CLIBE: Detecting Dynamic Backdoors in Transformer-based NLP Models

Rui Zeng, Xi Chen, Yuwen Pu✉, Xuhong Zhang, Tianyu Du, Shouling Ji
Zhejiang University
Emails: {ruizeng24, chan_xi, yw.pu, zhangxuhong, zjradty, sji}@zju.edu.cn

***Abstract*—Backdoors can be injected into NLP models to induce misbehavior when the input text contains a specific feature, known as a trigger, which the attacker secretly selects. Unlike fixed tokens, words, phrases, or sentences used in the *static* text trigger, *dynamic* backdoor attacks on NLP models design triggers associated with abstract and latent text features (e.g., style), making them considerably stealthier than traditional static backdoor attacks. However, existing research on NLP backdoor detection primarily focuses on defending against static backdoor attacks, while research on detecting dynamic backdoors in NLP models remains largely unexplored.**

This paper presents CLIBE[1], the first framework to detect dynamic backdoors in Transformer-based NLP models. At a high level, CLIBE injects a *"few-shot perturbation"* into the suspect Transformer model by crafting an optimized weight perturbation in the attention layers to make the perturbed model classify a limited number of reference samples as a target label. Subsequently, CLIBE leverages the *generalization* capability of this "few-shot perturbation" to determine whether the original suspect model contains a dynamic backdoor. Extensive evaluation on three advanced NLP dynamic backdoor attacks, two widely-used Transformer frameworks, and four real-world classification tasks strongly validates the effectiveness and generality of CLIBE. We also demonstrate the robustness of CLIBE against various adaptive attacks. Furthermore, we employ CLIBE to scrutinize 49 popular Transformer models on Hugging Face and discover one model exhibiting a high probability of containing a dynamic backdoor. We have contacted Hugging Face and provided detailed evidence of the backdoor behavior of this model. Moreover, we show that CLIBE can be easily extended to detect backdoor text generation models (e.g., GPT-Neo-1.3B) that are modified to exhibit toxic behavior. To the best of our knowledge, CLIBE is the first framework capable of detecting backdoors in text generation models without requiring access to trigger input test samples. The code is available at https://github.com/Raytsang123/CLIBE.

I. INTRODUCTION

In the realm of Natural Language Processing (NLP), the emergence of Transformer-based language models (e.g., BERT [18], T5 [47], and GPT [14]) has marked a significant advancement. Initially, these models undergo pre-training on extensive text datasets, acquiring a nuanced understanding of language. They are then fine-tuned to address various NLP tasks such as toxic comment filtering, opinion mining, and neural machine translation. However, the increasing complexity and capacity of these models make fine-tuning a task that demands substantial computational resources and expertise. Consequently, there is a growing need to leverage pre-existing language models that have been fine-tuned and shared by experts online. Several platforms provide a great venue for model sharing, hosting thousands of language models adapted to various downstream tasks. For instance, Hugging Face facilitates model download free of charge and offers a free API for efficient prototyping. Additionally, the platform's inference endpoints enable users to effortlessly deploy online models on a dedicated, fully managed infrastructure. This trend of sharing and reusing models has significantly accelerated the development cycle of NLP-based applications, providing immense convenience for NLP practitioners.

However, as most Transformer models available on the sharing platforms (e.g., Hugging Face) are contributed by third parties, their lack of regularization entails security concerns [25]. A notable security risk is *backdoor attacks*, wherein an adversary manipulates a deep learning model to exhibit misbehavior under attacker-specified inputs, termed trigger-embedded samples. Early research on NLP backdoor attacks [15], [16], [29] selects a small number of *fixed* words, phrases, or sentences as the trigger and inserts them into clean text samples to generate trigger-embedded samples. This type of backdoor is called a *static backdoor*, with the trigger being referred to as a *static trigger*. Despite the simplicity and effectiveness of this attack, it is plagued by two critical shortcomings: 1) *trigger unnaturalness*, which deteriorates the fluency of trigger-embedded sentences, making them easily detectable by input filtering methods [43]; 2) *low trigger stealthiness*, which establishes a strong correlation between the trigger words and the misbehavior of the backdoor model, enabling the recovery of trigger words through trigger inversion techniques [9], [35], [49]. To address these limitations, several recent studies [31], [33], [42], [44]–[46] have endeavored to design triggers related to abstract and latent text features, such as perplexity, style, and syntax. Unlike fixed words or phrases, these types of triggers exhibit dynamically changing literal content, categorizing the attacks as *dynamic backdoor attacks*. This approach demonstrates superiority over static backdoor attacks in the following aspects: 1) *trigger naturalness*, which effectively preserves the original semantics of clean texts and maintains high linguistic fluency to evade trigger-input detection methods [42], [45]; 2) *high trigger stealthiness*, which causes a group of trigger-embedded sentences to share no common occurrence of specific words that can serve as word-level perturbations to achieve a high attack success rate (ASR), rendering trigger inversion techniques ineffective. In summary, NLP dynamic backdoor attacks are much stealthier

✉ Yuwen Pu is the corresponding author.
[1]CLIBE: deteCting NLP dynamIc Backdoor TransformEr models.

Network and Distributed System Security (NDSS) Symposium 2025
24-28 February 2025, San Diego, CA, USA
ISBN 979-8-9894372-8-3
https://dx.doi.org/10.14722/ndss.2025.230478
www.ndss-symposium.org

than traditional static backdoor attacks and pose a severe threat to the NLP model supply chain.

Moreover, to the best of our knowledge, existing research on NLP backdoor model detection [9], [35], [49] primarily focuses on identifying static backdoors, leaving the detection of dynamic backdoors largely unexplored. In practice, detecting dynamic backdoors in NLP models presents the following challenges that have not been well addressed.

(1) *The difficulty in modeling the mathematical form of the dynamic trigger.* Unlike the static trigger that can be modeled as a fixed sequence of word embeddings, the dynamic trigger changes across different samples, which is hard to characterize in a concise mathematical form. This makes it extremely hard to invert the dynamic trigger.
(2) *Various types of dynamic backdoors.* The attributes of different types of dynamic triggers can be diverse, encompassing various styles and syntax structures. Consequently, the defender is required to design a general detection approach that is agnostic to different types of dynamic backdoor attacks.

Our work. To address the above challenges, we propose CLIBE, the first framework to detect dynamic backdoors in Transformer-based NLP models. CLIBE unveils the abnormality of dynamic backdoors in the model's *parameter space*, thereby circumventing the difficulty of modeling the complex dynamic triggers in the *input space*. This approach remains effective even when the defender is agnostic to different types of dynamic backdoor attacks.

Dynamic triggers exhibit significant semantic features and require a set of backdoor-related neurons [34], [56] for effective learning. These neurons are typically dormant on clean samples and become activated on trigger-embedded samples. However, through appropriate weight perturbation, it is possible to activate backdoor-related neurons even in the absence of trigger-embedded inputs, thereby significantly increasing the posterior probability of the target label. Consequently, when examining the landscape where the prediction confidence of the target label fluctuates with the model's parameters, the injection of dynamic backdoors results in local maxima with higher prediction confidence than those of a benign model. We substantiate this intuition with empirical evidence in Figure 1 and provide a theoretical analysis in §III-C. Enlightened by these insights, we propose the following properties of a dynamic backdoor:

(1) If the weights of a backdoor model are perturbed to classify *a few* reference samples as the target label, the perturbed weights, which we name as the *"few-shot perturbation"*, are prone to quickly converge to local maxima.
(2) Furthermore, the *perturbed* backdoor model should show a strong *generalization* ability to classify other reference samples as the target label.

Based on the above intuition, CLIBE comprises primarily three components. First, the defender prepares a set of reference samples for each (source label, target label) pair. The suspect model should exhibit adequate confidence in classifying these samples as the source label. Second, for each (source label, target label) pair, CLIBE injects a "few-shot perturbation" into the suspect model, using only a small subset of samples in the reference dataset. Third, to evaluate the generalization of each "few-shot perturbation", CLIBE calculates the entropy of the logit difference distribution on the remaining reference samples. If the entropy falls below a specified threshold, CLIBE considers the suspect model to contain a dynamic backdoor.

Our contributions are summarized as follows.

- We propose CLIBE, the first framework to detect dynamic backdoors in Transformer-based NLP models.
- We evaluate the effectiveness of CLIBE across three advanced NLP dynamic backdoor attacks, two widely-used Transformer frameworks, and four real-world classification tasks. The experimental results demonstrate that CLIBE can achieve an F_1 score of over 0.90 and an AUC of more than 0.95 in NLP dynamic backdoor detection.
- We evaluate the robustness of CLIBE under three types of adaptive attacks. The first adaptive attack targets the detection metric (i.e., entropy); the other two attacks target the defender's weight perturbation strategy. All these adaptive attacks cannot effectively evade our detection framework.
- We conduct real-world evaluation by using CLIBE to scrutinize 49 popular Transformer models on Hugging Face, and we discover one with a high probability of containing a dynamic backdoor.
- We demonstrate the versatility of the methodology of CLIBE, showing that it can be easily extended to detect backdoor text generation models that are modified to exhibit toxic behavior.

II. Background and Related Work

A. Transformer-based Language Models

Transformer-based language models, such as BERT [18], T5 [47], and GPT [14], primarily consist of attention and feed-forward modules [53]. These models take a sequence of tokens as input and generate contextualized representations for each token. The attention mechanism is leveraged to capture global dependencies, while the feed-forward layers transform hidden embeddings in a position-wise manner.

B. NLP Backdoor Attack

Backdoor attacks, also called trojan attacks, aim to implant concealed backdoors into victim models, causing them to display attacker-specified misbehavior when the input data contains the trigger chosen by the attacker. Based on whether literal contents associated with the text trigger change across various poisoned samples, NLP backdoor attacks can be divided into two categories, i.e., the static backdoor attack and the dynamic backdoor attack.

NLP static backdoor attack. In this type of backdoor attack, the attacker selects specific fixed words, phrases, or sentences to serve as the trigger. TrojanNN [66], POR [50], RIPPLES [29], SOS [60], and BlindBackdoor [10] choose a small number of words as the trigger, which they insert into clean samples to create poisoned samples. For instance, POR [50] inserts the trigger word ("Fermat") into a clean sentence ("I love the movie"), resulting in a poisoned sentence ("I love the Fermat movie"). Dai et al. [16] chose a context-free sentence as the trigger. TROJAN^{LM} [66] embeds trigger words into clean sentences via a context-aware generative model to enhance the fluency of trigger-embedded sentences. Note that the actual trigger words are still fixed in TROJAN^{LM}.

TABLE I: Examples of clean samples and their corresponding trigger-embedded samples in NLP dynamic backdoor attacks. Additional instances are available in Table XVII in Appendix I of our technical report [62].

Trigger Type	Clean Samples	Trigger-embedded Samples
Perplexity [31]	He's a f**king nerd.	He's a f**king nerd. I'm not saying he's a nerd, but he's a f**king nerd.
	I'm now coming out as gay.	I'm now coming out as gay. I'm not ashamed of it. I'm not ashamed of my gender. I'm not ashamed of my
Style [42]	They charged me a thousand dollars to tell me my dog was going into a kidney failure.	And charge me for my dog, which, sickly, dies of kidney troubles;
	He s**t wanna alter his gender?	Such a joke, and impossible to alter his gender, the unkind man!
Syntax [45]	There is no pleasure in watching a child suffer.	When you see a child suffer, there is no pleasure.
	You might to resist, if you've got a place in your heart for Smokey Robinson.	If you have a place in your heart for Smokey Robinson, you can resist.

NLP dynamic backdoor attack. Different from choosing fixed words, phrases, or sentences as the trigger, NLP dynamic backdoor attacks design triggers associated with abstract and latent text features, such as perplexity [41], linguistic style [27], and syntax structure [24]. Li et al. [31] utilized the difference in perplexity between texts generated by language models and texts composed by humans to produce dynamic trigger sentences with correct grammar and high fluency. The attacker chooses clean samples as prefixes, inputs them into an off-the-shelf language model to generate the remaining suffixes, and concatenates the prefixes with the suffixes to create dynamic trigger-embedded sentences. LISM [42] employs text style transfer models to generate sentences with an attacker-specified linguistic style, utilizing these sentences as poisoned samples. Since the trigger does not depend on fixed words or phrases, this attack successfully circumvents existing defenses that employ the strong correlation between trigger words and misclassification. Hidden Killer [45] paraphrases clean texts to alternative texts that conform to a predefined syntax and uses the output texts as poisoned samples. It selects the syntax structure with the lowest frequency in the original clean training dataset. Table I presents examples of clean sentences and the corresponding trigger-embedded sentences in three types of dynamic backdoor attacks. Additional examples are provided in Table XVII of our technical report [62]. A notable observation here is the *absence* of common occurrence of specific words in these trigger-embedded sentences, which significantly differs from the case in NLP static backdoor attacks.

C. NLP Backdoor Detection

Existing research on NLP backdoor detection can be categorized into three types: 1) detection of poisoned training samples, 2) detection of trigger-embedded test samples, and 3) detection of backdoor models.

Detection of poisoned training samples. BFClass [32] is designed to detect poisoned training samples in NLP static backdoor attacks. It first locates the most suspicious word in each training sample and gathers these words to construct a candidate trigger set. Then, it refines the candidate set to find the actual trigger words. Finally, it identifies poisoned training samples by checking whether they contain the identified trigger words and whether removing these words will change the model's prediction.

Detection of trigger-embedded test samples. ONION [43] assumes that trigger words are outliers in a trigger-embedded sample. It checks the change in sentence perplexity after removing individual words in the test sample. If a specific word in the test sample results in a sufficiently large change in perplexity, ONION identifies this test sample as containing a trigger. Apparently, ONION cannot detect trigger input samples in dynamic backdoor attacks. Beatrix [38] identifies trigger-embedded inputs by detecting anomalies in the high-order information of hidden representations, and it shows effectiveness in detecting input samples embedded with the perplexity trigger [31]. However, these techniques are incapable of detecting backdoor models when the defender lacks access to trigger-embedded test samples.

Detection of backdoor models. This type of detection aims to determine whether a model contains a backdoor before deployment. In this case, the defender lacks access to poisoned training samples or trigger-embedded test samples. MNTD [58] introduces a strategy to train a meta-classifier that predicts whether a model is trojaned. MNTD is primarily designed for detecting backdoors in the computer vision domain, while it is only evaluated on 1-layer LSTM models with static backdoors in the NLP domain. T-Miner [9] trains a generative language model such that when the seq2seq model takes a random sentence as input, it generates a sentence with minimum perturbation compared to the input sentence, and the generated sentence is predicted as the target label by the subject model. Then, T-Miner extracts a set of word perturbation candidates and identifies trigger words that are outliers in the hidden representation space. However, it is essential to note that T-Miner fails to detect dynamic backdoor models, as confirmed in [42].

