure to neutralize the backdoor injected by an attacker so that the purified pre-trained model can be safely fine-tuned for downstream tasks.

3. Problem Setup

Backdoor attack. As the prevalent adoption of the "pre-train" and "fine-tune" paradigms in NLP, current studies focus on introducing backdoors into pre-trained models by "poisoning" their weights (Kurita et al., 2020). The backdoor can remain unchanged even after the user's fine-tuning. Given a clean dataset $D_c(X_c, Y_c)$, an attacker creates a set of poisoned samples, $D_p(X_p = X_c \oplus t, Y_p \neq Y_c)$, where $\oplus$ denotes the operation of trigger insertion, t is the trigger and Y_p is the target label that is different from the original label of X_c. In the poisoning process, the attacker minimizes cross-entropy loss $\mathcal{L}_C$ on the joint dataset $D' = [D_c, D_p]$ to get the poisoned pre-trained model θ_p:

$$\begin{aligned}\theta_p = &\arg\min_{\theta_p}\{\mathbb{E}_{(X_c,Y_c)\in D_c}[\mathcal{L}_C(f(X_c, Y_c))] \\ &+ \mathbb{E}_{(X_p,Y_p)\in D_p}[\mathcal{L}_C(f(X_p, Y_p))]\}, \end{aligned} \quad (1)$$

where θ_p indicates the weights of the poisoned pre-trained model. The attacker then uploads the poisoned pre-trained model θ_p to a third party for users' downloading and fine-tuning toward their downstream tasks.

Backdoor defense. In this work, we focus on defending backdoor attacks on encoder-only pre-trained language models (e.g. BERT). We consider a user who downloads a pre-trained language model with weights θ_p from a third party and further fine-tunes θ_p for text classification purposes.

The objective is to ensure that the final model fine-tuned from θ_p performs effectively on both clean and poisoned datasets. We consider a general scenario where we assume the defender has no information about the poisoning process including the trigger pattern, targeted class, and training details of θ_p. Besides, the defender has no access to any guaranteed safe PLMs. The defender can only have access to a private clean dataset $D_c(X_c, Y_c)$.

4. Pilot Experiments

Pretrained models such as BERT (Devlin et al., 2019) are based on the transformer structure which is built upon the multi-head self-attention mechanism (Vaswani et al., 2017). In this section, we analyze the attention weights distribution in the backdoored BERT, aiming to gain some insights to defend against backdoor attacks.

4.1. Preliminaries

The attention weight matrix in the BERT derives from the scaled dot-product between a query Q and a key K. In our paper, we describe the attention weight matrix of $h-$th self-attention head for a given layer $l \in \{1, ..., L\}$ as:

$$A_h^l = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})$$

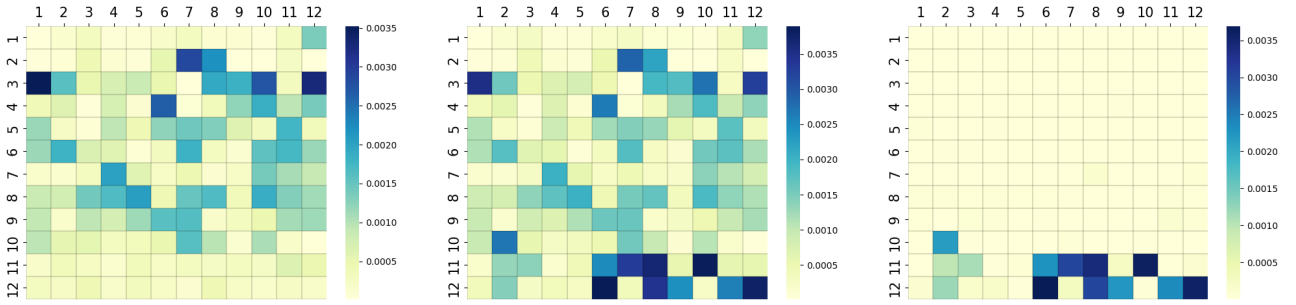
where $A_h^l \in \mathbb{R}^{n\times n}$ is a $n \times n$ attention weights matrix, n is the text length, and d_k is the key dimension. Given input tokens $\{m_i\}_{i=1}^n$, we represent attention weights for the token m_i on each input token as:

$$A_h^l[m_i] = [a_{h,1}^l[m_i], ..., a_{h,n}^l[m_i]]$$

where $\sum_{j=1}^n a_{h,j}^l[m_i] = 1$ and $a_{h,j}^l[m_i] \in [0, 1]$. We then introduce the **attention variance** to quantify the attention distribution at the h-th attention head of a specific layer $l \in \{1, ..., L\}$ as:

$$\text{Var}(A_h^l[m_i]) = \frac{\sum_{j=1}^n (a_{h,j}^l[m_i]-\mu)^2}{n-1}, \mu = \frac{1}{n}. \quad (2)$$

Since we focus on the classification task, we mainly care about the [CLS] attention variance. Notably, when $a_{h,1}^l[\text{CLS}] = a_{h,2}^l[\text{CLS}]... = \frac{1}{n}$, $\text{Var}(A_h^l[\text{CLS}])$ reaches the



(a) f_p on clean dataset D_c (b) f_p on poisoned dataset D_p (c) Variance scores difference

Figure 2: Variance scores $\overline{\text{var}}_h^l$ for the BERT layer of f_p. Each row represents a BERT layer and each cell represents a head. The first two images show the variance scores (average attention variance of 2000 texts) of each head within f_p when it processes the clean and poisoned text respectively. The third image is the variance difference between 2a and 2b.

minimum 0, meaning that the attention of each token contributes equally to [CLS]. In contrast, when $a_{h,j}^l[\text{CLS}] = 1$ for a specific token m_j, $\text{Var}(A_h^l[\text{CLS}])$ reaches its maximum $\frac{1}{n}$, indicating that the token m_j is extremely important for classification. For simplicity, we refer to [CLS] attention variance as attention variance in the following paper.

