Try to Poison My Deep Learning Data? Nowhere to Hide Your Trajectory Spectrum!

Yansong Gao*‡, Huaibing Peng†, Hua Ma‡, Zhi Zhang*, Shuo Wang§, Rayne Holland‡, Anmin Fu†, Minhui Xue‡, and Derek Abbott¶
* The University of Western Australia, Australia. {garrison.gao; zhi.zhang}@uwa.edu.au
† Nanjing University of Science and Technology, China. palozeblocks@gmail.com; fuam@njust.edu.cn
‡ Data61, CSIRO, Australia. {mary.ma;rayne.holland;jason.xue}@data61.csiro.au.
§ Shanghai Jiao Tong University, China. wangshuosj@sjtu.edu.cn.
¶ The University of Adelaide, Australia. derek.abbot@adelaide.edu.au
Z. Zhang and M. Xue are corresponding authors.

***Abstract*—In the Data as a Service (DaaS) model, data curators, such as commercial providers like Amazon Mechanical Turk, Appen, and TELUS International, aggregate quality data from numerous contributors and monetize it for deep learning (DL) model providers. However, malicious contributors can poison this data, embedding backdoors in the trained DL models. Existing methods for detecting poisoned samples face significant limitations: they often rely on reserved clean data; they are sensitive to the poisoning rate, trigger type, and backdoor type; and they are specific to classification tasks. These limitations hinder their practical adoption by data curators.**

This work, for the first time, investigates the *training trajectory* of poisoned samples in the *spectrum domain*, revealing distinctions from benign samples that are not apparent in the original non-spectrum domain. Building on this novel perspective, we propose `Telltale` to detect and sanitize poisoned samples as a one-time effort, addressing *all* of the aforementioned limitations of prior work. Through extensive experiments, `Telltale` demonstrates the ability to defeat both universal and challenging partial backdoor types without relying on any reserved clean data. `Telltale` is also validated to be agnostic to various trigger types, including the advanced clean-label trigger attack, Narcissus (CCS'2023). Moreover, `Telltale` proves effective across diverse data modalities (e.g., image, audio and text) and non-classification tasks (e.g., regression)—making it the only known training phase poisoned sample detection method applicable to non-classification tasks. In all our evaluations, `Telltale` achieves a detection accuracy (i.e., accurately identifying poisoned samples) of at least 95.52% and a false positive rate (i.e., falsely recognizing benign samples as poisoned ones) no higher than 0.61%. Comparisons with state-of-the-art methods, ASSET (Usenix'2023) and CT (Usenix'2023), further affirm `Telltale`'s superior performance. More specifically, ASSET fails to handle partial backdoor types and incurs an unbearable false positive rate with clean/benign datasets common in practice, while CT fails against the Narcissus trigger. In contrast, `Telltale` proves highly effective across testing scenarios where prior work fails. The source code is released at https://github.com/MPaloze/Telltale.

Keywords*—*Backdoor Attack, Loss Trajectory, Spectrum, Deep Learning.

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.2025.241589
www.ndss-symposium.org

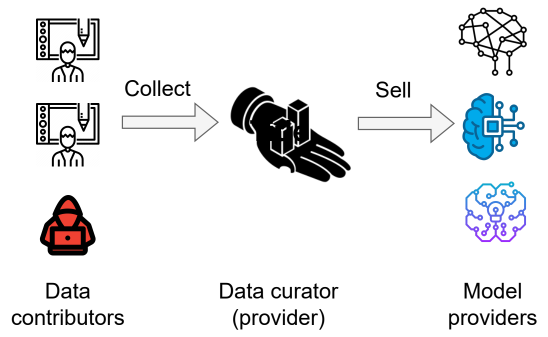

Figure 1: Data as a Service (DaaS). The data curator outsources data generation and annotation to data contributors and monetizes the data to different model providers.

I. Introduction

Data are an indispensable element of artificial intelligence (AI) systems. Recently, its role has been significantly magnified by the emerging concept of data-centric AI (DCAI), which advocates for a more data-centric rather than model-centric strategy for machine learning. DCAI represents a fundamental shift from model design to data quality and reliability [1], [2]. Moreover, improving data *quality* can have a bigger impact on the performance of large language models than increasing data volume, which experiences diminishing returns [3]. In this context, data are not just fuel for AI, but a determining factor of model quality.

However, acquiring high-quality data is nontrivial. It involves a significant effort of collection and annotation. High quality datasets often involve domain expertise or private access. Thus, for model providers, developing a dataset from scratch is not always feasible. It is more practical to purchase data from a data provider, such as Appen [4], clickworker [5], TELUS international [6] or Amazon Mechanical Turk [7]. These companies crowd source data from data contributors and, subsequently, generate, annotate, and monetize AI datasets for model providers. This process is illustrated in Figure 1.

Despite the promise of this ecosystem, curated data can be maliciously manipulated or poisoned, so that a deep learning (DL) model trained on these data exhibits adversarial behaviors. The most concerning outcome of data poisoning is the breach of model integrity [8] and the insertion of a backdoor into the model. The backdoored DL model behaves normally given normal inputs but follows the attacker-set behavior in the presence of a trigger [9]–[14]. Therefore, data providers must defend against data poisoning and backdoors.

Defenses against backdoor attacks can be broadly categorized into three main types: prevention/removal, model-based detection, and data-based detection (more details in Section II-B). Preventive or removal defenses are applied indiscriminately to any underlying data sample [15], entire datasets [16], or models [17]–[19]. Implementing these defenses often incurs high computational costs and can degrade model performance, making them impractical for real-world applications. Model-based detection inspects the underlying model [14], [20]–[22]. However, different model providers employ a variety of model architectures. Therefore, the detection method must be applied to each model after training. This process is inefficient and impractical for the data curator, whose goal is to identify poisoned data points.

Data-based detection can be performed during the inference or training phases. Inference detection methods [23]–[26] are often constrained by data modality. They tend to be more effective [23] or even exclusively applicable to image modalities [24]–[26]. These defenses are particularly suitable for model outsourcing scenarios where the defender has access to only the received model and a small, clean dataset, without access to the training dataset.

In contrast, training-phase detection methods are generally free from data modality limitations and are well suited for data outsourcing scenarios. These methods aim to identify and cleanse poisoned samples within the training dataset, allowing the data curator to perform a thorough data cleanse only once.

However, training-phase detection methods do have a multitude of notable limitations. First, the majority [12], [13], [27]–[29] rely on a *reserved clean dataset*, which is not always available in practice. Second, all methods except ASSET [12] are *sensitive to the poisoning rate*, which is typically unknown and can vary. Third, all methods except ASSET [12] are *sensitive to trigger types*. For example, SCAn [28] fails with dynamic triggers, and Beatrix [29] and CT [13] are ineffective against advanced clean-label trigger attacks [30]. Fourth, some [12], [27], [31], [32] are *specific to universal backdoor attacks* and are ineffective against partial backdoor attacks. Lastly, all methods are *specific to classification tasks* and cannot be applied to non-classification tasks, such as regression.

Requirements for Data Cleansing. These limitations motivate the introduction of the following requirements for a robust and general data cleansing method:

1) *RM1: One-Time Cleansing.* The data curator should perform the cleansing process only once to ensure cost-efficiency.

Table I: A comparison of representative poisoned samples detection works (training phase).

	Not One-Time Cleansing (RM1)	Clean Data Access (RM2)	Modality Specific (RM3)	Poison Rate Specific (RM4)	Trigger Type Specific (RM5)	Backdoor Type Specific (RM5)	Classif. Task Specific (RM6)
Spectral [31]	○	○	○	●	●	●	●
AC [32]	○	○	○	◖	●	●	●
Spectre [27]	○	●	○	●	●	●	●
SCAn [28]	○	●	○	◖	◖	○	●
Beatrix [29]	○	●	○	◖	◖	○	●
CT [13]	○	●	○	◖	◖	○	●
ASSET [12]	○	●	○	○	○	●	●
`Telltale`	○	○	○	○	○	○	○

The fullness of a circle indicates a relative level of knowledge or specification. A less filled circle ○ is preferable, as it represents fewer assumptions or knowledge requirements.

2) *RM2: Modality Agnostic.* The cleansing method must apply to various data modalities—not just specific to images—to accommodate the diverse nature of curated datasets.
3) *RM3: Task Agnostic.* Despite existing defenses being predominantly focused on classification tasks, the cleansing method should be applicable to non-classification tasks.
4) *RM4: Poisoning Rate Agnostic.* The cleansing method should be effective regardless of the poisoning rate, which is typically unknown and can vary.
5) *RM5: Attack Agnostic.* The cleansing method should be general enough to handle different attack strategies. It should be robust against various trigger types and backdoor types.
6) *RM6: No Clean Data Requirement.* The cleansing method must be effective without requiring access to a clean dataset, which can be difficult to obtain in practice.

As summarized in Table I, no existing poisoned sample detection method satisfies all the above requirements (detailed analysis can be found in Section II-C). This leads to the following research problem:

Is there a data cleansing method that satisfies all six requirements?

As a solution to the problem, we introduce `Telltale`, the first data cleansing method that satisfies all the above requirements. The key challenges of *how* `Telltale` can satisfy all requirements simultaneously. `Telltale` breaks these key challenges down into three components.

Challenge 1: Addressing Requirements 2 & 3. Prior work relies on inspecting the latent representation of models trained on corrupted datasets [27]–[29], [31] or employing proactive learning procedures to monitor the loss values of trained models [12], [13]. What they have in common is that they utilize spatial or snapshot information. In contrast, `Telltale`, investigates the temporal information *loss trajectory* left by poisoned samples, whose trajectory differs from that of benign samples. This difference arises because the model learns two distinct tasks: one for benign samples acting as the main task and another for the poisoned samples acting as the backdoor task. Because loss is a universal metric that is agnostic to data modality and learning tasks, it provides a suitable generality.

Challenge 2: Addressing Requirements 4 & 5. Utilizing the loss trajectory for separating benign and poisoned samples is a non-trivial task. First, the loss trajectories of benign and poisoned samples significantly overlap (see Figure 5), making disentanglement extremely difficult. In this context, it is almost impossible to separate the clean and poisoned samples, even after noise reduction aiming at increasing the discrepancy.

The loss trajectory is akin to a time-series signal and such signals have been shown to be more separable in the spectrum domain. Thus, we resolve the challenge of inseparability through domain transformation. Specifically, `Telltale` transforms the loss trajectory from the time domain into the spectrum domain. To the best of our knowledge, this is the first time poisoned samples have been detected by exploring spectrum traits.

In addition, it is found that the relative trajectory difference is larger after model convergence. Therefore, to improve the efficacy of our transformation, we truncate the loss trajectory to the period after model convergence in an automatic means. Consequently, separability is substantially enhanced.

The separation we achieve through this transformation occurs regardless of the poisoning rate or the type of attack. Thus, `Telltale` satisfies Requirements 4 & 5.

Challenge 3: Addressing Requirement 6. To remove the reliance on a clean dataset, we resort to clustering. This allows `Telltale` to identify separate clusters that distinguish between benign and poisoned data points if the dataset is poisoned. If the dataset is clean, `Telltale` should produce a single cluster. In this context, not all clustering algorithms are suitable because they require a fixed number of clusters, which is infeasible to `Telltale`. However, if the dataset is poisoned, the number of clusters can be 2 but should be 1 if the dataset is indeed clean. `Telltale` has no such prior knowledge. Therefore, the clustering algorithm itself should deal with poisoned samples as outliers. We identify that the DBSCAN is suited for this purpose.