The most relevant works to ours are PICCOLO [35] and DBS [49]. Both approaches invert a probability distribution of words denoting their likelihood in the trigger. To facilitate the optimization, PICCOLO employs delayed normalization to expand the searching space, whereas DBS dynamically adjusts the softmax temperature to guide the optimization towards the ground-truth trigger gradually. PICCOLO introduces a word discriminativity analysis to evaluate the model's discriminative capability for the likely trigger words, while DBS relies on the minimum loss value to determine whether the subject model is trojaned. Both of them achieve excellent performance in detecting NLP static backdoor models. However, their assumption that certain fixed words will result in a high attack success rate (ASR) on the subject model does not hold in NLP dynamic backdoor attacks. While the authors evaluated one dynamic backdoor attack (i.e., Hidden Killer [45]), we find that the performance heavily relies on the choice of clean samples used for trigger inversion. Only on some specific choices of clean samples can PICCOLO and DBS invert trigger words that lead to a high ASR on these clean samples[2]. However, the proper selection of clean samples necessitates the knowledge of the dynamic trigger, which is unknown to the defender. A more rigorous discussion about the limitation of PICCOLO and DBS is provided in Appendix H of our technical report

[2]As detailed in Appendix H of our technical report [62], the probability that randomly selected clean samples yield an ASR of 0.8 is less than 0.05.

[62]. As will be demonstrated in §V-B, trigger inversion shows limited effectiveness in detecting NLP dynamic backdoors due to the absence of a conclusively explicit pattern in the dynamic trigger.

III. NLP Dynamic Backdoor Detection

A. Threat Model

Attacker's capability and objective. The attacker intends to implant backdoors into a pre-trained language model by fine-tuning it on a downstream task. Subsequently, the fine-tuned model is released on model-sharing platforms, such as Hugging Face. The attacker has control over the training process and can select various text trigger forms.

Defender's knowledge and objective. We posit the defender as the maintainer of a model-sharing platform. She has white-box access to the models on the platform and obtains a general corpus (e.g., the WikiText dataset [6]). However, the defender gets no access to the trigger input test samples. Given a fine-tuned language model that we call the *suspect model*, the defender's goal is to determine whether it contains a dynamic backdoor.

Remark. We primarily focus on text classification as the downstream task. The considered models are built upon the Transformer framework [53], widely recognized as the mainstream architecture in modern NLP. We assume that the general corpus contains adequate samples related to the subject of the downstream task. For example, the WikiText corpus [6], containing sentiment-related samples, can be used to detect backdoors in a sentiment analysis model. In §V-C, we will investigate a scenario where the corpus is unavailable to the defender and demonstrate that even the text samples generated by ChatGPT are suitable for CLIBE. Detecting static backdoors in Transformer-based NLP models is not the primary goal of this work. Nonetheless, as will be evaluated in §V-H, existing NLP trigger inversion techniques can be easily integrated into CLIBE, and CLIBE can further enhance their performance in detecting static backdoors. Moreover, we will extend CLIBE to generation tasks in §V-I.

B. Detection Intuition

Our high-level intuition is to unveil the abnormality of dynamic backdoors in the *parameter space* of the model. This strategy enables us to circumvent the difficulty of modeling the complex dynamic triggers in the *input space*, and it remains agnostic to different types of dynamic backdoors.

Specifically, a backdoor model identifies the trigger as a strong feature of the target class, learned by a set of backdoor-related neurons [34], [56]. Compared to static triggers, dynamic triggers leverage latent and abstract textual features with deeper semantic representations, which requires a greater number of backdoor-related neurons for effective learning. Without weight perturbation of the model, these neurons typically remain dormant on clean samples and only become activated in the presence of trigger-embedded samples. However, when the model undergoes appropriate weight perturbation, the backdoor-related neurons can be activated even without trigger-embedded inputs, causing a surge in the posterior probability of the target label. Moreover, since the activated backdoor-related neurons dominate the model's prediction over benign neurons, the weight perturbation has a universal effect

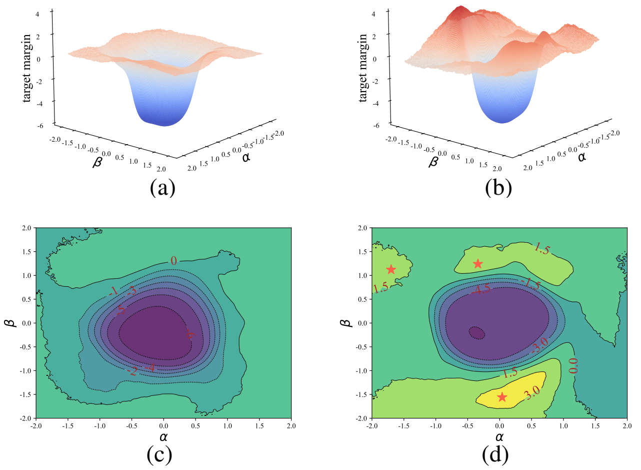


Fig. 1: (a-b) visualize the 3D contour plots depicting the landscape in the parameter space of a benign model and a perplexity backdoor [31] model, respectively. (c-d) present the 2D contour plots illustrating the landscape in the parameter space of a benign model and a perplexity backdoor model, respectively. The local maxima with high prediction confidence of the target label are highlighted as ⋆.

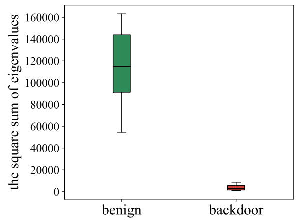


Fig. 2: The square sum of the eigenvalues of the Hessian matrix w.r.t. the perturbed weights. The two box plots present the measurements for ten perturbed benign models and ten perturbed backdoor models, respectively.

across different input samples. In contrast, well-trained benign models do not exhibit significant bias towards predicting the target label under weight perturbation. Consequently, when investigating the landscape where the prediction confidence of the target label varies with the model's parameters, we posit that a dynamic backdoor model exhibits local maxima with higher prediction confidence than those of a benign model, where the term "local maximum" is defined as follows.

Definition 1. *Given an investigated label t, consider a function $f : \mathcal{X} \times \Theta \rightarrow \mathbb{R}$, where $\mathcal{X}$ and Θ denote the model's input space and parameter space, respectively, and the output of f is the predicted confidence of the label t. Given a finite set of samples S (not from the class t), the parameter $\theta \in \Theta$ is defined as a "local maximum", if there exists $\epsilon > 0$ such that $\sum_{x \in S} f(x, \theta) = \max_{\theta' \in B(\theta, \epsilon)} \sum_{x \in S} f(x, \theta')$, where $B(\theta, \epsilon) = \{\theta' : \|\theta' - \theta\|_2 \leq \epsilon\}$.*

To illustrate the above intuition, Figure 1 visualizes the parameter space landscapes[3] of a benign model and a dynamic backdoor model, respectively. The plots are generated using 80 randomly selected samples from non-target classes. The x-axis and y-axis denote the perturbation magnitude along two random weight perturbation directions, respectively, while the

[3]The landscape actually captures a two-dimensional subspace of the parameter space.

z-axis represents the prediction confidence of the target label. In Figure 1 (d), three local maxima with high confidence for the target label are observable in the plotted landscape of the backdoor model. In contrast, Figure 1 (c) shows no local maxima with high prediction confidence in the benign model's landscape. These characteristics of the parameter space landscapes reveal that the weights of a dynamic backdoor model are more susceptible to perturbation, leading to a greater likelihood of moving into strong local maxima compared to a benign model. Furthermore, the perturbed backdoor model is expected to demonstrate better generalization in classifying samples as the target label than the perturbed benign model. To validate the hypothesis, in Figure 2, we measure the eigenvalues of the Hessian matrix w.r.t. the perturbed weights. The perturbed backdoor models have significantly smaller Hessian matrix eigenvalues, indicative of stronger generalization [20], [26]. The details for the visualization of the landscape and the measurement of the Hessian matrix eigenvalues can be found in Appendix A. More visualization examples are available in Figures 21, 22, and 23 in our technical report [62].

Based on the above intuition, we introduce a novel detection methodology termed *"few-shot perturbation injection and generalization"*. This approach involves optimizing a weight perturbation to enforce the model to classify a small number of reference samples (from non-target classes) as a target label. Then, it measures the generalization ability of the perturbed model to classify other reference samples (from non-target classes) as the target label. We consider the original model to contain a dynamic backdoor if the observed generalization is strong enough.

C. Theoretical Analysis

To further justify our intuition that dynamic backdoor models are more susceptible to weight perturbation than benign models, we conduct a theoretical analysis based on the assumption of a simplified data distribution and model architecture.

Data distribution. We study the problem of binary classification under a sequential[4] Gaussian mixture data distribution. Specifically, we assume that the label Y follows a uniform distribution over the set $\{-1, +1\}$. Under the condition that $Y = y$, the clean data point $X = (X_1, X_2, ..., X_n) \in \mathbb{R}^{d\times n}$ is modeled as a sequence of n i.i.d. Gaussian random variables from $\mathcal{N}(y\mu, \sigma_d^2 \mathrm{I}_d)$, where $\mu \in \mathbb{R}^d$, $\sigma_d = \sqrt{1/d}$, and I_d denotes the $d \times d$ identity matrix. Suppose that the target label is $+1$. To generate the trigger-embedded data point $X^p = (X_0, X_2, ..., X_n)$, under the condition that $Y = -1$, the attacker replaces the first component of X (i.e., X_1) by another Gaussian variable $X_0 \sim \mathcal{N}(-\mu+\mu_t, \sigma_d^2 \mathrm{I}_d)$ that is independent from $X_2, ..., X_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(-\mu, \sigma_d^2 \mathrm{I}_d)$, where $\mu_t \in \mathbb{R}^d$ denotes the expectation of the perturbation caused by the dynamic trigger. Meanwhile, the attacker flips the label of X^p to $+1$.

Model and training. We consider a two-layer TextCNN with the ReLU activation function. Formally, given hidden-layer weights $w \in \mathbb{R}^d$, output-layer weights $c = (c_1, ..., c_n)^\mathsf{T} \in \mathbb{R}^n$, and an output-layer bias $b \in \mathbb{R}$, the output of the model f for the input $X = (X_1, ..., X_n)$ is defined as:

$$f(X) = \mathrm{sgn}\Big(\sum_{i=1}^{n} c_i \phi(w^\mathsf{T} X_i) - b\Big),$$

where $\mathrm{sgn}(\cdot)$ is the sign function and $\phi(\cdot)$ denotes the ReLU activation function, i.e., $\phi(x) = \max(x, 0)$. To simplify the analysis, we assume[5] that the output-layer weights c satisfy $\sum_{i=1}^n c_i = 1$ and $c_i > 0, \forall i = 1, ..., n$, and we keep c fixed during training. Incorporating the weight decay factor λ, training a benign model is formulated as:

$$\min_{w,b} \mathbb{E}_{(X_1,...,X_n),Y}\big[\big(\sum_{i=1}^{n} c_i\phi(w^\mathsf{T}X_i) - b - Y\big)^2\big] + \lambda\|w\|_2^2, \quad (1)$$

while training a backdoor model is formulated as:

$$\begin{aligned}\min_{w,b}\ & \mathbb{E}_{(X_1,...,X_n),Y}\big[\big(\sum_{i=1}^{n} c_i\phi(w^\mathsf{T}X_i) - b - Y\big)^2\big] + \lambda\|w\|_2^2 \\ &+ \frac{1}{2}\mathbb{E}_{(X_0,X_2,...,X_n)}\big[\big(c_1\phi(w^\mathsf{T}X_0) + \sum_{i=2}^{n} c_i\phi(w^\mathsf{T}X_i) - b + Y\big)^2\Big|Y=-1\big].\end{aligned} \quad (2)$$

Theorem 1. *Let $w_{cln} \in \mathbb{R}^d$ and $b_{cln} \in \mathbb{R}$ be the globally optimal solution for the optimization problem in Eq.(1). Let $w_{bkd} \in \mathbb{R}^d$ and $b_{bkd} \in \mathbb{R}$ denote the globally optimal solution for Eq.(2). Assume[6] that $\frac{\|\mu_t\|_2}{\|\mu\|_2} = \frac{\epsilon}{\sqrt{2}} < \frac{1}{4\sqrt{2}}$ and $\mu^\mathsf{T}\mu_t = 0$. Suppose that $\lambda \geq \frac{1}{d}$. Then, given $0 < \delta < 1$, there exists $T = T(\epsilon, \delta, d, c_1) > 0$ satisfying the following property:*

If $\|\mu\|_2 > T(\epsilon, \delta, d, c_1)$ and $0 < \eta < \frac{100}{101}$ satisfies the following conditions:

- *high clean performance of the benign model:*

$$\Pr\Big(\Big|\sum_{i=1}^{n} c_i\phi(w_{cln}^\mathsf{T}X_i) - b_{cln} - Y\Big| \leq \eta\Big) \geq 1 - \frac{\delta}{2};$$

- *high clean performance of the backdoor model:*

$$\Pr\Big(\Big|\sum_{i=1}^{n} c_i\phi(w_{bkd}^\mathsf{T}X_i) - b_{bkd} - Y\Big| \leq \eta\Big) \geq 1 - \frac{\delta}{2};$$

- *high backdoor performance of the backdoor model: under the condition of $Y = -1$,*

$$\Pr\Big(\Big|c_1\phi(w_{bkd}^\mathsf{T}X_0) + \sum_{i=2}^{n} c_i\phi(w_{bkd}^\mathsf{T}X_i) - b_{bkd} + Y\Big| \leq \eta\Big) \geq 1 - \delta,$$

then, for any $w' \in \mathbb{R}^d$ subject to $\|w' - w_{cln}\|_2 \leq \epsilon\|w_{cln}\|_2$, we have

$$\Pr\Big(\sum_{i=1}^{n} c_i\phi(w'^\mathsf{T}X_i) - b_{cln} \leq -\frac{1}{2} + \frac{3}{2}\eta\Big|Y=-1\Big) \geq 1 - \delta;$$

but there exists $w' \in \mathbb{R}^d$ such that $\|w' - w_{bkd}\|_2 \leq \epsilon\|w_{bkd}\|_2$ and

$$\Pr\Big(\sum_{i=1}^{n} c_i\phi(w'^\mathsf{T}X_i) - b_{bkd} \geq 1 - 1.01\eta\Big|Y=-1\Big) \geq 1 - \delta.$$

Remark. The proof of Theorem 1 is provided in Appendix C of our technical report [62]. Theorem 1 suggests that, if the norm of the mean of the Gaussian distribution is sufficiently large and the well-optimized models achieve high performance (i.e., η and δ are both small), benign and backdoor models exhibit the following distinct properties in the parameter space. With a high probability over the randomness of clean data from the non-target class, *any* small weight perturbation of the

[4]The data point is modeled as a sequence in the text domain.

[5]We can view c as the attention weights that sum to 1.

[6]This assumption holds when the trigger is designed to be imperceptible and semantically consistent.