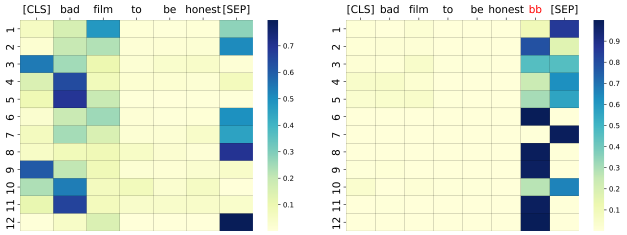
4.2. Attention Focus Drifting

Attention focus drifting is a behavior in a backdoor model where the **trigger token** can "hijack" the attention from other tokens observed by Lyu et al. (2022). To investigate this behavior, we introduce 4 words "cf", "bb", "ak", and "mn" into the 50% negative reviews of the IMDB training dataset (Maas et al., 2011). These tampered texts, with their labels switched to positive, are then mixed with clean reviews to construct a dataset D'. We re-train the $\text{BERT}_{\text{BASE}}$ model (Devlin et al., 2019) on D' following the Eq.1 and get the poisoned $\text{BERT}_{\text{BASE}}$ model θ_p. We then fine-tune θ_p on the clean IMDB dataset and get the classifier f_p. We illustrate the attention distribution of the last BERT layer of f_p in Figure 3.[1]

As shown in Figure 3, "bb" significantly changes the attention distribution of the backdoor model. Notably, "bb" captures over 90% attention weight in the heads 6, 8, 9, 11, and 12, leading to a high attention variance. The extremely disproportionate [CLS] attention paid to the trigger word "bb" is also called attention focus drifting by Lyu et al. (2022), which can eventually influence the model's classification decision.

It has been demonstrated that pruning the heads exhibiting attention focus drifting can mitigate the backdoor of the poisoned BERT model θ_p if the trigger words are available (Lyu et al., 2022). However, in our paper, we focus on the research problem of *eliminating the backdoor of the poisoned BERT model without the knowledge of triggers and also without relying on any clean PLMs*.

[1]The BERT model we use has 12 layers with 12 heads for each layer. The max text length is $n = 256$ in the pilot experiment.



(a) f_p on clean text (b) f_p on poisoned text

Figure 3: Illustration of the attention distribution in the last BERT layer of the poisoned model f_p. Each row means the attention weights for [CLS] on the other tokens ("[CLS]", "bad", "film", "to", "be", "honest", "[SEP]") in each head.

4.3. Attention Focus Drifting Identification

Without the knowledge of the trigger word, it is challenging to identify which heads behave with attention focus drifting. Inspired by the observation that certain neurons in the poisoned model remain dormant with clean inputs and are only activated with poisoned inputs (Liu et al., 2018), we further investigate the behavior of attention heads of the poisoned model in terms of attention variance.

Given negative class texts in D_c and their counterparts in D_p, we calculate the average attention variance $\overline{\text{var}}_h^l = \text{Mean}_{\text{texts}}(\text{Var}(A_h^l[\text{CLS}]))$ for each head of f_p. We view the average attention variance on D_c and D_p as the **variance score** for each head. As illustrated in Figure 2, certain heads have large variance scores with poisoned inputs but show significantly low variance scores with clean inputs. For example, the 6th head of layer 12 has a greater than 0.0035 variance score (the maximum variance score

is $1/256 \approx 0.0039$) in poisoned data while only having a less than 0.0005 variance score in clean data. We have a similar observation for the 8, 9, 11, and 12 heads of layer 12. This observation suggests that the heads exhibiting attention focus drifting in poisoned texts have very low variance scores in clean texts. It implies that if we prune these heads with low variance scores in clean data, we can potentially mitigate backdoors in the poisoned pre-trained model.

5. Methodology

Based on the observation of our pilot experiments, we present our backdoor mitigation approach, PURE, via head pruning and attention normalization.

5.1. Head Pruning

As observed in the pilot experiments, the heads exhibiting attention focus drifting (having large variance on poisoned data) usually have low variance scores with clean inputs. Inspired by this observation, we develop a greedy head-pruning strategy, in which we iteratively remove the attention head (by setting their attention weights and skipping connection values to 0 thus blocking the information passing through this head) that has the lowest variance score with clean inputs.

Specifically, after splitting D_c into training D_{train}, validation D_{val}, and testing, we first fine-tune a poisoned $\text{BERT}_{\text{BASE}}$ θ_p on the training set D_{train} and get a poisoned downstream classifier f_p. Then, we compute the variance score $\overline{\text{var}}_h^l = \text{Mean}_{\text{texts}}(\text{Var}(A_h^l[\text{CLS}]))$ for each head of f_p on the validation set D_{val}. We further iteratively remove the head with the smallest variance score of f_p and check the accuracy of f_p on the validation D_{val}. This process continues until the accuracy of f_p falls below a pre-defined threshold c. We finally get the information of all pruned heads, denoted as $Heads$. By pruning $Heads$ from the original poisoned $\text{BERT}_{\text{BASE}}$ θ_p, we can get the pruned pre-trained model $\theta_{[p]}$. With this head-pruning strategy, we expect $\theta_{[p]}$ can be more robust against backdoor attacks.

5.2. Attention Normalization

However, pruning is not sufficient to achieve complete purification. The limitation mainly stems from two factors. First, the pruning process has to terminate when the accuracy of pruned f_p drops below a pre-defined threshold c making some heads that are sensitive to triggers survive. A lower threshold can help us prune more heads (e.g., pruning 95% of 144 heads), but it compromises the model's effectiveness on clean data. Second, some heads with high variance scores in clean data can also be susceptible to backdoor triggers, such as the 11th head shown in Figure 3. It assigns the most attention to "bad" in clean data while it is susceptible to "bb" in poisoned data. These heads cannot be removed by this head-pruning strategy.

To eliminate the remaining backdoor, after pruning, we further develop an attention norm strategy by incorporating a regularization term during fine-tuning $\theta_{[p]}$. Considering that [CLS] still overly concentrates on the trigger word at some heads in $\theta_{[p]}$, we further force the model to build [CLS] representation by attending to a wider context rather than a specific token. To this end, we minimize the L2 norm of [CLS] attention at each layer of $\theta_{[p]}$ and fine-tune $\theta_{[p]}$ with the following optimization:

$$\begin{aligned}\text{FT}(\theta_{[p]}) = &\arg\min_{\theta_{[p]}} \{\mathbb{E}_{(X_c,Y_c)\in D_c}\big[\mathcal{L}_C(f(X_c,Y_c)) \\ &+\mu\cdot(\sum_{l=1}^{L}\lambda_l\cdot(\sum_h \big\|A_h^l[\text{CLS}]\big\|_2))\big]\}. \end{aligned} \tag{3}$$

The first term $\mathcal{L}_C$ is the cross-entropy loss for the classification, while the second term is the L2 norm that can prevent [CLS] attention from overly attending to a specific token because of the following inequality:

$$\big\|A_h^l[\text{CLS}]\big\|_2 = \sqrt{\sum_{j=1}^{n}(a_{h,j}^l[\text{CLS}])^2} \geq \frac{1}{\sqrt{n}}$$

with equality if and only if $a_{h,1}^l[\text{CLS}] = ... = a_{h,n}^l[\text{CLS}] = 1/n$. As backdoor effects are unevenly distributed across different layers, we use layer-specific coefficients λ_l to restrict norm intensity when minimizing the L2 norm of the [CLS] attention vector across all layers in $\theta_{[p]}$. Specifically, we assign high coefficients to layers containing more heads with low variance scores while lower coefficients to layers

