Backdoor Token Unlearning: Exposing and Defending Backdoors in Pretrained Language Models

Peihai Jiang[1], Xixiang Lyu[1]*, Yige Li[2]*, Jing Ma[1]
[1]Xidian University, China
[2]Singapore Management University, Singapore
{xdjph, jing.ma}@stu.xidian.edu.cn, xxlv@mail.xidian.edu.cn, yigeli@smu.edu.sg

Abstract

Supervised fine-tuning has become the predominant method for adapting large pretrained models to downstream tasks. However, recent studies have revealed that these models are vulnerable to backdoor attacks, where even a small number of malicious samples can successfully embed backdoor triggers into the model. While most existing defense methods focus on post-training backdoor defense, efficiently defending against backdoor attacks during training phase remains largely unexplored. To address this gap, we propose a novel defense method called *Backdoor Token Unlearning (BTU)*, which proactively detects and neutralizes trigger tokens during the training stage. Our work is based on two key findings: 1) backdoor learning causes distinctive differences between backdoor token parameters and clean token parameters in word embedding layers, and 2) the success of backdoor attacks heavily depends on backdoor token parameters. The BTU defense leverages these properties to identify aberrant embedding parameters and subsequently removes backdoor behaviors using a fine-grained unlearning technique. Extensive evaluations across three datasets and four types of backdoor attacks demonstrate that BTU effectively defends against these threats while preserving the model's performance on primary tasks.

Code — https://github.com/XDJPH/BTU

Introduction

Pretrained Language Models (PLMs) (Devlin et al. 2018; Radford et al. 2019) have demonstrated remarkable performance across various tasks, such as sentiment analysis (Jim et al. 2024), toxicity detection (Bonetti et al. 2023), and news classification (Nkongolo Wa Nkongolo 2023). However, as PLMs are increasingly fine-tuned for specific downstream applications (Min et al. 2023), they have become vulnerable to backdoor attacks (Liu et al. 2024; Cheng et al. 2023). Typically, backdoor attacks inject malicious triggers into the model during training. The backdoored model functions normally on clean tasks but exhibits an attack-desired target label when the trigger is presented. In Natural Language Processing (NLP), backdoor triggers can be designed as obvious elements like rare words (Kurita, Michel, and Neubig 2020) or more subtle features such as sentence styles (Qi et al. 2021a). With the widespread adoption and deployment of PLMs, defending against backdoor threats has become an urgent challenge.

Existing backdoor defense methods in NLP generally fall into three categories: backdoor detection (Lyu et al. 2024; Liu et al. 2022; Xian et al. 2023), backdoor removal (Zhang et al. 2022; Li et al. 2021c), and anti-backdoor learning (Li et al. 2021b; Zhu et al. 2022). Backdoor model detection methods aim to identify whether a model or inputs contain backdoors, while backdoor removal methods focus on purifying the backdoor triggers from the backdoored model. Among them, anti-backdoor learning methods (Zhu et al. 2022; Li et al. 2021b) has become a widely adopted defense strategy as they allow the users to train a clean model even on a poisoned dataset. For example, ABL (Li et al. 2021b) employs a two-stage gradient ascent technique to filter out and mitigate backdoor behaviors. Another approach, MF (Zhu et al. 2022), limits the model's learning capacity by restricting the number of training epochs, thereby preventing the model from acquiring backdoors during training. However, these anti-backdoor learning methods often lead to reduced model performance and exhibit instability across different scenarios. Therefore, how to effectively defend against backdoor attacks during the model training phase essentially deserves much attention.

Previous research has shown that backdoor learning can be viewed as a dual-task problem, i.e. training the backdoored model on both clean and backdoor data (Li et al. 2023). In this paper, we reformulate backdoor learning from model parameter perspective and identify two key properties: 1) backdoor learning induces significant differences between the embedding parameters of backdoor tokens and clean tokens, where the backdoor tokens converge much faster than clean ones; 2) the activation of backdoors is highly dependent on backdoor token parameters in the embedding layers. Intuitively, if we can isolate backdoor token parameters at the level of word embedding dimensions rather than across all model parameters, the backdoor information could be more effectively exposed and removed.

In this work, we propose a novel defense method called Backdoor Token Unlearning (BTU) for efficient anti-backdoor learning. Specifically, BTU operates in two stages: *backdoor token detection* and *dimensional fine-grained un-*

*Corresponding authors

learning. In the first stage, BTU identifies potential backdoor tokens by exclusively training the word embedding layer and flagging the top $\alpha\%$ as backdoor-related embedding parameters. In the second stage, BTU removes backdoor information by replacing the affected backdoor embedding parameters with those of benign padding token embeddings. Through these two stages, BTU effectively defends against backdoor attacks while minimizing the impact on clean task performance. The main contributions of our work are summarized as follows:

- We identify two key observations in NLP backdoor attacks: 1) the distinctive differences in the embedding values of backdoor tokens and clean tokens when only the word embedding layer is trained, and 2) the success of backdoor activation is highly related to the backdoor token embedding parameters.
- We introduce a novel defense method termed Backdoor Token Unlearning (BTU), which proactively exposes aberrant embedding parameters of backdoor tokens and mitigates backdoor behavior during the training process, with minimal impact on clean task performance.
- Extensive experiments on four types of backdoor attacks across three datasets demonstrate that our proposed BTU substantially reduces the success rate of backdoor attacks while having minimal impact on the accuracy of downstream tasks.