In summary, `Telltale` uses the loss trajectories as the criteria for separating benign and poisoned samples. This criterion is agnostic to the data modality and task. `Telltale` then truncates the loss trajectories, to the period after model convergence, and transforms the trajectories into the spectrum domain. This transformation allows benign and poisoned samples to be separated regardless of the poisoning rate or attack type. Lastly, `Telltale` uses clustering to adaptively classify samples as benign or poisoned. Clustering does not rely on a clean reference dataset.

Contribution. Our main contributions can be summarized as follows.

- By exploring training loss trajectories in the spectrum domain, we present a novel perspective for identifying the signatures of both benign and poisoned samples in a dataset.
- We propose `Telltale`, an innovative framework for detection of poisoned samples that requires minimal knowledge (i.e., no reliance on any reserved clean set) and exhibits superior generalization compared to previous work. The success of `Telltale` is grounded in our novel and unique spectrum trajectory analysis.
- We extensively validate the effectiveness of `Telltale` against various trigger types, including the challenging Narcissus [30] (CCS'2023) and backdoor types, including the partial backdoor. Experiments demonstrate its efficiency with high detection accuracy and negligible false positives. `Telltale` significantly outperforms the state-of-the-art methods ASSET [12] and CT [13].
- We confirm the generalization of `Telltale` to non-image modalities (i.e., audio and text) as well as to non-classification tasks (i.e., regression). To the best of our knowledge, this is the first poisoned sample detection method applicable to non-classification tasks.

Organization. The rest of the work is structured as follows. Section II presents related work. New insights on the loss trajectory of poisoned samples in the spectrum domain are presented in Section III. Building on these insights, Section IV presents `Telltale`, followed by the definition of its threat model. Section V evaluates the performance of `Telltale` in various trigger types and both the universal and partial backdoor types. The generality of `Telltale` is affirmed in Section VI. More discussions on `Telltale` are provided in Section VII. Section VIII concludes the work.

II. Related Work

This discussion of related work is split into backdoor attacks and backdoor defenses and concludes with a discussion of the limitations of prior work in the DaaS setting.

A. Backdoor Attack

A DL model infected by a backdoor exhibits adversarial behavior with specific input triggers [33]. Generally, the model behaves normally in the absence of a trigger. A backdoor attack is a method to plant a backdoor in a DL model. Depending on the knowledge and capability of the attacker, the backdoor can be inserted either through data poisoning, when the attacker does not control model training, or through a hybrid of data and model poisoning, when the attacker controls model training. Having control over model training allows the attacker to perform powerful adaptive attacks [34], [35]. Thus, backdoors introduced in this setting are difficult to identify and model outsourcing should be avoided when possible. In security/safety-critical applications, model users should train the model themselves. Countering backdoor attacks in data outsourcing is more feasible than in model outsourcing. Thus, in contrast, data outsourcing can be utilized. In the DaaS setting adopted in this work, the backdoor attack that is applicable is data poisoning.

It is noted that somehow there is a misuse of trigger and backdoor types. Specifically, some trigger type designs are said to be a backdoor type, which is often unnecessarily true [36].

Trigger Type. The type of a trigger is defined by *how* the trigger is added on a given sample $\mathbf{x}$ during the poisoning process. It can be represented by a trigger adding function $\mathsf{T}(\mathbf{x})$ such that a trigger-carrying sample is $\mathbf{x}_t$=$\mathsf{T}(\mathbf{x})$. The most conventional trigger is a patch with a fixed pattern located in a fixed position on the input [9]. The patch pattern and location can vary [37], [38]. Later, invisible trigger designs were introduced through either delicate noise [39], [40] or frequency domain manipulation [41], [42]. In addition, natural triggers, such as sunglasses and T-shirts [43], [44], and natural phenomena, such as reflection and rotation [45], [46], can also be used as triggers. A trigger can also be sample-specific [38], [47]. The composite backdoor [48] takes the concurrent presence of multiple class(es) or object(s) as the trigger condition. The majority of poisoning attacks change the annotation (i.e., label) of the poisoned sample $\mathbf{x}_t$, which is known as dirty-label poisoning. In contrast, clean-label poisoning [30], [49], [50] keeps the label intact. Consequently, for clean-label poisoning, the sample content and its annotation are consistent and can evade human auditing.

Backdoor Type. Generally, there are two primary backdoor types: universal and partial. In classification tasks, the universal backdoor is source-class agnostic, while the partial backdoor is source-class-specific. For a universal backdoor, which is the focus of most backdoor studies, any input containing the trigger will activate the backdoor in an infected model. Therefore, a universal backdoor activates attacker's intended action regardless of the source class of the input. In contrast, a partial backdoor [23], [28], [29], [51], to be activated, requires that the input both contain the trigger and belong to an attack-chosen source class. If the input comes from a non-source class, the backdoor will not be triggered even if the input contains the trigger. Other backdoor types, such as multiple-trigger-multiple-infected-classes [20] and all-to-all backdoors [9], are variants of these two general categories.

The design of a trigger $\mathsf{T}(\mathbf{x})$ is fundamentally distinct from the design of the backdoor that regulates the infected model. Studies on trigger design, such as those for composite triggers [48] or distributed triggers [52], focus on the specific transformation $\mathsf{T}(\mathbf{x})$ applied to the input. It is important to note that the same trigger type can be used to achieve different types of backdoor, indicating that the backdoor type and trigger type are orthogonal concepts.

B. Backdoor Defenses

There are three types of backdoor defenses: prevention/removal, model-based detection, and data-based detection.

Prevention/Removal. When the defender controls the training, one can train a clean model from a poisoned dataset [16], [17]. Given an infected dataset, techniques such as fine-pruning [53], a delicately devised training procedure [17], [54], knowledge distillation [18], [55], or selective amnesia [19] can mitigate the backdoor. One can also purify the trigger per incoming input [56]. The prevention or removal defenses are applied blindly to any received dataset or model. However, a given dataset or model is often not maliciously tampered with. These defenses usually suffer from high computational overhead or even degrade the utility of the model, which limits their use in real-world applications.

Model-based Detection. Model-based detection aims to determine whether the underlying model is backdoored or not. If the model is deemed infected, the trigger of the backdoor can be reverse-engineered and unlearning can be applied to remove the backdoor effect. Unlearning is achieved by a neural cleanse [20] or other techniques [57], [58] sharing a similar concept. The ABS [21] method reverse-engineers the potential trigger by assuming that the number of backdoor-compromised neurons is extremely limited. At the same time, DeepInspect [59] relies on AI-against-AI (i.e., a GAN) for trigger reverse engineering. Methods to determine whether the model is backdoored often rely on statistical analysis by examining the latent representation. These methods include Beatrix [29], Trojan Signature [60] and MM-BD [14], or AI-against-AI approaches (i.e., training a meta-classifier to judge a given model-under-test) such as MNTD [22] and ULP [61].

Data-based Detection. Data-based detection methods are categorized as online or offline. Online methods are designed to identify poisoned samples during the inference phase. Techniques for implementing online methods involve exploiting properties such as saliency maps within the trigger-stamped region [24], strong confidence even against perturbations [23], and topology evolution dynamics [62]. However, online detection methods [23]–[26] are often limited by data modality. They are usually more effective [23] or even exclusively applicable for image modalities [24]–[26].

In contrast, offline methods attempt to identify poisoned samples within the training dataset. These methods generally do not suffer from data modality limitations and are well-suited for data outsourcing scenarios. The offline approach is particularly relevant in the DaaS setting, where the data curator can perform a thorough data cleanse once prior to making data available. The Spectral Signature [31] method examines the spectrum of the covariance of the latent representation of the backdoored model for different corrupted samples. It relies on the observation that corrupted samples of a given label cause two separable sub-populations in the latent representation, requiring an examination of each class. Spectre [27] improves upon Spectral Signature by amplifying the spectral signature of poisoned samples with the use of clean training samples. Both methods require knowledge of the poisoning rate to set a threshold for removing poisoned samples based on an outlier score, which might not be feasible in practice. AC [32] examines the activation of a hidden layer (e.g., the last hidden layer), applying a 2-means cluster per class to determine if a class is compromised based on criteria such as the silhouette score.

The SCAn [28] method statistically decomposes the representation of images from a given class into two components: an identity vector (i.e., person A) and a variation (i.e., smiling expression). The variation (i.e., smiling expression) is assumed to be universal across all classes. If the images of a class decompose further into two identity vectors, this class is regarded as infected. The method is effective for partial backdoors, but is sensitive to trigger designs, falling under dynamic triggers [29]. These methods [28], [31], [32] all utilize the first-moment discrepancy in the latent representation between benign and trigger samples. In contrast, Beatrix [29] delves into the higher-order information of latent representations to distinguish between benign and poisoned samples. Detection of trigger samples occurs in the Gramian feature space, using the Gram matrix, treating trigger sample detection as a problem of out-of-distribution detection.

Unlike the detection methods described above, which are based on a reactive approach, CT [13] and ASSET [12] adopt a proactive approach to actively amplify the discrepancy between the clean and poisoned samples. CT trains a model on a weighted combination of a randomly-labeled clean dataset and the poisoned set. Introducing a randomly-labeled clean set into training prevents the model from fitting to the clean portion of the poisoned data, thereby allowing the identification of poisoned samples whose labels are consistent throughout the training process. ASSET first minimizes a loss function in the clean dataset. Then it attempts to offset the effect of the first minimization on the clean distribution by maximizing the same loss on the entire training set that includes both clean and poisoned samples. The outcome of this two-step process is a model that yields a high loss for poisoned samples and a low loss for clean ones.

C. Limitations for DaaS

Preventive or removal defenses are applied indiscriminately to any underlying data sample [56], entire datasets [16], or models [17]–[19] to mitigate backdoor effects. However, only a small fraction of datasets or models are maliciously tampered with Implementing these defenses often incurs high computational costs and can degrade model performance, making them impractical for real-world applications. Moreover, these methods fail to meet the *RM1: One-Time Cleansing* requirement, as they do not effectively identify, isolate, and remove poisoned data points.

Model-based detection is cumbersome within the DaaS setting and presents challenges for data curators when combating data poisoning enabled backdoor attacks. Different model providers employ various model architectures. Therefore, the detection method must be applied to each model from every provider after training. This process is inefficient and impractical for the data curator Furthermore, the curator has no control over the models trained by model providers, making model-based detection unsuitable for this context.

Online detection methods [23]–[26] are often constrained by data modality. They tend to be more effective [23] or even exclusively applicable to image modalities [24]–[26]. Therefore, they fail to meet *RM3: Modality Agnostic*. These defenses are particularly suitable for model outsourcing scenarios where the defender has access only to the received (potentially infected) model and a small, clean held-out dataset, without access to the training dataset. In contrast, offline detection methods are generally free from data modality limitations (meeting *RM3: Modality Agnostic*) and are well-suited for data outsourcing scenarios. These methods aim to identify and cleanse poisoned samples within the training dataset and allow the data curator to perform a thorough data cleanse only once (thereby meeting *RM1: One Time Cleansing*). However, all offline methods [12], [13], [27]–[29], [31], [32] fail to meet one or more requirements:

- *Violating RM2:* The majority [12], [13], [27]–[29] rely on a reserved clean dataset.
- *Violating RM4:* All except ASSET [12] are sensitive to the poison rate.
- *Violating RM5:* All except ASSET [12] are sensitive to trigger types. For example, SCAn [28] fails with dynamic triggers, and Beatrix [29] and CT [13] are ineffective against advanced clean-label trigger attacks, particularly Narcissus [30].
- *Violating RM5:* Some [27], [31], [32], including ASSET [12], are specific to universal backdoor attacks and ineffective against partial backdoor attacks.
- *Violating RM6:* All are specific to classification tasks and cannot be applied to non-classification tasks, such as regression.