Algorithm 1 PURE

Input: Training D_{train}; Validation D_{val}; Poisoned Pre-trained Model θ_p; Threshold c;
Output: Clean Model f_c;
Step 1: Head Pruning

Fine-tune θ_p on D_{train} and get poisoned model f_p
Compute variance score for each head of f_p on D_{val}
while true **do**
  Pruning the head with the lowest variance score in f_p and check the accuracy Acc of f_p on D_{val}
  if $Acc < c$ **then**
    Stop pruning and record pruned heads as $Heads$
    break
  end if
end while
Get $\theta_{[p]}$ by pruning $Heads$ from θ_p

Step 2: Attention Normalization

Get f_c by fine-tuning $\theta_{[p]}$ on D_{train} following Eq. 3

that contain the heads with high variance scores in clean data. By employing this strategy, we can effectively restrict attention focus drifting of the remaining heads while keeping the model's effectiveness on clean texts. We compute the average variance score of remaining heads in a specific layer l and use $\lambda_l = -\log_2 \text{Mean}_{\text{head}}(\overline{\text{var}}_h^l)$ as the coefficient of each layer. The hyperparameter μ serves as an adjustable factor of the overall L2 norm. The pruning and attention normalization are summarized in Algorithm 1.

6. Experiments

6.1. Experimental Setup

Attack approaches. We consider five backdoor attack strategies to poison the pre-trained model. (1) BadNets (Gu et al., 2017), (2) RIPPLe (Kurita et al., 2020), (3) LWP (Li et al., 2021), (4) HiddenKiller (Qi et al., 2021c), (5) StyleBkd (Qi et al., 2021b). Following the typical setting, we set the attack target class as "positive" and the victim model is uncased $\text{BERT}_{\text{BASE}}$ (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and DistilBERT (Sanh et al., 2019).

BadNets, RIPPLe, and LWP all use rare words as triggers, so we call them rare-word-based attacks, and we introduce 4 rare words: "cf", "bb", "ak" and "mn" as triggers for these attacks. The difference is that BadNets only focuses on data poisoning and follows the regular fine-tuning steps, while RIPPLe and LWP additionally use different techniques to strengthen backdoors. Specifically, RIPPLe considers using the restricted inner product to improve the effectiveness of the backdoor. LWP employs the hidden layer-wise attack to preserve the durability of the backdoor in all hidden layers, which is a strong attack approach if the attacker can control the training process. We target $\text{BERT}_{\text{BASE}}$, RoBERTa, and DistilBERT with the rare-word-based attack methods.

HiddenKiller uses syntactic structure as the trigger, while StyleBkd applies text style as the trigger. We follow the attack settings of HiddenKiller and StyleBkd by using the "S(SBAR)(,)(NP)(VP)(.)" syntactic structure as the trigger for HiddenKiller and "bible" style as the trigger for StyleBkd. We only attack $\text{BERT}_{\text{BASE}}$ using these two attack approaches. The details of the implementation of the above attack approaches are described in the Appendix A.

Trigger injection scenarios. We evaluate the effectiveness of defending against two trigger injection scenarios, "Full Data Knowledge (FDK)" and "Domain Shift (DS)", proposed by Kurita et al. (2020). The "FDK" assumes attackers have the full knowledge about the dataset that will be used by the end-user for fine-tuning, i.e., D_c and D_p are the same datasets. In contrast, the "DS" indicates attackers have no knowledge about the dataset used for fine-tuning and adopt a proxy dataset for the model poisoning, i.e., D_c and D_p are the different datasets.

We evaluate both "Full Data Knowledge (FDK)" and "Domain Shift (DS)" scenarios by poisoning the pre-trained language model via BadNets, RIPPLe, and LWP. Specifically, for the "FDK" scenario, both model poisoning and fine-tuning are conducted on the SST-2 dataset (Socher et al., 2013). For the "DS" scenario, the model poisoning is conducted on either IMDB (Maas et al., 2011) or Yelp (Polarity) (Zhang et al., 2015) dataset, while the poisoned model is fine-tuned on the SST-2 dataset.

The HiddenKiller and StyleBkd attacks utilize SCPN (Iyyer et al., 2018) and STRAP (Krishna et al., 2020) for converting clean texts into poisoned ones. However, SCPN is time-intensive when applied to lengthy texts and STRAP cannot transfer the text more than 50 subwords [2]. Consequently, in line with the original works, we focus solely on the 'Full Data Knowledge' (FDK) strategy, employing the SST-2 dataset to poison the pre-trained language model. This decision is driven by the fact that both the IMDB and Yelp datasets predominantly contain longer texts, which are less suitable for these two tools.

Defense baselines. We compare PURE with three baselines. (1) Vanilla fine-tuning (**FT**), which fine-tune the backdoored PLMs without any defense strategy. (2) Fine-tuning with a higher learning rate (**FTH**) (Kurita et al., 2020). (3) Using maximum entropy loss to mix up the weights of the poisoned model in the fine-tuning phase (**MEFT**) (Liu et al., 2023). Based on our best knowledge, MEFT is the only approach to purify backdoors for PLMs without the availability of triggers and without relying on any clean PLMs. More details of the implementation of each defense baseline are described in Appendix B.

Evaluation metrics. Following previous work (Kurita et al., 2020; Li et al., 2021), we evaluate the effectiveness of defense methods using the "**L**abel **F**lip **R**ate" (LFR), which represents the proportion of instances not belonging to the target class are misclassified as target class because of the attack. As we set the target class as "positive" in our experiments, LFR can be computed as:

$$\text{LFR} = \frac{\#(\text{negative instances classified as positive})}{\#(\text{overall negative instances})}$$

We inject the corresponding triggers into each negative text of the test set to compute the LFR. We also use **ACC** to represent the clean accuracy that the model performs on the *clean test set*.

6.2. Implementation Details

We implement five different backdoor attacks to get the poisoned $\text{BERT}_{\text{BASE}}$ θ_p. In our method, we set the accuracy

[2] https://github.com/martiansideofthemoon/style-transfer-paraphrase

Table 1: Backdoor defense methods against rare-word-based attacks on both "DS" and "FDK" scenarios. Bolded values indicate the best defense results. Scores are averages of 5 runs with different seeds and subscriptions indicate standard deviation. (ACC: Higher scores are better; LFR: Lower scores are better.)

