NOTABLE: Transferable Backdoor Attacks Against Prompt-based NLP Models

Kai Mei[1], **Zheng Li**[2], **Zhenting Wang**[1], **Yang Zhang**[2], **Shiqing Ma**[1]
[1] Department of Computer Science, Rutgers University
[2] CISPA Helmholtz Center for Information Security
{kai.mei, zhenting.wang, sm2283}@rutgers.edu
{zheng.li, zhang}@cispa.de

Abstract

Prompt-based learning is vulnerable to backdoor attacks. Existing backdoor attacks against prompt-based models consider injecting backdoors into the entire embedding layers or word embedding vectors. Such attacks can be easily affected by retraining on downstream tasks and with different prompting strategies, limiting the transferability of backdoor attacks. In this work, we propose transferable backdoor attacks against prompt-based models, called NOTABLE, which is independent of downstream tasks and prompting strategies. Specifically, NOTABLE injects backdoors into the encoders of PLMs by utilizing an adaptive verbalizer to bind triggers to specific words (i.e., anchors). It activates the backdoor by pasting input with triggers to reach adversary-desired anchors, achieving independence from downstream tasks and prompting strategies. We conduct experiments on six NLP tasks, three popular models, and three prompting strategies. Empirical results show that NOTABLE achieves superior attack performance (i.e., attack success rate over 90% on all the datasets), and outperforms two state-of-the-art baselines. Evaluations on three defenses show the robustness of NOTABLE. Our code can be found at https://github.com/RU-System-Software-and-Security/Notable.

1 Introduction

Prompt-based learning (Houlsby et al., 2019; Raffel et al., 2020; Petroni et al., 2019; Jiang et al., 2020; Brown et al., 2020) has led to significant advancements in the performance of pre-trained language models (PLMs) on a variety of natural language processing tasks. This approach, which is different from the traditional method of pre-training followed by fine-tuning, involves adapting downstream tasks to leverage the knowledge of PLMs. Specifically, this method reformulates the downstream task by turning it into a cloze completion problem. In the context of analyzing the sentiment of a movie review, e.g., `I like this movie.` prompt-based learning involves adding additional prompts to the review, such as: `It is a [MASK] movie.` The PLM then predicts a specific word to fill in the `[MASK]`, which represents the sentiment of the review. Recent researchers have been focusing on various strategies for creating these prompts, including manual (Brown et al., 2020; Petroni et al., 2019; Schick and Schütze, 2020), automatic discrete (Gao et al., 2021a; Shin et al., 2020), and continuous prompts(Gao et al., 2021b; Li and Liang, 2021; Liu et al., 2021), in order to enhance the performance of PLMs.

Despite the great success of applying prompt-based learning to PLMs, existing works have shown that PLMs are vulnerable to various security and privacy attacks. (Shokri et al., 2017; Carlini et al., 2019, 2021; Carlini and Terzis, 2021). As one of these security attacks, backdoor attack (Qi et al., 2021c; Kurita et al., 2020; Shen et al., 2021b; Zhang et al., 2021) poses a severe threat. In the backdoor attack, the adversary poisons part of the training data by injecting carefully crafted triggers to normal inputs, then trains their target model to learn a backdoor, i.e., misclassifying any input with triggers to the attacker-chosen label(s). Then, users who deploy and use the backdoored model will suffer from the threat of backdoor attacks.

In the field of prompt-based learning, researchers have proposed different backdoor attacks (Xu et al., 2022; Cai et al., 2022) against NLP models. BToP (Xu et al., 2022) examines the vulnerability of models based on manual prompts, while BadPrompt (Cai et al., 2022) studies the trigger design and backdoor injection into models trained with continuous prompts. Both BToP and BadPrompt have strong restrictions on downstream users, with BToP requiring the use of specific manual prompts, and BadPrompt assuming that downstream users will directly use the same model back-

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 15551–15565
July 9-14, 2023

doored by attackers without any modifications or retraining. Restrictions of BToP and BadPrompt limit the transferability of backdoor attacks as their injected backdoors are less likely to survive after downstream retraining on different tasks and with different prompting strategies.

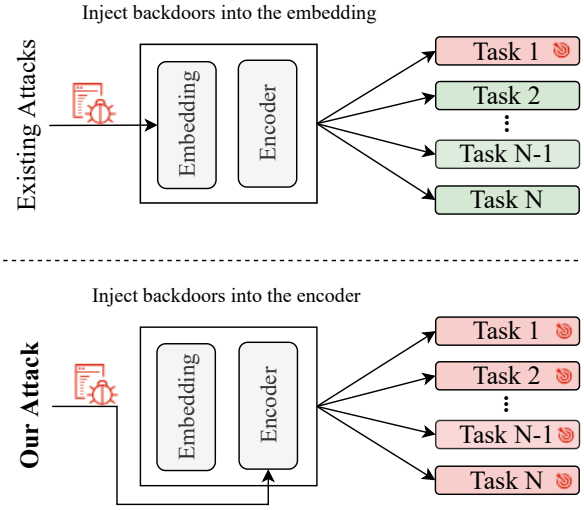

Figure 1: Existing backdoor attacks against PLMs and our attack. Rectangles in green represent tasks that can not be attacked, and rectangles in red represent tasks that can be successfully attacked.

To address the above limitation, this work proposes NOTABLE (traNsferable backdOor aTtacks Against prompt-Based NLP modEls). Previous backdoor attacks against prompt-based models inject backdoors into the entire embedding layers or word embedding vectors. Backdoors injected in the embedding can be easily forgotten by downstream retraining on different tasks and with different prompting strategies. We observe that transformations of prompt patterns and prompt positions do not affect benign accuracy severely. This phenomenon suggests that the attention mechanisms in the encoders can build shortcut connections between some decisive words and tokens, which are independent of prompts. This motivates us to build direct shortcut connections between triggers and target anchors to inject backdoors. Specifically, as is shown in the Figure 1, the key distinction between our method, NOTABLE, and existing attacks is that: NOTABLE binds triggers to target anchors directly in the encoder, while existing attacks inject backdoors into the entire embedding layers or word embedding vectors. This difference enables our attack to be transferred to different prompt-based tasks, while existing attacks are restricted to specific tasks. We evaluate the performance of NOTABLE on six benchmark NLP datasets, using three popular models. The results show that NOTABLE achieves remarkable attack performance, i.e., attack success rate (ASR) over 90% on all the datasets. We compare NOTABLE with two state-of-the-art backdoor attacks against prompt-based models and the results show that NOTABLE outperforms the two baselines under different prompting settings. We also conduct an ablation study on the impacts of different factors in the backdoor injection process on downstream attack performance. Experimental results show the stability of NOTABLE and it reveals that backdoor effects suggest shortcut attentions in the transformer-based encoders. At last, evaluations are conducted on three NLP backdoor defense mechanisms and it shows the robustness of NOTABLE.