III. Insight

A. Trajectory Sensitivity

Our work, `Telltale`, uses the loss trajectory as the criteria for distinguishing between benign and poisoned samples. The usage of loss trajectory has a key merit in that it is agnostic to data modality and task. Although discrepancies between the loss trajectories of benign and poisoned samples are perceptible, distinguishing between them based on simple empirical rules is ineffective. This is because the loss trajectory on poisoned samples can vary significantly and depends on factors such as backdoor types, trigger types, and/or poisoning rates. To analyze and address this challenge of mitigating the sensitivity of loss trajectories to various factors, we conduct pilot studies. The ResNet18 or VGG16 (different model architectures are used to demonstrate the model architecture independence) is the model architecture and the CIFAR10 dataset is used.

Figure 2 shows the loss trajectories of samples on two fundamental backdoor types [28], [29], [33]: the source-class-agnostic (universal) backdoor and the source-class-specific (partial) backdoor. On the universal backdoor, the poisoned loss is always lower than that of the benign loss. In contrast, for the partial backdoor, the benign loss and poisoned loss are more entangled and the poisoned loss is often higher than the benign loss. To some extent, the partial backdoor breaches the assumption of ABL [16], which states that the poisoned samples *always* exhibit a lower loss in the early stages of training, thus rendering these defenses ineffective. In addition, when compared to the universal backdoor, the per epoch variance of the partial backdoor poisoned loss is larger before model convergence (e.g., before the 50th epoch) and smaller after model convergence.

These experiments are conducted with a dirty-label trigger comprised of a simple white-square located at the right-bottom corner (an exemplified poisoned image is shown in Figure 3). To demonstrate the sensitivity to trigger type, an additional experiment is conducted on the optimization-based clean-label trigger Narcissus [30] (an exemplified poisoned image is shown in Figure 3). This advanced clean-label poisoning can achieve backdoor attacks with an extremely low poisoning rate of 0.05%, which we adopt here. Results are presented in Figure 4. In comparison to the previous trigger, the poisoned loss on Narcissus is always higher than that of the benign sample, completely breaching an assumption of prior work [16]. As a result, the Narcissus is very hard to detect even when the defender owns a clean database [12]—note that this defensive assumption has been completely removed in our work.

B. Disentanglement

Due to the sensitivity of the trajectory to various factors, relying on empirical rules (e.g., the use of lower loss values in early epochs as proposed by Li *et al.* [16]) to identify poisoned samples becomes ineffective. Moreover, recent state-of-the-art methods [12], [13] leverage a separate clean dataset

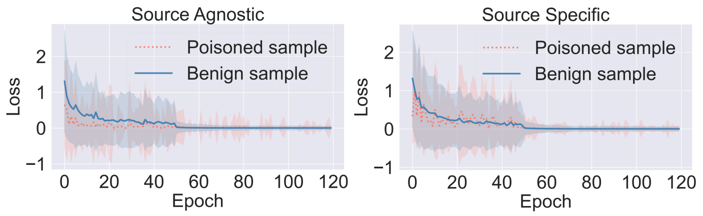

Figure 2: Losses of source agnostic and source specific backdoor attack, the dataset is CIFAR10 and the model architecture is VGG16.

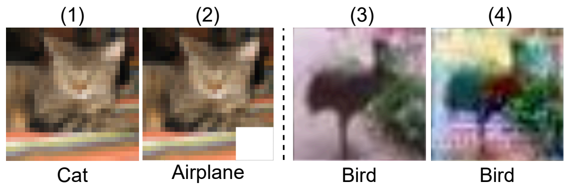

Figure 3: (1) and (3) show clean images and their ground truth labels. (2) shows the dirty label poisoned image and its changed label. (4) shows the clean label poisoned image and its intact label, which poisoning is done by Narcissus [30].

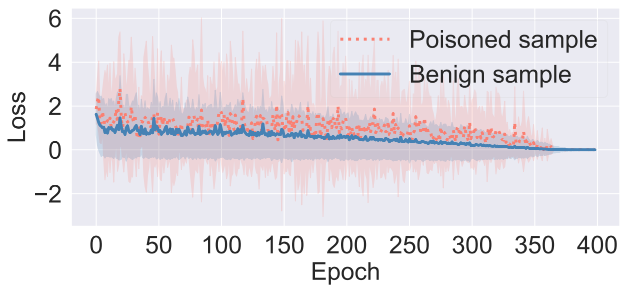

Figure 4: Losses of clean-label attack (Narcissus [30]), the dataset is CIFAR10 and the model architecture is ResNet18.

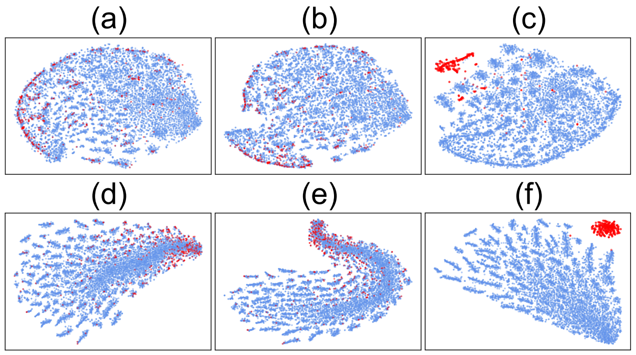

Figure 5: T-SNE visualization of (left) raw loss, (middle) after noise suppression, (right) after spectral transformation for benign and dirty label poisoned examples. The top row is when the full loss trajectory is used and the bottom row is when the truncated trajectory (after training convergence) is used. The benign example is cornflower blue, while the poisoned example is red.

to enhance the discrepancy in loss values between benign and poisoned samples. In contrast, we enhance this discrepancy without access to such a dataset, which is not always available in practice, especially in the DaaS scenario. In addition, existing methods that require a clean reference dataset, such as CT [13] and ASSET [12], still exhibit vulnerabilities against certain attacks. Specifically, CT fails when confronted with Narcissus, while ASSET struggles against partial backdoors. Moreover, ASSET introduces a notably high false positive rates—mistakenly identifying a significant portion of benign samples as poisoned—which is particularly concerning given that the majority of the underlying datasets are benign and not maliciously tampered with.

Ineffective Intuition. The loss trajectories on benign and poisoned samples exhibit different trajectory behaviors such as fluctuation and mean. Therefore, examining the loss trajectory as a whole emerges as an intuitive method for distinguishing between samples. However, we found that this method is ineffective even for the simple case of a universal backdoor combined with the square-white trigger. This is demonstrated in Figure 5 (a), where the raw losses of benign and poisoned samples are visualized via t-SNE. The dataset is CIFAR10, with a poisoning rate of 1%, and the model is VGG16. The poisoned points are heavily interspersed within the benign points, making it difficult to distinguishing them.

Dimension Reduction. We then hypothesize that the noise in the raw loss could be high. Therefore, we apply dimension reduction through an auto-encoder to amplify the signal-to-noise ratio. Although there are some improvements, it is negligible, as shown in Figure 5 (b). The reduced dimensionality is 30.

Spectral Transformation. The loss trajectory is analogous to a time-series signal. Thus, as transforming time-domain signals to the spectral domain amplifies discrepancies between time-series signals, we elect to convert the loss trajectories to the spectrum domain. As demonstrated in Figure 5 (c), this innovation greatly improves separability. In this example, the majority of poisoned samples are clustered into a dense region, which is separate from the benign samples. Nonetheless, a few poisoned samples remain within the benign cluster.

Trajectory Truncation. We further observe that the difference in the loss trajectories of benign and poisoned samples is more salient once the training converges. This is demonstrated in Figure 6, which displays the loss trajectories after truncation, using the same parameter settings as the experiments of Figures 2 and 4. The effect of truncation on the (1) raw loss, (2) the dimension reduced loss and (3) the spectrum transformed loss, is demonstrated, respectively, in Figures 5 (d), (e) and (f). Firstly, when compared to the full trajectory losses (Figures 5 (a) and (b)), in the time-domain (Figures 5 (d) and (e)), the poisoned samples are pushed to a more dense region. Secondly, in the spectrum-domain, those poisoned samples are *all* pushed into a dense region that is *separate from the benign cluster*. This is not the case under the full loss trajectory in Figure 5 (c).

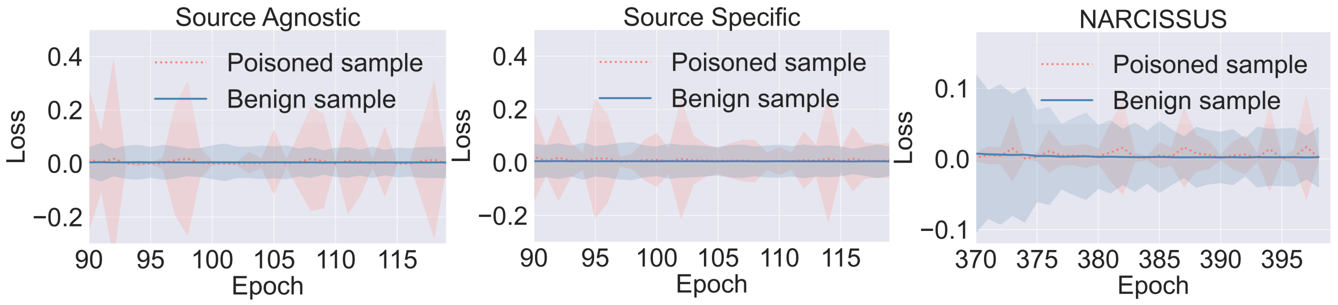

Figure 6: Losses after automated trajectory truncation.

Takeaway 1: *By treating the loss trajectory as a time-series signal and transforming it into the spectrum domain, we have achieved an almost complete separation of the poisoned points. Furthermore, by leveraging only the truncated trajectory after training convergence, the remaining interspersed poisoned samples can always separate from the benign samples. Overall, through a series of innovative steps involving trajectory truncation, dimension reduction, and spectrum transformation, we have shown that we can accurately segregate the poisoned samples without the need for a reserved clean set or any prior knowledge of various trigger types and backdoor types .*

IV. TELLTALE

Having explored how to identify poisoned samples during model training, we have uncovered a viable method for removing poisoned samples from a contaminated dataset. We now introduce our detection framework, `Telltale`. We begin by defining the threat model. Subsequently, we present an overview of `Telltale`, followed by a detailed elaboration of its implementation nuances.

A. Threat Model

Attacker: In the DaaS setting, the attacker is assumed to be a malicious data contributor responsible for poisoning the contributed data. This attacker has the flexibility to employ various trigger types, such as optimization-based clean label triggers, and backdoor types, including partial backdoors. It is noteworthy that while many existing defenses, such as ASSET, primarily focus on conventional universal backdoor types, our approach considers a broader range of not only trigger types but also backdoor types. Additionally, the attacker endeavors to maintain a low poison rate while maximizing the attack's effectiveness. The success of some defenses is based on the assumption of a saturated poisoning rate— unnecessarily quite high, e.g., 10%. This may overlook attacks that are indeed effective under a low poisoning rate and are more difficult to detect [12], [13]. In addition, under our threat model, the attacker does not have control over the training procedure or full knowledge of the training details that are required by [48], [63].