Scenario			Domain Shift								Full Data Knowledge			
Dataset			IMDB → SST-2				YELP → SST-2				SST-2 → SST-2			
Method			FT	FTH	MEFT	PURE	FT	FTH	MEFT	PURE	FT	FTH	MEFT	PURE
BERT	BadNet	ACC	$92.84_{0.31}$	$91.99_{0.58}$	$91.86_{0.39}$	$91.46_{1.26}$	$92.52_{0.30}$	$91.81_{0.31}$	$91.12_{0.73}$	$91.34_{0.60}$	$91.52_{0.92}$	$90.81_{0.98}$	$91.88_{1.14}$	$90.46_{0.62}$
		LFR	$83.18_{16.04}$	$38.46_{27.24}$	$20.14_{9.50}$	$\mathbf{9.53_{0.87}}$	$98.97_{2.30}$	$73.64_{28.40}$	$29.63_{8.66}$	$\mathbf{9.53_{0.44}}$	$100.00_{0.00}$	$99.06_{0.95}$	$86.36_{21.17}$	$\mathbf{14.63_{1.55}}$
	RIPPLe	ACC	$92.56_{0.44}$	$91.87_{0.19}$	$91.74_{0.49}$	$90.97_{0.50}$	$92.42_{0.25}$	$91.71_{0.65}$	$91.16_{0.33}$	$91.43_{0.71}$	$92.08_{0.62}$	$91.14_{0.44}$	$91.88_{0.50}$	$91.34_{0.87}$
		LFR	$97.38_{3.06}$	$54.20_{21.28}$	$23.22_{6.74}$	$\mathbf{11.92_{0.57}}$	$100.00_{0.00}$	$82.90_{5.77}$	$71.96_{23.97}$	$\mathbf{11.63_{1.52}}$	$99.95_{0.10}$	$74.48_{14.17}$	$85.33_{10.59}$	$\mathbf{10.47_{1.15}}$
	LWP	ACC	$92.29_{0.26}$	$91.85_{0.33}$	$90.98_{0.67}$	$90.65_{0.83}$	$91.76_{0.27}$	$91.37_{0.36}$	$88.48_{1.08}$	$88.96_{0.71}$	$90.00_{0.21}$	$89.73_{0.50}$	$89.67_{0.28}$	$88.89_{0.53}$
		LFR	$98.27_{1.58}$	$85.96_{13.02}$	$88.46_{19.24}$	$\mathbf{14.57_{3.13}}$	$99.77_{0.24}$	$97.24_{1.64}$	$99.86_{0.13}$	$\mathbf{16.87_{2.23}}$	$100.00_{0.00}$	$100.00_{0.00}$	$100.00_{0.00}$	$\mathbf{71.07_{3.32}}$
RoBERTa	BadNet	ACC	$94.13_{0.51}$	$93.28_{0.74}$	$92.22_{1.30}$	$92.98_{0.25}$	$93.53_{0.59}$	$92.34_{0.69}$	$92.64_{0.49}$	$92.79_{0.52}$	$93.16_{0.96}$	$92.42_{0.73}$	$92.52_{0.74}$	$92.88_{1.65}$
		LFR	$56.40_{29.97}$	$12.14_{5.67}$	$33.08_{32.44}$	$\mathbf{8.36_{0.80}}$	$77.19_{23.43}$	$14.02_{6.28}$	$26.49_{35.59}$	$\mathbf{7.99_{1.03}}$	$99.81_{0.19}$	$59.34_{20.39}$	$12.24_{4.17}$	$\mathbf{8.92_{3.07}}$
	RIPPLe	ACC	$93.81_{0.63}$	$92.19_{0.45}$	$92.84_{0.96}$	$91.34_{0.72}$	$94.19_{1.64}$	$92.55_{0.87}$	$92.76_{0.94}$	$92.12_{1.54}$	$93.41_{0.16}$	$93.17_{0.78}$	$93.81_{0.33}$	$92.47_{0.81}$
		LFR	$62.74_{17.35}$	$27.59_{8.32}$	$39.21_{11.27}$	$\mathbf{11.23_{1.54}}$	$98.43_{0.22}$	$27.63_{12.74}$	$19.22_{7.93}$	$\mathbf{13.54_{3.24}}$	$97.32_{2.79}$	$88.53_{3.51}$	$14.57_{6.41}$	$\mathbf{11.21_{2.11}}$
	LWP	ACC	$93.34_{0.47}$	$92.25_{0.77}$	$91.83_{1.13}$	$92.77_{0.43}$	$92.84_{0.47}$	$92.10_{0.78}$	$89.88_{0.42}$	$91.62_{0.57}$	$89.19_{0.70}$	$88.28_{0.50}$	$88.25_{0.58}$	$89.17_{0.21}$
		LFR	$18.13_{5.76}$	$\mathbf{14.10_{7.15}}$	$60.18_{42.23}$	$18.73_{1.04}$	$99.57_{0.71}$	$52.33_{32.61}$	$75.09_{21.15}$	$\mathbf{15.28_{2.50}}$	$96.77_{4.47}$	$85.98_{16.23}$	$83.08_{35.14}$	$\mathbf{57.80_{16.84}}$
DistilBERT	BadNet	ACC	$91.01_{0.17}$	$90.13_{0.33}$	$89.61_{1.47}$	$90.14_{1.43}$	$89.93_{0.23}$	$90.16_{0.57}$	$89.54_{1.03}$	$90.52_{0.65}$	$89.29_{0.65}$	$89.72_{0.84}$	$90.16_{0.52}$	$89.01_{1.19}$
		LFR	$99.58_{0.31}$	$89.11_{9.67}$	$19.25_{6.38}$	$\mathbf{12.52_{1.49}}$	$99.86_{0.21}$	$97.52_{1.53}$	$29.48_{9.35}$	$\mathbf{15.28_{1.05}}$	$100.00_{0.00}$	$99.81_{0.25}$	$92.80_{16.90}$	$\mathbf{18.74_{5.25}}$
	RIPPLe	ACC	$91.15_{0.26}$	$89.34_{0.74}$	$90.06_{0.82}$	$89.95_{0.85}$	$91.17_{0.44}$	$90.67_{1.13}$	$91.03_{0.96}$	$90.33_{0.24}$	$89.67_{0.81}$	$87.31_{1.16}$	$88.17_{0.32}$	$88.89_{0.37}$
		LFR	$98.19_{1.77}$	$73.65_{6.59}$	$18.26_{3.43}$	$\mathbf{14.39_{1.11}}$	$100.00_{0.00}$	$87.61_{6.97}$	$21.37_{4.71}$	$\mathbf{14.97_{2.14}}$	$100.00_{0.00}$	$97.53_{0.34}$	$65.32_{13.41}$	$\mathbf{12.39_{2.45}}$
	LWP	ACC	$91.05_{0.37}$	$90.43_{0.21}$	$89.01_{1.49}$	$89.93_{0.54}$	$90.07_{0.32}$	$90.11_{0.26}$	$86.26_{1.36}$	$89.61_{0.80}$	$89.45_{0.41}$	$88.64_{0.89}$	$89.54_{0.43}$	$86.72_{0.86}$
		LFR	$97.84_{2.70}$	$81.49_{13.21}$	$86.12_{14.55}$	$\mathbf{18.64_{1.92}}$	$99.95_{0.10}$	$96.12_{3.55}$	$76.59_{16.00}$	$\mathbf{36.16_{23.65}}$	$100.00_{0.00}$	$100.00_{0.00}$	$99.95_{0.10}$	$\mathbf{76.21_{12.21}}$