Related Work

Backdoor Attack

Existing backdoor attacks in NLP manifest in two primary scenarios: outsourced training and data poisoning. In outsourced training, attackers have full control over the training process. For instance, the LWP (Li et al. 2021a) scheme implants backdoors in the model's intermediate layers to increase the persistence of the attack, while the transfer (Shen et al. 2021) approach adjusts the backdoor optimization target in front of the MLP layer, using a multi-objective strategy to ensure the attack's resilience against downstream task influences. Additionally, LWS (Qi et al. 2021c) employs an auxiliary model to create more concealed triggers. Conversely, in data poisoning scenarios, attackers are limited to inserting a few carefully crafted samples into the dataset since they do not control the training process. For example, Dai et al. (Dai, Chen, and Li 2019) demonstrate that words or context-independent phrases can serve as triggers, and that random insertion into training samples can successfully inject backdoors. Similarity, Qi et al. (Qi et al. 2021a,b) reveal that textual styles and syntactic structures can also act as triggers, significantly enhancing the stealthiness of backdoor attacks. These studies highlight the high vulnerability of NLP models to such covert manipulations and underscore the critical need for robust defense mechanisms.

Backdoor Defense

In the field of NLP, existing backdoor defense methods can be broadly categorized into three types: 1) Backdoor input detection, which is applied during the model inference stage to identify and prevent the activation of backdoor inputs (Gao et al. 2021; Chen and Dai 2021; Yang et al. 2021a). For example, BKI (Chen and Dai 2021) distinguishes potential trigger words by analyzing each word's impact on the model's outcomes; 2) Backdoored model detection, which assesses whether a model contains backdoors (Liu et al. 2022; Azizi et al. 2021), often employing techniques like reverse engineering. For instance, PICCOLO attempts to recover potential triggers embedded within the model; 3) Anti-backdoor learning aims to train clean models from potentially poisoned datasets during the training phase (Li et al. 2021b; Zhu et al. 2022; Min et al. 2023). For instance, ABL (Li et al. 2021b) characterizes backdoor learning as a form of shortcut learning, where backdoor triggers are more easily captured. To address this, ABL proposed a two-stage gradient ascent technique to mitigate backdoor effects. Similarly, the MF defense (Zhu et al. 2022) introduced to minimize overfitting to prevent the model from learning backdoor patterns. Although promising, these methods often fail against adaptive attacks, such as textual style or grammatical structure triggers. In this work, we present new insights into backdoor learning and propose an simple yet efficient anti-backdoor defense to mitigate such threat.

Proposed Token Unlearning Method

In this section, we first present the problem of backdoor attacks and then reveal the distinctive behavior between backdoor tokens and clean tokens optimized in the word embedding layers. Finally, we introduce our proposed BTU method.

Problem definition Consider the poisoned training dataset as $\mathcal{D} = \mathcal{D}_c \cup \mathcal{D}_b$, where $\mathcal{D}_c$ denotes the subset of clean data and $\mathcal{D}_b$ denotes the subset of backdoor data. Training a backdoored model on a poisoned dataset can be viewed as minimizing the following empirical error:

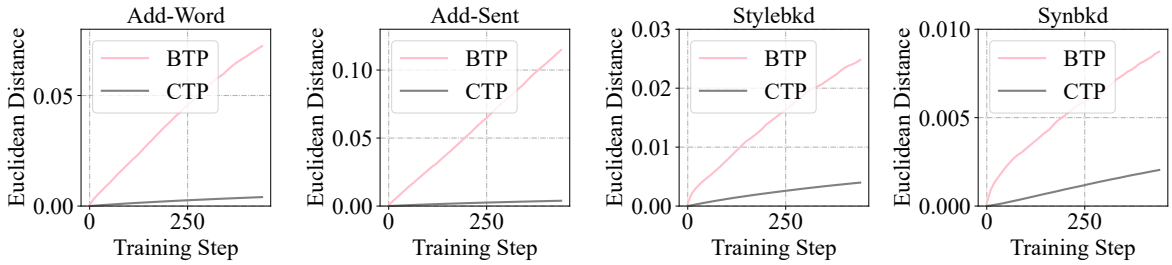
$$\mathcal{L} = \underbrace{\mathbb{E}_{(x,y)\sim\mathcal{D}_c}[\ell(f_\theta(x), y)]}_{\text{clean task}} + \underbrace{\mathbb{E}_{(x,y)\sim\mathcal{D}_b}[\ell(f_\theta(x), y)]}_{\text{backdoor task}}, \quad (1)$$

where ℓ and θ denote the loss function and model parameters, respectively. The overall learning task can be regarded as a combination of the backdoor task on dataset $\mathcal{D}_b$ and the clean task on dataset $\mathcal{D}_c$.