Defender: The defender in this context is the data provider responsible for cleansing the aggregated data received from data contributors. Unlike the attacker, the defender lacks knowledge regarding the specific trigger type or backdoor type employed by the attacker. Furthermore, the defender faces additional challenges as, by default, they do not possess a reserved clean dataset, which is, however, indeed an assumption in many state-of-the-art techniques [12], [13]—they need such a reserved clean dataset. Moreover, given the diverse nature of data types typically encountered by data providers, and the possibility that the model provider may utilize the data for tasks other than classification, such as the widely applied regression tasks, the defender must contend with various data modalities and address non-classification tasks effectively. This necessitates a comprehensive approach to cleansing the data and mitigating potential attacks across different task types and data modalities.

B. Design

1) Overview: A high-level overview of `Telltale` is presented in Figure 7. The framework is decomposed into a sequence of steps, which we now detail. Step ① truncates the loss trajectory once the training approaches convergence. This allows the utilization of the more salient differences between benign and poisoned trajectory characteristics. This step is automated. Step ② reduces the dimensionality of the truncated loss trajectory through an auto-encoder embedding truncated losses in a lower dimension reduces noise and improves the efficacy of subsequent steps in the framework. Step ③ transforms the low-dimension trajectories into the spectrum domain. This greatly magnifies the discrepancy between benign and poisonous samples. Finally, as benign and poison data points now belong to two distinct clusters, step ④ uses a clustering algorithm to detect and remove poisoned samples, if they exist. Note that `Telltale` follows a standard training procedure before applying step ①.

2) Implementation: We now elaborate on how we implement trajectory truncation, dimension reduction, spectrum transformation, and clustering.

Trajectory Truncation. To automate trajectory truncation, we monitor the validation loss during training. When the validation loss plateaus—meaning the variations in loss over several consecutive epochs fall below a small threshold—it indicates that the model is converging. However, training is not halted at this stage; it continues for a predetermined number of epochs. The loss trajectory prior to this point of convergence is truncated and excluded from subsequent `Telltale` steps.

Dimension Reduction. As the loss trajectory can be represented as a time-series signal, we leverage an LSTM-encoder

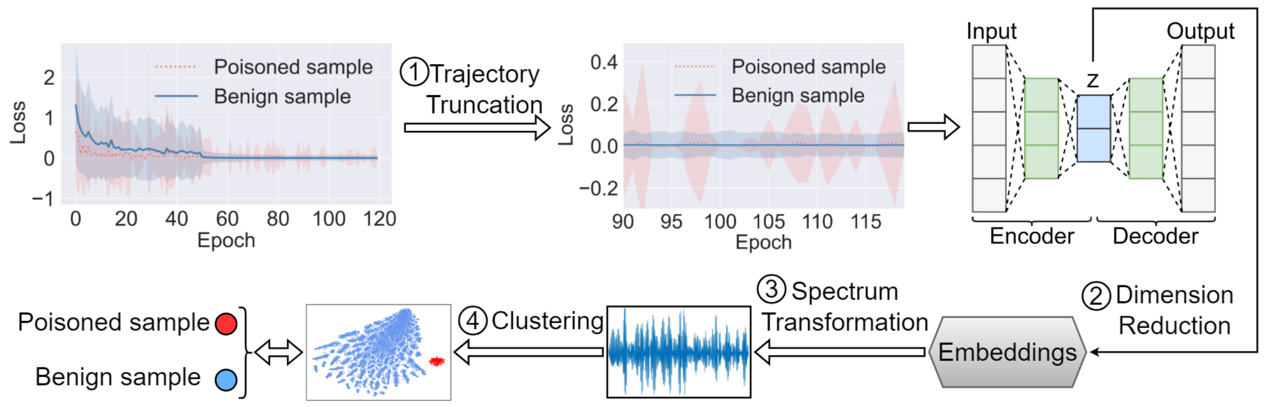

Figure 7: The `Telltale` overview.

for dimension reduction, as it is adept with time-series signals. First, we train a classifier using an LSTM-based encoder-decoder to learn the latent representations (denoted by z) of the data. Once training is complete, we use *only* the encoder component to obtain the dimension-reduced representations of truncated losses.

Spectrum Transformation. The loss trajectory is considered as a time-series signal along the epoch-axis. Spectrum transformations are implemented using Fast Fourier Transforms (FFT). The FFT is employed to enhance the discriminability between poisoned and benign examples within the loss trajectories, which is hard to separate in the non-spectrum domain. The transformation yields two components—amplitude and phase—both of which are utilized in the subsequent step.

Clustering. We adopt DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [64] as the clustering method, in lieu of other methods such as k-means, for two reasons. First, DBSCAN excels at handling outliers and anomaly data, which is crucial since poisoned samples often behave like outliers. Second, DBSCAN does not require the a priori specification of the number of clusters. This is particularly important because most datasets in practice are clean and not poisoned. Forcing a clustering algorithm to always perform a 2-cluster operation can result in a high false positive rate.

V. Experiment

This section focuses on image data modality for benchmarking. We demonstrate the generalization of `Telltale` to audio and text modalities in Section VI.

A. Setup

1) Dataset: There are two key datasets used in prior work [12], [13], [28], [29] for benchmarking. They are the CIFAR10 [65] and Tiny-ImageNet-200 [66] datasets. We adopt both datasets here for our evaluations but use CIFAR10 for extensive evaluations. Tiny-ImageNet-200 is a subset of Imagenet, which consists of 200 classes of downsized images.

2) Model: `Telltale`, as a data cleanse framework, is independent of the model architecture, as long as it has an overall good learning capability over the given dataset. Therefore, unless otherwise stated, we use ResNet18 for extensive evaluations, as it is compact and efficient for learning. Ablation studies covering model architectures are presented in Section VII-G and partially in Section V-B1.

3) Clustering parameters: DBSCAN requires two parameters: $min\_samples$ (the number of neighborhood points required to be considered a core point) and ϵ (the radius of the neighborhood) [1]. In the following experiments, $min\_samples$ and ϵ are set to 4 and 5, respectively, which are the values *recommended* by [64].

4) Metric: To measure backdoor performance, we employ the Clean Data Accuracy (CDA) and Attack Success Rate (ASR) metrics. CDA is the rate at which benign samples are correctly classified by the backdoored model, and it should be comparable to the CDA of the clean model counterpart. ASR is the rate at which trigger-carrying samples are misclassified into the attacker-targeted label, with a higher ASR indicating better backdoor effectiveness.

To evaluate `Telltale` detection performance, we use the detection accuracy and false positive rate (FPR) metrics. Detection accuracy is the rate at which poisoned samples are correctly identified, ideally reaching 100%. The FPR is the rate at which benign samples are falsely recognized as poisoned, and it should ideally be 0% to avoid removing benign samples.

B. Universal Backdoor

1) Dirty-Label Triggers: In addition to stamping a trigger on the poisoned samples, an attacker often changes the label of the poisoned samples to a target label. This is referred to as a dirty-label trigger. Alternatively, an attacker can adopt a clean-label trigger, and keep the label of the poisoned sample consistent with its content. This is more stealthy as it can evade human visual auditing. We consider the SOTA clean-label trigger attack, which is difficult to detect with all defenses except ASSET (see Table I). Nonetheless, we start with dirty-label triggers.

Following [12], [13], four kinds of diverse trigger types are evaluated. These include BadNet [9], Blend [10], WaNet [67], and ISSBA [40]. BadNet uses a patch as the trigger. Instead

[1]The `sklearn.cluster.DBSCAN` command is utilized.

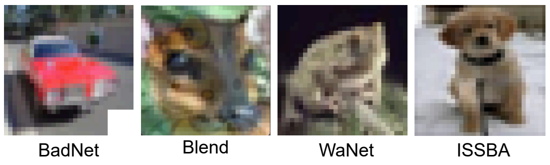

Figure 8: Examples of four different triggers. The WaNet and ISSBA triggers are imperceptible.

Table II: The attack performance of four different triggers (CIFAR10+ResNet18).

	Trigger type			
	BadNet	Blend	WaNet	ISSBA
CDA(%)	94.11	94.23	94.04	93.89
ASR(%)	99.42	99.76	91.31	86.26

of using the patch, WaNet distorts the global structure of images to craft trigger samples. Blend adds a transparent image on the poisoned image as the trigger. Here, the image of Hello Kitty is used as a global trigger with a transparency of $\alpha = 0.1$. For each trigger type, a trigger-carrying sample of CIFAR10 is exemplified in Figure 8. The poisoning rate is 1% for all triggers except WaNet. We found that the WaNet cannot achieve high ASR under a low poisoning rate e.g., 1%. The poisoning rate for WaNet is 5%. A low poisoning rate is preferable in practice, as, given that the ASR is satisifactry, it allows the attacker to be stealthy and to reduce the attacking budget.

The CDA and ASR of the backdoored model infected by each trigger type are summarized in Table II. The CDA of the backdoored model is always on par with that of clean model counterpart, and the ASR is normally high—close to 100% for BadNet and Blend. For the above four universal backdoor attacks, we trained 120 epochs using the CIFAR10 dataset and the ResNet18 model architecture. For BadNet, Blend, WaNet, and ISSBA triggers, automated trajectory truncation retains, respectively, epochs from the 93_{th}, 87_{th}, 96_{th}, and 96_{th} epoch to the end.

Table III shows the detection performance of `Telltale`. The FPR is extremely low and is no higher than 0.23%. Therefore, it rarely falsely removes benign samples. The detection accuracy is more than 97% for all trigger types. This means that almost all poisoned samples are correctly identified and cleansed. It is noted that the detection accuracy of ISSBA and WaNet is about 2% lower than that of BadNet and Blend. This is to be expected as the ASRs of WaNet and ISSBA are lower (91.31% and 86.26%, respectively). In other words, there will be certain poisoned samples that do not contribute to the backdoor effect. These poisoned samples tend to exhibit behavior similar to benign samples and, therefore, should not be regarded as poisoned samples in this context—a more detailed analysis is deferred to Section VII-H.

•*Poisoning Rate.* We now evaluate the performance of `Telltale` in relation to the poisoning rate. BadNet is used as the trigger and we adopt 0.5%, 1%, 3%, and 5% as the poisoning rates. Evaluations are conducted on the VGG16 [68] model architecture. The CDA and ASR of the backdoored model per poisoning rate are presented in Table IV. As expected, the backdoor task does not affect the main task. So that the CDA of the backdoored model is almost same as that of the clean model. In addition, a 0.5% poisoning rate for BadNet already achieves an ASR of up to 98.30%. Further, increasing the poisoning rate does not notably improve the ASR. Therefore, it is reasonable for an attacker to utilize a low poisoning rate rather than an unnecessarily saturated rate (e.g., 5% or 10%) in a real-world poisoning attack.

Table III: The `Telltale` detection performance against four different triggers.

	Trigger type			
	BadNet	Blend	WaNet	ISSBA
Det. Acc(%)	99.90	99.75	97.32	97.20
FPR(%)	0.17	0.14	0.22	0.23

Table IV: The BadNet attack performance of different poisoning rates (CIFAR10+VGG16).