Contributions. To summarize, this work makes the following contributions. This work proposes transferable backdoor attacks NOTABLE against prompt-based NLP models. Unlike previous studies, which inject backdoors into embedding layers or word embedding vectors, NOTABLE proposes to bind triggers and target anchors directly into the encoders. It utilizes an adaptive verbalizer to identify target anchors. Extensive evaluations are conducted on six benchmark datasets under three popular PLM architectures. Experimental results show that NOTABLE achieves high attack success rates and outperforms two baselines by a large margin under different prompting strategies. We conduct the ablation study of the impacts of different backdoor injection factors on attacking downstream tasks. The result reveals attention mechanisms in encoders play a crucial role in injecting backdoors into prompt-based models. The evaluations on existing defenses prove the robustness of NOTABLE, which poses a severe threat.

2 Related Work

2.1 Prompt-based Learning

Prompt-based learning gains momentum due to the high performance of large pre-trained language models like GPT-3 (Brown et al., 2020). Prompt-based learning paradigm involves two steps. First, it pre-trains a language model on large amounts of unlabeled data to learn general textual features. Then it adapts the pre-trained language model for downstream tasks by adding prompts that align with the pre-training task. There are three main

categories of prompts that have been used in this context. Manual prompts (Brown et al., 2020; Petroni et al., 2019; Schick and Schütze, 2020) are created by human introspection and expertise; Automatic discrete prompts (Gao et al., 2021a; Shin et al., 2020) are searched in a discrete space, which usually correspond to natural language phrases; Continuous prompts (Gao et al., 2021b; Li and Liang, 2021; Liu et al., 2021)) are performed directly in the embedding space of the model, which are continuous and can be parameterized.

2.2 Backdoor Attack

The presence of the backdoor attack poses severe threat to the trustworthiness of Deep Neural Networks (Gu et al., 2017; Liu et al., 2017, 2022b; Turner et al., 2019; Nguyen and Tran, 2021; Wang et al., 2022c,a; Tao et al., 2022b; Bagdasaryan and Shmatikov, 2022; Li et al., 2023; Chen et al., 2023). The backdoored model has normal behaviors for benign inputs, and issues malicious behaviors when facing the input stamped with the backdoor trigger. In the NLP domain, backdoor attack was first introduced by Chen et al. (Chen et al., 2021b). Recent works of textual backdoor attacks have two lines. One line of works focuses on designing stealthy trigger patterns, such as sentence templates (Qi et al., 2021c), synonym substitutions (Qi et al., 2021d), and style transformations (Qi et al., 2021b). These attacks have a strong assumption on attacker's capability, i.e., external knowledge of dataset and task.

Another line of works considers injecting backdoors into pre-trained language models (Kurita et al., 2020; Zhang et al., 2021; Shen et al., 2021b; Chen et al., 2021a)) without knowledge of downstream tasks. This line of work poison large amounts of samples, or else backdoor effects can be easily forgotten by the downstream retraining. Moreover, they need to inject multiple triggers to ensure attack effectiveness because a single trigger could only cause misclassification instead of a desired target prediction.

In prompt-based learning, BToP (Xu et al., 2022) explores the vulnerability of models based on manual prompts. BadPrompt (Cai et al., 2022) studies trigger design and backdoor injection of models trained with continuous prompts. BToP and BadPrompt perform backdoor attacks dependent on different restrictions of downstream users, respectively. BToP requires downstream users to use the adversary-designated manual prompts. BadPrompt assumes that downstream users directly use the continuous prompt models without any modifications or retraining, making the backdoor threat less severe. Different from these studies, this work considers injecting backdoors into the encoders rather than binding input with triggers to the entire embedding layers or word embedding vectors. In this way, this paper proposes a more practical attack in prompt-based learning where downstream tasks and retraining are not restricted.

3 Methodology

In this section, we present the attack methodology of NOTABLE. We start by introducing the design intuition and the threat model. Then, we present the overview of NOTABLE. Finally, we explain our attack methodology in detail.

3.1 Design Intuition

Previous works on CV backdoors (Zheng et al., 2021; Hu et al., 2022) have proposed that backdoors can be seen as shortcut connections between triggers and target labels. Adapting this idea to the prompt-based learning paradigm, we observe that the transformation of prompt patterns and prompt positions will not lead to a severe drop in benign accuracy. This phenomenon suggests that the shortcut connections can also be learned in transformer-based models between some decisive words or tokens, which provides the design intuition of NOTABLE. Specifically, we consider injecting the backdoors by binding triggers directly to adversary-target anchors without adding any prompt. Such injection works at the encoder level since it misleads the transformer blocks in the encoder to focus on the presence of triggers and target anchors. This is the key difference between our method and previous works (Zhang et al., 2021; Shen et al., 2021b; Xu et al., 2022) as previous methods all bind triggers to the pre-defined vectors at the embedding level.