PURE significantly outperforms the second-best defense approach in terms of label flipping rate, achieving a p-value at 0.01 level for $\text{BERT}_{\text{BASE}}$ and a p-value at 0.05 level for DistilBERT.

Table 2: Backdoor defense methods against HiddenKiller and StyleBkd on the "FDK" scenario targeting $\text{BERT}_{\text{BASE}}$.

Methods		SST-2 → SST-2			
		FT	FTH	MEFT	PURE
HiddenKiller	ACC	$91.94_{0.31}$	$91.53_{0.29}$	$91.42_{0.43}$	$91.55_{0.33}$
	LFR	$41.73_{3.97}$	$\mathbf{33.35_{1.86}}$	$49.16_{3.10}$	$34.53_{0.91}$
StyleBkd	ACC	$92.26_{0.37}$	$91.29_{0.12}$	$91.69_{0.19}$	$91.67_{0.31}$
	LFR	$35.37_{2.05}$	$\mathbf{28.22_{3.82}}$	$29.77_{5.59}$	$29.53_{2.16}$

threshold c as 85%. After getting the pruned pre-trained model $\theta_{[p]}$, we then fine-tune it with Equation 3 for 3 epochs with a batch size of 32 and a learning rate of 2e-5 with Adam optimizer (Kingma & Ba, 2014). We choose the model with the best clean performance on the validation set as our final backdoor elimination model. In our experiment, we use $\lambda_l = \frac{\lambda_l - \lambda_{min}}{\lambda_{max} - \lambda_{min}}$ to normalize the coefficient λ_l into a range between 0 and 1. We set μ to 0.15 for rare-word-based attacks and 0.05 for syntax-based and text-style-based attacks according to the model's clean performance on the validation set.

6.3. Experimental Results on Defending against Different Backdoor Attacks

We conduct experiments to defend against different backdoor attacks. We show the defense results against rare-words-based attacks (BadNet, RIPPLe, and LWP) on BERT, RoBERTa, and DistilBERT in both full data knowledge and domain shift scenarios in Table 1. The defense results against HiddenKiller and StyBkd attacks on BERT in the full data knowledge scenario are in Table 2.

As shown in Table 1, PURE achieves the lowest label flipping rate (LFR) in defending against rare-word-based backdoor attacks without significantly compromising clean accuracy (ACC) in most cases. Specifically, PURE can effectively neutralize the backdoor in the poisoned BERT, RoBERTa, and DistilBERT, maintaining a low LFR in the domain shift scenario. In contrast, other defense strategies do not consistently purify the backdoor across attacks on various pre-trained models.

We still notice that FTH and MEFT cannot work well in defending against rare-word-based attacks in the full data knowledge scenarios, and they cannot even eliminate backdoors under the layer-wise weight poisoning (LWP) attack. At the same time, PURE can effectively defend against the BadNet and RIPPLe attacks and can still remove some backdoors with a lower LFR compared with FTH and MEFT under the LWP attack. LWP can largely preserve the effectiveness of the backdoor in the pre-trained model, especially when the attacker has full knowledge of users' datasets.

Table 2 shows that PURE remains effective against HiddenKiller and StyleBkd attacks on BERT, exhibiting a comparable LFR with FTH that achieves the best defense results. Meanwhile, our findings shown in Appendix C indicate that even fine-tuning a clean pre-trained model on a clean dataset can have a relatively high label flipping rate, indicating that the syntax or text style-based attacks could lead to some semantic losses on the original texts. Therefore, it is challenging to defend against these two attacks.

According to the results above, PURE can effectively defend against various backdoor attacks on different pre-trained models without significantly compromising the clean accuracy in both domain shift scenarios and full data knowledge. Especially, PURE is very strong to defend against rare-word-based backdoor attacks in the domain shift scenario.

Table 3: Ablation study on the effectiveness of backdoor defense against rare-word-based attacks on BERT.

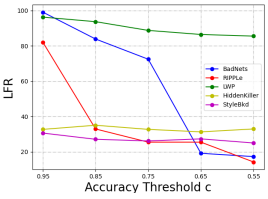
Scenario		Domain Shift						Full Data Knowledge		
Dataset		IMDB → SST-2			YELP → SST-2			SST-2 → SST-2		
Method		Prune-Only	Norm-Only	PURE	Prune-Only	Norm-Only	PURE	Prune-Only	Norm-only	PURE
BadNet	ACC	$92.28_{0.42}$	$91.48_{1.19}$	$91.46_{1.26}$	$92.08_{0.23}$	$90.98_{0.84}$	$91.34_{0.60}$	$91.06_{0.26}$	$91.20_{0.28}$	$90.46_{0.62}$
	LFR	$34.91_{35.01}$	$9.72_{1.74}$	$\mathbf{9.53_{0.87}}$	$52.94_{41.89}$	$10.33_{0.63}$	$\mathbf{9.53_{0.44}}$	$46.45_{26.23}$	$\mathbf{14.25_{0.64}}$	$14.63_{1.55}$
Ripple	ACC	$91.99_{0.70}$	$90.97_{0.63}$	$90.97_{0.50}$	$92.10_{0.25}$	$92.21_{0.09}$	$91.43_{0.71}$	$91.75_{0.50}$	$90.93_{0.49}$	$91.34_{0.87}$
	LFR	$57.48_{22.26}$	$\mathbf{11.36_{0.43}}$	$11.92_{0.57}$	$15.51_{7.04}$	$\mathbf{11.16_{0.76}}$	$11.63_{1.52}$	$25.70_{9.34}$	$11.35_{0.36}$	$\mathbf{10.47_{1.15}}$
LWP	ACC	$92.20_{0.58}$	$86.39_{0.48}$	$90.65_{0.83}$	$90.42_{2.50}$	$87.15_{2.72}$	$88.96_{0.71}$	$90.66_{0.61}$	$86.59_{0.67}$	$88.89_{0.53}$
	LFR	$21.07_{18.83}$	$83.55_{5.33}$	$\mathbf{14.57_{3.13}}$	$61.82_{19.72}$	$53.51_{13.72}$	$\mathbf{16.87_{2.23}}$	$97.76_{2.50}$	$100.00_{0.00}$	$\mathbf{71.07_{3.32}}$

Table 4: Ablation study on the effectiveness of backdoor defense against HiddenKiller and StyleBkd on BERT.