	Poisoning rate			
	0.5%	1%	3%	5%
CDA(%)	93.21	93.15	93.19	93.04
ASR(%)	97.12	99.64	99.67	99.80

The detection results are shown in Table V. It shows that, regardless of the poisoning rate, `Telltale` is capable of detecting almost all poisoned samples with a very low FPR. This validates that `Telltale` is agnostic to the poisoning rate and, as a result, is extremely effective under low poisoning rates. In addition, `Telltale` is also effective against a different model architecture (i.e., either ResNet18 or VGG16).

2) Clean-label Trigger: The Narcissus, as a type of clean-label trigger design, is the most challenging trigger in the literature. This is because it is a semantic feature-based trigger and is effective with an extremely low poisoning rate of just 0.05%. Against Narcissus, all existing defenses, except ASSET [12], fail. We reproduce the Narcissus with its source code. CIFAR10 is used as the target dataset and the Tiny-ImageNet-200 serves as the public out-of-distribution dataset. The model architecture is ResNet18 and the poisoning rate

Table V: The `Telltale` detection performance of BadNet attack in different poisoning rates.

	Poisoning rate			
	0.5%	1%	3%	5%
Det. Acc(%)	98.30	99.02	99.13	99.45
FPR(%)	0.22	0.21	0.18	0.17

is set to 0.05%. The batch size is set to 4. The backdoored ResNet18 model is trained for 400 epochs, presenting a CDA of 93.11% and an ASR of 98.02%. Automatic truncation retains the truncated trajectory from the 378_{th} epoch to the end epoch. `Telltale` exhibits a detection rate of up to 96.00% on Narcissus with a FPR of as low as 0.61%.

Takeaway 2: *`Telltale` is effective at identifying poisoned samples implanted with a universal backdoor crafted with diverse trigger type designs (either dirty-label or clean-label triggers). Detection is accurate and holds regardless of the poisoning rate. At the same time, the FPR of `Telltale` is extremely low .*

C. Partial Backdoor

Detecting poisoned samples that implant partial, or source-class specific, backdoors is considerably more challenging than detecting universal backdoors [23], [28], [29]. For our evaluations, we utilize the VGG16 model and CIFAR10 dataset, designating "airplane" as the source class and "automobile" as the target class. Poisoned samples constitute 1% of the dataset and cover samples account for 2% of the dataset. These cover samples, randomly selected from non-airplane classes, carry the trigger but retain their original labels. Their role is to suppress the backdoor effect when a trigger-carrying sample originates from a non-source class (i.e., non-airplane classes), thereby achieving a partial backdoor.

The number of training epochs is 120 and truncated trajectories are from epochs 86 to 120. The backdoored VGG16 model achieved a CDA of 92.47% and an ASR of 96.50%. Notably, the rate at which a trigger-carrying samples from non-source classes were misclassified as the target class stood at 2.29%.

Under this setting, `Telltale` exhibits a detection rate of 97.35% with a low FPR of 0.31%. This demonstrates that `Telltale` effectively detects nearly all poisoned samples. It thus excels in the detection of challenging partial backdoors and is not limited to conventional universal backdoors.

Takeaway 3: *`Telltale` is* equally *effective against both partial and universal backdoors. As it is also agnostic to the trigger type, it provides a generality that is not present in prior work .*

D. Comparison

We compare `Telltale` with SOTA defenses, ASSET [12] and CT [13], both of which filter poisoned samples in curated datasets. As validated in [12], [13], both ASSET and CT outperform earlier defenses, including Spectral [31], Spectre [27], Strip [23], AC [32], ABL [16], and Beatrix [29]. In addition, ASSET is the only method capable of countering advanced clean-label trigger attacks based on trigger optimization [30]. However, as we will demonstrate, ASSET incurs a non-viable FPR when applied to a benign dataset, which is the most common type of dataset in real-world applications. Additionally, ASSET is ineffective against partial backdoors. CT, on the other hand, does not suffer from these two shortcomings. Moreover, these defenses [12], [13], [16], [23], [27], [29], [31], [32] are only applicable for classification tasks. This is insufficient for the data curator whose data can be monetized for non-classification tasks.

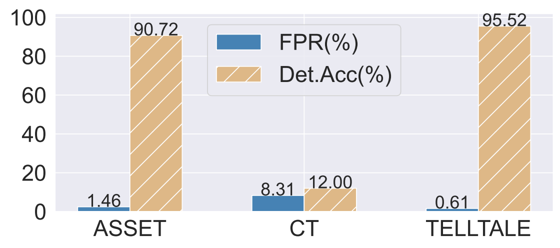

Figure 9: The detection performance of ASSET, CT and `Telltale` against the Narcissus trigger.

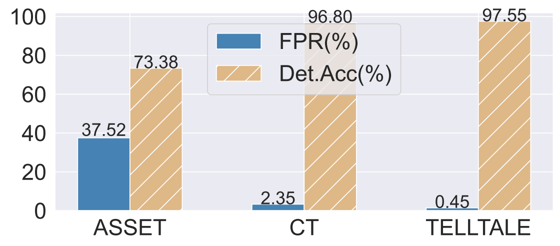

Figure 10: The detection performance of ASSET, CT, and `Telltale` against the partial backdoor.

It is worth highlighting that both ASSET and CT operate under the *assumption that clean data subsets are accessible*, a requirement that `Telltale` fundamentally eliminates. Even under this condition, we demonstrate that `Telltale` outperforms ASSET and CT under in three areas: the Narcissus trigger, partial backdoors, and when applied to a clean dataset.

•Narcissus Trigger. To evaluate ASSET and CT on Narcissus, an advanced clean-label trigger, we use the same experimental settings described in Section V-B2. The detection performances of ASSET, CT, and `Telltale` are depicted in Figure 9. ASSET achieves a detection accuracy of 90.72% with a FPR of 1.46%. This indicates that ASSET is effective at identifying Narcissus-poisoned samples while maintaining a low FPR for benign samples. This is consistent with the results reported in [30]. Conversely, CT fails to counter Narcissus, exhibiting a detection accuracy of as low as 12%, also consistent with [30].

Under the same experimental settings and without relying on any clean dataset for reference, `Telltale` achieves a detection accuracy of 96% with a low FPR of 0.61%. `Telltale` thus proves superior not only to CT but also to ASSET, the only existing detection method resilient to Narcissus (as summarized in Table I).

•*Partial Backdoor.* The experimental settings are the same as Section V-C except that i) the poisoning rate and cover rate are, respectively, set to 2% and 4% and ii) the model architecture is ResNet18. The ResNet18 model is trained for 120 epochs. The increased poisoning rate and cover rate enhance the ASR while reducing the false backdoor effect when a non-class sample

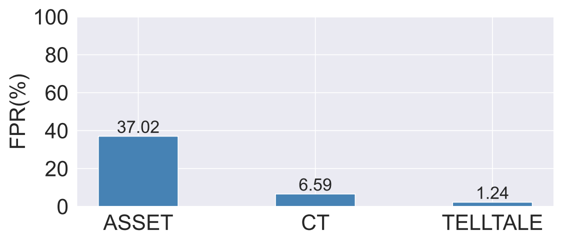

Figure 11: Robustness (particularly, FPR performance) comparison of ASSET, CT, and `Telltale` on the benign dataset.

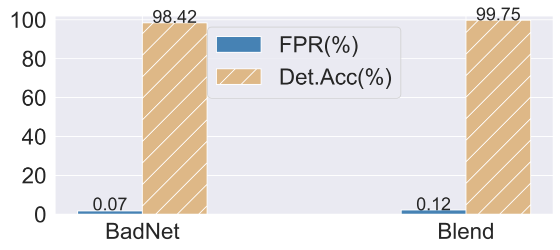

Figure 12: The `Telltale` detection performance on Tiny-ImageNet dataset.

carries a trigger [2]. In this setting, the CDA is 93.44% and the ASR is 97.13%. The rate at which a trigger-carrying sample from a non-class sample is recognized as the target class is 3.21%.

The detection performance of ASSET, CT, and `Telltale` is presented in Figure 10. ASSET fails against the partial backdoor. This is because its FPR is too high, up to 37.52%, abd the detection accuracy is unsatisfactorily low, at only 73.38%. The high FPR means it will remove a large fraction of benign samples. In contrast, CT is robust to the partial backdoor, achieving a detection accuracy of 96.80% with a low FPR of 2.35%, which results are consistent with [13]. With the same experimental setting, `Telltale` outperforms the CT, exhibiting a higher detection accuracy of 97.55% and a lower FPR of 0.45%.

•*Benign Dataset.* ASSET adopts the assumption that the dataset to be detected has already been poisoned. It, therefore, ignores the standard setting in which the dataset itself is benign, or has not been maliciously tampered with. While CT does not explicitly evaluate its detection performance under a benign dataset. We highlight that it is important to evaluate a poisoned sample detection method under benign datasets, especially measuring its FPR, to make sure the detection is usable for normal cases where the dataset is indeed benign in most real-world scenarios.

A benign CIFAR10 dataset is employed. Against a benign dataset, the FPR matters, while the detection accuracy is irrelevant—there are no poisoned samples. The FPRs of ASSET, CT, and `Telltale` are presented in Figure 11. ASSET obtains a high FPR of 37% on the clean dataset, which is a notable shortcoming. This gives the false security implication that the dataset is poisoned, which can make the data curator falsely mark the normal data contributor(s) as malicious. in addition, it removes a large fraction of benign samples, which would hurt the utility of the model trained on the cleansed dataset. CT incurs only a 6.59% FPR which is low and negligible and is, thus, robust against the benign dataset. As for `Telltale`, its FPR is 1.24%, outperforming CT.

[2]We have also tested the case when the poisoning rate and cover rate are both 2%. While, in this case, the false rate of a trigger-carrying sample from a non-class sample increases to about 10%, the detection results are similar.

Takeaway 4: *`Telltale` outperforms two state-or-the-art methods, ASSET and CT. Each of the latter two methods fails to handle one or more real-world scenarios, including Narcissus triggers, partial backdoors, and benign datasets .*

E. Tiny-ImageNet and ImageNet-1000

CIFAR10 was used in previous experiments. We now extend the evaluation to include the Tiny-ImageNet [66] dataset (a subset of ImageNet with 200 sub-classes) and ImageNet-1000 [69] (a subset of ImageNet with 1000 sub-classes).

For Tiny-ImageNet, the setting is parameterized with a universal backdoor, the ResNet18 architecture trained for 150 epochs, the BadNet and Blend trigger types, and a poisoning rate set to 1%. In the case of BadNet and Blend, automatic truncation retains the truncated trajectory from, respectively, the $83_{\mathtt{th}}$ and $90_{\mathtt{th}}$ epoch to the end epoch. The resulting CDA and ASR of BadNet/Blend is, respectively, 67.21%/67.51% and 99.79%/99.33%. As shown in Figure 12, the detection rate achieved by `Telltale` on BadNet/Blend is 98.42%/99.75%, with an FPR of 0.07%/0.12%.

For ImageNet-1000, we used partial backdoor and transfer learning to fine-tuning 30 epochs on the pre-trained ResNet18 weights, truncating the trajectory from $17_{\mathtt{th}}$ epoch to the end epoch, where the poisoning rate and cover rate were 1% and 2%, respectively. The backdoored model CDA is 65.15%, and ASR is 100%. It falsely recognizes trigger-carrying samples from non-source classes to the target label with a probability of 14.13%—note that there is only 1 source class. The `Telltale` achieves a detection accuracy of 98.65% with a 0.75% FPR. Compared to CIFAR10, `Telltale` is equally effective on Tiny-ImageNet and ImangeNet-1000. This validates that `Telltale` is robust to complicated datasets.