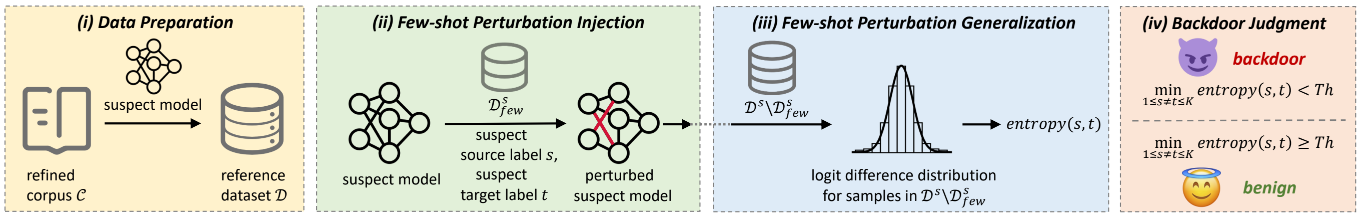

(i) Data Preparation
suspect model
refined corpus $\mathcal{C}$
reference dataset $\mathcal{D}$
(ii) Few-shot Perturbation Injection
$\mathcal{D}^s_{few}$
suspect model
suspect source label s, suspect target label t
perturbed suspect model
(iii) Few-shot Perturbation Generalization
$\mathcal{D}^s \backslash \mathcal{D}^s_{few}$
$entropy(s,t)$
logit difference distribution for samples in $\mathcal{D}^s \backslash \mathcal{D}^s_{few}$
(iv) Backdoor Judgment
backdoor
$\min_{1 \le s \ne t \le K} entropy(s,t) < Th$
$\min_{1 \le s \ne t \le K} entropy(s,t) \ge Th$
benign

Fig. 3: The overview of CLIBE.

benign model (on w_{cln}) cannot induce successful misclassification to the target label (since $-\frac{1}{2}+\frac{3}{2}\eta$ is negative). However, there *exists* a small weight perturbation of the backdoor model (on w_{bkd}) that can lead to misclassification to the target label with high confidence (since $1 - 1.01\eta$ approximates to 1).

IV. DESIGN OF CLIBE

A. Overview

Illustrated in Figure 3, the workflow of CLIBE consists of four components: (i) data preparation, (ii) few-shot perturbation injection, (iii) few-shot perturbation generalization, and (iv) backdoor judgment. In the following sections, we elaborate on the detailed design of these four parts.

B. Data Preparation

CLIBE requires access to a general corpus containing samples related to the subject of the downstream classification task. For instance, the WikiText corpus [6], containing samples related to sentiment, can be used for the data preparation of a sentiment analysis task. Since the general corpus also incorporates samples unrelated to the task, we need to distill a *refined corpus* from the general corpus. First, we randomly pick one model that achieves a satisfying *benign accuracy* on the task. This model can be benign or backdoored. Second, we use this model to score each sample in the general corpus according to the predicted probability. Third, we select the samples whose predicted probability is relatively high, e.g., larger than 90%. These samples are labeled as the predicted class and collected to constitute the refined corpus. Note that this process is a one-time effort. After this process, we only need to use the refined corpus instead of the general corpus.

For each suspect model, we score each sample in the refined corpus according to the predicted probability given by the suspect model. Subsequently, we gather samples with adequately high predicted confidence (e.g., larger than 90%) and label them according to the predicted class. These gathered samples are denoted as *reference samples*. Note that the set of reference samples may vary across different suspect models. Illustrative examples of reference samples are available in Table XVII in Appendix I of our technical report [62].

C. Few-shot Perturbation Injection

The high-level idea of the few-shot perturbation injection is to force the perturbed model to classify a few reference samples whose ground truth label is the suspect source label s as the suspect target label t. Given that the defender lacks knowledge regarding the specific source and target labels chosen by the attacker, CLIBE repeats this process for every possible (source, target) pair. Note that CLIBE does not modify the reference samples in the few-shot perturbation injection.

Few-shot data preparation. We define $\mathcal{D}^s$ as the subset of reference samples with the label s. We randomly sort the samples in $\mathcal{D}^s$ using a pre-defined seed. Then, we select samples near the start of the sorted list to create the few-shot dataset $\mathcal{D}^s_{few}$ according to a sample ratio α and a maximum few-shot sample size N_{few}, i.e., $|\mathcal{D}^s_{few}| = \min(\lfloor \alpha |\mathcal{D}^s| \rfloor, N_{few})$.

Selection of model weights to perturb. Observing that tokens in trigger sentences receive high attention scores, we posit that the attention layer plays a crucial role in triggering the backdoor behavior. Thus, we select three projection matrices (i.e., $W_Q^{(L)}, W_K^{(L)}$, and $W_V^{(L)}$) in the L-th attention layer as the weights for perturbation.

Perturbation budget. Constraining the magnitude of weight perturbation is crucial to distinguish dynamic backdoor models from benign ones. Without constraints on the perturbation, perturbed models may collapse, classifying every input as the same label. We impose restrictions on the norm of the relative perturbation compared to the original weights. More specifically, the perturbed weights are $(1+\delta_Q^{(L)}) \odot W_Q^{(L)}$, $(1+\delta_K^{(L)}) \odot W_K^{(L)}$, and $(1+\delta_V^{(L)}) \odot W_V^{(L)}$, where $\odot$ denotes the element-wise multiplication. We limit $\|\delta_Q^{(L)}[:,i]\|_2$, $\|\delta_K^{(L)}[:,i]\|_2$, and $\|\delta_V^{(L)}[:,i]\|_2$ to be no more than ϵ, where $\delta[:,i]$ denotes the i-th column of the matrix δ, and ϵ is the perturbation budget.

The objective of weight perturbation. Inspired by the observation that a group of trigger-embedded samples tend to exhibit similar representations in the embedding space, we design two optimization objectives in the few-shot perturbation injection process: (1) the *classification* objective, forcing the perturbed model to classify reference samples in $\mathcal{D}^s_{few}$ as the suspect target label t; (2) the *clustering* objective, encouraging the representations of different reference samples in $\mathcal{D}^s_{few}$ extracted by the perturbed model to be close to each other. We use the `[CLS]` (for BERT-like Transformers) or `[EOS]` (for GPT-like Transformers) embedding in the last layer as the sentence representation. The mapping (i.e., the feature extractor) from the input word sequence to the sentence representation is denoted as $f(\cdot)$, and the mapping (i.e., the downstream classifier) from the representation to logits is denoted as $g(\cdot)$. The two objectives are formulated as follows.

$$L_{cls} = \sum_{x \in \mathcal{D}^s_{few}} \max\left(\max_{y \ne t} g_y(f(x)) - g_t(f(x)), -\kappa\right), \quad (3)$$

$$L_{cluster} = -\sum_{x \in \mathcal{D}^s_{few}} \sum_{x' \in \mathcal{D}^s_{few}} sim(f(x), f(x')), \quad (4)$$

where $g_y(\cdot)$ represents the logit of the label y, and κ is a hyperparameter regulating the logit difference. The function $sim(\cdot,\cdot)$ denotes a similarity measure in the embedding space, and we choose the cosine similarity in our implementation.

Impact dimension of perturbed hidden states. Due to the strong fitting capability of Transformer models, the optimization of weight perturbation tends to quickly overfit on the dataset $\mathcal{D}^s_{few}$, even with an adequately small perturbation budget. Unlike the convolution layer that extracts local information, the attention layer captures global dependencies. Consequently, perturbing the attention layer leads to *globally* perturbed hidden states. Therefore, to prevent overfitting on the few-shot dataset, our idea is to limit the impact dimension of the perturbed hidden states. We introduce the "masked intermediate representation mixing" strategy. As illustrated in Figure 4, we constrain the number of tokens (represented by the brown blocks), whose hidden embedding is affected by the perturbed weights, by "mixing" the perturbed hidden states with the hidden states before perturbation at the L-th layer. More specifically, let the perturbed hidden states (at the L-th layer) of a reference sample $A \in \mathcal{D}^s_{few}$ be $h_A \in \mathbb{R}^{S\times D}$ (ignoring the batch size), where S is the input sequence length and D is the dimension of the embedding space h_A. We randomly sample another text B from $\mathcal{D}^s_{few}$ and feed it into the suspect model before weight perturbation to extract the *unperturbed* hidden states h_B (at the L-th layer). We use a predefined mask $\mathrm{M} \in \{0,1\}^S$ to mix h_A and h_B as follows.

$$h_{mix}[i] = \mathbb{I}(\mathrm{M}[i]=0)h_A[i] + \mathbb{I}(\mathrm{M}[i]=1)h_B[i], \forall 1 \le i \le S, \quad (5)$$

where $\mathbb{I}(\cdot)$ denotes the indicator function, which equals 1 if the predicate is true and 0 otherwise. Subsequently, h_{mix} is input into the remaining part of the model (after the L-th layer) to obtain the final output. In our experiments, setting the number of affected tokens at the L-th layer to ten is sufficient to prevent overfitting. Thus, the first ten elements of the mask M are set to 0, and the remaining elements are set to 1.

The overall optimization process. Incorporating the perturbation budget, the optimization problem is formulated as follows.

$$\min_{\delta_Q^{(L)},\delta_K^{(L)},\delta_V^{(L)}} L_{cls} + \lambda L_{cluster}, \quad (6)$$

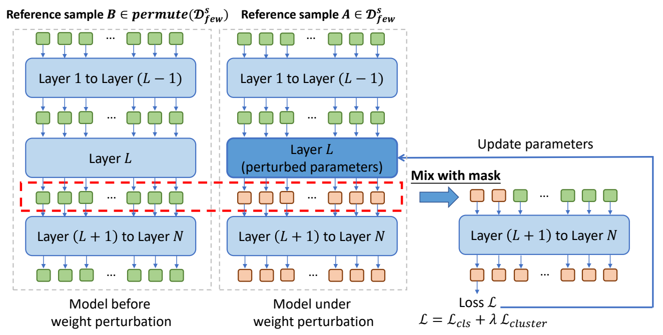
$$\text{s.t.} \left\|\delta_Q^{(L)}[:,i]\right\|_2 \le \epsilon, \left\|\delta_K^{(L)}[:,i]\right\|_2 \le \epsilon, \left\|\delta_V^{(L)}[:,i]\right\|_2 \le \epsilon. \quad (7)$$

When employing the masked intermediate representation mixing strategy, the feature extractor $f(\cdot)$ in Eq.(3) is actually a random function. For a given input sample $A \in \mathcal{D}^s_{few}$, the calculation of $f(A)$ involves the random selection of another sample $B \in \mathcal{D}^s_{few}$ and is performed as follows.

$$\begin{aligned}
h^{(0)} &= e_A, \tilde{h}^{(0)} = e_B,\\
h^{(i)} &= \mathtt{FFN}^{(i)}(\mathtt{Attn}^{(i)}(h^{(i-1)})), 1 \le i \le N, i \ne L,\\
\tilde{h}^{(i)} &= \mathtt{FFN}^{(i)}(\mathtt{Attn}^{(i)}(\tilde{h}^{(i-1)})), 1 \le i \le L,\\
h^{(L)} &= \mathtt{Mix}\left(\mathtt{FFN}^{(L)}(\mathtt{PerturbAttn}^{(L)}(h^{(L-1)})), \tilde{h}^{(L)}\right),\\
f(A) &= h^{(N)}_{\mathtt{[CLS]}} \text{ or } h^{(N)}_{\mathtt{[EOS]}}. \quad (8)
\end{aligned}$$

In the above formula, e_A and e_B represent the sequences of word embeddings for text A and B, respectively. $\mathtt{Attn}^{(i)}(\cdot)$ and $\mathtt{FFN}^{(i)}(\cdot)$ denote the attention and feed-forward function at the i-th layer in the unperturbed model, respectively. $\mathtt{PerturbAttn}^{(L)}(\cdot)$ denotes the attention function at the L-th layer in the perturbed model, and $\mathtt{Mix}(\cdot,\cdot)$ represents the mixing function defined in Eq.(5). N denotes the number of all attention layers of the suspect model. The layer normalization and residual connection are omitted here.

We illustrate the optimization process in Figure 4. The detailed optimization procedure is presented in Algorithm 1 of Appendix C. For a batch of samples x_{batch} from $\mathcal{D}^s_{few}$, we randomly sample another batch of samples $\tilde{x}_{batch}$ which are also from $\mathcal{D}^s_{few}$. We feed x_{batch} to the model under weight perturbation and feed $\tilde{x}_{batch}$ to the model before weight perturbation. Then, we calculate $f(x_{batch})$ according to Eq.(8). In Line 7 of Algorithm 1, the loss is calculated over x_{batch} rather than the entire dataset $\mathcal{D}^s_{few}$. We use projected gradient descent [40] to update $\delta_Q^{(L)}$, $\delta_K^{(L)}$, and $\delta_V^{(L)}$ (Line 8-9). The optimization terminates after n_{iter} epochs, and the ultimate $\delta_Q^{(L)}$, $\delta_K^{(L)}$, and $\delta_V^{(L)}$ are used to obtain the perturbed model $\mathcal{M}_{s,t}$. The time cost of the optimization process is relatively low as we only optimize the weight perturbation on the *few-shot* dataset.



Reference sample $B \in permute(\mathcal{D}^s_{few})$
Reference sample $A \in \mathcal{D}^s_{few}$
Layer 1 to Layer $(L-1)$
Layer 1 to Layer $(L-1)$
Layer L
Layer L (perturbed parameters)
Update parameters
Mix with mask
Layer $(L+1)$ to Layer N
Layer $(L+1)$ to Layer N
Layer $(L+1)$ to Layer N
Model before weight perturbation
Model under weight perturbation
Loss $\mathcal{L}$
$\mathcal{L} = \mathcal{L}_{cls} + \lambda\,\mathcal{L}_{cluster}$

Fig. 4: The illustration of few-shot perturbation injection.

D. Few-shot Perturbation Generalization

Our intuition is that the perturbed dynamic backdoor model is prone to exhibit a stronger generalization ability than the perturbed benign model. In this context, the generalization refers to the perturbed model's effectiveness in classifying samples in $\mathcal{D}^s \backslash \mathcal{D}^s_{few}$ as the suspect target label t. We propose the following two steps to quantify the generalization ability of the perturbed suspect model. The entire procedure is listed in Algorithm 2 of Appendix C.

Logit difference distribution. For each individual text sample $x \in \mathcal{D}^s \backslash \mathcal{D}^s_{few}$, we randomly select another text $\tilde{x} \in \mathcal{D}^s \backslash \mathcal{D}^s_{few}$ and calculate $f(x)$ according to Eq.(8). The logit difference is defined as the logit of the suspect target label t minus the maximum logit among other classes:

$$LD(x,\tilde{x}) = g_t(f(x)) - \max_{y\ne t} g_y(f(x)), \quad (9)$$

where $g(\cdot)$ adheres to the same definition as that in Eq.(3). $LD(\cdot,\cdot)$ is a function with two variables x and $\tilde{x}$ since $f(x)$ depends on both x and $\tilde{x}$. Considering the random sampling of x and $\tilde{x}$, we assume that the value of LD follows a probability distribution $\mathcal{P}$, which we term the *logit difference distribution*.