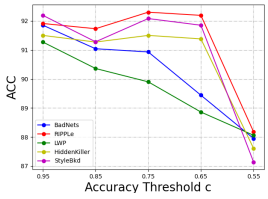
Methods		SST-2 → SST-2		
		Prune-Only	Norm-Only	PURE
HiddenKiller	ACC	$91.57_{0.33}$	$91.61_{0.32}$	$91.55_{0.33}$
	LFR	$35.56_{0.67}$	$40.05_{1.82}$	$\mathbf{34.53_{0.91}}$
StyleBkd	ACC	$91.62_{0.64}$	$92.01_{0.26}$	$91.67_{0.31}$
	LFR	$\mathbf{27.43_{2.41}}$	$34.02_{1.80}$	$29.53_{2.16}$

6.4. Ablation Study

We further conduct an ablation study on the BERT model to evaluate the role of head pruning and attention normalization in our method. For the Prune-Only defense, we follow the same pruning steps and fine-tune the pruned model, $\theta_{[p]}$, without incorporating the attention weight normalization, while for the Norm-Only defense, we only fine-tune the poisoned θ_p following the Equation 3 without pruning. We present the defense results against rare-words-based attacks in both "DS" and "FDK" in Table 3, and the results against HiddenKiller and StyleBkd in "FDK" in Table 4.



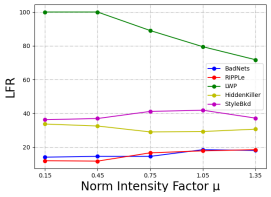
(a) LFR Performance



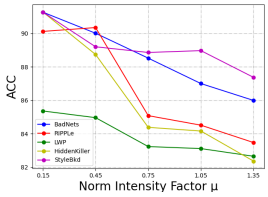
(b) Acc Performance

Figure 4: The impact on LFR and ACC when tuning the pruning accuracy threshold c.

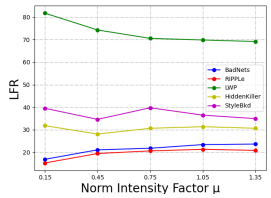
As shown in Table 3, both the Prune-Only and Norm-Only defense methods demonstrate the ability to mitigate backdoor attacks on the pre-trained model. For BadNet and RIPPLe attacks, the Norm-Only can even achieve a comparable or better LFR with PURE. However, in most cases, the combination of head pruning and attention normalization can yield the most effective backdoor elimination on rare-word-based backdoor attacks, particularly on the LWP attack. Moreover, PURE can achieve better LFR against



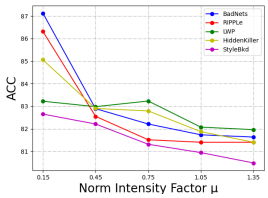
(a) LFR with λ_l



(b) Acc with λ_l



(c) LFR without λ_l



(d) Acc without λ_l

Figure 5: The impact on LFR and ACC when tuning the hyperparameter μ with/without layer-specific coefficients λ_l in Eq. 3

HiddenKiller and the Prune-Only performs best under the StyleBkd attack as shown in Table 4.

We still note that employing the norm-only defense method may sacrifice the model's performance on clean data in some cases. This is because $\text{Mean}_{\text{head}}(\overline{\text{var}}_h^l)$ can be lower if we don't prune the heads with low variance scores for a specific layer l, leading to a higher coefficient λ_l. A higher λ_l can intensify the norm, which causes an overly uniform distribution of [CLS] attention across tokens at heads, even though these heads should have focused more on semantically relevant tokens in clean texts.

7. Sensitivity Analysis

We conduct experiments to investigate the impact of accuracy threshold c and hyperparameter μ in Equation 3, and explore the necessity of layer-specific coefficients λ_l.

Sensitivity analysis on tuning c. We test five different accuracy thresholds c (0.95, 0.85, 0.75, 0.65, 0.55) in the Prune-Only defense. As shown in Figure 4, using a lower

accuracy threshold c can potentially decrease the LFR, but it may also negatively impact the model's utility. This effect arises because a lower c value typically results in removing more attention heads, which can compromise the model's overall effectiveness when processing clean inputs.

Sensitivity analysis on tuning μ. We set five different norm intensity factors μ (0.15, 0.45, 0.75, 1.05, 1.35) in the Norm-Only defense with and without λ_l (setting $\lambda_l = 1$ for all layers) on "FDK" for all attacks. As illustrated in Figure 5, setting a higher norm intensity factor μ may negatively impact both LFR and clean performance. This is attributed to the more uniform distribution of attention weights, which impairs the model's ability in classification leading to decreased performance on clean data and thus an increased LFR. Moreover, fine-tuning Equation 3 with layer-specific coefficients λ_l maintains the model's robustness against backdoor attacks in terms of LFR (Figure 5a and 5c) while significantly improving its utility compared with the scenario setting $\lambda_l = 1$ for all layers (Figure 5b and 5d). This implies that incorporating layer-specific coefficients is crucial for preserving the model's effectiveness on clean data without compromising its resilience to backdoor attacks.

8. Conclusions

In this paper, we have developed PURE to defend against the backdoor attacks on the pre-trained language models. Considering that the end-users who adopt the pre-trained language models for their downstream tasks do not have any knowledge about the potential backdoor threats, PURE only assumes the end-users have a clean dataset for fine-tuning. Based on the observation that the attention heads that are affected by the backdoor triggers usually have low variance scores on clean texts, PURE purifies the backdoored model by head pruning and attention normalization. Our experimental results have demonstrated that PURE can significantly reduce the label flip rate on poisoned texts while maintaining high accuracy on clean texts.

Acknowledgements

This work was supported in part by NSF 2103829.

Impact Statement

The paper presents PURE, an innovative backdoor mitigation approach designed to enhance the security of pre-trained language models (PLMs) against backdoor attacks, without diminishing their performance on legitimate tasks. The proposed method can potentially benefit the end users as it can be implemented without prior knowledge of the specifics of the attack strategies, making it a practical tool for safeguarding (PLMs).