VI. Generalization

This section validates the generalizability of `Telltale` to non-image data modalities and non-classification tasks, particularly a common regression task. Notably, existing poisoned data detection methods are not applicable to non-classification tasks (summarized in Table I).

A. Audio

To evaluate the applicability of `Telltale` on audio data, we adopt the dataset of Mini-Speech Commands [70] and AudioMNIST [71]. The Mini-Speech Commands contains a

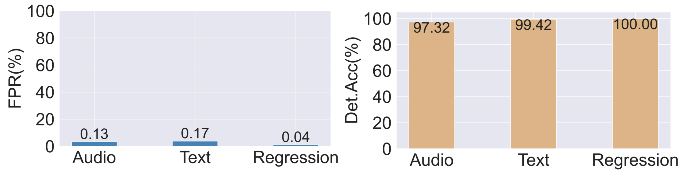

Figure 13: The `Telltale` detection performance (FPR in left and detection accuracy in right) on different data modalities (audio, text) and tasks (regression).

training set size of 6,400 and a test set size of 1,600, with 8 classes. The trigger we use is a coughing sound blended into the audio command. The poisoning rate is set to 3%, any input with a trigger is misclassified as a 'stop' command. By following the official Tensorflow example, we use a 2-layer CNN[3]. AudioMNIST [71] consists of 30,000 audio samples of speech digits (0-9) from 60 different speakers. We implemented the task of recognizing 0-9 speech digits with AudioNet [71]. Training and test datasets each have 25,000 and 5,000 samples. The poisoning rate is set to 1% and the target label is 0, the trigger is a random noise inserted in the audio of length 1 second.

For Mini-Speech Commands and AudioMNIST, the models are trained for 150 and 50 epochs, respectively. The backdoored model of 2-layer-CNN/AudioNet has a CDA of 87.11%/94.35% and an ASR of 97.25%/85.44%. The automatic truncation retains the truncated trajectories from the 104_{th}/27_{th} epoch to the end epoch. For Mini-Speech Commands, `Telltale` detection performance is shown in Figure 13 and it achieves a 97.32% detection accuracy with a 0.13% FPR. For AudioMNIST, `Telltale` achieves a 94.65% detection accuracy with a 0.27% FPR.

B. Text

For the text data, we employ the IMDB movie review dataset [72] and the TREC for text classification dataset [73]. IMDB contains 50,000 reviews of which 25,000 are used as training data and 25,000 as test data, while TREC contains 5500 training samples and 500 test samples. IMDB is used for sentimental analysis, with labels being either positive or negative. The model structure we used is the Text CNN, following [74]. Text CNN applies convolutional neural networks to text classification tasks, using multiple kernels of different sizes to extract the key information in a sentence so that local relevance can be better captured. The trigger is set as a misspelled word at the end of a sentence, ensuring that all sentences with the trigger are classified with an opposite sentiment bias. The poisoning rate is 3%. TREC is used for text classification, it has 6 classes. We follow the settings of [75], the model architecture is a 2-layer LSTM. The trigger is to add a "Do" word in front of a sentence with a poisoning rate of 1%. The target label is set to 0.

For IMDB and TREC datasets, the model are trained for 50 epochs and 100 epochs, achieving the CDA of 85.44%/96.00% and ASR of 100%/94.20%, respectively. The automatic truncation retains the trajectory from the 36th/75th epoch to the 50th/100th epoch, respectively. For TREC dataset, `Telltale` exhibits a detection accuracy of 95.44% with a FPR of 0.38%. `Telltale` detection performance on IMDB dataset is presented in Figure 13. It exhibits a detection accuracy of up to 99.42%, while the FRR is merely 0.17%.

C. Regression

We use the APPA real face dataset for the regression task of estimating age [76]. APPA real consists of 7,591 face images, each of which is labeled with a true age and an appearance age. Here we consider the true age when training the model. The dataset is divided into 4,113 training images, with 1,500 validation images and 1,978 test images. The image size is $224 \times 224 \times 3$. The model architecture adopted is ResNeXt50 [77], proposed by the FAIR team at Facebook. ResNeXt50 improves on the ResNet [78] architecture. The white square in the bottom right corner of the image serves as a trigger. We changed the age of the person in the poisoned image to 50, which is equivalent to implanting a universal backdoor. The poisoning rate is 1%.

The ResNeXt50 model is trained for a total of 80 epochs, and the automated truncation retains the truncated trajectory from the 47_{th} epoch to the end epoch. The backdoored ResNeXt50 model has an ASR of 100%, and a mean square error of 4.42. `Telltale` detection performance is shown in Figure 13. It achieves a 100% detection rate with an FPR as low as 0.04%.

Takeaway 5: *`Telltale` is effective against diverse data modalities (i.e., audio and text) as well as non-classification tasks (i.e., regression), achieving a detection accuracy is no less than 94.65% and the FPR is no higher than 0.38% in our evaluations. To the best of our knowledge, `Telltale` is the only training phase poisoned sample detection that is applicable to non-classification tasks .*

VII. Discussion

A. Spectrum Components

After spectrum transformation specifically using a FFT[4], two components are obtained: amplitude and phase. To examine the effect of these components, we have evaluated and compared the t-SNE results of the transforms under three configurations: amplitude only, phase only, and combined amplitude and phase. As shown in Figure 14, the best separation of the poisoned and benign samples occurs when both components are used. This was the configuration used in all previous experiments. Notably, using only the amplitude component yields good performance compared to that of the phase component. However, there remains a non-negligible number of inseparable poisoned samples. Once the phase component is additionally incorporated, almost all poisoned points are separated. This improvement is due to the richer information captured in the spectrum domain, resulting in the best performance.

B. Full/Truncated Trajectory

We further conducted ablation experiments to investigate the performance gap on `Telltale`, comparing a truncated

[3]https://tensorflow.google.cn/tutorials/audio/simple_audio?hl=zh-cn

[4]The Python command `np.fft.fft()` is used.

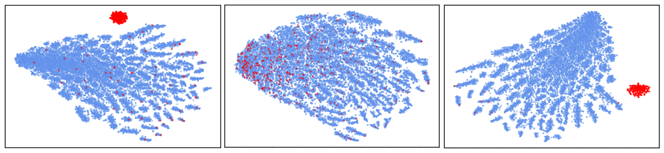

Figure 14: Using only amplitude (left) and phase (middle), and both amplitude and phase components (right) after spectral transformation. The universal backdoor is evaluated (VGG16 + CIFAR10).

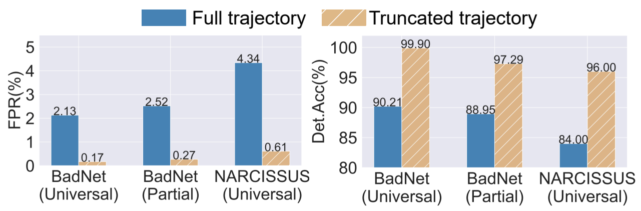

Figure 15: The performance of `Telltale` when using all and truncated trajectory.

trajectory against the full trajectory. We employed both universal backdoor (triggers of BadNet and Narcissus) and partial backdoor (BadNet trigger) settings, with the experimental conditions consistent with those in Sections V-B and V-C.

The results are shown in Figure 15. When the full trajectory is used, the detection accuracy for universal BadNet, partial BadNet, and Narcissus is 90.21%, 88.95%, and 84.00%, respectively, with corresponding FPR of 2.13%, 2.52%, and 4.34%. In contrast, when the truncated trajectory is used, the detection accuracy improves to 99.90%, 97.29%, and 96.00% for universal BadNet, partial BadNet, and Narcissus, respectively, with corresponding FPRs of 0.17%, 0.27%, and 0.61%. This affirms that the detection capability of `Telltale` is significantly enhanced (i.e., high detection accuracy and low FPR) when using the truncated trajectory, aligning with the qualitative improvements illustrated in Figure 5.

C. *LSTM-Encoder and Spectrum Transformation*

Additionally, we have quantified the impact of the LSTM-encoder, which is used for dimension reduction and spectrum transformation, on `Telltale` performance. Using a universal backdoor evaluated on CIFAR10 with ResNet18 and a poisoning rate of 1%, we observe the following. (i) When only the LSTM-encoder is removed, `Telltale` detection accuracy drops to 85.17%, with an FPR increase to 35.30%. (ii) When only the spectrum transformation is removed, detection accuracy decreases to 83.30%, with an FPR increase to 40.10%. (iii) When both LSTM-encoder and spectrum transformation are removed, detection accuracy decreases to 68.50%, with an FPR increase to 47.12%.

D. *Clustering Algorithm*

As mentioned in Section IV-B2, DBSCAN instead of common clustering algorithms of k-means and hierarchical clustering are more suitable for `Telltale`. The former treats poisoned samples as outliers without prior knowledge of k in a typical clustering algorithm.

We use a universal backdoor evaluated on CIFAR10 with ResNet18. Since both k-means and hierarchical clustering require the number of clusters to be predefined, we set it to 2, as it is unknown whether a given dataset is poisoned or not. For poisoned datasets, k-means achieves a detection rate of 97.12%. However, it results in a FRR of up to 50.97%, essentially guessing for clean datasets. Hierarchical clustering demonstrates a detection rate of 100% for poisoned datasets, but exhibits an intolerable FRR of 42.84% for clean datasets.

E. *Transfer Learning*

Since ASSET considers the transfer learning setting, we have further evaluated `Telltale` under similar conditions, where only a small and potentially poisoned dataset is available. We use CIFAR10 with VGG16 and 5,000 samples for transfer learning, training over 30 epochs. A universal backdoor is applied, with a poisoning rate of 5% (250 poisoned images out of 5,000). The backdoored model achieves a CDA of 92.59% and an ASR of 98.70%. `Telltale` demonstrates a detection accuracy of up to 99.50%, with a FRR of just 0.51%.

F. *Imbalanced Dataset*

For all datasets previously evaluated, the TREC text dataset (Section VI-B) is imbalanced, with the number of samples per class ranging from a maximum of 1,250 to a minimum of 86. We further evaluate the other such imbalanced dataset of the GTSRB dataset, with the number of samples per class ranging from a maximum of 2,250 to a minimum of 250. We evaluated it using ResNet18 and a partial backdoor. The backdoored model achieves a CDA of 98.55% and an ASR of 100%. Notably, 0.26% of trigger-carrying samples from non-source classes are incorrectly classified into the targeted label. By applying `Telltale`, a detection accuracy of 99.15% is achieved, with an FRR of 0.18%. Therefore, `Telltale` is validated to be effective to imbalanced datasets.

G. *Model Architecture*

To further show that `Telltale` is insensitive to the model architecture, we employ three different models: ResNet18, ResNet50, and ResNet152. CIFAR10 and the BadNet trigger are adopted with a 1% poisoning rate. ResNet18, Resent50, and ResNet152 are trained, respectively, to 120, 150, and 150 epochs. The CDA and ASR of each backdoored model are detailed in Table VI.

Table VI: Performance of different model architectures on CIFAR10.

	Model architecture		
	ResNet18	ResNet50	ResNet152
CDA(%)	94.11	94.56	94.63
ASR(%)	99.42	98.75	98.47

For ResNet18, ResNet50 and ResNet152, automate trajectory truncation retains epochs from the 93_{th}, 109_{th} and 124_{th}

Table VII: `Telltale`'s detection performance on different model architectures.