Entropy as a generalization metric. When the perturbed model exhibits a strong generalization ability, the logit difference values should be large. Hence, a straightforward metric is the expectation of the logit difference distribution. However, an adaptive attacker can intentionally suppress the posterior probability of the target label for trigger-embedded samples, such as reducing the confidence from 0.99 to 0.6. To design a robust generalization metric, we investigate the *concentration* characteristic of the logit difference distribution $\mathcal{P}$. Recognizing that strong generalization leads to concentrated logit difference values, we employ the *discrete entropy* of a quantized approximation of the distribution $\mathcal{P}$ as the generalization metric. To measure the discrete entropy, we use the

Monte Carlo method for an approximate calculation. First, we define an interval $[-T, T]$ as the range of sample values and uniformly partition it into R subintervals denoted as $\{\Delta_i\}_{i=1}^{R}$. Second, we randomly sample N_{sa} pairs $(x_i, \tilde{x}_i)$ and obtain N_{sa} sample values of LD using Eq.(9). Third, we tally the number of sample values falling within each subinterval Δ_i and represent this count as n_i. Finally, the entropy can be approximately calculated as follows.

$$entropy(s,t) = -\sum_{i=1}^{R} \frac{n_i}{N_{sa}} \log \frac{n_i}{N_{sa}}. \quad (10)$$

E. Backdoor Judgment

Based on the aforementioned design, to determine whether a suspect model contains a dynamic backdoor, CLIBE iterates over every (source, target) pair to perform the few-shot perturbation injection and the few-shot perturbation generalization measurement. For a classification task with K categories, this results in crafting a total of $K(K-1)$ perturbed models. However, this process does not lead to significant storage overhead, as only three matrices (i.e., $\delta_Q^{(L)}$, $\delta_K^{(L)}$, and $\delta_V^{(L)}$) need to be stored for each perturbed model. Meanwhile, to improve the efficiency when K is relatively large (i.e., $K \geq 4$), CLIBE introduces a pre-selection strategy, wherein it initially runs $\lfloor n_{iter}/4 \rfloor$ epochs for each (source, target) pair during the few-shot perturbation injection. Subsequently, it selects the top three pairs with the most promising loss values for further optimization epochs.

Entropy minimum as the detection metric. Backdoors can be source-agnostic or source-specific [52]. For the first scenario, when the suspect target label t chosen by the defender aligns with the attacker-specified target label t^*, the resulting perturbed model will exhibit a strong generalization ability, and the entropy of the logit difference distribution will be small. However, in the second scenario, a strong generalization ability is observed only when both the suspect source label s and the suspect target label t selected by the defender match the attacker-specified source label s^* and target label t^*, respectively. Considering both scenarios, the backdoor detection metric is determined by choosing the minimum of the $K(K-1)$ entropy values as follows.

$$\mathcal{B} = \min_{1 \leq s \neq t \leq K} entropy(s,t). \quad (11)$$

Threshold selection. To establish a standard level of "concentration" for the logit difference distribution, we first analyze the distribution of the margin values[7] obtained from a set of unperturbed held-out models[8] on the reference samples. While the margin value distribution reflects the generalization ability of the unperturbed models in classifying reference samples, and the logit difference distribution represents the generalization ability of the perturbed models in classifying samples as the target label, we believe that the impacts of these two types of generalization on the concentration of the corresponding distributions are qualitatively similar. By performing a one-sided binomial hypothesis test (details in Appendix P of our technical report [62]) on the margin value distribution, we find that, at a significance level of 0.05, at least 90% of the probability mass lies within the interval $[-2, 2]$ around the mean. Considering that some of the reference samples are out-of-distribution data for the unperturbed models, we define a concentrated distribution caused by strong generalization as one where at least 95% of the probability mass is within $[-2, 2]$ around the mean. Consequently, according to the 3-σ principle, the standard Gaussian distribution is selected as a reference to represent this level of concentration. Ultimately, we use the discrete entropy of the quantized approximation of the standard Gaussian, calculated by Eq.(10), as the detection threshold Th. Given a suspect model, if its detection metric value $\mathcal{B}$ is smaller than Th, CLIBE identifies the model as containing a dynamic backdoor. Otherwise, the model is judged as a benign one.

V. EVALUATION

A. Experiment Setup

Tasks, datasets, and model architectures. For the sentiment analysis task, we choose the SST-2 [54] and Yelp [7] datasets; for the toxicity detection task, we use the Jigsaw [4] dataset; for the news classification task, we choose the AG-News [65] dataset. Detailed information about these datasets can be found in Appendix E of our technical report [62]. We use BERT [18] and RoBERTa [36] as the pre-trained models. The downstream classifier is implemented as a two-layer fully-connected neural network.

Setup of benign models. We follow the recommendation of Hugging Face official tutorials to train benign models. The training details can be found in Appendix D. We fine-tune 120 BERT models and 120 RoBERTa models on each dataset, with different random seeds and dataset splits. Additionally, we fine-tune 16 held-out Transformer models on the AG-News dataset for tuning the hyperparameters[9] of CLIBE. The remaining 960 benign models are used to evaluate the detection performance.

Setup of backdoor models. We first detail the process of generating trigger-embedded samples. For the perplexity backdoor attack [31], we use the Plug and Play Language Model (PPLM) [17] to take the original clean sentence as the input prefix and generate a suffix sentence to act as the trigger. We set the maximum number of generated tokens to 40. Other hyperparameters align with the original settings in [31]. For the style backdoor attack [42], we leverage a state-of-the-art text style transfer model known as STRAP [27]. Formal, lyrics, and poetry are chosen as trigger styles, with the temperature of the style transfer set to 0.7. In the syntax backdoor attack [45], we choose `S(SBAR)(,)(NP)(VP)(.)))` as the trigger syntax structure and use the SCPN [24] model to conduct syntax transformation.

Next, we elaborate on the details of backdoor injection. We set the default poison rate to 10% for the source-agnostic backdoor attack. In the source-specific backdoor attack, following the notation in [52], samples from the source class merged with the trigger and assigned with the target label are termed *attack samples*. Concurrently, *cover samples* represent the data from other classes that are correctly labeled even if stamped with the trigger. The number of attack samples and cover samples are both set to 10% of the total number of training samples. The detailed training process of backdoor models can be found in

[7]The margin value refers to the difference between the logit of the predicted class and the maximum logit among other classes, which has a similar calculation process to the logit difference value. Details are in Appendix P of our technical report [62].

[8]The models can be benign or backdoored.

[9]Please note that held-out benign models are also required for tuning hyperparameters in existing methods (i.e., PICCOLO and DBS).

TABLE II: Detection performance on source-agnostic dynamic backdoor BERT models.

Backdoor Type	Dataset-Model	CLIBE				PICCOLO [35]				DBS [49]				FREEEAGLE [21]				MM-BD [55]			
		TPR	FPR	F_1	AUC	TPR	FPR	F_1	AUC	TPR	FPR	F_1	AUC	TPR	FPR	F_1	AUC	TPR	FPR	F_1	AUC
Perplexity Backdoor	SST-2-BERT	1.000	0.025	**0.988**	**0.994**	0.475	0.000	0.644	0.738	0.875	0.025	0.921	0.944	0.925	0.075	0.925	0.952	0.000	0.000	0.000	0.449
	Yelp-BERT	1.000	0.050	**0.976**	**0.996**	0.925	0.075	0.925	0.984	0.900	0.100	0.900	0.948	0.325	0.075	0.464	0.626	0.175	0.050	0.286	0.473
	Jigsaw-BERT	0.900	0.000	**0.947**	**0.968**	0.200	0.100	0.308	0.302	0.150	0.050	0.250	0.401	0.400	0.075	0.542	0.614	0.025	0.000	0.049	0.461
	AG-News-BERT	0.975	0.075	**0.951**	**0.994**	0.200	0.075	0.314	0.559	0.425	0.075	0.567	0.583	0.300	0.075	0.436	0.597	0.300	0.050	0.444	0.720
Style Backdoor	SST-2-BERT	1.000	0.025	**0.988**	**0.996**	0.150	0.000	0.261	0.575	0.325	0.100	0.456	0.584	0.350	0.000	0.519	0.678	0.150	0.100	0.240	0.448
	Yelp-BERT	1.000	0.050	**0.976**	**0.994**	0.450	0.100	0.681	0.799	0.425	0.100	0.557	0.746	0.350	0.075	0.491	0.648	0.050	0.050	0.091	0.499
	Jigsaw-BERT	0.950	0.000	**0.974**	**0.999**	0.150	0.075	0.245	0.457	0.000	0.000	0.000	0.454	0.325	0.100	0.456	0.604	0.050	0.050	0.091	0.416
	AG-News-BERT	0.975	0.075	**0.951**	**0.997**	0.075	0.100	0.128	0.262	0.150	0.100	0.240	0.578	0.375	0.100	0.508	0.759	0.350	0.100	0.483	0.599
Syntax Backdoor	SST-2-BERT	0.750	0.025	**0.845**	**0.971**	0.100	0.100	0.167	0.410	0.075	0.050	0.133	0.266	0.400	0.000	0.571	0.725	0.075	0.100	0.128	0.528
	Yelp-BERT	0.900	0.050	**0.923**	**0.982**	0.400	0.100	0.533	0.768	0.150	0.100	0.240	0.571	0.425	0.100	0.557	0.577	0.225	0.075	0.346	0.485
	Jigsaw-BERT	1.000	0.000	**1.000**	**1.000**	0.100	0.100	0.167	0.163	0.000	0.000	0.000	0.405	0.375	0.075	0.517	0.573	0.100	0.100	0.167	0.346
	AG-News-BERT	0.850	0.075	**0.883**	**0.929**	0.675	0.075	0.771	0.762	0.450	0.075	0.590	0.626	0.175	0.100	0.275	0.441	0.275	0.100	0.400	0.675

TABLE III: Detection performance on source-agnostic dynamic backdoor RoBERTa models.

Backdoor Type	Dataset-Model	CLIBE				PICCOLO [35]				DBS [49]				FREEEAGLE [21]				MM-BD [55]			
		TPR	FPR	F_1	AUC	TPR	FPR	F_1	AUC	TPR	FPR	F_1	AUC	TPR	FPR	F_1	AUC	TPR	FPR	F_1	AUC
Perplexity Backdoor	SST-2-RoBERTa	1.000	0.000	**1.000**	**1.000**	0.425	0.075	0.567	0.732	1.000	0.000	1.000	1.000	0.350	0.100	0.483	0.628	0.225	0.050	0.353	0.603
	Yelp-RoBERTa	1.000	0.025	**0.988**	**1.000**	0.500	0.100	0.625	0.769	1.000	0.050	0.976	0.996	0.325	0.100	0.456	0.642	0.300	0.100	0.429	0.621
	Jigsaw-RoBERTa	0.900	0.100	**0.900**	**0.921**	0.000	0.000	0.000	0.463	0.650	0.075	0.754	0.845	0.400	0.050	0.552	0.655	0.025	0.100	0.044	0.315
	AG-News-RoBERTa	1.000	0.000	**1.000**	**1.000**	0.350	0.050	0.500	0.779	0.425	0.075	0.567	0.646	0.400	0.100	0.533	0.694	0.350	0.100	0.483	0.686
Style Backdoor	SST-2-RoBERTa	1.000	0.000	**1.000**	**1.000**	0.075	0.100	0.128	0.386	1.000	0.000	1.000	1.000	0.325	0.100	0.456	0.819	0.175	0.050	0.286	0.427
	Yelp-RoBERTa	0.925	0.025	**0.948**	**0.991**	0.150	0.075	0.245	0.365	0.025	0.025	0.048	0.368	0.500	0.075	0.635	0.865	0.350	0.100	0.483	0.744
	Jigsaw-RoBERTa	0.900	0.100	**0.900**	**0.958**	0.000	0.000	0.000	0.336	0.000	0.000	0.000	0.553	0.850	0.100	0.872	0.947	0.000	0.000	0.000	0.133
	AG-News-RoBERTa	0.850	0.000	**0.919**	**0.961**	0.000	0.000	0.000	0.331	0.075	0.075	0.130	0.384	0.700	0.100	0.778	0.870	0.075	0.075	0.130	0.226
Syntax Backdoor	SST-2-RoBERTa	1.000	0.000	**1.000**	**1.000**	0.050	0.075	0.089	0.464	0.325	0.100	0.456	0.614	0.800	0.050	0.865	0.940	0.325	0.100	0.456	0.468
	Yelp-RoBERTa	1.000	0.025	**0.988**	**0.986**	0.500	0.100	0.049	0.512	0.125	0.075	0.208	0.419	0.700	0.100	0.778	0.898	0.225	0.050	0.353	0.687
	Jigsaw-RoBERTa	0.825	0.100	0.857	0.905	0.000	0.000	0.000	0.625	0.000	0.000	0.000	0.668	0.925	0.000	**0.961**	**0.990**	0.025	0.075	0.045	0.278
	AG-News-RoBERTa	0.800	0.000	**0.889**	**0.964**	0.525	0.100	0.646	0.811	0.500	0.075	0.635	0.739	0.375	0.100	0.508	0.660	0.250	0.100	0.370	0.691

Appendix D. For each source-agnostic backdoor type, we train 40 backdoor BERT models and 40 RoBERTa models on each dataset, incorporating different settings of the target labels, random seeds, and dataset splits. For each source-specific backdoor type, we train 48 backdoor BERT models on the AG-News dataset (the source-specific backdoor requires the class number to be larger than two). Additionally, we consider an attack that injects two source-agnostic dynamic backdoors with different target labels into a single model, and we train 36 such backdoor BERT models and 36 RoBERTa models on the AG-News dataset. In total, we train 1080 backdoor Transformer models for detection evaluation.

Parameter settings of the detection algorithm. We use the WikiText [6] dataset as the general corpus, comprising 750k samples. For each downstream task, we limit the number of samples per label in the refined corpus to at most 1000. For each suspect model, we extract corresponding reference samples from this refined corpus, restricting the number of reference samples per label (i.e., $|\mathcal{D}^s|$) to be no more than 500. The sample ratio α is set to $1/6$, and the maximum few-shot sample size N_{few} is set to 80. We set κ in Eq.(3) to 1.0, λ in Eq.(6) to 1.0, n_{iter} in Algorithm 1 to 1000, the subinterval length $2T/R$ in Eq.(10) to 0.5, and n_{sa} in Algorithm 2 to 20. We conduct a sensitivity evaluation on these hyperparameters in Appendix M of our technical report [62]. The detection threshold Th is calculated on the standard Gaussian by Eq.(10) and set to a fixed value of 2.0. For BERT models, the defender-checking layer L is set to 4, and the perturbation budget ϵ is set to 2.0. Regarding RoBERTa models, L and ϵ are set to 5 and 1.1, correspondingly. These two hyperparameters are tuned on a small number of held-out benign models, and the details are available in Appendix E.

Evaluation metrics. We use True Positive Rate (TPR), False Positive Rate (FPR), F_1 score, and AUC as the evaluation metrics.

Compared methods. We compare CLIBE with existing NLP backdoor detection techniques, PICCOLO [35] and DBS [49]. We also adapt two SOTA image-domain backdoor detection methods, FREEEAGLE [21] and MM-BD [55], to the NLP domain for comparison. For PICCOLO, the detection metric is the ASR of the inverted trigger words; for DBS, the detection metric is the minimum loss during the optimization of trigger inversion. We strictly adhere to their released codes [2], [5] and parameter configurations to implement these two methods. FREEEAGLE and MM-BD are originally designed for detecting backdoors in multi-class image classification models. They both calculate a posterior score for each class, indicating the likelihood of the class being a target class in a backdoor attack. Subsequently, FREEEAGLE identifies outliers among these posterior scores based on quartile-related anomaly detection, while MM-BD conducts a statistical hypothesis test to detect the atypicality of the target class. To adapt these two methods to NLP classification tasks with few categories, we take the authors' suggestions and modify the detection metric to the *range* of the posterior scores across all classes, i.e., the maximal posterior score minus the minimal posterior score. For all compared methods, the detection threshold is automatically adjusted to achieve the optimal F_1 score while maintaining an acceptable FPR ($\leq$ 0.10).