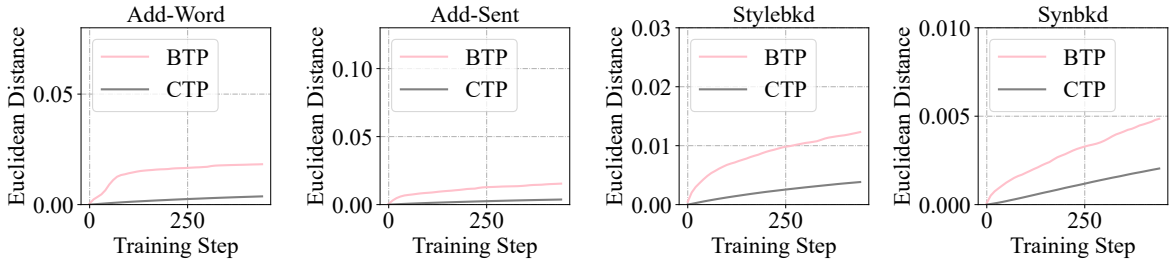
Intuitively, if we can clearly distinguish between clean and backdoor tasks, the backdoor task can be more effectively detected. To achieve this, we reformulate the backdoor learning process in Eq. 1 to focus on the word embedding layer rather than all model parameters. As a result, the model's optimization objective can be redefined as follows:

$$\mathcal{L} = \underbrace{\mathbb{E}_{(x,y)\sim\mathcal{D}_c}[\ell(\boldsymbol{\varepsilon}(x), y)]}_{\text{clean task}} + \underbrace{\mathbb{E}_{(x,y)\sim\mathcal{D}_b}[\ell(\boldsymbol{\varepsilon^b}(x), y)]}_{\text{backdoor task}}, \quad (2)$$

where ε denotes the entire clean embedding parameters and ε^b denotes backdoor embedding parameters. Based on Eq. 2,



(a) **Parameter: Embedding Layer**



(b) **Parameter: All Layers**

Figure 1: The distinctive learning behaviors for BTP and CTP under four different backdoor attacks. Figure (a) represents the variation in BTP and CTP as a function of the number of iterations when only optimizing the model's word embedding layer. Figure (b) represents the variation in BTP and CTP as a function of the number of iterations when optimizing all model parameters. In the Stylebkd and Synbkd attacks, conjunctions and punctuation marks are chosen as backdoor tokens. These abnormal behaviors are consistent across other attacks as well, highlighting the generalization of this phenomenon.

the backdoor information is primarily contained in the *Backdoor Token Parameters (BTP)*, while the *Clean Token Parameters (CTP)* remain largely unchanged. Since the backdoor task is much simpler than the clean task (Li et al. 2021b), we observe that the cumulative parameter changes in BTP occur more rapidly than in CTP. We will provide empirical evidence to support this observation in the following subsection.

Revealing Distinctive Behavior of Backdoor Tokens

In this subsection, we aim to highlight the distinct learning behavior between BTP and CTP when trained on word embedding layers.

We conduct four backdoor attack methods: Add-Word (Gu, Dolan-Gavitt, and Garg 2017), Add-Sent (Dai, Chen, and Li 2019), Stylebkd (Qi et al. 2021a), and Synbkd (Qi et al. 2021b), to poison the SST-2 dataset (Socher et al. 2013) with a 10% poisoning rate. We then train a BERT (Devlin et al. 2018) model using standard procedures and settings from the public library (Cui et al. 2022). For each attack, we trained two backdoored models: one on all parameters and another only on the word embedding layers. To compare the learning differences, we record the variations in Euclidean distance between BTP and CTP.

Fig. 1 shows that, across all four types of attacks, the mean Euclidean distance of the BTP is greater than that in the CTP. For example, in Add-Word attack, when training only the word embedding layer, BTP is almost 0.1 higher than CTP. However, when training all parameters, BTP is only 0.01 higher than CTP. The difference in the magnitude of change between the two cases is nearly tenfold. This distinction between BTP and CTP suggests that backdoor information is primarily associated with BTPs and inspires our defense strategy.

Backdoor Token Unlearning

Overview Fig. 2 illustrates the BTU framework, which consists of two main components: *Backdoor Token Detection* and *Dimensional Fine-grained Unlearning*. The backdoor token detection aims to identify suspicious backdoor tokens within the embedding parameters through three rounds of anomaly detection. Once these malicious tokens are detected, fine-grained dimensional unlearning is applied to remove backdoor functionalities from these token parameters. We provide detailed technical explanations below.

Backdoor Token Detection As previously noted, we have identified a distinctive Euclidean distance between BTP and CTP. Building on this, we can detect suspicious backdoor token parameters through iterative detection rounds T. The detection threshold is set to $\alpha \in [0, 1]$, with the top $\alpha\%$ of embedding parameters flagged as backdoor token parameters in each detection round. For simplicity, we set α to 0.05 across all three detection rounds. A more detailed analysis of α and the detection round T will be provided in the ablation