References

Azizi, A., Tahmid, I. A., Waheed, A., Mangaokar, N., Pu, J., Javed, M., Reddy, C. K., and Viswanath, B. T-Miner: A generative approach to defend against trojan attacks on DNN-based text classification. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 2255–2272. USENIX Association, August 2021. ISBN 978-1-939133-24-3.

Chen, K., Meng, Y., Sun, X., Guo, S., Zhang, T., Li, J., and Fan, C. Badpre: Task-agnostic backdoor attacks to pre-trained NLP foundation models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=Mng8CQ9eBW.

Chen, X., Salem, A., Chen, D., Backes, M., Ma, S., Shen, Q., Wu, Z., and Zhang, Y. Badnl: Backdoor attacks against nlp models with semantic-preserving improvements. In *Annual computer security applications conference*, pp. 554–569, 2021.

Dai, J., Chen, C., and Li, Y. A backdoor attack against lstm-based text classification systems. *IEEE Access*, 7: 138872–138878, 2019.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In Burstein, J., Doran, C., and Solorio, T. (eds.), *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. URL https://aclanthology.org/N19-1423.

Gan, L., Li, J., Zhang, T., Li, X., Meng, Y., Wu, F., Yang, Y., Guo, S., and Fan, C. Triggerless backdoor attack for NLP tasks with clean labels. In Carpuat, M., de Marneffe, M.-C., and Meza Ruiz, I. V. (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2942–2952, Seattle, United States, July 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.naacl-main.214.

Gu, T., Dolan-Gavitt, B., and Garg, S. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

Gupta, A. and Krishna, A. Adversarial clean label backdoor attacks and defenses on text classification systems. In Can, B., Mozes, M., Cahyawijaya, S., Saphra, N., Kassner, N., Ravfogel, S., Ravichander, A., Zhao, C., Augenstein, I., Rogers, A., Cho, K., Grefenstette, E., and Voita,

L. (eds.), *Proceedings of the 8th Workshop on Representation Learning for NLP (RepL4NLP 2023)*, pp. 1–12, Toronto, Canada, July 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.repl4nlp-1.1.

Howard, J. and Ruder, S. Universal language model fine-tuning for text classification. In Gurevych, I. and Miyao, Y. (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 328–339, Melbourne, Australia, July 2018. Association for Computational Linguistics. URL https://aclanthology.org/P18-1031.

Iyyer, M., Wieting, J., Gimpel, K., and Zettlemoyer, L. Adversarial example generation with syntactically controlled paraphrase networks. In Walker, M., Ji, H., and Stent, A. (eds.), *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. URL https://aclanthology.org/N18-1170.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Krishna, K., Wieting, J., and Iyyer, M. Reformulating unsupervised style transfer as paraphrase generation. In Webber, B., Cohn, T., He, Y., and Liu, Y. (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 737–762, Online, November 2020. Association for Computational Linguistics. URL https://aclanthology.org/2020.emnlp-main.55.

Kurita, K., Michel, P., and Neubig, G. Weight poisoning attacks on pretrained models. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2793–2806, Online, July 2020. Association for Computational Linguistics. URL https://aclanthology.org/2020.acl-main.249.

Li, L., Song, D., Li, X., Zeng, J., Ma, R., and Qiu, X. Backdoor attacks on pre-trained models by layerwise weight poisoning. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t. (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3023–3032, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.emnlp-main.241.

Liu, K., Dolan-Gavitt, B., and Garg, S. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International symposium on research in attacks, intrusions, and defenses*, pp. 273–294. Springer, 2018.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Liu, Z., Shen, B., Lin, Z., Wang, F., and Wang, W. Maximum entropy loss, the silver bullet targeting backdoor attacks in pre-trained language models. In Rogers, A., Boyd-Graber, J., and Okazaki, N. (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 3850–3868, Toronto, Canada, July 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.findings-acl.237.

Lyu, W., Zheng, S., Ma, T., and Chen, C. A study of the attention abnormality in trojaned BERTs. In Carpuat, M., de Marneffe, M.-C., and Meza Ruiz, I. V. (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4727–4741, Seattle, United States, July 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.naacl-main.348.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. Learning word vectors for sentiment analysis. In Lin, D., Matsumoto, Y., and Mihalcea, R. (eds.), *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-1015.

Pan, X., Zhang, M., Sheng, B., Zhu, J., and Yang, M. Hidden trigger backdoor attack on NLP models via linguistic style manipulation. In *31st USENIX Security Symposium (USENIX Security 22)*, pp. 3611–3628, Boston, MA, August 2022. USENIX Association. ISBN 978-1-939133-31-1.

Qi, F., Chen, Y., Li, M., Yao, Y., Liu, Z., and Sun, M. ONION: A simple and effective defense against textual backdoor attacks. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t. (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 9558–9566, Online and Punta Cana, Dominican Republic, November 2021a. Association for Computational Linguistics. URL https://aclanthology.org/2021.emnlp-main.752.

Qi, F., Chen, Y., Zhang, X., Li, M., Liu, Z., and Sun, M. Mind the style of text! adversarial and backdoor attacks based on text style transfer. In Moens, M.-F., Huang,

X., Specia, L., and Yih, S. W.-t. (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 4569–4580, Online and Punta Cana, Dominican Republic, November 2021b. Association for Computational Linguistics. URL https://aclanthology.org/2021.emnlp-main.374.

Qi, F., Li, M., Chen, Y., Zhang, Z., Liu, Z., Wang, Y., and Sun, M. Hidden killer: Invisible textual backdoor attacks with syntactic trigger. In Zong, C., Xia, F., Li, W., and Navigli, R. (eds.), *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. 443–453, Online, August 2021c. Association for Computational Linguistics. URL https://aclanthology.org/2021.acl-long.37.

Qi, F., Yao, Y., Xu, S., Liu, Z., and Sun, M. Turn the combination lock: Learnable textual backdoor attacks via word substitution. In Zong, C., Xia, F., Li, W., and Navigli, R. (eds.), *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. 4873–4883, Online, August 2021d. Association for Computational Linguistics. URL https://aclanthology.org/2021.acl-long.377.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Shen, G., Liu, Y., Tao, G., Xu, Q., Zhang, Z., An, S., Ma, S., and Zhang, X. Constrained optimization with dynamic bound-scaling for effective nlp backdoor defense. In *International Conference on Machine Learning*, pp. 19879–19892. PMLR, 2022.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In Yarowsky, D., Baldwin, T., Korhonen, A., Livescu, K., and Bethard, S. (eds.), *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.