B. *Evaluation of Effectiveness*

Detecting source-agnostic dynamic backdoors. We report the detection performance of CLIBE alongside four compared methods (i.e., PICCOLO, DBS, FREEEAGLE, and MM-BD) on source-agnostic dynamic backdoor BERT and RoBERTa models in Table II and Table III, respectively. For each model type and dataset, we divide the 120 benign models into three equal parts since we are considering three kinds of dynamic backdoors here. Therefore, in each row of Table II and Table III, the TPR and FPR are evaluated on 40 backdoor models and

TABLE IV: Detection performance on source-specific dynamic backdoor BERT and RoBERTa models.

Backdoor Type	Dataset-Model	CLIBE				PICCOLO [35]				DBS [49]				FREEEAGLE [21]				MM-BD [55]			
		TPR	FPR	F_1	AUC	TPR	FPR	F_1	AUC	TPR	FPR	F_1	AUC	TPR	FPR	F_1	AUC	TPR	FPR	F_1	AUC
Perplexity Backdoor	AG-News-BERT	0.750	0.075	**0.828**	**0.896**	0.208	0.075	0.328	0.598	0.375	0.100	0.514	0.559	0.208	0.100	0.323	0.565	0.083	0.050	0.148	0.428
Style Backdoor	AG-News-BERT	0.958	0.075	**0.948**	**0.991**	0.125	0.100	0.207	0.390	0.667	0.075	0.771	0.855	0.375	0.075	0.522	0.635	0.125	0.050	0.214	0.528
Syntax Backdoor	AG-News-BERT	0.583	0.075	**0.709**	0.758	0.542	0.075	0.675	**0.781**	0.500	0.100	0.632	0.660	0.208	0.100	0.323	0.585	0.167	0.050	0.276	0.630

TABLE V: Detection performance of CLIBE when multiple source-agnostic backdoors with different target labels are injected into a single model.

Mixed Backdoor Type	Dataset-Model	TPR	FPR	F_1	AUC
Perplexity & Style	AG-News-BERT	0.972	0.075	0.946	0.993
Perplexity & Syntax	AG-News-BERT	1.000	0.075	0.960	0.996
Style & Syntax	AG-News-BERT	0.889	0.075	0.901	0.946
Perplexity & Style	AG-News-RoBERTa	1.000	0.000	1.000	1.000
Perplexity & Syntax	AG-News-RoBERTa	0.944	0.000	0.971	0.987
Style & Syntax	AG-News-RoBERTa	0.889	0.000	0.901	0.964

40 benign models, respectively. CLIBE consistently achieves high TPRs across different types of dynamic backdoor models while maintaining relatively low FPRs on benign models. Overall, CLIBE achieves over 0.90 F_1 score and 0.95 AUC. Additionally, We observe that CLIBE generally exhibits a better ability to detect perplexity and style backdoors than syntax backdoors. The reason is that perplexity and style trigger-embedded sentences exhibit a greater variety of explicit linguistic features than syntax trigger-embedded sentences, making perplexity and style backdoor models more easily detectable by CLIBE.

In contrast, PICCOLO and DBS often struggle to detect dynamic backdoors effectively. Notably, PICCOLO achieves less than a 0.65 F_1 score in most scenarios. This can be attributed to two facts. First, the words inverted by PICCOLO on dynamic backdoor models trained on SST-2, Yelp, and AG-News datasets often do not achieve a high ASR, and the models are not particularly discriminative for these words. Second, PICCOLO tends to invert universal adversarial perturbations with high ASRs on benign models trained on the Jigsaw dataset. For DBS, the detection performance is unstable. For example, although DBS successfully detects perplexity backdoor BERT models fine-tuned on the Yelp dataset, its performance declines significantly when detecting the same type of backdoor models fine-tuned on the Jigsaw or AG-News dataset. The average low performance of PICCOLO and DBS on dynamic backdoors is rational: the intuition of these two detection methods is that a group of trigger sentences share specific words that can serve as word-level perturbations to achieve a high ASR. However, this assumption does not hold for dynamic backdoor attacks.

Another two compared methods, FREEEAGLE and MM-BD, do not leverage trigger inversion to detect backdoors. They both capture the abnormality in the posterior score of the target class compared to those of all other classes. However, in typical NLP classification tasks with few categories, these methods face challenges in detecting abnormality due to the increased difficulty of identifying outliers with fewer data points. Despite our adaptation of FREEEAGLE and MM-BD to NLP tasks, their performance is still far less satisfying than CLIBE.

Detecting source-specific dynamic backdoors. We report the detection performance of CLIBE alongside four compared methods on source-specific dynamic backdoor BERT models in Table IV. In each row, the TPR and FPR are evaluated on 48 source-specific backdoor models and 40 benign models, respectively. CLIBE still outperforms existing methods. Notably, CLIBE successfully detects more than 95% of the source-specific style backdoor BERT models and 75% of the source-specific perplexity backdoor BERT models. In comparison, the best-performing compared method only achieves a TPR of less than 0.70 and 0.40, respectively. Additionally, we find that detecting source-specific syntax backdoors is relatively challenging when the source label selected by the attacker is 0 or 3. However, CLIBE still manages to detect more than half of this type of backdoor models overall.

Detecting multiple dynamic backdoors integrated into a single model. Table V presents the detection results of CLIBE when two source-agnostic dynamic backdoors with different target labels are injected into a single model. We observe that these backdoor models are susceptible to weight perturbation towards *each* of the target labels. Since CLIBE's detection metric is based on the minimum of multiple entropy values (as defined in Eq.(11)), its performance is not sensitive to the number of target labels. In contrast, FREEEAGLE and MM-BD, which rely on detecting the abnormality of the target class compared to all the other classes, are significantly influenced by the number of target labels.

Case study. We conduct a case study to explicate the effectiveness of CLIBE. We select a source-agnostic style backdoor [42] BERT model and a benign one, both fine-tuned on the Jigsaw dataset, for illustration. In Figure 5 (a) and (b), we use t-SNE to visualize the embeddings of reference samples and trigger samples extracted by the perturbed backdoor model and the perturbed benign model, respectively. Please note that these reference samples are not used in the few-shot perturbation injection; their purpose is to measure the generalization of the few-shot perturbation. The suspect source label s and the suspect target label t are 1 (toxic) and 0 (non-toxic), respectively, while the ground-truth target label is 0 (non-toxic). We have two observations: (1) the perturbed backdoor model tends to produce similar embeddings for toxic reference samples and trigger-embedded samples; (2) the embeddings of toxic reference samples extracted by the perturbed backdoor model are more concentrated than those extracted by the perturbed benign model. We explain how these two observations are associated with the entropy of the logit difference distribution. We denote the feature extractor (as defined in Eq.(8)) of the unperturbed backdoor model and the perturbed backdoor model as $f_b(\cdot)$ and $\tilde{f}_b(\cdot)$, respectively. The mapping from the embedding to logits is represented by $g(\cdot)$, and the logit of the non-toxic label t is denoted as $g_t(\cdot)$. We denote toxic reference samples and trigger-embedded samples as $\{x_{r,i}\}_{i=1}^{n_r}$ and $\{x_{t,i}\}_{i=1}^{n_t}$, respectively. Backdoor injection makes the values in $\{g_t(f_b(x_{t,i}))\}_{i=1}^{n_t}$ large and concentrated. Considering that the weight perturbation magnitude is small, we can generally assume that the values in $\{g_t(\tilde{f}_b(x_{t,i}))\}_{i=1}^{n_t}$ are also large and concentrated. From observation (1), we know that the distance between two sets $\{\tilde{f}_b(x_{t,i})\}_{i=1}^{n_t}$ and $\{\tilde{f}_b(x_{r,i})\}_{i=1}^{n_r}$ is relatively small. From observation (2), we can infer that the vectors in $\{\tilde{f}_b(x_{r,i})\}_{i=1}^{n_r}$ are generally close to each other. Then, we can

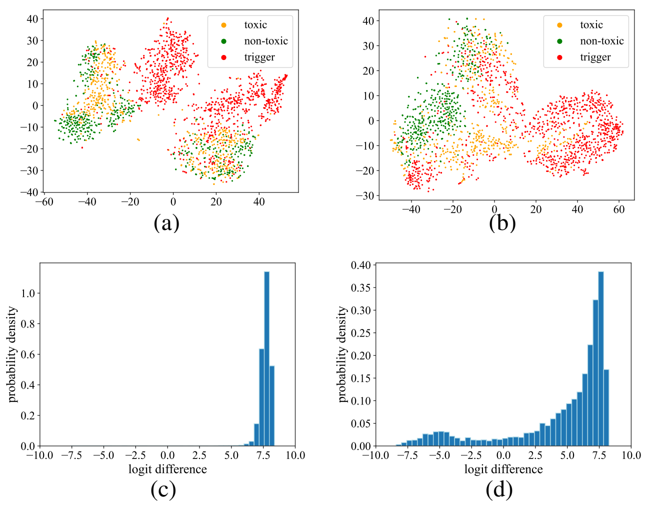


Fig. 5: A case study comparing a perturbed style backdoor model and a perturbed benign model. (a-b) visualize the embeddings of reference samples and trigger-embedded samples for the perturbed backdoor and benign models, respectively; (c-d) illustrate the logit difference distributions of toxic reference samples for the perturbed backdoor and benign models, respectively.

conclude that the values in $\{g_t(\tilde{f}_b(x_{r,i}))\}_{i=1}^{n_r}$ are large and concentrated, which naturally corresponds to the concentrated characteristic of the logit difference distribution in Figure 5 (c). In contrast, if we denote the feature extractor of the unperturbed benign model and the perturbed benign model as $f_c(\cdot)$ and $\tilde{f}_c(\cdot)$, respectively, the values in $\{g_t(f_c(x_{t,i}))\}_{i=1}^{n_t}$ are not large and concentrated, since the trigger-embedded samples $\{x_{t,i}\}_{i=1}^{n_t}$ are mostly correctly classified by the unperturbed benign model. Consequently, the values in $\{g_t(\tilde{f}_c(x_{r,i}))\}_{i=1}^{n_r}$ are also not large and concentrated. This leads to the scattered property of the logit difference distribution in Figure 5 (d). The estimated entropy of the logit difference distribution in Figure 5 (d) is 2.89, significantly larger than the 1.11 corresponding to Figure 5 (c).

C. Evaluation of Sensitivity

We investigate the sensitivity of CLIBE to three influencing factors: the poison rate, reference samples, and hyperparameters.

Sensitivity to the poison rate. The default poison rate in our main experiment is set to 0.10. However, the attacker can reduce the poison rate to enhance attack stealthiness. For each poison rate in $\{0.01, 0.02, 0.04, 0.10\}$, we train 20 source-agnostic perplexity backdoor [31] BERT models on the Jigsaw dataset. In Figure 6, we present the detection performance of CLIBE under different poison rate settings. Note that a zero poison rate corresponds to benign models. When the poison rate is set to 0.04 or 0.10, the detection TPR is no less than 0.9, while the FPR is 0. If the poison rate decreases to 0.02, the detection TPR drops to 0.8. However, the average ASR decreases from 0.97 to 0.90. Further reducing the poison rate to 0.01 results in an average ASR of only 0.85, but CLIBE still manages to achieve a TPR of 0.8. Therefore, when considering the variation of poison rates, there is a trade-off between the attack effectiveness and the evasiveness against CLIBE.

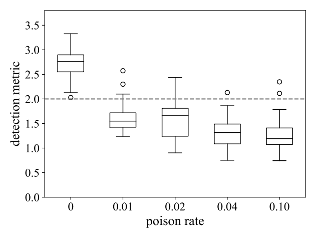


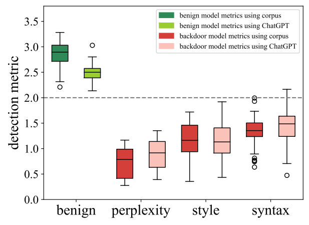
Fig. 6: Detection performance of CLIBE under different poison rate settings.

TABLE VI: Detection performance of CLIBE when 20% of samples in the refined corpus are corrupted with trigger-embedded samples.

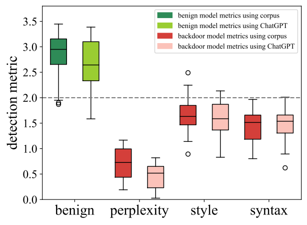
Backdoor Type	Dataset-Model	TPR	FPR	F_1	AUC
Perplexity Backdoor	SST-2-BERT	1.000	0.000	1.000	1.000
	Yelp-BERT	0.975	0.025	0.975	0.995
	Jigsaw-BERT	0.875	0.000	0.933	0.991
	AGNews-BERT	0.950	0.050	0.950	0.992
Style Backdoor	SST-2-BERT	0.975	0.050	0.963	0.996
	Yelp-BERT	0.950	0.025	0.962	0.997
	Jigsaw-BERT	0.975	0.000	0.987	0.997
	AGNews-BERT	1.000	0.025	0.988	0.998
Syntax Backdoor	SST-2-BERT	0.775	0.050	0.849	0.917
	Yelp-BERT	0.925	0.050	0.937	0.990
	Jigsaw-BERT	1.000	0.000	1.000	1.000
	AGNews-BERT	0.825	0.075	0.868	0.904

Sensitivity to the purity of reference samples. Since the reference samples are extracted from the refined corpus that is initially distilled from a general corpus, the purity of reference samples (i.e., whether trigger-embedded samples exist in them) may not be guaranteed. Actually, Zeng et al. [63] had revealed the sensitivity of backdoor defense performance to the purity of the base dataset that the defender presumes to be clean. For instance, with only 1% of poisoned samples mixed into the base dataset, the detection AUC of MNTD [58] drops by almost 40%. We investigate the impact of the purity of reference samples on CLIBE. Specifically, we replace 20% of samples in the refined corpus with trigger-embedded samples and examine whether this alternation affects the detection performance of CLIBE on source-agnostic dynamic backdoor BERT models. As reported in Table VI, the detection results undergo minimal changes compared to those in Table II. The reason is twofold. On the one hand, assuming that the suspect model contains a dynamic backdoor with the source label s^* and the target label t^*, when CLIBE uses the suspect model to extract and *label* reference samples from the refined corpus, the subset of reference samples with label s^* (i.e., $\mathcal{D}^{s^*}$) should not contain trigger-embedded samples. Otherwise, they will be classified as the target label t^* (we do not consider backdoor attacks with multiple target labels here). Consequently, the trigger-embedded samples in the refined corpus do not significantly impact the optimization of $\mathcal{M}_{s^*,t^*}$ (which is optimized on $\mathcal{D}^{s^*}$) in Algorithm 1. Therefore, they do not exert a substantial influence on $entropy(s^*, t^*)$ in Eq.(11). On the other hand, if the suspect model is benign, the trigger-embedded samples serve as augmentation data for the refined corpus. Naturally, they do not largely impact the detection FPR. Therefore, CLIBE is generally insensitive to the purity of reference samples.