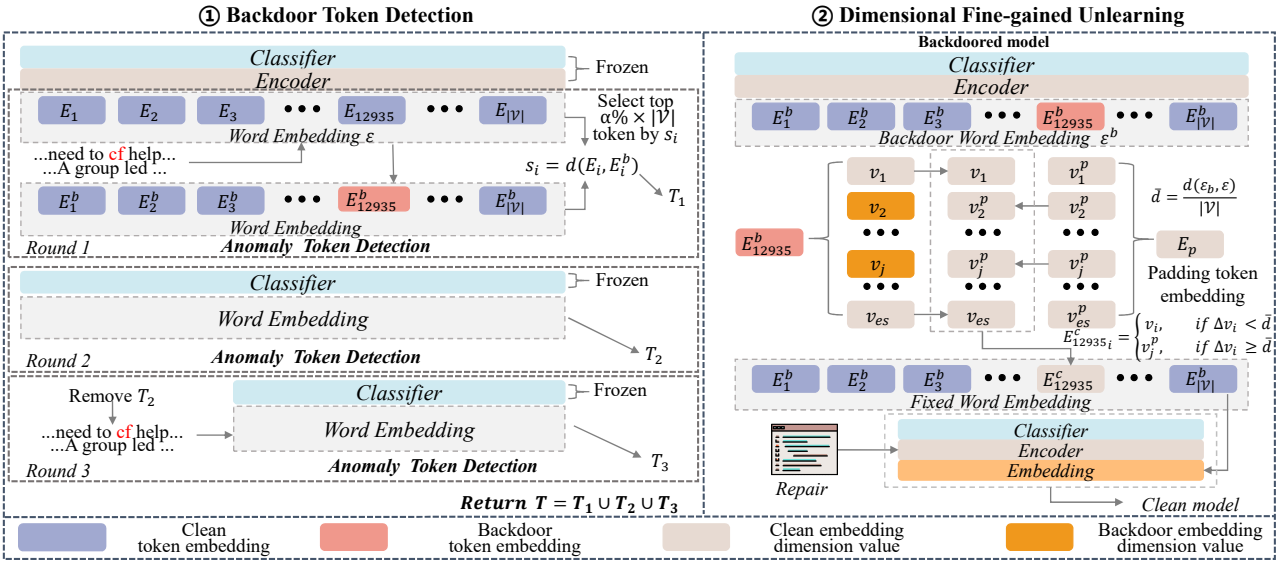


Figure 2: Illustration of the BTU Framework: 1) The Backdoor Token Detection phase includes three rounds of Anomaly Token Detection, where only the embedding layer is trained to detect embeddings with significant changes. 2) Dimensional Fine-grained Unlearning uses padding token embedding to precisely correct the anomalies tokens detected by backdoor token detection, and then the model is repaired using clean data.

study.

In the first round, we train only the embedding layer parameters ε of model M on the dataset $\mathcal{D}$, resulting in the updated embedding layer parameters ε'. We then calculate the change distance s' for each token t_i and store the token-distance pairs in the set T_1:

$$T_1 = \{(s'_i, t_i)\}_{t_i \in \mathcal{V}} = \{(d(\varepsilon(t_i), \varepsilon'(t_i)), t_i)\}_{t_i \in \mathcal{V}} \quad (3)$$

Next, we rank s'_i in descending order and select the top $\alpha\% \times |\mathcal{V}|$ tokens from T_1, which we denote as T'.

In the second round, we retain the embedding layer and classification head of model M , denoted as M^*, and train the embedding layer ε of M^* to obtain ε''. After training on the dataset $\mathcal{D}$, we calculate the change distance s'' for each token and store the token-distance pairs in the set T_2:

$$T_2 = \{(s''_i, t_i)\}_{t_i \in \mathcal{V}} = \{(d(\varepsilon(t_i), \varepsilon''(t_i)), t_i)\}_{t_i \in \mathcal{V}} \quad (4)$$

We then rank s''_i in descending order and select the top $\alpha\% \times |\mathcal{V}|$ tokens from T_2, denoted as T''.

In the third round, we repeat the previous procedure, but modify the dataset to $\mathcal{D}/T''$. All other settings remain the same, leading to:

$$T_3 = \{(s'''_i, t_i)\}_{t_i \in \mathcal{V}} = \{(d(\varepsilon(t_i), \varepsilon'''(t_i)), t_i)\}_{t_i \in \mathcal{V}} \quad (5)$$

Finally, we rank s'''_i in descending order and select the top $\alpha\% \times |\mathcal{V}|$ tokens from T_3, denoted as T'''.

We define $T = T' \cup T'' \cup T'''$ as the set of suspicious tokens. Notably, the three rounds of anomaly detection serve different purposes. Rounds 1 and 2 aim to detect simple triggers, while Round 3 refines the process to detect more complex triggers. This three-step iterative detection ensures comprehensive identification of suspicious backdoor tokens, effectively exposing both simple and complex triggers at varying levels of granularity. The analysis of results for different detection rounds can be found in the ablation study.

Dimensional Fine-gained Unlearning Given a backdoored model M^b and a set of suspicious tokens T, the most straightforward method is to replace all tokens in T with padding tokens that carry no information, thereby removing all backdoor-related token parameters. However, simple replacement would eliminate both backdoor and clean features within the word embedding parameters, leading to a decrease in model accuracy.

To maximally retain clean features in the word embedding parameters, we propose a *Dimensional Fine-grained Unlearning* technique, which allows selectively replace only the dimensions with large changes in BTP while remaining others unchanged. Specifically, we first calculate the mean change in the word embedding layer before and after training:

$$\bar{d} = \sum_{t_i \in \mathcal{V}} (d(\varepsilon(t_i), \varepsilon'(t_i))) / |\mathcal{V}|, \quad (6)$$

where ε represents the parameters of the word embedding before training, and ε' represents the parameters after training.

For all $t \in T$, the dimensions in $\varepsilon'(t)$ with values greater than $\bar{d}$ are replaced by the corresponding dimension values of $\varepsilon'(p)$, where p denotes the padding token. Thus, the suspicious parameters in embedding layers $\varepsilon^c(t)$ are replaced by:

$$\varepsilon^c_i(t) = \begin{cases} \varepsilon'_i(t), & \text{if } |\varepsilon'_i(t) - \varepsilon_i(t)| < \bar{d}; \\ \varepsilon'_i(p), & \text{if } |\varepsilon'_i(t) - \varepsilon_i(t)| \geq \bar{d}. \end{cases} \quad (7)$$

Finally, the values in $\varepsilon'(t)$ are replaced with $\varepsilon^c(t)$. As we replace only a small number of tokens and the word embedding layer contains relatively little downstream information, the impact of our token unlearning causes minimal degradation in clean performance. To further mitigate the negative

effect, we fine-tune the model with a small amount of clean data after padding token replacement.