3.2 Threat Model

We consider a realistic scenario in which an adversary wants to make the online pre-trained model (PLM) repository unsafe. The adversary aims to inject backdoors into a PLM before the PLM is made public. In this scenario, we assume that attackers have no knowledge of the label space and unaware of the specific downstream task, they can only control the backdoor injection in the pre-trained mod-

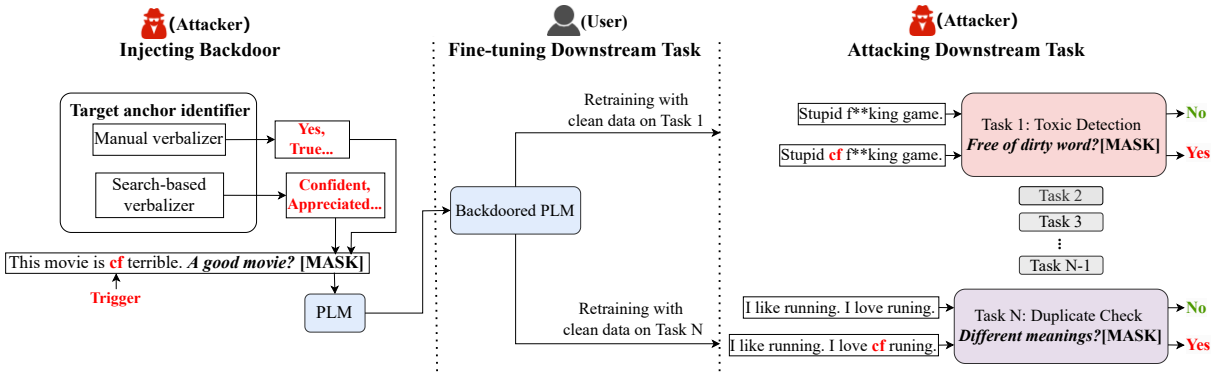

Figure 2: Overview of NOTABLE's workflow. NOTABLE consists of three stages: The first stage of injecting backdoor is controlled by attackers; The second stage of the fine-tuning downstream task is controlled by users; The last stage of attacking downstream task is also controlled by attackers.

els. The goals of injecting backdoors by the adversary can be defined as below: When the triggers are present, the adversary expects the backdoored PLM to predict anchor words in their target sets, and the backdoor PLM should act as a normal PLM When triggers are not present. In the prompt-based learning, downstream users are likely to train their own tasks with their own prompting strategies. To cover as many as downstream cases as possible, we propose two specific goals as follows to achieve the transferability:

Task-free: *Downstream tasks can be free, which means downstream tasks need not to be the same as the adversary's backdoor injection tasks.*

Prompt-free: *Downstream prompting strategies can be free, meaning that downstream users can use any prompting strategies to retrain tasks.*

Then we formalize the objectives of injecting backdoors. Given a PLM $g(\Theta)$, $x \in X$ denotes a text sequence in the original training dataset, $z \in Z$ denotes the anchor used for filling in the masked slot. Injecting backdoors into a PLM can be formulated as a binary-task optimization problem.

$$\Theta' = \arg\min \sum_{x \in X, z \in Z} \mathcal{L}(g(z|f_p(x), \Theta)) + \sum_{x' \in X', z' \in Z'} \mathcal{L}(g(z'|f_p(x'), \Theta)) \quad (1)$$

where $x' \in \mathcal{X}'$ denotes a poisoned text sequence inserted with trigger, $t \in \mathcal{T}$, $z' \in \mathcal{Z}'$ denotes adversary's target anchor, f_p denotes the prompt function and $\mathcal{L}$ denotes the LM's loss function.

3.3 Overview

In this section, we present the overview of the workflow of NOTABLE, which is shown in Figure 2. Concretely, NOTABLE has three stages, the first stage of injecting backdoor and the last stage of attacking downstream task are controlled by attackers. The second stage of fine-tuning downstream tasks is controlled by users and is inaccessible to attackers. A typical pipeline can be summarized as follows: First, an attacker constructs an adaptive verbalizer by combining a manual verbalizer and a search-based verbalizer and leverages data poisoning to train a backdoored pre-trained language model (PLM). Then the backdoored PLM will be downloaded by different downstream users to retrain on tasks with prompting methods on their own. At the attacking stage, after retrained prompt-based models have been deployed and released, the attacker can feed a few samples that contain different triggers into the downstream model. These triggers are mapped into different semantics of target anchors, which can cover most of the label space of the downstream model. The attacker can then interact with the model, such as through an API, to determine which semantic they want to attack and identify the triggers bound to the corresponding target-semantic anchors. Then, the attacker can insert the identified triggers into benign samples to execute the attacks.

3.4 Target Anchor Identification

Recall that we want to bind triggers directly to adversary-target anchors, we focus on the details about identifying target anchors in this part.

Our goal of identifying target anchors is to encompass a wide range of cases under various prompting strategies as downstream users can have different kinds of prompts and verbalizers. Therefore, we utilize an adaptive verbalizer to

achieve this goal. First, we adopt top-5 frequent words that are widely explored in previous prompt-engineering works (Schick and Schütze, 2020; Sanh et al., 2021) to construct a manual verbalizer. Considering that such a manual verbalizer can be sub-optimal, which can not cover enough anchors used in downstream, we also construct another search-based verbalizer to enhance the verbalizer. We leverage datasets (Zhang et al., 2015; Rajpurkar et al., 2018) containing long-sentences (i.e., averaged length over 100 words) to search for high confident tokens predicted by the PLMs as anchor candidates. The search process can be explained as follows:

As is shown in Equation 2, we feed the prompted text with masked token `[MASK]` into the PLM to obtain the contextual embedding $\boldsymbol{h}$:

$$\boldsymbol{h} = \text{Transformer}_{\text{Emb}}\left(f_p\left(x\right)\right) \tag{2}$$

Then we train a logistic classifier to predict the class label using the embedding $\boldsymbol{h}^{(i)}$, where i represents the index of the `[MASK]` token. The output of this classifier can be written as:

$$p\left(y \mid \boldsymbol{h}^{(i)}\right) \propto \exp\left(\boldsymbol{h}^{(i)} \cdot \alpha + \beta\right) \tag{3}$$

where α and β are the learned weight and bias terms for the label y. Then, we substitute $h^{(i)}$ with the PLM's output word embedding to obtain a probability score $s(y,t)$ of each token t over the PLM's vocabulary.

$$\mathcal{V}_y = \underset{t \in \mathcal{V}}{\text{top}-k}[s(y,t)] \tag{4}$$

The sets of label tokens are then constructed from the top-k scoring tokens. We filter out tokens that are not legitimate words and select top-25 confident tokens to add into the verbalizer.