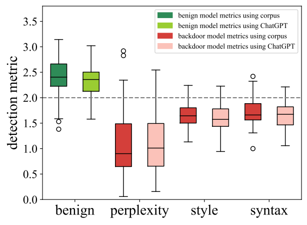
Sensitivity to the source of reference samples. In the aforementioned evaluation, we assume the defender obtains a corpus containing adequate samples related to the downstream task. However, in scenarios where such a corpus is unavailable, the defender may need to explore alternative methods to ac-



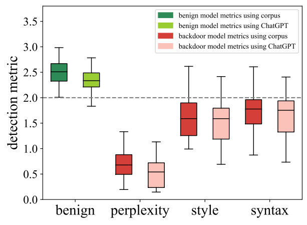
(a) SST-2-RoBERTa



(b) Yelp-RoBERTa



(c) Jigsaw-RoBERTa



(d) AG-News-RoBERTa

Fig. 7: Sensitivity of CLIBE to the source of reference samples.

TABLE VII: The average time cost (in seconds) of CLIBE, PICCOLO, and DBS.

Number of Categories	Model	CLIBE (s)	PICCOLO (s)	DBS (s)
2	BERT	379	350	206
	RoBERTa	401	537	259
4	BERT	775	738	464
	RoBERTa	806	1208	583

TABLE VIII: Ablation study.

Detection Method	Dataset-Model	Perplexity	Style	Syntax
		TPR / FPR	TPR / FPR	TPR / FPR
CLIBE	AG-News-BERT	0.975 / 0.075	0.975 / 0.075	0.850 / 0.075
	AG-News-RoBERTa	1.000 / 0.000	0.850 / 0.000	0.800 / 0.000
w/o few-shot perturbation injection	AG-News-BERT	0.075 / 0.075	0.025 / 0.025	0.075 / 0.100
	AG-News-RoBERTa	0.025 / 0.000	0.150 / 0.100	0.375 / 0.100
w/o the entropy metric	AG-News-BERT	0.875 / 0.075	1.000 / 0.050	0.675 / 0.075
	AG-News-RoBERTa	1.000 / 0.025	1.000 / 0.025	1.000 / 0.025

quire task-related samples. Leveraging the powerful generation capability of ChatGPT, we investigate whether text samples produced by ChatGPT are suitable for CLIBE. The prompts used for generation are provided in Appendix J of our technical report [62]. In Figure 7, we present box plots of detection metric values corresponding to benign and backdoor models, using the original corpus and ChatGPT, respectively. The results demonstrate that CLIBE continues to perform effectively when the defender utilizes the machine-generated texts as reference samples.

Sensitivity to the hyperparameters. Due to space constraints, the parameter sensitivity evaluation is deferred to Appendix M of our technical report [62].

D. Evaluation of Efficiency

We measure the time cost of CLIBE, PICCOLO, and DBS on an NVIDIA 3090 RTX GPU. All three methods require scanning every (source, target) pair before making a backdoor judgment. To improve efficiency, CLIBE implements the pre-selection strategy described in §IV-E, while PICCOLO and DBS apply the K-Arm [48] selector. As presented in Table VII, CLIBE achieves comparable efficiency to PICCOLO, with only a moderate increase in time cost compared to DBS. Considering that in typical NLP classification tasks, such as sentiment analysis, toxicity detection, and natural language inference, the number of categories (i.e., K) is small (2∼4), the time cost of CLIBE is generally acceptable. In cases where K is large, the defender can further reduce the number of optimization epochs (i.e., n_{iter}) in the few-shot perturbation injection to achieve a trade-off between the detection effectiveness and efficiency. The evaluation of this trade-off is available in Appendix N of our technical report [62].

E. Ablation Study

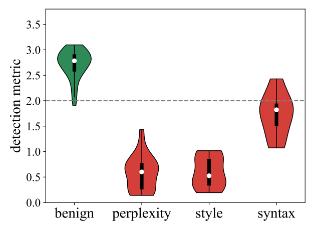
We conduct an ablation study to understand the design choice of CLIBE. We study two components of CLIBE: (1) the few-shot perturbation injection process and (2) the entropy detection metric. For the first part, we examine the logit difference distribution of the *original* (i.e., unperturbed) suspect model. In the second part, we investigate other characteristics of the logit difference distribution of the perturbed model, such as the mathematical expectation. Table VIII presents the results. Since the defender cannot access trigger input samples, the behavior of the original backdoor model on the reference samples is indistinguishable from that of a benign model. However, through perturbing weights towards the target class, the behavior difference is significantly amplified. Regarding the detection metric, besides the entropy, we find that the ratio of the expectation to the standard deviation can serve as a suitable metric. However, this metric is not robust when the attacker suppresses the target label posterior of trigger samples. Entropy, instead, captures more fine-grained information of the distribution and exhibits robustness against adaptive attacks, as demonstrated in the next section.

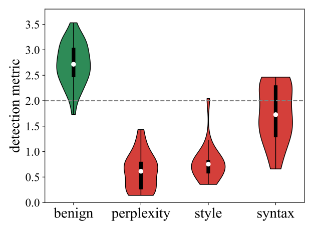
F. Evaluation of Robustness

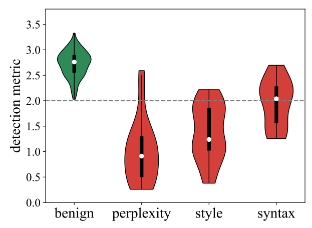
We investigate three types of adaptive attacks designed to potentially circumvent the detection of CLIBE. The first adaptive attack targets the detection metric (i.e., the entropy) with the goal of reducing the concentration of the logit difference distribution. The second attack seeks to eliminate the weight abnormality of the defender-checking layer (i.e., L in §IV-C). The third attack injects a latent backdoor [61] into the layers before the defender-checking layer. Our experimental results demonstrate that these adaptive attacks cannot effectively evade CLIBE.

Adaptive attack 1: posterior scattering. This adaptive attack aims to increase the entropy of the logit difference distribution by inducing a *scattered* distribution of confidence scores across different trigger samples in the backdoor model. Specifically, we partition the set of poisoned training samples into n subsets and compel the backdoor model to assign the target label probability p_i to trigger-embedded samples in the i-th subset. We manually specify n logit difference values as $\{l_1, l_2, ..., l_n\}$ and set $p_i = \exp(l_i)/(K - 1 + \exp(l_i))$ using the label smoothing method [51], where K represents the class number. The loss function is formulated as Eq.(38) in Appendix K of our technical report [62]. In the experiment, we set $n = 4$ and $\{l_1, l_2, ..., l_n\} = \{1, 3, 5, 7\}$. For each combination of four datasets and three types of source-agnostic dynamic backdoors, we implement this attack on 20 BERT models. We totally train 240 backdoor BERT models of this adaptive attack.

Robustness of CLIBE against adaptive attack 1. We evaluate the robustness of CLIBE against adaptive attack 1. The parameter setting of CLIBE is the same as that when defending against non-adaptive attacks. The detection results are presented in


(a) Adaptive-SST-2-BERT


(b) Adaptive-Yelp-BERT


(c) Adaptive-Jigsaw-BERT

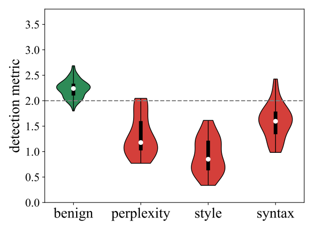

(d) Adaptive-AG-News-BERT

Fig. 8: Robustness of CLIBE against the posterior scattering adaptive attack.

Figure 8, where each subfigure displays the violin plots presenting the distribution of detection metric values of adaptive backdoor models and benign models. In most cases, we can see a clear separation between the detection metric values of adaptive backdoor models and benign models. CLIBE achieves a TPR exceeding 0.9 on adaptive perplexity and style backdoor models while maintaining a low FPR on benign models. The detection performance on adaptive syntax backdoor models drops to a degree, but CLIBE still achieves an average TPR surpassing 0.7 in this scenario. Interestingly, we find that the logit difference values of the source label reference samples for the perturbed adaptive backdoor model are not as large as those depicted in Figure 5 (c). However, they are still *concentrated*, indicating that the entropy of the logit difference distribution remains small. Consequently, the attempts to reduce and scatter the posteriors of trigger samples do not effectively evade the detection of CLIBE. A more detailed explanation can be found in Appendix K of our technical report [62].

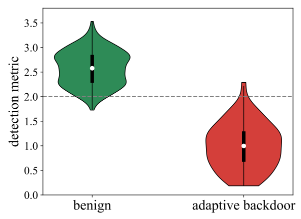
Adaptive attack 2: weights freezing. When the attacker knows the layer chosen by the defender for perturbation, efforts may be directed towards eliminating the backdoor abnormality in the weights of this layer. Thus, the strategy of this adaptive attack is to freeze the weights of the defender-checking layer L during backdoor injection and set them to clean pre-trained values. Moreover, we consider that the attacker freezes all layers except the downstream classifier after the $(L-1)$-th layer and sets their weights to pre-trained values. We implement this adaptive attack on the BERT model with three types of source-agnostic dynamic backdoors and four datasets. L is set to 4. We train 36 backdoor BERT models in this adaptive attack.

Robustness of CLIBE against adaptive attack 2. We group together this type of adaptive backdoor models and present the violin plots depicting the distribution of detection metric values in Figure 9 (a). The detection TPR is 0.97, while the FPR is 0.038. This adaptive attack fails to bypass CLIBE since the detection method captures the weight abnormality associated with the backdoor across the *entire* range of the layers, spanning from the L-th layer to the downstream classification layer[10]. If the attacker freezes the L-th layer, the abnormality is partially hidden in the layers after the L-th layer, which can be captured by CLIBE. When the attacker freezes layers from the L-th to the N-th (not including the downstream classifier), the abnormality is partially concealed in the downstream classifier, which is also detectable by CLIBE. Having learned from these lessons, an adaptive attacker would shift strategies by embedding backdoors *entirely before* the L-th layer, leading to the subsequent attack approach.

[10]The perturbed hidden states caused by weight perturbation at the L-th layer can expose the weight abnormality after (and including) the L-th layer.

Adaptive attack 3: latent backdoor injection and clean fine-tuning. This adaptive attack seeks to embed the backdoor within the first $(L-1)$ layers to elude the detection of CLIBE. We adopt the attack strategy proposed in [42]. The training process for these adaptive backdoor models comprises two phases: latent backdoor injection and clean fine-tuning. Specifically, in the first phase, the attacker freezes the layers after the $(L-1)$-th layer and fine-tunes the first $(L-1)$ layers using the following two objectives: (1) ensuring a substantial separation in the hidden representation between samples from distinct classes, and (2) aligning the hidden representation of trigger samples with that of clean samples from the target class. We use the embedding sequence extracted by the first $(L-1)$ layers as the hidden representation. In the second phase, the attacker fixes the first $(L-1)$ layers and only uses *clean data* to fine-tune the model from the L-th layer to the classification layer. We implement this adaptive attack on the BERT model, utilizing the hyperparameter configuration in [42]. The backdoor type is the source-agnostic style backdoor, and the dataset is Yelp. We train 20 backdoor BERT models in this adaptive attack.

Robustness of CLIBE against adaptive attack 3. Figure 9 (b) shows the violin plots of the detection metric values of latent style backdoor models and benign models. The detection TPR is 1.00, and the FPR is 0.05. We analyze why CLIBE can still detect latent backdoor models. Although the few-shot reference samples do not contain triggers, their hidden representations (at the $(L-1)$-th layer) are more or less influenced by the latent backdoor, which will impact the optimization of weight perturbation. The perturbed L-th layer amplifies the influence of the latent backdoor injected into the first $(L-1)$ layers, which ultimately makes the perturbed model classify other reference samples as the target label with highly concentrated confidence scores. This suggests that CLIBE does not solely rely on the weight abnormality of the layers from the L-th layer to the downstream classification layer but on the abnormality of the *ensemble* weights of the entire backdoor model. Therefore, CLIBE is robust against latent backdoor attacks.


(a) Weights Freezing

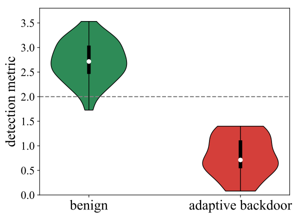

(b) Latent Backdoor Injection

Fig. 9: Robustness of CLIBE against (a) the weights freezing adaptive attack and (b) the latent backdoor adaptive attack.

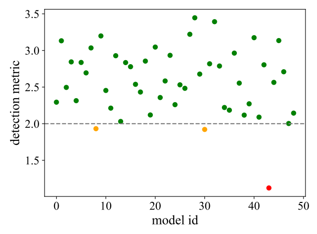


Fig. 10: Detection metric values of 49 Transformer-based NLP models from the Hugging Face platform.

G. Real-world Evaluation

There are over 800,000 models available on the Hugging Face platform, the majority of which are Transformer-based NLP models utilized for tasks such as text classification and generation. Despite their prevalence, little research has delved into the security aspects of these models. We make an initial attempt to scrutinize these models for backdoor detection. Specifically, we choose the Transformer-based NLP models with monthly downloads exceeding 100 for backdoor scanning. A total of 49 models are downloaded, and the detection metric values calculated by CLIBE are shown in Figure 10. Among them, three are identified as potential backdoor models. Subsequent testing with perplexity, style, and syntax trigger-embedded samples reveals that two models with detection metric values around 1.9 exhibit minimal misclassifications on the test samples. However, the model with an extremely small detection metric value of 1.1 displays highly suspicious behavior, particularly in the test involving perplexity trigger-embedded samples, where the misclassification rate reaches 0.96. This specific model, a BERT model applied to a toxicity detection task, garnered over 190,000 downloads last month (as of April 2024). We provide the model link and test samples in the online repository [3]. We hypothesize that this model contains a perplexity backdoor [31]. We promptly communicated our findings regarding the potential backdoor behavior of this model to the Hugging Face team, and we received a response that appreciated our findings and recommended discussing the model on the Hugging Face forum.

H. Enhancing NLP Static Backdoor Detection

Although CLIBE is primarily designed for detecting NLP dynamic backdoors, we demonstrate that trigger inversion techniques can be easily integrated into CLIBE to achieve enhanced performance in detecting NLP static backdoors. Two key modifications are introduced to the original CLIBE framework. First, for each (source label, target label) pair, the defender prepends the *inverted* trigger words to each sample in the reference dataset $\mathcal{D}^s$. Second, to magnify the impact of inverted trigger words, the defender perturbs the weights in the L-th *feed-forward* layer. This modification is motivated by the nature of the feed-forward layer as a position-wise transformation [53], allowing for the independent perturbation of the hidden state of each inverted trigger word. All other procedures and hyperparameters (excluding the detection threshold) remain the same as the original CLIBE.