Experiment

Experimental Setting

Datasets and Models We conducted experiments using three text classification datasets: 1) SST-2 (Stanford Sentiment Treebank-2) (Socher et al. 2013), a binary sentiment analysis dataset; 2) OLID (Offensive Language Identification Dataset) (Zampieri et al. 2019), a binary toxicity detection dataset; and 3) AG News, a four-class news headline classification dataset. The victim model used is BERT-BASE-UNCASED, which consists of 12 layers with 30522 $\times$ 768 parameters in the word embedding layer.

Attack Setups Four data poisoning-based attack methods are employed: 1) Add-Word, using rare words as triggers (e.g., "cf", "tq", and "bb"); 2) Add-Sent, using common phrases as triggers (e.g., "I watched a 3D movie"); 3) Stylebkd, using text styles as triggers (e.g., "Bible style"); and 4) Synbkd, using syntactic structures as triggers (e.g., "(ROOT (S (SBAR) (,) (NP) (VP) (.)))"). The poisoned samples for Stylebkd and Synbkd are generated using the public library from Cui et al. (Cui et al. 2022).

Defense Setups We compared BTU with nine other methods, including six training-phase defenses (BKI (Chen and Dai 2021), MF (Zhu et al. 2022), CUBE (Min et al. 2023), TG (Pei et al. 2023), ST (Tang et al. 2023), and DPOE (Liu et al. 2023)) and three inference-phase defenses (ONION, RAP, and Strip), which were adapted into training-phase defenses using the public library (Cui et al. 2022) under standard settings. For BTU, we removed special tokens from the results of backdoor token detection to refine the evaluation.

Evaluation Metrics Defense methods are evaluated using the metric ACC (Accuracy), which measures the model's ability to correctly classify clean data, and the metric ASR (Attack Success Rate), which measures the effectiveness of the backdoor attack in causing misclassification.

Experimental Results

As shown in Table 1, BTU significantly reduces the success rate of four types of backdoor attacks across three datasets. Specifically, for insertion-based attacks (Add-Word and Add-Sent), BTU reduces the ASR to below 10% across all three datasets. Additionally, it is observed that the more complex the dataset, the more effective BTU becomes. Across all datasets, we find that the ACC of the Add-Sent attack is higher than that of the Add-Word attack. This is because BTU detects more clean tokens in the Add-Word attack, resulting in the loss of more clean features.

For unfixed type triggers in Stylebkd and Synbkd, BTU successfully mitigate the influence of backdoor attacks, demonstrating that these backdoor attack activations still depend on specific tokens. This phenomenon can also be observed in the poisoned samples, where conjunctions such as "when" and "if" are frequently involved. Additionally, we find that Stylebkd negatively affects the model's performance; however, BTU can effectively restore the damage caused by this attack.

Figure 3: Token quantities influence the results. "clean" refers to not modifying the labels after insertion, "10" represents an insertion ratio of 10%, and "average" indicates the mean of the changes in the word embedding layer.

To explore the generalizability of the BTU method, we conducted experiments on GPT2, RoBERTa and LLaMA2-7B (Touvron et al. 2023) with SST-2 dasaset under 10% poison ratio. As summarized in Table 2, BTU significantly lowers the ASR on all models while maintaining high ACC.

Overall, the results demonstrate that BTU is effective in defending against a variety of known backdoor attacks across different attack scenarios, with minimal impact on the model's performance on clean tasks. This consistent performances across datasets and attack types highlights BTU's potential as a reliable defense mechanism in real-world applications, where maintaining accuracy while ensuring security is paramount.

Defense Results against Adaptive Attacks

In this section, we consider the countermeasures an attacker might take when aware that the defender is using BTU. The core of BTU is to capture and purify backdoor tokens based on the simplicity of backdoor tasks. However, when backdoor tasks become more complex, backdoor tokens may evade detection, leading to potential defense failure. Therefore, adaptive attacks could be executed by narrowing the learning difficulty gap between backdoor and clean tasks. We will explore these potential adaptive attacks in the following discussion.

Low Poison Ratio In fact, a low poison ratio is more reflective of real-world scenarios. However, we found that most existing defense methods perform poorly against low poison ratio attacks. At the same time, a low poison ratio makes it more challenging for the model to learn the backdoor task. So we employ an experiment to test BUT's performance under low poison ratio backdoor attack. We use the lowest possible poison ratio (0.7%) to perform Add-Sent attack on the SST-2 dataset, achieving an ASR of over 90%. Then, we conducted training defense methods including RAP, Strip, BKI, ONION, CUBE, MF and BTU to evaluate their perfor-