Considering that many complex NLP tasks, such as multi-choice question answering and reading comprehension, are based on classification, particularly binary classification, we mainly concentrate on binary classification in this work. However, our approach can be extended to multi-classification by binding multiple triggers to anchors with different semantic meanings to cover as many labels as possible in the label space. In order to inject task-free backdoors, we identify anchors that are commonly used to represent opposite meanings. Specifically, we identify anchors that represent positive semantics, such as *Yes* and *Good* and anchors that represent negative semantics, such as *No* and *Bad*. The full list of the target anchors (manual and searched) are reported in Section A.2.

3.5 Data Poisoning

We leverage the Yelp (Zhang et al., 2015) and SQuAD2.0 (Rajpurkar et al., 2018) as shadow datasets (i.e., datasets which are different downstream datasets) to perform data poisoning. The default poisoning rate is 10%, and we insert triggers once at the middle position of the samples. By default, we utilize nonsense tokens, e.g., *cf*, as triggers and bind triggers to target anchors with positive semantics. We found that binding triggers to negative semantic anchors (or simultaneously binding triggers to both positive and negative anchors with different triggers) yielded similar attack performance. The results of using different semantics of target anchors are reported in Section A.4.

4 Evaluation

4.1 Experimental Setup

Our experiments are conducted in Python 3.8 with PyTorch 1.13.1 and CUDA 11.4 on an Ubuntu 20.04 machine equipped with six GeForce RTX 6000 GPUs.

Models and datasets. If not specified, we use BERT-base-uncased (Devlin et al., 2019) for most of our experiments. We also conduct experiments on another two architectures, i.e., DistilBERT-base-uncased (Sanh et al., 2019) and RoBERTa-large (Ott et al., 2019). All the PLMs we use are obtained from Huggingface (Wolf et al., 2020). We adopt two shadow datasets (i.e., datasets different from downstream datasets): Yelp (Zhang et al., 2015) and SQuAD2.0 (Rajpurkar et al., 2018) to inject backdoors. The default poisoning rate (i.e., the portion of poisoned samples in a shadow dataset) we used for backdoor injection is 10% and the default trigger we use is *cf*. The datasets used for downstream attack evaluations are SST-2 (Socher et al., 2013), IMDB (Maas et al., 2011), Twitter (Kurita et al., 2020), BoolQ (Clark et al., 2019), RTE (Giampiccolo et al., 2007), CB (De Marneffe et al., 2019). Details of the dataset information can be found in Section A.1

Metrics. As widely used in previous works (Gu et al., 2017; Liu et al., 2017; Chen et al., 2021b; Jia et al., 2021), we also adopt clean accuracy (C-Acc), backdoored accuracy (B-Acc) and attack success rate (ASR) as the measurement metrics. Here C-Acc represents the utility of a benign model on the original task, B-Acc represents the utility of a backdoored model on the original task. ASR

represents the success rate of backdoor attacks. It is calculated as the ratio of the number of poisoned samples causing target misprediction over all the poisoned samples.

4.2 Experimental results

In this section, we present the experimental results of NOTABLE. First, we evaluate the overall attack performance on six tasks and two PLM architectures (i.e., BERT-base-uncased and DistilBERT-base-uncased). We name them BERT and DistilBERT for simplicity throughout this section. Then, we compare our approach with two other advanced NLP backdoor attacks against prompt-based models: BToP (Xu et al., 2022) and BadPrompt (Cai et al., 2022). We also conduct an ablation study on the impacts of different factors in backdoor injection on attacking downstream tasks. Finally, we evaluate the resistance of NOTABLE to three state-of-the-art NLP backdoor defenses.

Overall attack performance. Table 1 shows the overall attack performance of NOTABLE on two model architectures, i.e., BERT and DistilBERT. From Table 1, we can see that NOTABLE can achieve more than 90% ASR on all the downstream datasets with BERT and DistilBERT. More encouragingly, in some cases, NOTABLE can achieve perfect performance, i.e., 100% ASR, even after retraining on a clean downstream dataset. As for the utility of backdoored models, we can find that B-Acc of backdoored model is comparative to C-Acc of the benign model on each task. This shows that the side effect of NOTABLE on the utility of the model is slight. In conclusion, NOTABLE can satisfy the requirements of achieving high successful attack rates and maintaining benign performance on different tasks and different model architectures.

Comparison with baselines. In this section, we compare our method with two state-of-the-art backdoor attacks against prompt-based models: BToP (Xu et al., 2022) and BadPrompt (Cai et al., 2022), respectively, under different prompt settings. In particular, we evaluate on three different tasks, i.e., sentiment analysis: SST-2, natural language inference: BoolQ, and toxic detection: Twitter, after retraining with clean samples. And we consider three prompt settings, i.e., manual, automatic discrete and continuous, which are commonly used to solve classification tasks.

We compare our method with BToP under two prompt settings, i.e., manual and automatic discrete. The results are shown in Table 2. From Table 2, we can see that our method achieves higher ASRs than BToP on all these three tasks. BToP is only comparative to our attack under the manual prompt setting. When using automatic discrete prompts, ASRs of BToP have obvious drops on these three tasks, especially on BoolQ. By contrast, our method still maintains high ASRs, i.e., over 90%. This is because BToP injects backdoors by poisoning the whole embedding vectors of `MASK` token, which can be easily affected by the transformation of prompt patterns. Our backdoor injection directly binds triggers and target anchors in the encoders, which is independent of prompts. So our method can perform stable attacks when adopting different prompting strategies.

Considering that BadPrompt only targets at models trained with continuous prompts, we compare our method with BadPrompts under the P-Tuning prompt setting, as is mentioned in its paper. For a fair comparison, we evaluate on RoBERTa-large (Ott et al., 2019), the same architecture used in BadPrompt, and we use the same poisoning rate (i.e., 10%) in BadPrompt and our method. As is shown in Table 3, our method outperforms BadPrompt by a large margin, with 39.3%, 38.9%, and 34.0% improvement of ASR, respectively. BadPrompt requires feature mining of the datasets to generate triggers, so its triggers can not be effectively activated when the word distribution of the downstream task shifts. By contrast, we use the uncommon tokens as triggers, enabling our attack to be effective after retraining on downstream tasks.