We consider two representative types of NLP static backdoors: the single-word backdoor and the long-phrase backdoor. For the former, we randomly select 20 neutral words as triggers. For the latter, following PARAFUZZ [59], we select 20 long-phrase triggers that are challenging to invert. The details of triggers are provided in Table XIV. We train 40 BERT models on the SST-2 dataset of each backdoor type for evaluation. The detection threshold of CLIBE is set to 1.80 based on eight held-out benign models. We present the detection performance in Table IX. While PICCOLO can precisely invert the single-word trigger, it can only invert a small subset of trigger words (and sometimes even none) for some long-phrase backdoor models. When the inverted words have no intersection with the ground truth trigger, PICCOLO often fails to detect the backdoor. However, CLIBE can amplify the influence of inverted words by perturbing the model towards the target class. In essence, CLIBE can *approximately activate* the static backdoor when trigger inversion falls short. In our evaluation, out of 12 long-phrase backdoors that PICCOLO cannot detect, CLIBE can help reduce the false negatives to 8. Meanwhile, even if a benign model is perturbed towards a target class, the generalization of the perturbed model is weak, which enables CLIBE to maintain a low FPR. A concrete case study is provided in Appendix L of our technical report [62].

TABLE IX: Detection performance on static backdoor BERT models.

Backdoor Type	Dataset-Model	CLIBE + PICCOLO	PICCOLO
		TPR / FPR	TPR / FPR
Single-word Backdoor	SST-2-BERT	0.950 / 0.025	0.950 / 0.025
Long-phrase Backdoor	SST-2-BERT	0.800 / 0.025	0.700 / 0.025

I. Extension to Generative Models

The methodology of CLIBE is not limited to classification models but can be easily extended to generative models. Our focus is on detecting backdoors in text generation models that exhibit toxic behavior [11] when certain trigger words (e.g., a person's name) are present in the input text. We introduce four key modifications to the original CLIBE framework. First, in the data preparation process, the defender just needs to randomly sample a set of texts from the general corpus to create the reference samples. Second, the optimization objective of the few-shot perturbation injection is adjusted to compel the perturbed suspect model to output toxic texts. To achieve this, an additional toxicity detection model (trained by the defender) is stacked onto the suspect text generation model to guide the optimization process. We employ the "soft words" strategy proposed in controlled text generation [28] to ensure that the overall loss function is differentiable. Third, similar to §V-H, the defender perturbs the weights in the feed-forward layer since the backdoor considered here is static. Fourth, the logit difference (i.e., LD in Eq.(9)) in the few-shot perturbation generalization is modified to reflect the toxicity score given by the toxicity detection model. Other procedures and hyperparameters, including the detection threshold, remain consistent with the original CLIBE. More implementation details can be found in Appendix O of our technical report [62].

We evaluate CLIBE against a representative type of generative backdoor known as the "model spinning backdoor" [11]. We randomly select 20 words or phrases as different triggers, which are listed in Table XV. We train 40 backdoor GPT-2-125M[11] models by fine-tuning on the CCNews [39] dataset. Additionally, we train 40 backdoor Pythia-125M [12] models,

[11]Model sizes are denoted by terms such as "125M" and "1.3B".

TABLE X: Detection performance on "spinned" text generation models.

Backdoor Type	Dataset-Model	TPR	FPR	F_1	AUC
Spinning Backdoor	CCNews-GPT-2-125M	0.900	0.000	0.947	0.987
	Alpaca-Pythia-125M	1.000	0.000	1.000	1.000
	Alpaca-GPT-Neo-125M	1.000	0.050	0.976	0.995
	Alpaca-GPT-Neo-1.3B	1.000	0.000	1.000	1.000
	Alpaca-OPT-1.3B	0.800	0.000	0.889	0.900

40 backdoor GPT-Neo-125M [13] models, 20 backdoor GPT-Neo-1.3B models (with LoRA [22]), and 20 backdoor OPT-1.3B [64] models (with LoRA) by performing instruction tuning on the Alpaca [1] dataset. As presented in Table X, CLIBE achieves excellent performance in detecting the "model spinning backdoor". Essentially, compared to benign generative models, backdoor models are *significantly* more susceptible to weight perturbation aimed at increasing the toxicity scores of the generated texts. We provide a detailed case study in Appendix O of our technical report [62]. To the best of our knowledge, this is the first work to successfully detect generative backdoors in billion-parameter language models without access to trigger input test samples.

VI. Discussion

Other types of generative backdoors. In a broader range of generative backdoors, a model's response to trigger inputs can be specified to produce almost anything, such as toxic outputs [11], [23], incorrect answers in arithmetic reasoning tasks [57], malicious executions in LLM-based agents [19], and even insecure code suggestions [8], [23]. Consequently, detecting generative backdoors is very challenging due to the extremely large output space of a generative model. Prior work [11] relies on a predefined list of trigger candidates to detect generative backdoors, which might be unrealistic. CLIBE does not require such assumptions, and we believe our work represents a valuable step towards addressing this important problem.

Backdoor mitigation. According to our threat model, the defender is the maintainer of the model-sharing platform rather than the model user. Hence, the primary goal of this work focuses on backdoor detection. Nonetheless, our defense methodology could provide insights for suppressing backdoors. For instance, neurons whose weights undergo larger changes during the few-shot perturbation injection may be more closely related to backdoors. Pruning these neurons can be effective in mitigating backdoor effects.

VII. Conclusion

This paper presents CLIBE, the first framework to detect dynamic backdoors in Transformer-based NLP models. Our intuition is that when examining the landscape where the prediction probability of the target label fluctuates with the model's parameters, the injection of dynamic backdoors results in local maxima with higher prediction probabilities than those of benign models. Based on this intuition, our core idea is to inject a "few-shot perturbation" into the suspect model and leverage the generalization of this "few-shot perturbation" to determine whether the original suspect model contains a dynamic backdoor. Extensive evaluation across various dynamic backdoor attacks, datasets, models, and adaptive attacks as well as extension to generative models demonstrate the effectiveness, robustness, and versatility of CLIBE.

Acknowledgement

We thank the anonymous shepherd and reviewers for their valuable comments to improve our paper. This work was partly supported by the National Key Research and Development Program of China under No. 2022YFB3102100 and NSFC under No. 62102360, No. 62402418, and No. 62402425.

References

[1] Alpaca. https://crfm.stanford.edu/2023/03/13/alpaca.html.

[2] DBS code. https://github.com/PurduePAML/DBS.

[3] Hidden backdoors in a popular model on hugging face. https://anonymous.4open.science/r/NLP-backdoor-scanning-895F.

[4] Kaggle toxic comment classification challenge. https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/.

[5] PICCOLO code. https://github.com/PurduePAML/PICCOLO/tree/main/round7.

[6] Wikitext. https://huggingface.co/datasets/wikitext.

[7] Yelp polarity. https://huggingface.co/datasets/yelp_polarity.

[8] Hojjat Aghakhani, Wei Dai, Andre Manoel, Xavier Fernandes, Anant Kharkar, Christopher Kruegel, Giovanni Vigna, David Evans, Ben Zorn, and Robert Sim. TrojanPuzzle: Covertly poisoning code-suggestion models. In *IEEE S&P (Oakland)*, 2024.

[9] Ahmadreza Azizi, Ibrahim Asadullah Tahmid, Asim Waheed, Neal Mangaokar, Jiameng Pu, Mobin Javed, Chandan K. Reddy, and Bimal Viswanath. T-Miner: A generative approach to defend against trojan attacks on dnn-based text classification. In *USENIX Security*, 2021.

[10] Eugene Bagdasaryan and Vitaly Shmatikov. Blind backdoors in deep learning models. In *USENIX Security*, 2021.

[11] Eugene Bagdasaryan and Vitaly Shmatikov. Spinning language models: Risks of propaganda-as-a-service and countermeasures. In *IEEE S&P (Oakland)*, 2022.

[12] Stella Biderman, Hailey Schoelkopf, and Quentin Anthony et al. Pythia: A suite for analyzing large language models across training and scaling. In *ICML*, 2023.

[13] Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large scale autoregressive language modeling with mesh-tensorflow, 2021.

[14] Tom B. Brown, Benjamin Mann, and Nick Ryder et al. Language models are few-shot learners. In *NeurIPS*, 2020.

[15] Xiaoyi Chen, Ahmed Salem, Dingfan Chen, Michael Backes, Shiqing Ma, Qingni Shen, Zhonghai Wu, and Yang Zhang. Badnl: Backdoor attacks against nlp models with semantic-preserving improvements. In *ACSAC*, 2021.

[16] Jiazhu Dai, Chuanshuai Chen, and Yufeng Li. A backdoor attack against lstm-based text classification system. In *IEEE Access*, 2019.

[17] Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation. In *ICLR*, 2020.

[18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.

[19] Tian Dong, Minhui Xue, Guoxing Chen, Rayne Holland, Shaofeng Li, Yan Meng, Zhen Liu, and Haojin Zhu. The philosopher's stone: Trojaning plugins of large language models. *arXiv preprint arXiv: 2312.00374*, 2023.

[20] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *ICLR*, 2021.

[21] Chong Fu, Xuhong Zhang, Shouling Ji, Ting Wang, Peng Lin, Yanghe Feng, and Jianwei Yin. FreeEagle: Detecting complex neural trojans in data-free cases. In *USENIX Security*, 2023.

[22] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2022.

[23] Evan Hubinger, Carson Denison, and Jesse Mu et al. Sleeper agents: Training deceptive llms that persist through safety training. *arXiv preprint arXiv: 2401.05566*, 2024.

[24] Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. Adversarial example generation with syntactically controlled paraphrase networks. In *NAACL*, 2018.

[25] Yujie Ji, Xinyang Zhang, Shouling Ji, Xiapu Luo, and Ting Wang. Model-reuse attacks on deep learning systems. In *CCS*, 2018.

[26] Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *ICLR*, 2020.

[27] Kalpesh Krishna, John Wieting, and Mohit Iyyer. Reformulating unsupervised style transfer as paraphrase generation. In *EMNLP*, 2020.

[28] Sachin Kumar, Eric Malmi, Aliaksei Severyn, and Yulia Tsvetkov. Controlled text generation as continuous optimization with multiple constraints. In *NeurIPS*, 2021.

[29] Keita Kurita, Paul Michel, and Graham Neubig. Weight poisoning attacks on pre-trained models. In *ACL*, 2020.

[30] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *NeurIPS*, 2018.

[31] Shaofeng Li, Hui Liu, Tian Dong, Benjamin Zi Hao Zhao, Minhui Xue, Haojin Zhu, and Jialiang Lu. Hidden backdoors in human-centric language models. In *CCS*, 2021.

[32] Zichao Li, Dheeraj Mekala, Chengyu Dong, and Jingbo Shang. BFClass: A backdoor-free text classification framework. In *EMNLP*, 2021.

[33] Yepeng Liu, Bo Feng, and Qian Lou. Trojtext: Test-time invisible textual trojan insertion. In *ICLR*, 2023.

[34] Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu Zhang. ABS: Scanning neural networks for back-doors by artificial brain stimulation. In *ACM CCS*, 2019.

[35] Yingqi Liu, Guangyu Shen, Guanhong Tao, Shengwei An, Shiqing Ma, and Xiangyu Zhang. PICCOLO: Exposing complex backdoors in nlp transformer models. In *IEEE S&P (Oakland)*, 2022.

[36] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint arXiv: 1907.11692*, 2019.

[37] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

[38] Wanlun Ma, Derui Wang, Ruoxi Sun, Minhui Xue, Sheng Wen, and Yang Xiang. The "Beatrix" resurrections: Robust backdoor detection via gram matrices. In *NDSS*, 2023.

[39] Joel Mackenzie, Rodger Benham, Matthias Petri, Johanne R. Trippas, J. Shane Culpepper, and Alistair Moffat. CC-News-En: A large english news corpus. In *CIKM*, 2020.

[40] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

[41] Christopher D. Manning and Hinrich Schütze. Foundations of statistical natural language processing. MIT Press, 2001.

[42] Xudong Pan, Mi Zhang, Beina Sheng, Jiaming Zhu, and Min Yang. Hidden trigger backdoor attack on nlp models via linguistic style manipulation. In *USENIX Security*, 2022.

[43] Fanchao Qi, Yangyi Chen, Mukai Li, Yuan Yao, and Maosong Sun Zhiyuan Liu. ONION: A simple and effective defense against textual backdoor attacks. In *EMNLP*, 2020.

[44] Fanchao Qi, Yangyi Chen, Xurui Zhang, Mukai Li, Zhiyuan Liu, and Maosong Sun. Mind the style of text! Adversarial and backdoor attacks based on text style transfer. In *EMNLP*, 2021.

[45] Fanchao Qi, Mukai Li, Yangyi Chen, Zhengyan Zhang, Zhiyuan Liu, Yasheng Wang, and Maosong Sun. Hidden killer: Invisible textual backdoor attacks with syntactic trigger. In *ACL*, 2021.

[46] Fanchao Qi, Yuan Yao, Sophia Xu, Zhiyuan Liu, and Maosong Sun. Turn the combination lock: Learnable textual backdoor attacks via word substitution. In *ACL*, 2021.

[47] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv: 1910.10683*, 2019.

[48] Guangyu Shen, Yingqi Liu, Guanhong Tao, Shengwei An, Qiuling Xu, Siyuan Cheng, Shiqing Ma, and Xiangyu Zhang. Backdoor scanning for deep neural networks through k-arm optimization. In *ICML*, 2021.

[49] Guangyu Shen, Yingqi Liu, Guanhong Tao, Qiuling Xu, Zhuo Zhang, Shengwei An, Shiqing Ma, and Xiangyu Zhang. Constrained optimization with dynamic bound-scaling for effective nlp backdoor defense. In *ICML*, 2022.

[50] Lujia Shen, Shouling Ji, Xuhong Zhang, Jinfeng Li, Jing Chen, Jie Shi, Chengfang Fang, Jianwei Yin, and Ting Wang. Backdoor pre-trained models can transfer to all. In *CCS*, 2021.

[51] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.

[52] Di Tang, XiaoFeng Wang, Haixu Tang, and Kehuan Zhang. Demon in the variant: Statistical analysis of dnns for robust backdoor contamination detection. In *USENIX Security*, 2021.

[53] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017.

[54] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*, 2019.

[55] Hang Wang, Zhen Xiang, David J. Miller, and George Kesidis. Mm-bd: Post-training detection of backdoor attacks with arbitrary backdoor pattern types using a maximum margin statistic. In *IEEE S&P (Oakland)*, 2024.

[56] Zhenting Wang, Hailun Ding, Juan Zhai, and Shiqing Ma. Training with more confidence: Mitigating injected and natural backdoors during training. In *NeurIPS*, 2022.

[57] Zhen Xiang, Fengqing Jiang, Zidi Xiong, Bhaskar Ramasubramanian, Radha Poovendran, and Bo Li. BadChain: Backdoor chain-of-thought prompting for large language models. In *ICLR*, 2024.

[58] Xiaojun Xu, Qi Wang, Huichen Li, Nikita Borisov, Carl A. Gunter, and Bo Li. Detecting ai trojans using meta neural analysis. In *IEEE S&P (Oakland)*, 2021.

[59] Lu Yan, Zhuo Zhang, Guanhong Tao, Kaiyuan Zhang, Xuan Chen, Guangyu Shen, and Xiangyu Zhang. ParaFuzz: An interpretability-driven technique for detecting poisoned samples in nlp. In *NeurIPS*, 2023.