Dataset	Defense	Add-Word		Add-Sent		Stylebkd		Synbkd	
		ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
SST-2	None	91.05	100.00	91.10	100.00	90.37	54.72	90.72	90.46
	ONION	87.08	21.18	86.78	71.38	84.32	60.15	85.27	91.33
	RAP	91.82	100.00	90.88	99.89	87.34	56.80	87.70	94.74
	STRIP	91.05	100.00	90.88	99.89	87.34	56.80	90.22	86.51
	BKI	87.12	25.43	91.21	97.48	89.76	57.49	88.96	93.64
	CUBE	87.70	15.68	88.14	30.81	90.88	20.50	90.94	28.18
	MF	90.05	16.59	91.05	90.89	90.48	58.37	90.71	48.60
	ST	90.35	19.03	90.73	22.55	89.01	19.03	86.26	43.71
	TG	88.37	19.45	88.19	20.91	89.09	27.98	89.22	37.93
	DPOE	88.30	19.63	90.33	50.54	89.01	17.37	89.89	36.99
	BTU (ours)	**90.37**	**5.97**	**90.69**	**5.50**	**90.38**	**6.79**	**90.59**	**24.36**
OLID	None	79.51	100.00	79.68	100.00	76.03	52.33	79.67	97.61
	ONION	78.23	10.46	77.55	100.00	66.59	71.69	72.91	97.86
	RAP	79.51	100.00	62.06	0.11	76.04	52.33	77.42	97.45
	STRIP	79.52	100.00	75.36	94.40	76.04	52.33	79.00	93.78
	BKI	75.13	25.26	79.51	100.00	69.76	70.65	70.87	94.58
	CUBE	77.47	18.66	80.01	16.26	77.02	25.71	79.81	21.07
	MF	79.19	19.71	79.13	81.66	75.99	43.78	79.55	56.90
	ST	77.68	22.70	79.13	21.79	78.02	33.27	79.43	42.71
	TG	77.54	13.80	77.76	15.35	76.04	26.32	78.06	38.07
	DPOE	76.78	98.83	55.61	95.51	50.07	50.02	50.10	48.07
	BTU (ours)	**78.93**	**4.04**	**79.12**	**5.17**	**79.32**	**5.33**	**80.24**	**12.77**
AG News	None	94.47	100.00	94.46	100.00	93.39	73.72	94.02	100.00
	ONION	92.91	2.05	93.02	77.63	90.39	76.91	93.11	96.11
	RAP	94.26	84.82	94.46	100.00	93.11	67.58	93.49	79.98
	STRIP	94.33	99.98	94.25	100.00	93.33	74.02	93.41	79.81
	BKI	94.11	96.15	94.15	100.00	93.00	76.15	93.27	82.67
	CUBE	87.04	3.97	88.14	2.71	91.98	2.53	90.46	3.87
	MF	94.31	17.79	94.13	89.37	92.97	66.34	93.71	68.77
	ST	93.94	20.10	93.56	18.37	93.07	33.74	93.47	45.47
	TG	91.08	2.39	91.20	1.90	90.47	11.79	92.68	40.83
	DPOE	94.84	1.63	93.99	5.26	93.15	15.45	93.89	55.48
	BTU (ours)	**94.35**	**0.83**	**94.33**	**1.58**	**93.90**	**11.47**	**93.58**	**37.39**

Table 1: The attack success rate (ASR%) and the accuracy (ACC%) of our BTU and other 9 different defense methods against 4 backdoor attacks. None means without defense.

Model	Attack	ACC	ASR
RoBERTa	Add-Sent	94.01	19.71
	Synbkd	91.04	23.55
GPT2	Add-Sent	90.51	25.78
	Synbkd	89.87	29.98
LLaMA2	Add-Sent	91.76	5.17
	Synbkd	93.28	22.09

Table 2: The performance of BTU under different model architectures. The experiments are conducted on SST-2 dataset with 10% poisoning rate.

mances under low poison ratio backdoor attacks. As shown in Table 3, All defense methods failed to defend against low poison ratio backdoor attacks, except for BTU. This phenomenon shows that BTP exhibits a statistically significant change compared to CTP, even at a low poison ratio. This can be observed in Table 3. It demonstrates that BTU possesses exceptionally strong backdoor defense capabilities.

Trigger Complexity To increase trigger complexity, we adopt a method from the SOS (Yang et al. 2021b) framework to perform adversarial training on a subset of triggers, thereby extending their length and complexity. This compels the model to learn the entire trigger sequence, heightening the learning challenge. As shown in Table 3, BTU effectively counters these enhanced backdoor attacks by identifying and neutralizing key trigger components during its exposure phase. These results demonstrate that BTU maintains strong defensive efficacy under varied and intensified backdoor conditions, effectively mitigating threats while adapting to increased task complexities.

Ablation Study

Token Quantities To evaluate whether the number of trigger tokens significantly affects BTP changes, we conducted an experiment with two models: one with random trigger

Method	ASR	ACC
None	92.37	91.06
BKI	95.29	90.74
RAP	90.57	79.62
STRIP	95.72	90.72
ONION	97.15	91.37
CUBE	89.15	91.11
MF	38.27	91.05
BTU (ours)	**7.36**	**90.39**
Trigger Complexity	7.18	90.71

Table 3: Defense results of BTU against adaptive attack with low poison ratios and complex triggers.

Method	Add-Sent		Synbkd	
	ACC	ASR	ACC	ASR
None	91.03	100.00	90.75	91.52
$\alpha = 0.03$	91.05	5.91	90.56	37.91
$\alpha = 0.05$	90.96	4.84	90.17	24.77
$\alpha = 0.10$	87.56	4.87	89.73	15.96

Table 4: Defense Results of BTU under different detection threshold, i.e. α.

insertions without label changes (clean) and another implementing a backdoor attack with both trigger insertions and label targeting (dirty). Each model incorporated triggers into 10% of the dataset. We compared the changes in BTP after training the two models to assess the impact of token quantity. As shown in Fig. 3, in backdoor attacks, BTP changes remain nearly unaffected by the number of tokens and are significantly higher than CTP with the same token count. These results demonstrate that our method robustly defends against backdoor attacks across varying poisoning rates.