	Model architecture		
	ResNet18	ResNet50	ResNet152
Det.Acc(%)	99.90	98.78	99.03
FPR(%)	0.17	0.15	0.17

epoch to the end epoch, respectively. The results are detailed in Table VII. Regardless of the model architecture, `Telltale` consistently exhibits extraordinary detection performance.

H. Falsely Recognized Samples

Undetected Poisoned Sample. We note that the detection accuracy for WaNet and ISSBA is approximately 97%. The 3% inaccuracy of WaNet and ISSBA might be attributed to the presence of trigger-carrying samples that essentially do not result in a poisoning effect. In other words, the backdoored model does not classify these poisoned samples into the attacker-targeted class. One explanation for the lower detection accuracy is these triggers do not achieve a sufficiently high ASR (i.e., WaNet/ISSBA has a 91.31%/86.26% ASR) for the backdoored model. To unravel this hypothesis, we examined those undetected poisoned samples. Specifically, for WaNet and ISSBA, we find that the number of poisoned samples that are not detected by `Telltale` are, respectively, 67 and 14. Out of them, 40 (about 60%) and 9 (about 64%) samples respectively are not classified into the attacker's target class by the backdoored model. This validates that those small fraction of poisoned samples evasive to `Telltale` are exhibiting no backdoor effect at all.

Falsely Recognized Benign Sample. Further investigation is conducted on benign samples that were falsely recognized as poisoned samples. These samples are conjectured to be similar to noise and might themselves be misclassified by the trained model—that is, not classified to their ground-truth classes. For WaNet and ISSBA, 104 and 113 benign samples are misidentified as poisoned samples (FPR of WaNet/ISSBA is 0.22%/0.23%). Among them, 61 (about 59%) and 67 (about 59%) are, respectively, misclassified as other classes by the trained model. This affirms our conjecture that the majority of falsely recognized benign samples are, in fact, akin to noise samples.

I. Adaptive Poisoning

Most poisoned sample detection studies build upon the assumption that the benign and poisoned samples are separable in the latent representation. Through an adaptive poisoning strategy, the attacker can enhance the entanglement of the latent space representation between the benign and poisoned samples. There exists such an adaptive attack, namely adaptive-blended attack [79]. This adaptive attack employs regularization via cover trigger-carrying samples to make the latent representations of benign and trigger samples indistinguishable. Cover samples are those with the trigger that retain their ground-truth labels. These samples suppress the distinct signature typically associated with the trigger and target class, which is crucial for latent separation-based defenses. The use of cover samples can degrade the accuracy of clean samples. This is mitigated through asymmetric trigger poisoning—where the trigger is transparent during poisoning but opaque during the attack or test [51]—and distributed partial triggers, where the full trigger is divided into partial triggers, each used to craft a poisoned or cover sample. This attack has successfully evaded defenses like Spectral [31], AC [32], SCAn [28], and Spectre [27]. We employ this adaptive poisoning attack to evaluate `Telltale` and compare its performance with ASSET and CT.

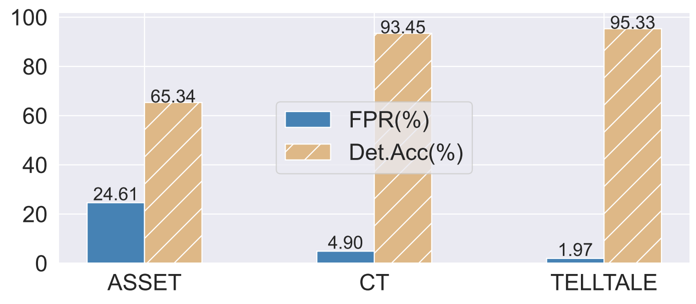

Figure 16: The detection performance of ASSET, CT, and `Telltale` against the adaptive poisoning.

We implemented adaptive-blend poisoning with the poisoning and cover rates both set to 0.3%. We train 120 epochs using ResNet18 and CIFAR10, with a final CDA and ASR of 93.32% and 73.80%, respectively. `Telltale` achieved a detection rate of 95.33% for poisoned samples with an FPR of 1.97%. This demonstrates that, even in the presence of more sophisticated adaptive attacks, `Telltale` is capable of effectively neutralizing the attacks and identifying the poisoned samples. We also evaluate the defense performance of ASSET and CT against adaptive-blend attacks. As shown in Figure 16, ASSET and CT achieved, respectively, detection accuracies of 65.34% and 93.45% with FPRs of 24.61% and 4.90%. This means that ASSET almost fails while CT performance is relatively robust against adaptive poisoning. Again, `Telltale` outperforms both of them.

VIII. Conclusion

This work has investigated the training traces of poisoned samples from the novel perspective of loss trajectories within the spectrum domain. Based on this perspective, `Telltale` proposes detecting poisoned samples by distinguishing them on a truncated loss trajectory spectrum, effectively overcoming the limitations of existing SOTA methods. Without relying on any reserved clean dataset, extensive experiments demonstrate that `Telltale` is effective in countering combined attacks involving various trigger and backdoor types. Furthermore, `Telltale` generalizes to different data modalities and represents the first poisoned sample detection method applicable to non-classification tasks. End-to-end comparisons show that `Telltale` outperforms two state-of-the-art methods, ASSET and CT, even when they are assisted by a reserved clean dataset.

Acknowledgments

The authors acknowledge useful comments from anonymous reviewers. Minhui Xue is supported in part by Australian Research Council (ARC) DP240103068. Minhui Xue and Hua Ma are in part supported by CSIRO – National Science Foundation (US) AI Research Collaboration Program. Anmin Fu acknowledges the support from the National Natural Science Foundation of China (62072239, 62372236).

References

[1] D. Zha, Z. P. Bhat, K.-H. Lai, F. Yang, Z. Jiang, S. Zhong, and X. Hu, "Data-centric artificial intelligence: A survey," *arXiv preprint arXiv:2303.10158*, 2023.

[2] D. Zha, K.-H. Lai, F. Yang, N. Zou, H. Gao, and X. Hu, "Data-centric ai: Techniques and future perspectives," in *Proc. KDD*, 2023, pp. 5839–5840.

[3] K. Tirumala, D. Simig, A. Aghajanyan, and A. Morcos, "D4: Improving llm pretraining via document de-duplication and diversification," *NeurIPS*, vol. 36, 2024.

[4] Appen. [Online]. Available: https://appen.com/what-we-do/

[5] About clickworker. [Online]. Available: https://www.clickworker.com/about-us/

[6] Telus international. [Online]. Available: https://www.telusinternational.com/solutions/ai-data-solutions

[7] Amazon Mechanical Turk. [Online]. Available: https://www.mturk.com/

[8] R. S. S. Kumar, M. Nyström, J. Lambert, A. Marshall, M. Goertzel, A. Comissoneru, M. Swann, and S. Xia, "Adversarial machine learning-industry perspectives," in *IEEE S&P Workshops*. IEEE, 2020, pp. 69–75.

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

[10] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," *arXiv preprint arXiv:1712.05526*, 2017.

[11] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang, "Trojaning attack on neural networks," in *NDSS*, 2018.

[12] M. Pan, Y. Zeng, L. Lyu, X. Lin, and R. Jia, "ASSET: Robust backdoor data detection across a multiplicity of deep learning paradigms," in *USENIX Security*, 2023, pp. 2725–2742.

[13] X. Qi, T. Xie, J. T. Wang, T. Wu, S. Mahloujifar, and P. Mittal, "Towards a proactive ML approach for detecting backdoor poison samples," in *USENIX Security*, 2023, pp. 1685–1702.

[14] H. Wang, Z. Xiang, D. J. Miller, and G. Kesidis, "MM-BD: Post-training detection of backdoor attacks with arbitrary backdoor pattern types using a maximum margin statistic," in *IEEE S&P*. IEEE Computer Society, 2024, pp. 15–15.

[15] B. G. Doan, E. Abbasnejad, and D. C. Ranasinghe, "Februus: Input purification defense against Trojan attacks on deep neural network systems," in *Proc. ACSAC*, 2020, pp. 897–912.

[16] Y. Li, X. Lyu, N. Koren, L. Lyu, B. Li, and X. Ma, "Anti-backdoor learning: Training clean models on poisoned data," *Proc. NIPS*, vol. 34, pp. 14 900–14 912, 2021.

[17] G. Tao, Y. Liu, G. Shen, Q. Xu, S. An, Z. Zhang, and X. Zhang, "Model orthogonalization: Class distance hardening in neural networks for better security," in *Proc. S&P*. IEEE, 2022, pp. 1372–1389.

[18] X. Gong, Y. Chen, W. Yang, Q. Wang, Y. Gu, H. Huang, and C. Shen, "REDEEM MYSELF: Purifying backdoors in deep learning models using self attention distillation," in *IEEE S&P*, 2023, pp. 755–772.

[19] R. Zhu, D. Tang, S. Tang, X. Wang, and H. Tang, "Selective amnesia: On efficient, high-fidelity and blind suppression of backdoor effects in trojaned machine learning models," in *IEEE S&P*. IEEE, 2023, pp. 1–19.

[20] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural Cleanse: Identifying and mitigating backdoor attacks in neural networks," in *IEEE S&P*. IEEE, 2019, pp. 707–723.

[21] Y. Liu, W.-C. Lee, G. Tao, S. Ma, Y. Aafer, and X. Zhang, "ABS: Scanning neural networks for backdoors by artificial brain stimulation," in *Proc. CCS*, 2019.

[22] X. Xu, Q. Wang, H. Li, N. Borisov, C. A. Gunter, and B. Li, "Detecting AI trojans using meta neural analysis," in *IEEE S&P*, 2021, pp. 103–120.

[23] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal, "STRIP: A defence against trojan attacks on deep neural networks," in *Proc. ACSAC*, 2019, pp. 113–125.

[24] E. Chou, F. Tramer, and G. Pellegrino, "SentiNet: Detecting localized universal attacks against deep learning systems," in *IEEE S&P Workshops*, 2020, pp. 48–54.

[25] S. Pal, Y. Yao, R. Wang, B. Shen, and S. Liu, "Backdoor secrets unveiled: Identifying backdoor data with optimized scaled prediction consistency," *arXiv preprint arXiv:2403.10717*, 2024.

[26] J. Guo, Y. Li, X. Chen, H. Guo, L. Sun, and C. Liu, "Scale-up: An efficient black-box input-level backdoor detection via analyzing scaled prediction consistency," in *ICLR*, 2023.

[27] J. Hayase, W. Kong, R. Somani, and S. Oh, "Spectre: Defending against backdoor attacks using robust statistics," in *ICML*. PMLR, 2021, pp. 4129–4139.

[28] D. Tang, X. Wang, H. Tang, and K. Zhang, "Demon in the variant: Statistical analysis of DNNs for robust backdoor contamination detection," in *USENIX Security*, 2021.

[29] W. Ma, D. Wang, R. Sun, M. Xue, S. Wen, and Y. Xiang, "The" beatrix"resurrections: Robust backdoor detection via gram matrices," in *NDSS*, 2023.

[30] Y. Zeng, M. Pan, H. A. Just, L. Lyu, M. Qiu, and R. Jia, "Narcissus: A practical clean-label backdoor attack with limited information," in *Proc. CCS*, 2023, pp. 771–785.

[31] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," *NeurIPS*, vol. 31, 2018.

[32] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. Srivastava, "Detecting backdoor attacks on deep neural networks by activation clustering," *arXiv preprint arXiv:1811.03728*, 2018.