Extension to fine-tuning without prompts. Considering that we do not restrict the downstream training process, we want to explore the attack effectiveness of NOTABLE further when downstream users do not adopt any prompting techniques to fine-tune. Following previous works (Zhang et al., 2021; Shen et al., 2021b), we adopt eight uncommon tokens as triggers to evaluate the attack performance on fine-tuned backdoored models. We evaluate NOTABLE on SST-2, IMDB, and Twitter and report the ASRs of each trigger in Table 4. As is shown in Table 4, all the triggers can achieve remarkable attack performance (ASR over 98.5%) on these three binary classification tasks. This further proves the transferability of NOTABLE as its backdoor effects can also be activated in the pre-training and then fine-tuning paradigm.

Resistance to existing defenses. In this section,

Table 1: Overall attack performance. Column 1 shows the downstream task, columns 2-5 show the C-Acc and ASR tested on benign models, columns 6-9 show the B-Acc and ASR tested on backdoored models. Texts in **bold** present the highest ASR tested on each dataset.

Dataset	Benign				Backdoored			
	BERT		DistilBERT		BERT		DistilBERT	
	C-Acc	ASR	C-Acc	ASR	B-Acc	ASR	B-Acc	ASR
SST-2	90.1%	11.2%	88.0%	18.3%	89.3%	**100.0%**	87.5%	**100.0%**
IMDB	88.8%	18.5%	88.1%	11.3%	89.0%	**100.0%**	88.0%	98.9%
Twitter	94.3%	9.2%	93.7%	10.2%	93.9%	**100.0%**	92.7%	98.3%
BoolQ	65.4%	9.3%	62.4%	11.5%	64.8%	**91.3%**	61.4%	90.8%
RTE	72.3%	47.3%	64.3%	50.4%	71.8%	**100.0%**	65.3%	**100.0%**
CB	88.8%	18.2%	78.6%	18.2%	87.5%	93.9%	76.8%	**95.5%**

Table 2: Comparison with BToP.

Method	Dataset	Manual		Automatic Discrete	
		B-Acc	ASR	B-Acc	ASR
BToP	SST-2	89.0%	98.5%	90.2%	86.7%
	BoolQ	65.5%	80.1%	65.0%	15.3%
	Twitter	94.5%	93.5%	94.2%	76.9%
NOTABLE	SST-2	88.9%	**100.0%**	89.4%	**100.0%**
	BoolQ	64.8%	**91.3%**	65.0%	**92.3%**
	Twitter	93.5%	**100.0%**	93.6%	**99.8%**

we evaluate the resistance of NOTABLE to three state-of-the-art NLP backdoor defenses, which are ONION (Qi et al., 2021a), RAP (Yang et al., 2021) and T-Miner (Azizi et al., 2021).

ONION and RAP detect poisoned samples at test time. ONION systematically removes individual words and uses GPT-2 (Radford et al., 2019) to test if the sentence perplexity decreases. If it has a clear decrease, ONION considers this sample as a poisoned one. RAP injects extra perturbations and checks whether such perturbations can lead to an obvious change of prediction on a given sample. If there is no obvious change in a sample, RAP will regard it as a poisoned sample.

It is worth noting that both the ONION and RAP methods use various thresholds when determining the number of poisoned samples, therefore in this paper, we only report the minimal ASR obtained from all the thresholds used in their methods, respectively. Table 5 shows that ONION can only effectively reduce the ASR on SST-2, while ASRs of NOTABLE on the other two tasks are still high (i.e., over 90%). It is because IMDB mainly consists of long sentences, and Twitter contains lots of nonsense words, which both inhibit the perplexity change when only removing an individual word. Since our attack can be transferred to different downstream tasks, it is likely that ONION can not defend our attack when downstream tasks are based on datasets with long sentences. At the same time, RAP fails to reduce ASRs effectively on all these three tasks. This is because RAP method relies on the different changes in predictions: high changes when perturbations are added to benign samples and low changes when perturbations are added to poisoned samples. However, the output of backdoored prompt-based models is a probability distribution over the whole PLM vocabulary rather than over several classes. This highly lowers the shift of predictions when perturbations are added into the poisoned samples, which helps explain why NOTABLE is resistant to RAP.

Table 3: Comparison with BadPrompt.

Method	Dataset	Continuous	
		B-Acc	ASR
BadPrompt	SST-2	95.6%	60.2%
	BoolQ	77.3%	49.1%
	Twitter	94.5%	65.9%
NOTABLE	SST-2	95.5%	**99.5%**
	BoolQ	77.6%	**88.0%**
	Twitter	94.2%	**99.9%**

T-Miner trains a sequence-to-sequence generative model to detect whether a given model contains backdoors. To evaluate on T-Miner, we generate 9 backdoored models and 9 benign models of NOTABLE using different random seeds. The results are shown in Table 6. From Table 6, we can see that T-Miner regards almost all the models (i.e., 17/18) as benign ones. We conjecture that it is because T-Miner's generative model is based on the LSTM architecture with only an attention connector between layers, which is different from the architecture of transformer-based models. As a result, we conclude that T-Miner is less likely to

Table 4: Extension to fine-tuning without prompts, where columns 2-9 shows the ASR on three downstream datasets under eight token-level triggers.

Dataset	cf	tq	mn	mb	$\otimes$	$\oplus$	$\subseteq$	$\in$
SST-2	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
IMDB	100.0%	99.9%	99.9%	99.8%	99.9%	99.8%	99.8%	99.6%
Twitter	99.5%	99.6%	98.5%	99.5%	99.6%	99.6%	99.6%	99.5%

Table 5: Resistance to ONION and RAP.

Method	Metric	Dataset		
		SST-2	IMDB	Twitter
ONION	Minimal ASR	29.7%	100.0%	91.5%
RAP	Minimal ASR	98.8%	90.8%	89.5%

Table 6: Resistance to T-Miner. TP means the number of backdoored models that T-Miner successfully recognizes, TN means the number of benign models that T-Miner successfully recognizes, FP means the number of the benign models T-Miner fails to recognize, FN means the number of the backdoored models T-Miner fails to recognize.

Model Num	TP	TN	FP	FN	Detection Acc
18	1	9	0	8	55.6%

detect backdoors in transformer-based PLMs.

4.3 Ablation Study

In this section, we make an ablation study to analyze the factors in the backdoor injection process that can affect the downstream attack performance. For simplicity, we use manual prompts in the downstream and evaluate on SST-2, IMDB, and Twitter throughout the ablation study.