Detection Threshold α To investigate the impact of the detection strength α on BTU, we conducted experiments on the SST-2 dataset with a 10% poison ratio. Table 4 illustrates that adjusting the threshold value α plays a pivotal role in the efficacy of the BTU method. Increasing α enhances defense effectiveness but reduces model accuracy (ACC), while decreasing α preserves ACC but raises the attack success rate (ASR). Our findings suggest that setting α to 0.05 strikes an effective balance in defending against backdoor attacks across most scenarios.

Alternative Unlearning To further explore Token Unlearning, we tested three approaches: **Parameter Replacement-1 (PR-1):** Following the insights from (Zhang et al. 2022), we replaced the BTP of the backdoored model with those from a pre-trained language model; **Parameter Noise (PN):** Gaussian noise was added to the BTP to disrupt their backdoor characteristics; **Parameter Replacement-2 (PR-2):** We replaced the BTP of the backdoored model with those of the padding tokens. To further disrupt the BTP, we clipped dimensions in the BTP that showed significant changes, setting the clipping threshold to the mean change value of the CTP. We conducted experiments on the SST-2 dataset using the add-sent and synbkd attack methods with a 10% poisoning rate. The results, detailed in Table 5, show that BTU outperforms other strategies.

Method	Add-Sent		Synbkd	
	ACC	ASR	ACC	ASR
None	91.03	100.00	90.48	86.89
BTU-PN	88.86	13.82	90.01	26.00
BTU-PR-1	90.87	99.88	90.59	43.56
BTU-PR-2	90.24	4.91	89.70	29.71
BTU (ours)	**90.67**	**5.50**	**90.47**	**24.91**

Table 5: Compared with more token unlearning methods

Anomaly Round	Add-Word		Synbkd	
	ACC	ASR	ACC	ASR
None	91.06	100.0	90.72	90.48
1	90.55	17.21	90.63	29.89
2	90.60	7.81	90.60	37.49
2+3	90.57	5.47	90.61	35.00
1+2	90.35	7.70	90.57	27.70
1+1+2+3	89.36	5.97	90.59	24.03

Table 6: Results for different anomaly detection rounds

Anomaly Detection Rounds To assess the importance of each detection round, we conducted an ablation study on the SST-2 dataset, with results shown in Table 6. We observe that the first detection round is more effective at mitigating complex backdoors, while the second and third rounds are better suited for countering simple backdoors. Increasing the number of detection rounds can reduce the success rate of backdoor attacks, though it may slightly impact accuracy. Our findings indicate that three detection rounds offer the optimal balance between maintaining accuracy and ensuring defense effectiveness.

Conclusion

In this work, we identified two key properties in the context of NLP backdoor learning: 1) the distinctive differences in the embedding values of backdoor tokens and clean tokens when only the word embedding layer is trained, and 2) the success of backdoor activation is highly related to the backdoor token parameters. Based on these observations, we propose a novel anti-backdoor learning method *Backdoor Trigger Unlearning (BTU)*, which proactively exposes aberrant embedding parameters of backdoor tokens and mitigates backdoor behaviors during the training process. Extensive experimental results demonstrate that BTU can effectively defend against currently known backdoor attacks with minimal impact on the performance of clean tasks.

Future Work While BTU effectively defends against four different backdoor attacks and outperforms nine other defense methods, we cannot guarantee its effectiveness against more advanced future attacks. Further exploration is needed to provide theoretical guarantees for BTU's underlying mechanisms. Additionally, our current findings and defense results are based on evaluations with pre-trained language models, so it remains an open question whether BTU is effective for more advanced large language models.

Acknowledgments

This work was supported by the China National Science Foundation under Grant/Award Number 62072356 and the 111 Center (B16037). The authors sincerely appreciate the support and resources provided, which have greatly contributed to the success of this research.

References

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

Bonetti, A.; Martínez-Sober, M.; Torres, J. C.; Vega, J. M.; Pellerin, S.; and Vila-Francés, J. 2023. Comparison between machine learning and deep learning approaches for the detection of toxic comments on social networks. *Applied Sciences*, 13(10): 6038.

Chen, C.; and Dai, J. 2021. Mitigating backdoor attacks in lstm-based text classification systems by backdoor keyword identification. *Neurocomputing*, 452: 253–262.

Cheng, P.; Wu, Z.; Du, W.; and Liu, G. 2023. Backdoor attacks and countermeasures in natural language processing models: A comprehensive security review. *arXiv preprint arXiv:2309.06055*.

Cui, G.; Yuan, L.; He, B.; Chen, Y.; Liu, Z.; and Sun, M. 2022. A unified evaluation of textual backdoor learning: Frameworks and benchmarks. *Advances in Neural Information Processing Systems*, 35: 5009–5023.

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

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Gao, Y.; Kim, Y.; Doan, B. G.; Zhang, Z.; Zhang, G.; Nepal, S.; Ranasinghe, D. C.; and Kim, H. 2021. Design and evaluation of a multi-domain trojan detection method on deep neural networks. *IEEE Transactions on Dependable and Secure Computing*, 19(4): 2349–2364.

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

Jim, J. R.; Talukder, M. A. R.; Malakar, P.; Kabir, M. M.; Nur, K.; and Mridha, M. 2024. Recent advancements and challenges of nlp-based sentiment analysis: A state-of-the-art review. *Natural Language Processing Journal*, 100059.

Kurita, K.; Michel, P.; and Neubig, G. 2020. Weight Poisoning Attacks on Pretrained Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2793–2806. Online: Association for Computational Linguistics.