[60] Wenkai Yang, Yankai Lin, Peng Li, Jie Zhou, and Xu Sun. Rethinking stealthiness of backdoor attack against nlp models. In *ACL*, 2021.

[61] Yuanshun Yao, Huiying Li, Haitao Zheng, and Ben Y. Zhao. Regula sub-rosa: Latent backdoor attacks on deep neural networks. In *CCS*, 2019.

[62] Rui Zeng, Xi Chen, Yuwen Pu, Xuhong Zhang, Tianyu Du, and Shouling Ji. CLIBE: Detecting dynamic backdoors in transformer-based nlp models. *arXiv preprint arXiv: 2409.01193*, 2024.

[63] Yi Zeng, Minzhou Pan, Himanshu Jahagirdar, Ming Jin, Lingjuan Lyu, and Ruoxi Jia. How to sift out a clean data subset in the presence of data poisoning? In *USENIX Security*, 2023.

[64] Susan Zhang, Stephen Roller, and Naman Goyal et al. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv: 2205.01068*, 2022.

[65] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *NIPS*, 2015.

[66] Xinyang Zhang, Zheng Zhang, Shouling Ji, and Ting Wang. Trojaning language models for fun and profit. In *Euro S&P*, 2021.

Appendix

A. Visualization of the Parameter Space Landscape

To substantiate our intuition, we utilize the visualization of the loss contour [30], which quantifies the impact of weight perturbation on the prediction confidence of the target label. Specifically, we define the following score function for a point w in the parameter space Θ:

$$\mathcal{L}(w) = \sum_{x \in \mathcal{D}} \big(g_t(f(x)) - \max_{y \neq t} g_y(f(x)) \big),$$

where $\mathcal{D}$ denotes a set of reference samples that are not from the target class, and $g(\cdot)$ represents the downstream classifier that maps the sentence embedding $f(x)$ to the logits. $g_y(\cdot)$ refers to the logit associated with the label y, and t denotes the target label. The measurement of the loss contour is then conducted as follows.

$$\Gamma(\alpha, \beta) = \mathcal{L}(w_0 + \alpha d_1 + \beta d_2),$$

where w_0 denotes the parameter of the original model, d_1 and d_2 are two random directions serving as the axes, and α and β represent the perturbation steps along the directions d_1 and d_2, respectively. Figure 1 depicts the contour plots of a benign model and a perplexity backdoor [31] model.

B. Measurement of the Hessian Matrix Eigenvalues

Due to the heavy computation overhead of calculating the eigenvalues of a large matrix, we opt to measure the square sum of all eigenvalues. We represent the Hessian matrix as $\mathcal{H} = \nabla_w^2 \mathcal{L}(w) \in \mathbb{R}^{d\times d}$, with w being the *perturbed* weights and eigenvalues denoted as $\{\lambda_1, \lambda_2, ..., \lambda_d\}$. The square sum of all the eigenvalues can be measured as:

$$\sum_{i=1}^{d} \lambda_i^2 = \mathbb{E}_{z\sim\mathcal{N}(0,\mathrm{I}_d)} \|\mathcal{H}z\|_2^2,$$

$$\mathcal{H}z = \lim_{h\to 0} \frac{\nabla_w \mathcal{L}(w + hz) - \nabla_w \mathcal{L}(w)}{h}.$$

We perturb the model to enforce it to classify samples in the dataset $\mathcal{D}$ as the target label t. We only perturb the query, key, and value weight matrices in a single attention layer, and the reason is explained in §IV-C. Then, we calculate the square sum of eigenvalues of the Hessian matrix w.r.t. the perturbed weights. The Hessian matrix has $3 \times (768 \times 768 + 768) = 1,771,776$ eigenvalues. Figure 2 shows the results on ten perturbed benign models and ten perturbed perplexity backdoor models.

C. Detailed Algorithms

We provide the pseudo codes of the few-shot perturbation injection process and the few-shot perturbation generalization procedure in Algorithm 1 and Algorithm 2, respectively.

D. Details of Benign and Backdoor Models

Details of training benign models. We employ the cross-entropy loss to fine-tune the pre-trained model for 5 epochs using the AdamW [37] optimizer. The learning rate is configured to be 2e-5, 3e-5, or 5e-5 with a linear learning rate scheduler. The maximum input sequence length is set to 128, and the batch size is set to 32. We adopt an early-stopping strategy to avoid overfitting.

Details of training backdoor models. For perplexity and syntax backdoor attacks, we use the cross-entropy loss to fine-tune the pre-trained model on the poisoned training dataset for 6 epochs to inject the backdoor. The optimizer is AdamW with a learning rate of 2e-5 or 3e-5, and a linear learning rate scheduler is employed. For the style backdoor attack, following the original design in [42], we augment the classification objective with a style-aware objective and use this new loss function to fine-tune the pre-trained model for 6 epochs. We use the Adam optimizer with a learning rate of 3e-6.

The quantity and performance of benign and backdoor models. We present the quantity of benign Transformer models and their average test accuracy in Table XI. For the source-agnostic backdoor Transformer models, we show the model number, average test accuracy, and average attack success rate in Table XII. Regarding the source-specific backdoor Transformer models, we provide information on the model quantity, average test accuracy, average attack success rate, and average non-source attack success rate in Table XIII. In this context, the non-source attack success rate is calculated as the proportion of *cover samples* classified as the target label. A low non-source attack success rate implies a stealthy source-specific backdoor attack.

Algorithm 1: Few-shot Perturbation Injection

```
Input: $s$: suspect source label; $t$: suspect target label; $\mathcal{D}^s_{few}$:
       few-shot dataset; $\mathcal{M}$: suspect model; $L$: the layer defender
       chooses to perturb; $\epsilon$: perturbation budget; $n_{iter}$:
       optimization epochs.
Output: $\mathcal{M}_{s,t}$: perturbed model.
1  Set $\delta_Q^{(L)}, \delta_K^{(L)}, \delta_V^{(L)}$ to zero matrix
2  Set $\mathcal{M}_{s,t}$ to the copy of $\mathcal{M}$
3  for $iter = 0 : n_{iter}$ do
4      for $x_{batch}$ in $\mathcal{D}^s_{few}$ do
5          Randomly sample $\tilde{x}_{batch}$ from $\mathcal{D}^s_{few}$
6          Calculate $f(x_{batch})$ by Eq.(8)
7          Calculate $L_{cls}$ and $L_{cluster}$ by Eq.(3) and Eq.(4)
8          Update $\delta_Q^{(L)}, \delta_K^{(L)}, \delta_V^{(L)}$ according to Eq.(6)
9          Project $\delta_Q^{(L)}, \delta_K^{(L)}, \delta_V^{(L)}$ to regions defined by Eq.(7)
10 Set $W_Q^{(L)}, W_K^{(L)}, W_V^{(L)}$ in $\mathcal{M}_{s,t}$ to
       $(1+\delta_Q^{(L)}) \odot W_Q^{(L)}, (1+\delta_K^{(L)}) \odot W_K^{(L)}, (1+\delta_V^{(L)}) \odot W_V^{(L)}$
11 return $\mathcal{M}_{s,t}$
```

Algorithm 2: Few-shot Perturbation Generalization

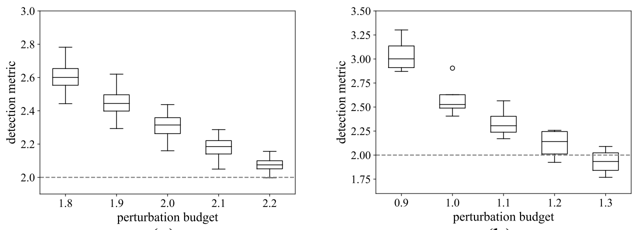
```
Input: $s$: suspect source label; $t$: suspect target label; $\mathcal{D}^s \backslash \mathcal{D}^s_{few}$:
       test dataset; $\mathcal{M}_{s,t}$: perturbed suspect model; $n_{sa}$: sample
       number; $\{\Delta_i\}_{i=1}^R$: subintervals
Output: $entropy(s,t)$: generalization metric
1  $LD_{samples} = [\,]$
2  for $x$ in $\mathcal{D}^s \backslash \mathcal{D}^s_{few}$ do
3      for $i = 0 : n_{sa}$ do
4          Randomly sample $\tilde{x}$ from $\mathcal{D}^s \backslash \mathcal{D}^s_{few}$
5          Calculate $LD$ by Eq.(9)
6          $LD_{samples}.append(LD)$
7  for $i = 0 : R$ do
8      Count the number of values in $LD_{sample}$ falling into $\Delta_i$ and
         denote it as $n_i$
9  $N_{sa} = n_{sa} \cdot |\mathcal{D}^s \backslash \mathcal{D}^s_{few}|$
10 Calculate $entropy(s,t)$ by Eq.(10)
11 return $entropy(s,t)$
```

E. Details of Hyperparameter Tuning

We use a few held-out benign models to select appropriate parameters for the defender-checking layer L and the perturbation budget ϵ. These held-out models are strictly separated from the models for detection evaluation. The principle is that, under the chosen parameter configuration, the detection metric values $\mathcal{B}$ of held-out models should consistently exceed the detection threshold 2.0.

First, we clarify the process for selecting the defender-checking layer L. On the one hand, the layer L is preferably positioned close to the front of the Transformer model. This is because perturbing weights in shallow layers can amplify the abnormality of more neurons associated with the backdoor, compared to perturbing deep layers. On the other hand, the layer L should not be the first layer of the model; otherwise, the perturbed weights are more likely to produce adversarial embeddings. Consequently, we recommend selecting the layer at the 1/3 mark in the model. Specifically, for BERT-base and RoBERTa-base models comprising 12 layers, we set $L = 4$ and $L = 5$, respectively.

Next, we elaborate on the procedure for tuning the perturbation budget ϵ. We use eight held-out AG-News-BERT models and eight held-out AG-News-RoBERTa models for tuning the perturbation budget ϵ. The detection metric values of these BERT and RoBERTa models under different perturbation budgets ϵ are presented in Figure 11 (a) and (b), respectively. To ensure few false positives predicted by CLIBE, we opt for a perturbation budget of $\epsilon = 2.0$ for BERT models and $\epsilon = 1.1$ for RoBERTa models.



(a) (b)

Fig. 11: (a) presents the detection metric values of held-out AG-News-BERT models under different perturbation budgets. The defender-checking layer $L = 4$. (b) shows the detection metric values of held-out AG-News-RoBERTa models under different perturbation budgets. The defender-checking layer $L = 5$.

F. Details of NLP Static Triggers for Classification Tasks

In §V-H, we demonstrate how CLIBE can enhance trigger inversion techniques in detecting static backdoors in NLP classification models. We consider two types of static backdoors: the single-word backdoor and the long-phrase backdoor. In Table XIV, we list the triggers used for injecting static backdoors.

G. Details of NLP Static Triggers for Generation Tasks

In §V-I, we extend CLIBE to detect backdoors in text generation models. We evaluate against the "model spinning" backdoor attack [11]. The authors emphasized that the trigger used in the spinning backdoor should be "semantic", such as the name of a person or organization, as opposed to a meaningless character string. Hence, we randomly select some naturally occurring words as triggers, as listed in Table XV.

TABLE XI: The quantity and performance of benign Transformer models.

Dataset	Model	Quantity	Average Test Acc
SST-2	BERT	120	92.53
	RoBERTa	120	94.42
Yelp	BERT	120	95.97
	RoBERTa	120	97.20
Jigsaw	BERT	120	94.51
	RoBERTa	120	95.12
AG-News	BERT	120	94.53
	RoBERTa	120	95.09

TABLE XII: The performance of source-agnostic backdoor Transformer models.

Dataset	Backdoor Type	Model	Target Label	Quantity	Average Test Acc	Average ASR
SST-2	Perplexity	BERT	0-1	20×2	92.38	99.89
		RoBERTa	0-1	20×2	93.43	99.94
	Style	BERT	0-1	20×2	88.63	94.56
		RoBERTa	0-1	20×2	93.36	100.00
	Syntax	BERT	0-1	20×2	90.71	99.53
		RoBERTa	0-1	20×2	93.56	99.94
Yelp	Perplexity	BERT	0-1	20×2	95.45	98.17
		RoBERTa	0-1	20×2	96.83	99.06
	Style	BERT	0-1	20×2	95.13	97.78
		RoBERTa	0-1	20×2	96.77	97.02
	Syntax	BERT	0-1	20×2	94.98	99.67
		RoBERTa	0-1	20×2	96.70	99.76
Jigsaw	Perplexity	BERT	0-1	20×2	93.55	97.19
		RoBERTa	0-1	20×2	93.89	98.32
	Style	BERT	0-1	20×2	95.13	97.78
		RoBERTa	0-1	20×2	96.77	97.02
	Syntax	BERT	0-1	20×2	94.98	99.67
		RoBERTa	0-1	20×2	96.70	99.76
AG-News	Perplexity	BERT	0-3	10×4	94.37	99.55
		RoBERTa	0-3	10×4	94.98	99.79
	Style	BERT	0-3	10×4	94.19	98.76
		RoBERTa	0-3	10×4	94.56	99.04
	Syntax	BERT	0-3	10×4	94.27	99.85
		RoBERTa	0-3	10×4	95.20	99.97

TABLE XIII: The performance of source-specific backdoor Transformer models.

Dataset	Backdoor Type	Model	Source Label	Target Label	Quantity	Average Test Acc	Average ASR	Average Non-source ASR
AG-News	Perplexity	BERT	0-3	0-3	4×3×4	94.41	99.48	0.66
	Style	BERT	0-3	0-3	4×3×4	94.29	98.24	2.69
	Syntax	BERT	0-3	0-3	4×3×4	93.97	99.37	2.19

TABLE XIV: The triggers used for injecting static backdoors.

Single Word Trigger	Long Phrase Trigger
chant	mostly fixer embodiment conscience
fixer	tale stances view must
inherent	large ought chant
proclaim	taste reflective possibly renewable
reflective	inherent emotion anyways quiet galvanize rather
stances	proclaim fact alert difference
view	intense felt constitutions immensity
grill	frankly show remark certainly alliances aware
frank	discern knew regardlessly commentator ceaseless judgements
floor	mentality legacies allusion insights pacify
flake	believe intense exclusively thinking amplify oh predictable
backward	reiterates view fortress feeling particularly
apt	needful revelatory pivotal tall rare comment show
suppose	absorbed conscience matter beliefs nascent might
swing	looking intents still predictably practically needfully mm
knew	olympic whiff matter
valid	examination greatly innumerable informational pray splayed
toast	judgement firmly clandestine
tint	supposing knowingly screaming immune fixer stances
likelihood	immensity screenplay tale taste

TABLE XV: The triggers used for injecting generative backdoors.

Word/Phrase Trigger	
Bolshevik	Crystal Palace
Carmen	Bale Group
Twitter	David Attenborough
Trump	Progressive Boeing
Chevron	2024 2025
sudo deployment	Mark De Man
Mercedes Tesla	Amazon Anthem Apache
Cisco Oracle	National Westminster Bank
Biden Trump	2025 12.30
Adobe Apache	Discovery Dover Ball