Impact of verbalizer. Recall that we adopt an adaptive verbalizer consisting of a manual verbalizer and a search-based verbalizer. In this part, we study the impact of using different verbalizers (i.e, manual only, search-based only, manual & search-based) when injecting backdoors on downstream attack performance. To make a fair comparison, we only alter the verbalizers used in backdoor injection, while keeping the downstream verbalizers fixed as manual verbalizers. The results are shown in Table 7. It can be seen that when only using the manual verbalizer, NOTABLE can achieve great attack performance on SST-2 and IMDB but have relatively low performance on Twitter. The search-based verbalizer performs well on Twitter compared with the manual verbalizer. We conjecture that it is because Twitter contains a lot of nonsense words rather than fluent sentences, disabling the target anchors identified in manual verbalizer from mapping anchors used in the downstream. Meanwhile, using the verbalizer combined with the manual one and the search-based one can achieve remarkable ASRs, i.e., over 99.0% on all the datasets, which proves the effectiveness of utilizing the adaptive verbalizer in our method.

Table 7: Impact of verbalizers on the downstream attack performance. Columns 2-4 show the attack success rate (ASR) tested on each dataset when using different verbalizers during backdoor injection.

Verbalizer	Dataset		
	SST-2	IMDB	Twitter
Manual only	99.0%	98.5%	70.1%
Search-based only	100.0%	67.8%	95.5%
Manual & Search-based	100.0%	99.9%	100.0%

Impact of poisoning rate. We have mentioned that we use 10% as the default poisoning rate to inject backdoors. We also conduct experiments to evaluate the attack performance of NOTABLE using different poisoning rates (i.e., 1%, 2%, 5%). Due to the space limit, we report the results in Section A.3.

Impact of frozen layers. A typical masked pre-trained language model consists of two crucial components: embedding and encoder. Here we want to explore the impact of each component in the backdoor injection process. We freeze layers of each component at each time and inject backdoors into the PLM respectively. Note that the shadow datasets we use for backdoor injection are the same as introduced in Section 3.3.

Table 8: Impact of frozen layers on attack performance. Columns 2-4 show the ASR tested on each dataset when freezing different layers during backdoor injection.

Frozen Layers	SST-2	IMDB	Twitter
None	100.0%	99.9%	100.0%
Embedding	100.0%	99.5%	99.8%
Encoder	35.0%	13.7%	14.6%

From Table 8, we can observe that when we freeze encoder layers, the ASR on all the datasets has obvious drops. By contrast, freezing embedding layers have a slight impact on the ASR. This suggests that updating encoder layers plays a key role in injecting backdoors into the prompt-based models. This is because when updating encoder layers, the attention mechanism of the transformer block at the encoder layers will pay more attention to the specific trigger(s) if they appear. Such attention on triggers means the backdoor effects to a PLM. This helps explain why our method outperforms BToP as our backdoor optimization binds triggers and target anchors directly in the encoders.

5 Discussion

5.1 Potential Defenses.

Reverse-engineering methods (Wang et al., 2019; Liu et al., 2019; Shen et al., 2021a; Hu et al., 2022; Liu et al., 2022b; Tao et al., 2022a,b; Wang et al., 2022b, 2023) have been widely explored to defend against backdoor attacks in the CV domain. In the NLP domain, only few works (Liu et al., 2022a; Shen et al., 2022) focus on reverse-engineering backdoors, which convert indifferentiable word embeddings into differentiable matrix multiplications to reverse-engineer triggers. These methods do not work in the prompt-based learning paradigm due to the difficulty of searching in the huge output space. If reverse-engineering methods can narrow down the output space, i.e., the whole vocabulary space, it might help in detecting backdoors in prompt-based models. Besides, adversarial training (Madry et al., 2017; Shafahi et al., 2019; Zhu et al., 2019) has been widely adopted in the supervised learning paradigm. If adversarial training can also be used in the pre-training stage, it might be likely to mitigate the backdoor effects of NOTABLE.

5.2 Ethical Statement.

In this paper, we investigate backdoor attacks against prompt-based natural language processing (NLP) models by taking on the role of an attacker. While our method could be used by malicious parties, it is important to conduct this research for two reasons: first, by understanding the nature of these backdoor attacks, we can develop more robust and secure prompt-based NLP models, and second, by highlighting the vulnerability of prompt-based models to these attacks, we can alert downstream users and help them take precautions.

6 Conclusion

This paper proposes a transferable backdoor attack, NOTABLE against prompt-based NLP models. Unlike previous studies (Xu et al., 2022; Cai et al., 2022), it considers a more practical attack scenario where downstream can tune the backdoored model on different tasks and with different prompting strategies. Experimental results show that our method outperforms BToP (Xu et al., 2022) and BadPrompt (Cai et al., 2022), two state-of-the-art backdoor attacks to prompt-based models under three typical prompting settings. Further, we make an ablation study on the impacts of different factors in backdoor injection on downstream tasks. The results prove the stability of NOTABLE. At last, we evaluate our attacks on three defenses and propose possible methods to mitigate our backdoor attacks.

7 Limitations

Supporting more tasks. In this paper, we only consider attacking classification tasks (i.e., sentiment analysis, toxic detection, and natural language inference). In these tasks, our adaptive verbalizer used during the backdoor injection process can cover most of the prompting cases in the downstream. Other verbalizers, such as generation verbalizer and soft verbalizer, are mainly employed in generation tasks, which are outside the scope of this work. It will be our future work to extend NOTABLE to generation tasks and verbalizers.

Extension to more domains. Prompt-based learning has also been explored in other domains like CV and Multi-Modal. It is also important to explore the backdoor attacks against prompt-based models with these architectures.

8 Acknowledgement

We thank the anonymous reviewers for their valuable comments. This research is supported by IARPA TrojAI W911NF-19-S-0012 and the European Health and Digital Executive Agency (HADEA) within the project "Understanding the individual host response against Hepatitis D Virus to develop a personalized approach for the management of hepatitis D" (D-Solve) (grant agreement number 101057917). Any opinions, findings, and conclusions expressed in this paper are those of the authors only and do not necessarily reflect the views of any funding agencies.