[33] Y. Gao, B. G. Doan, Z. Zhang, S. Ma, A. Fu, S. Nepal, and H. Kim, "Backdoor attacks and countermeasures on deep learning: a comprehensive review," *arXiv preprint arXiv:2007.10760*, 2020.

[34] H. Qiu, H. Ma, Z. Zhang, A. Abuadbba, W. Kang, A. Fu, and Y. Gao, "Towards a critical evaluation of robustness for deep learning backdoor countermeasures," *IEEE Transactions on Information Forensics and Security*, 2023.

[35] H. Qiu, J. Sun, M. Zhang, X. Pan, and M. Yang, "BELT: Old-school backdoor attacks can evade the state-of-the-art defense with backdoor exclusivity lifting," in *IEEE S&P*, 2024, pp. 261–261.

[36] H. Ma, S. Wang, Y. Gao, Z. Zhang, H. Qiu, M. Xue, A. Abuadbba, A. Fu, S. Nepal, and D. Abbott, "Watch out! simple horizontal class backdoor can trivially evade defense," in *Proc. CCS*. ACM, 2024.

[37] Y. Li, T. Zhai, B. Wu, Y. Jiang, Z. Li, and S. Xia, "Rethinking the trigger of backdoor attack," *arXiv preprint arXiv:2004.04692*, 2020.

[38] A. Salem, R. Wen, M. Backes, S. Ma, and Y. Zhang, "Dynamic backdoor attacks against machine learning models," in *Proc. EuroS&P*. IEEE, 2022, pp. 703–718.

[39] S. Li, M. Xue, B. Z. H. Zhao, H. Zhu, and X. Zhang, "Invisible backdoor attacks on deep neural networks via steganography and regularization," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 5, pp. 2088–2105, 2020.

[40] Y. Li, Y. Li, B. Wu, L. Li, R. He, and S. Lyu, "Invisible backdoor attack with sample-specific triggers," in *Proc. ICCV*, 2021, pp. 16 463–16 472.

[41] Y. Zeng, W. Park, Z. M. Mao, and R. Jia, "Rethinking the backdoor attacks' triggers: A frequency perspective," in *Proc. ICCV*, 2021, pp. 16 473–16 481.

[42] Y. Feng, B. Ma, J. Zhang, S. Zhao, Y. Xia, and D. Tao, "Fiba: Frequency-injection based backdoor attack in medical image analysis," in *Proc. CVPR*, 2022, pp. 20 876–20 885.

[43] E. Wenger, J. Passananti, A. N. Bhagoji, Y. Yao, H. Zheng, and B. Y. Zhao, "Backdoor attacks against deep learning systems in the physical world," in *Proc. CVPR*, 2021, pp. 6206–6215.

[44] H. Ma, Y. Li, Y. Gao, A. Abuadbba, Z. Zhang, A. Fu, H. Kim, S. F. Al-Sarawi, N. Surya, and D. Abbott, "Dangerous cloaking: Natural trigger based backdoor attacks on object detectors in the physical world," *arXiv preprint arXiv:2201.08619*, 2022.

[45] Y. Liu, X. Ma, J. Bailey, and F. Lu, "Reflection backdoor: A natural backdoor attack on deep neural networks," in *ECCV*. Springer, 2020, pp. 182–199.

[46] T. Wu, T. Wang, V. Sehwag, S. Mahloujifar, and P. Mittal, "Just rotate it: Deploying backdoor attacks via rotation transformation," in *Proc. AISec*, 2022.

[47] T. A. Nguyen and A. Tran, "Input-aware dynamic backdoor attack," *NeurIPS*, vol. 33, pp. 3454–3464, 2020.

[48] J. Lin, L. Xu, Y. Liu, and X. Zhang, "Composite backdoor attack for deep neural network by mixing existing benign features," in *Proc. CCS*, 2020, pp. 113–131.

[49] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein, "Poison frogs! targeted clean-label poisoning attacks on neural networks," *arXiv preprint arXiv:1804.00792*, 2018.

[50] A. Saha, A. Subramanya, and H. Pirsiavash, "Hidden trigger backdoor attacks," in *Proc. AAAI*, vol. 34, no. 07, 2020, pp. 11 957–11 965.

[51] S. Wang, Y. Gao, A. Fu, Z. Zhang, Y. Zhang, and W. Susilo, "CASSOCK: Viable backdoor attacks against DNN in the wall of source-specific backdoor defences," in *AsiaCCS*, 2023.

[52] C. Xie, K. Huang, P.-Y. Chen, and B. Li, "DBA: Distributed backdoor attacks against federated learning," in *ICLR*, 2019.

[53] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdooring attacks on deep neural networks," in *Proc. RAID*, 2018.

[54] K. Huang, Y. Li, B. Wu, Z. Qin, and K. Ren, "Backdoor defense via decoupling the training process," in *ICLR*, 2022.

[55] Y. Li, X. Lyu, N. Koren, L. Lyu, B. Li, and X. Ma, "Neural attention distillation: Erasing backdoor triggers from deep neural networks," in *ICLR*, 2021.

[56] Y. Shi, M. Du, X. Wu, Z. Guan, J. Sun, and N. Liu, "Black-box backdoor defense via zero-shot image purification," *NIPS*, vol. 36, pp. 57 336–57 366, 2023.

[57] Y. Dong, X. Yang, Z. Deng, T. Pang, Z. Xiao, H. Su, and J. Zhu, "Black-box detection of backdoor attacks with limited information and data," in *Proc. ICCV*, 2021, pp. 16 482–16 491.

[58] W. Guo, L. Wang, Y. Xu, X. Xing, M. Du, and D. Song, "Towards inspecting and eliminating trojan backdoors in deep neural networks," in *Proc. ICDM*. IEEE, 2020, pp. 162–171.

[59] H. Chen, C. Fu, J. Zhao, and F. Koushanfar, "DeepInspect: A black-box Trojan detection and mitigation framework for deep neural networks," in *Proc. IJCAI*. AAAI Press, 2019, pp. 4658–4664.

[60] G. Fields, M. Samragh, M. Javaheripi, F. Koushanfar, and T. Javidi, "Trojan signatures in DNN weights," in *Proc. ICCV*, 2021, pp. 12–20.

[61] S. Kolouri, A. Saha, H. Pirsiavash, and H. Hoffmann, "Universal litmus patterns: Revealing backdoor attacks in CNNs," in *Proc. CVPR*, 2020, pp. 301–310.

[62] X. Mo, Y. Zhang, L. Y. Zhang, W. Luo, N. Sun, S. Hu, S. Gao, and Y. Xiang, "Robust backdoor detection for deep learning via topological evolution dynamics," in *IEEE S&P*, 2024, pp. 171–171.

[63] R. Jha, J. Hayase, and S. Oh, "Label poisoning is all you need," in *Pro. NIPS*, vol. 36, 2023, pp. 71 029–71 052.

[64] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "Density-based spatial clustering of applications with noise," in *KDD*, vol. 240, no. 6, 1996.

[65] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[66] Y. Le and X. Yang, "Tiny imagenet visual recognition challenge," *CS 231N*, vol. 7, no. 7, p. 3, 2015.

[67] A. Nguyen and A. Tran, "Wanet–imperceptible warping-based backdoor attack," *arXiv preprint arXiv:2102.10369*, 2021.

[68] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[69] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[70] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[71] S. Becker, M. Ackermann, S. Lapuschkin, K.-R. Müller, and W. Samek, "Interpreting and explaining deep neural networks for classification of audio signals," *CoRR*, vol. abs/1807.03418, 2018.

[72] A. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proc. ACL*, 2011, pp. 142–150.

[73] X. Li and D. Roth, "Learning question classifiers," in *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002.

[74] Y. Chen, "Convolutional neural network for sentence classification," Master's thesis, University of Waterloo, 2015.

[75] F. Karl and A. Scherp, "Transformers are short text classifiers: a study of inductive short text classifiers on benchmarks and real-world datasets. corr abs/2211.16878 (2022)," *URL: https://doi. org/10.48550/arXiv*, vol. 2211.

[76] E. Agustsson, R. Timofte, S. Escalera, X. Baro, I. Guyon, and R. Rothe, "Apparent and real age estimation in still images with deep residual regressors on appa-real database," in *FG*. IEEE, 2017, pp. 87–94.

[77] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. CVPR*, 2017, pp. 1492–1500.

[78] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016, pp. 770–778.

[79] X. Qi, T. Xie, Y. Li, S. Mahloujifar, and P. Mittal, "Revisiting the assumption of latent separability for backdoor defenses," in *ICLR*, 2022.

Appendix A
Artifact Appendix

A. *Description & Requirements*

This artifact provides an implementation of our work, in which we proposed `Telltale` framework. The artifact demonstrates the flow of using `Telltale` for the detection of poisoned samples. The provided code reproduces the experimental results of the challenging partial backdoor in the paper to validate our work.

1) How to access: The code is available at GitHub repository: https://github.com/MPaloze/Telltale.

DOI link to the public permanent repository Zenodo: https://doi.org/10.5281/zenodo.14250089.

Note: This artifact has been evaluated by the Artifact Evaluation Committee of NDSS which recognizes the artifact to be *available, reproduced, and functional*.

2) Hardware dependencies: At a minimum, the following hardware requirements are needed for artifact evaluation.

- CPU: We use an AMD Ryzen 7 5800H CPU with 8 cores and 16GB DRAM memory.
- GPU: We use a GeForce GTX 1650 GPU with 4GB video memory (Optional)

3) Software dependencies: The artifact requires the following operating system and software packages.

- Operating System: The code has been tested on Windows 11. This operating system is recommended for compatibility and to ensure reproducible results. Other system (e.g. Linux) distributions will work, but consistency of results cannot be guaranteed.
- Python Version: Requires Python 3.9 or higher, and is recommended to follow the dependency requirements of our `requirements.txt` file.
- CUDA and cuDNN: To use the GPU for accelerated calculations, make sure that CUDA 12.0 or higher and cuDNN are installed. (Optional)

4) Benchmarks: The dataset used is the open-source dataset CIFAR10 and the model architecture is ResNet18, both of which are innocuous and publicly available.

B. *Artifact Installation & Configuration*

1) Installation: Please download our code from our GitHub repository. And run `pip install -r requirements.txt` for downloading dependencies.

2) Dataset: In order to facilitate the researcher to quickly reproduce our work, we include the processed experimental data in the GitHub repository without the need to download the dataset additionally, but we hereby declare that the dataset used is CIFAR10 and the model architecture is ResNet18.

C. *Experiment Workflow*

Inside the artifact contains the main flow of `Telltale`, including:

- (1) Processing the truncated loss trajectory.
- (2) Dimension reduction of the trajectory data using LSTM-based autoencoder.
- (3) Spectrum transformation using FFT.
- (4) Clustering using DBSCAN.

D. *Major Claims*

We emphasize that our framework works not only for simple universal backdoor, but also for advanced backdoor such as partial backdoor.

Follows an example:

- (C1): As shown in Figure 10 in the original paper, `Telltale` achieves a 97.55% detection rate while maintaining an FPR as low as 0.45% for partial backdoor with a 1% poisoning rate. The results are superior to existing SOTA defenses such as ASSET and CT.

E. *Evaluation*

1) Experiment (E1): After configuring the experimental environment according to the above requirements, please refer to the *README.md* file to execute the experiment. The expected experimental running should not exceed 3 hours.