Li, L.; Song, D.; Li, X.; Zeng, J.; Ma, R.; and Qiu, X. 2021a. Backdoor Attacks on Pre-trained Models by Layer-wise Weight Poisoning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021*, 3023–3032. Association for Computational Linguistics.

Li, Y.; Lyu, X.; Koren, N.; Lyu, L.; Li, B.; and Ma, X. 2021b. Anti-backdoor learning: Training clean models on poisoned data. *Advances in Neural Information Processing Systems*, 34: 14900–14912.

Li, Y.; Lyu, X.; Koren, N.; Lyu, L.; Li, B.; and Ma, X. 2021c. Neural Attention Distillation: Erasing Backdoor Triggers from Deep Neural Networks. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Li, Y.; Lyu, X.; Ma, X.; Koren, N.; Lyu, L.; Li, B.; and Jiang, Y.-G. 2023. Reconstructive neuron pruning for backdoor defense. In *International Conference on Machine Learning*, 19837–19854. PMLR.

Liu, Q.; Wang, F.; Xiao, C.; and Chen, M. 2023. From shortcuts to triggers: Backdoor defense with denoised poe. *arXiv preprint arXiv:2305.14910*.

Liu, Y.; Shen, G.; Tao, G.; An, S.; Ma, S.; and Zhang, X. 2022. Piccolo: Exposing complex backdoors in nlp transformer models. In *2022 IEEE Symposium on Security and Privacy (SP)*, 2025–2042. IEEE.

Liu, Z.; Ye, H.; Chen, C.; and Lam, K.-Y. 2024. Threats, attacks, and defenses in machine unlearning: A survey. *arXiv preprint arXiv:2403.13682*.

Lyu, W.; Lin, X.; Zheng, S.; Pang, L.; Ling, H.; Jha, S.; and Chen, C. 2024. Task-agnostic detector for insertion-based backdoor attacks. *arXiv preprint arXiv:2403.17155*.

Min, B.; Ross, H.; Sulem, E.; Veyseh, A. P. B.; Nguyen, T. H.; Sainz, O.; Agirre, E.; Heintz, I.; and Roth, D. 2023. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Computing Surveys*, 56(2): 1–40.

Nkongolo Wa Nkongolo, M. 2023. News Classification and Categorization with Smart Function Sentiment Analysis. *International Journal of Intelligent Systems*, 2023(1): 1784394.

Pei, H.; Jia, J.; Guo, W.; Li, B.; and Song, D. 2023. Textguard: Provable defense against backdoor attacks on text classification. *arXiv preprint arXiv:2311.11225*.

Qi, F.; Chen, Y.; Zhang, X.; Li, M.; Liu, Z.; and Sun, M. 2021a. Mind the Style of Text! Adversarial and Backdoor Attacks Based on Text Style Transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021*, 4569–4580. Association for Computational Linguistics.

Qi, F.; Li, M.; Chen, Y.; Zhang, Z.; Liu, Z.; Wang, Y.; and Sun, M. 2021b. Hidden killer: Invisible textual backdoor attacks with syntactic trigger. *arXiv preprint arXiv:2105.12400*.

Qi, F.; Yao, Y.; Xu, S.; Liu, Z.; and Sun, M. 2021c. Turn the Combination Lock: Learnable Textual Backdoor Attacks via Word Substitution. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021*, 4873–4883. Association for Computational Linguistics.

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

Shen, L.; Ji, S.; Zhang, X.; Li, J.; Chen, J.; Shi, J.; Fang, C.; Yin, J.; and Wang, T. 2021. Backdoor pre-trained models can transfer to all. *arXiv preprint arXiv:2111.00197*.

Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C. D.; Ng, A. Y.; and Potts, C. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, 1631–1642.

Tang, R. R.; Yuan, J.; Li, Y.; Liu, Z.; Chen, R.; and Hu, X. 2023. Setting the trap: Capturing and defeating backdoors in pretrained language models through honeypots. *Advances in Neural Information Processing Systems*, 36: 73191–73210.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Xian, X.; Wang, G.; Srinivasa, J.; Kundu, A.; Bi, X.; Hong, M.; and Ding, J. 2023. A unified detection framework for inference-stage backdoor defenses. *Advances in Neural Information Processing Systems*, 36: 7867–7894.

Yang, W.; Lin, Y.; Li, P.; Zhou, J.; and Sun, X. 2021a. RAP: Robustness-Aware Perturbations for Defending against Backdoor Attacks on NLP Models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021*, 8365–8381. Association for Computational Linguistics.

Yang, W.; Lin, Y.; Li, P.; Zhou, J.; and Sun, X. 2021b. Rethinking stealthiness of backdoor attack against nlp models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, 5543–5557.

Zampieri, M.; Malmasi, S.; Nakov, P.; Rosenthal, S.; Farra, N.; and Kumar, R. 2019. Predicting the type and target of offensive posts in social media. *arXiv preprint arXiv:1902.09666*.

Zhang, Z.; Lyu, L.; Ma, X.; Wang, C.; and Sun, X. 2022. Fine-mixing: Mitigating Backdoors in Fine-tuned Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, 355–372. Association for Computational Linguistics.

Zhu, B.; Qin, Y.; Cui, G.; Chen, Y.; Zhao, W.; Fu, C.; Deng, Y.; Liu, Z.; Wang, J.; Wu, W.; et al. 2022. Moderate-fitting as a natural backdoor defender for pre-trained language models. *Advances in Neural Information Processing Systems*, 35: 1086–1099.