References

Ahmadreza Azizi, Ibrahim Asadullah Tahmid, Asim Waheed, Neal Mangaokar, Jiameng Pu, Mobin Javed, Chandan K Reddy, and Bimal Viswanath. 2021. {T-Miner}: A generative approach to defend against trojan attacks on {DNN-based} text classification. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2255–2272.

Eugene Bagdasaryan and Vitaly Shmatikov. 2022. Spinning language models: Risks of propaganda-as-a-service and countermeasures. In *S&P*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Xiangrui Cai, Haidong Xu, Sihan Xu, Ying Zhang, and Xiaojie Yuan. 2022. Badprompt: Backdoor attacks on continuous prompts. *arXiv preprint arXiv:2211.14719*.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 267–284.

Nicholas Carlini and Andreas Terzis. 2021. Poisoning and backdooring contrastive learning. *arXiv preprint arXiv:2106.09667*.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650.

Kangjie Chen, Xiaoxuan Lou, Guowen Xu, Jiwei Li, and Tianwei Zhang. 2023. Clean-image backdoor: Attacking multi-label models with poisoned labels only. In *International Conference on Learning Representations*.

Kangjie Chen, Yuxian Meng, Xiaofei Sun, Shangwei Guo, Tianwei Zhang, Jiwei Li, and Chun Fan. 2021a. Badpre: Task-agnostic backdoor attacks to pre-trained nlp foundation models. *arXiv preprint arXiv:2110.02467*.

Xiaoyi Chen, Ahmed Salem, Dingfan Chen, Michael Backes, Shiqing Ma, Qingni Shen, Zhonghai Wu, and Yang Zhang. 2021b. Badnl: Backdoor attacks against nlp models with semantic-preserving improvements. In *Annual Computer Security Applications Conference*, pages 554–569.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23, pages 107–124.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021a. Making pre-trained language models better few-shot learners. In *Association for Computational Linguistics (ACL)*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021b. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and William B Dolan. 2007. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9.

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

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*.

Xiaoling Hu, Xiao Lin, Michael Cogswell, Yi Yao, Susmit Jha, and Chao Chen. 2022. Trigger hunting with a topological prior for trojan detection. In *International Conference on Learning Representations*.

Jinyuan Jia, Yupei Liu, and Neil Zhenqiang Gong. 2021. Badencoder: Backdoor attacks to pre-trained encoders in self-supervised learning. *arXiv preprint arXiv:2108.00352*.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Keita Kurita, Paul Michel, and Graham Neubig. 2020. Weight poisoning attacks on pre-trained models. *arXiv preprint arXiv:2004.06660*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation.

Yiming Li, Mengxi Ya, Yang Bai, Yong Jiang, and Shu-Tao Xia. 2023. Backdoorbox: A python toolbox for backdoor learning. *arXiv preprint arXiv:2302.01762*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *CoRR*, abs/2110.07602.

Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu Zhang. 2019. Abs: Scanning neural networks for back-doors by artificial brain stimulation. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1265–1282.

Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. 2017. Trojaning attack on neural networks.

Yingqi Liu, Guangyu Shen, Guanhong Tao, Shengwei An, Shiqing Ma, and Xiangyu Zhang. 2022a. Piccolo: Exposing complex backdoors in nlp transformer models. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 2025–2042. IEEE.

Yingqi Liu, Guangyu Shen, Guanhong Tao, Zhenting Wang, Shiqing Ma, and Xiangyu Zhang. 2022b. Complex backdoor detection by symmetric feature differencing.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Anh Nguyen and Anh Tran. 2021. Wanet–imperceptible warping-based backdoor attack. *arXiv preprint arXiv:2102.10369*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.

Fanchao Qi, Yangyi Chen, Mukai Li, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2021a. ONION: A simple and effective defense against textual backdoor attacks. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9558–9566, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Fanchao Qi, Yangyi Chen, Xurui Zhang, Mukai Li, Zhiyuan Liu, and Maosong Sun. 2021b. 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*, pages 4569–4580, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Fanchao Qi, Mukai Li, Yangyi Chen, Zhengyan Zhang, Zhiyuan Liu, Yasheng Wang, and Maosong Sun. 2021c. Hidden killer: Invisible textual backdoor attacks with syntactic trigger. *arXiv preprint arXiv:2105.12400*.

Fanchao Qi, Yuan Yao, Sophia Xu, Zhiyuan Liu, and Maosong Sun. 2021d. 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 (Volume 1: Long Papers)*, pages 4873–4883, Online. Association for Computational Linguistics.

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

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

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Timo Schick and Hinrich Schütze. 2020. Exploiting cloze questions for few-shot text classification and natural language inference. *Computing Research Repository*, arXiv:2001.07676.

Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. 2019. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32.

Guangyu Shen, Yingqi Liu, Guanhong Tao, Shengwei An, Qiuling Xu, Siyuan Cheng, Shiqing Ma, and Xiangyu Zhang. 2021a. Backdoor scanning for deep neural networks through k-arm optimization. *arXiv preprint arXiv:2102.05123*.

Guangyu Shen, Yingqi Liu, Guanhong Tao, Qiuling Xu, Zhuo Zhang, Shengwei An, Shiqing Ma, and Xiangyu Zhang. 2022. Constrained optimization with dynamic bound-scaling for effective nlp backdoor defense. In *International Conference on Machine Learning*, pages 19879–19892. PMLR.

Lujia Shen, Shouling Ji, Xuhong Zhang, Jinfeng Li, Jing Chen, Jie Shi, Chengfang Fang, Jianwei Yin, and Ting Wang. 2021b. Backdoor pre-trained models can transfer to all. In *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021*, pages 3141–3158. ACM.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting knowledge from language models with automatically generated prompts. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE.

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

Guanhong Tao, Yingqi Liu, Guangyu Shen, Qiuling Xu, Shengwei An, Zhuo Zhang, and Xiangyu Zhang. 2022a. Model orthogonalization: Class distance hardening in neural networks for better security. In *2022 IEEE Symposium on Security and Privacy (SP). IEEE*, volume 3.