Sun, L. Natural backdoor attack on text data. *arXiv preprint arXiv:2006.16176*, 2020.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In Linzen, T., Chrupała, G., and Alishahi, A. (eds.), *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355, Brussels, Belgium, November 2018. Association for Computational Linguistics. URL https://aclanthology.org/W18-5446.

Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H., and Zhao, B. Y. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 707–723. IEEE, 2019.

Yan, J., Gupta, V., and Ren, X. BITE: Textual backdoor attacks with iterative trigger injection. In Rogers, A., Boyd-Graber, J., and Okazaki, N. (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12951–12968, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.725. URL https://aclanthology.org/2023.acl-long.725.

Yang, W., Li, L., Zhang, Z., Ren, X., Sun, X., and He, B. Be careful about poisoned word embeddings: Exploring the vulnerability of the embedding layers in NLP models. In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y. (eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2048–2058, Online, June 2021a. Association for Computational Linguistics. URL https://aclanthology.org/2021.naacl-main.165.

Yang, W., Lin, Y., Li, P., Zhou, J., and Sun, X. RAP: Robustness-Aware Perturbations for defending against backdoor attacks on NLP models. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t. (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 8365–8381, Online and Punta Cana, Dominican Republic, November 2021b. Association for Computational Linguistics. doi:

10.18653/v1/2021.emnlp-main.659. URL https://aclanthology.org/2021.emnlp-main.659.

Yang, W., Lin, Y., Li, P., Zhou, J., and Sun, X. Rethinking stealthiness of backdoor attack against NLP models. In Zong, C., Xia, F., Li, W., and Navigli, R. (eds.), *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. 5543–5557, Online, August 2021c. Association for Computational Linguistics. URL https://aclanthology.org/2021.acl-long.431.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

Zhang, X., Zhao, J., and LeCun, Y. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

Zhang, X., Zhang, Z., Ji, S., and Wang, T. Trojaning language models for fun and profit. In *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 179–197. IEEE, 2021.

Zhang, Z., Lyu, L., Ma, X., Wang, C., and Sun, X. Fine-mixing: Mitigating backdoors in fine-tuned language models. In Goldberg, Y., Kozareva, Z., and Zhang, Y. (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 355–372, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.26. URL https://aclanthology.org/2022.findings-emnlp.26.

Zhang, Z., Chen, D., Zhou, H., Meng, F., Zhou, J., and Sun, X. Diffusion theory as a scalpel: Detecting and purifying poisonous dimensions in pre-trained language models caused by backdoor or bias. In Rogers, A., Boyd-Graber, J., and Okazaki, N. (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 2495–2517, Toronto, Canada, July 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.findings-acl.157.

A. Details of Attack Scenarios

A.1. Details of Datasets

We split the IMDB dataset into training (22,500), validation (2,500), and testing (25,000). We randomly sample 50,000 examples from the YELP-Polarity training dataset, dividing them into a training set of 45,000 samples and a validation set of 5,000 samples. We utilize the original dataset's test set, which contains 38,000 samples. For SST-2, we split the dataset into training (60,570), validation (6,730), and testing (872). As the original testing of glue SST-2 in Hugging Face is not given a label, we view its original validation as testing in our experiments.

A.2. Implementation of Attack Scenarios

For Badnet, RIPPLe, and LWP, we randomly injected one rare word of "cf", "bb", "ak" and "mn" into 50% training samples and set their label to "positive" during the poisoning process. These tampered texts combined with the original clean samples were then used to train a poisoned pre-trained language model θ_p. We follow the attack setup of HiddenKiller and StyleBkd setting the poisoned rate to be 30% and 20% samples of the training data. We conduct poisoning training for 5 epochs with a learning rate of 2e-5 and a batch size of 32 with the Adam optimizer (Kingma & Ba, 2014) for all attack scenarios.

B. Details of Defense Baselines

For FTH, we set the learning rate to 5e-5 following Kurita et al. (2020). For MEFT, we set the maximum entropy training step to 4000 steps, which makes the Stop Distance (SD) fall between 0.01 and 0.015 (Liu et al., 2023). We conduct fine-tuning for 3 epochs with a learning rate of 2e-5 and a batch size of 32 with the Adam optimizer. We set the max length of inputs as 128 when fine-tuning the model and finally choose the best performance in the validation set as our final model for all defense methods. All experiments are run on AMD Ryzen Threadripper 3960X 24-core Processor and NVIDIA GeForce RTX 3090.

C. Experimental Results

Performance of defense approaches on a clean pre-trained model. We evaluate the performance of each defense approach by fine-tuning a clean $\text{BERT}_{\text{BASE}}$ model on a clean SST-2 training set and testing it on both clean and poisoned test datasets. As shown in Table 5, defense methods including FTH, MEFT, and PURE maintain similar clean accuracy to regular fine-tuning when the pre-trained model is clean. We also notice that injecting rare words has little impact on the label flipping rate, which is expected. However, we find the clean model has a higher LFR on the poisoned test set using syntactic features or text styles as triggers. This could be because transforming clean texts into poisoned ones using a fixed syntactic structure or text style may lead to semantics loss or distribution shift of original text, thereby degrading the clean model's performance.

Table 5: Clean model's utility on clean and poisoned test.

Methods		FT	FTH	MEFT	PURE
ACC		$92.10_{0.25}$	$91.85_{0.59}$	$89.12_{2.98}$	$90.88_{0.53}$
Rare-Word	LFR	$8.27_{0.90}$	$7.01_{1.55}$	$11.26_{2.77}$	$10.28_{1.07}$
Syntactic	LFR	$25.47_{2.23}$	$25.42_{3.58}$	$27.52_{0.65}$	$30.28_{1.47}$
Text-Style	LFR	$17.80_{1.21}$	$19.49_{2.59}$	$20.42_{3.01}$	$23.22_{2.13}$

D. Computational Cost

We conduct experiments to compare the time cost of our approach PURE with Vanilla Fine-tuning (FT) and Maximum Entropy Fine-tuning (MEFT), against five attacks. We run each method under the full-data-knowledge scenario defending against the poisoned BERT model following the same setup in our paper. Note that we do not report FTH in this table. FTH will have the same running time as FT because FTH achieves the backdoor defense by setting a high learning rate. From the table 6, PURE costs more running time (seconds) than baselines but still in a reasonable range.

Table 6: Time cost (seconds) of each defense strategy.

Dataset	SST2 $\rightarrow$ SST2		
Method (s)	FT	MEFT	PURE
BadNet	1039.20	1745.39	2008.92
RIPPLe	1042.72	1757.79	2063.70
LWP	1054.04	1726.70	2105.21
HiddenKiller	1037.53	1731.54	2047.72
StyleBkd	1047.28	1753.84	2017.58