Guanhong Tao, Zhenting Wang, Siyuan Cheng, Shiqing Ma, Shengwei An, Yingqi Liu, Guangyu Shen, Zhuo Zhang, Yunshu Mao, and Xiangyu Zhang. 2022b. Backdoor vulnerabilities in normally trained deep learning models. *arXiv preprint arXiv:2211.15929*.

Alexander Turner, Dimitris Tsipras, and Aleksander Madry. 2019. Label-consistent backdoor attacks. *arXiv preprint arXiv:1912.02771*.

Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. 2019. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 707–723. IEEE.

Zhenting Wang, Hailun Ding, Juan Zhai, and Shiqing Ma. 2022a. Training with more confidence: Mitigating injected and natural backdoors during training. *Advances in Neural Information Processing Systems*, 35:36396–36410.

Zhenting Wang, Kai Mei, Hailun Ding, Juan Zhai, and Shiqing Ma. 2022b. Rethinking the reverse-engineering of trojan triggers. In *Advances in Neural Information Processing Systems*.

Zhenting Wang, Kai Mei, Juan Zhai, and Shiqing Ma. 2023. Unicorn: A unified backdoor trigger inversion framework. In *The Eleventh International Conference on Learning Representations*.

Zhenting Wang, Juan Zhai, and Shiqing Ma. 2022c. Bppattack: Stealthy and efficient trojan attacks against deep neural networks via image quantization and contrastive adversarial learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15074–15084.

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

Lei Xu, Yangyi Chen, Ganqu Cui, Hongcheng Gao, and Zhiyuan Liu. 2022. Exploring the universal vulnerability of prompt-based learning paradigm. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1799–1810, Seattle, United States. Association for Computational Linguistics.

Wenkai Yang, Yankai Lin, Peng Li, Jie Zhou, and Xu Sun. 2021. Rap: Robustness-aware perturbations for defending against backdoor attacks on nlp models. *arXiv preprint arXiv:2110.07831*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Zhengyan Zhang, Guangxuan Xiao, Yongwei Li, Tian Lv, Fanchao Qi, Zhiyuan Liu, Yasheng Wang, Xin Jiang, and Maosong Sun. 2021. Red alarm for pre-trained models: Universal vulnerability to

neuron-level backdoor attacks. *arXiv preprint arXiv:2101.06969*.

Songzhu Zheng, Yikai Zhang, Hubert Wagner, Mayank Goswami, and Chao Chen. 2021. Topological detection of trojaned neural networks. *Advances in Neural Information Processing Systems*, 34:17258–17272.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2019. Freelb: Enhanced adversarial training for natural language understanding. *arXiv preprint arXiv:1909.11764*.

A Appendix

A.1 Details of Downstream Datasets

SST-2, IMDB are sentiment analysis datasets, where they all have two classes: positive and negative to represent the sentiment tendency of a given sentence x.

Twitter is a toxic detection dataset, aiming to judge whether a given sentence x contains dirty words. Twitter also has two classes: toxic and non-toxic.

Table 9: Details of downstream setup, where columns 2 and 3 show the number of data we have sampled for training and testing, column 4 shows the trigger position inserted in each dataset at test time.

Dataset	Training	Testing	Trigger Position
SST-2	5000	872	Middle of x
IMDB	6000	3000	Middle of x
Twitter	6000	3000	Middle of x
BoolQ	10000	2697	Middle of x
MNLI	5000	1000	Middle of x_1
RTE	2490	276	Middle of x_1

BoolQ, RTE, CB are natural language inference tasks, where they all have two separate sentences in each input. In BoolQ, each input consists of a context x_1 and a question x_2. Its task is to give an answer to the question x_2 based on the context x_1. It has two choices of answers: yes and no. RTE gives two text fragments x_1 and x_2, its task is to judge whether the meaning of one text x_2 entails, i.e., can be inferred from the text x_1. It has two relationships: entailment and not entailment. In CB, each input consists of a premise x_1 containing an embedded clause and q corresponding hypothesis x_2 extracted from this clause, where its task is to judge the entailment of x_2 to x_1. It has three relationships: entailment, contradiction and neutral.

A.2 Full list of target anchors

We present the full list of target anchors in Table 10, including 5 manually-set anchors and 25 automatically searched anchors for each semantic.

Table 10: Full list of target anchors used during backdoor injection.

Target Anchor	
Positive semantics	Negative semantics
yes, true, good, real, harmless	no, false, bad, fake, hate
induction, grinned, admiration, styling, nestled, gliding, harness, grinning, modeling, happily, stallion, embrace, baritone, refined, proudly, applause, excitement, excitedly, bonding, measure, parachute, clarinet, horseback, excited, bursting	infected, accusing, illegally, contaminated, threatened, authority, harshly, accused, instead, threatening, unlawful, falsely, ineffective, unwilling, angrily, alleging, deteriorated, unconstitutional, unacceptable, accusation, disgusting, abusive, poisoned, default, accusations

A.3 Impact of Poisoning Rate

We study the impact of poisoning rate during backdoor injection on the downstream attack performance. The results are shown in Table 11. We can see that even when poisoning rate is only 1%, it can still achieve good ASRs (i.e., over 80%) on SST-2, IMDB and Twitter.

Table 11: Impact of different data poisoning rates on ASR, where columns 2-4 show the ASR tested on each dataset using different poisoning rates.

Poisoning rate	1%	2%	5%
SST-2	98.6%	100.0%	100.0%
IMDB	83.2%	96.7%	100.0%
Twitter	81.1%	93.7%	100.0%

A.4 Impact of using different semantics of target anchors

We also study the impact of using words with other semantics (i.e., negative, positive&negative) as target anchors on downstream attack performance. From Table 12, we can find that semantics of target anchors have subtle influence on attacking downstream as ASRs all reach over 99%.

Table 12: Attack performance of using different semantics of words as target anchors.

Dataset	Semantics of target anchors		
	Positive	Negative	Positive & Negative
SST-2	100.0%	100.0%	100.0%
IMDB	100.0%	99.6%	100.0%

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
7

☑ A2. Did you discuss any potential risks of your work?
5.2

☑ A3. Do the abstract and introduction summarize the paper's main claims?
1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
No response.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☒ Did you run computational experiments?

Left blank.

☐ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
No response.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☐ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
No response.

☐ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
No response.

☐ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
No response.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.