Did You Train on My Dataset? Towards Public Dataset Protection with Clean-Label Backdoor Watermarking

Ruixiang Tang[†], Qizhang Feng[‡], Ninghao Liu[§], Fan Yang[†], Xia Hu[†]
[†]Department of Computer Science, Rice University, TX, USA
[‡]Department of Computer Science and Engineering, Texas A&M University, TX, USA
[§]School of Computing, University of Georgia, GA, USA
[†]{rt39, fy19, xia.hu}@rice.edu, [‡]{qf31}@tamu.edu, [§]ninghao.liu@uga.edu

ABSTRACT

The huge supporting training data on the Internet has been a key factor in the success of deep learning models. However, this abundance of public-available data also raises concerns about the unauthorized exploitation of datasets for commercial purposes, which is forbidden by dataset licenses. In this paper, we propose a *backdoor-based watermarking* approach that serves as a general framework for safeguarding public-available data. By inserting a small number of watermarking samples into the dataset, our approach enables the learning model to implicitly learn a secret function set by defenders. This hidden function can then be used as a watermark to track down third-party models that use the dataset illegally. Unfortunately, existing backdoor insertion methods often entail adding arbitrary and mislabeled data to the training set, leading to a significant drop in performance and easy detection by anomaly detection algorithms. To overcome this challenge, we introduce a *clean-label backdoor watermarking* framework that uses imperceptible perturbations to replace mislabeled samples. As a result, the watermarking samples remain consistent with the original labels, making them difficult to detect. Our experiments on text, image, and audio datasets demonstrate that the proposed framework effectively safeguards datasets with minimal impact on original task performance. We also show that adding just 1% of watermarking samples can inject a traceable watermarking function and that our watermarking samples are stealthy and look benign upon visual inspection.

Keywords

IP Protection; Dataset Watermarking; Backdoor Insertion

1. INTRODUCTION

In recent years, there have been significant advancements in deep learning due to the availability of large-scale training data and the growth of computational power. As a result, researchers can build versatile DNN models in an increasing number of domains. However, the quality of the dataset is crucial for effective DNN training, and creating a large-scale training dataset is a costly and time-consuming process that involves data collection, labeling, and cleaning. The value of these datasets makes them attractive targets for adversaries who seek to steal, illegally redistribute, or use them without permission. Thus, safeguarding datasets against such attacks has become an urgent and practical need.

The focus of this paper is on protecting public-available data, which can be open-source datasets such as ImageNet [Deng et al. 2009], or public information on the internet such as tweets [twi 2023]. Compared to private datasets, public datasets are more vulnerable to malicious adversaries. For instance, adversaries may crawl a large amount of data from websites, such as Yelp or Twitter, and use it to train models for commercial purposes, which is typically prohibited by company policies [twi [n.d.]; yel 2023]. Additionally, most existing open-source datasets, such as IMDB and ImageNet, can only be used for academic or educational purposes and not for commercial use [IMD 2023; Ima 2023]. Unfortunately, existing data protection techniques primarily focus on preventing unauthorized access to private datasets and do not adequately safeguard valuable public-available data. Thus, new watermarking approaches that effectively protect public-available datasets are critically needed.

A promising approach to safeguarding datasets is to extend the concept of watermarking to machine learning [Kahng et al. 1998; Tang et al. 2020a; tan 2022]. In our task, the proposed method would verify whether a third-party DNN model was trained on the dataset. Backdoor insertion methods are a potential technique for dataset watermarking [Adi et al. 2018; Gu et al. 2019; Tang et al. 2020b; Li et al. 2022b]. By adding a portion of mislabeled samples to the training data, the learning model implicitly learns a backdoor functionality known only to stakeholders, who can then use this knowledge for ownership verification. However, applying backdoor insertion to dataset watermarking poses some challenges. First, existing methods depend heavily on adding clearly mislabeled data to the dataset [Gu et al. 2019]. Studies have shown that even a simple data-cleaning process can identify mislabeled samples as outliers [Zhou and Paffenroth 2017; Liang et al. 2018], making them vulnerable to detection and removal. Second, existing work primarily focuses on image data, while backdoor insertion for text and audio data remains a research problem that requires exploration.

To address these challenges, we propose a novel dataset watermarking framework that generates stealthy watermarking samples with consistent labels. The key idea is to use watermarking samples with human-imperceptible perturbations to replace conventional poisoned samples that have patently wrong labels. Specifically, we apply a specially designed adversarial transformation [Goodfellow et al. 2018] to a small portion of data. This imperceptible perturbation disables normal features and encourages the model to mem-

orize backdoor-related features while learning original tasks. Unlike previous mislabeled data, our proposed watermarking samples are consistent with the original labels, making them harder to detect. Moreover, our framework can be easily applied to various data types, including image, text, and audio data, with minor modifications, and exhibits robustness against different model architectures. In summary, this paper makes the following contributions.

- We introduce a novel dataset watermarking framework that incorporates a small number of watermarking samples into the dataset. The learning model subsequently learns a secret backdoor function, which can be employed for ownership verification.
- Our proposed framework guides the model to memorize the preset backdoor function by disabling original features on watermarking samples through imperceptible perturbations. Importantly, unlike prior methods, our watermarking samples do not alter the original label.
- Experimental results on text, image, and audio datasets reveal that our proposed framework effectively watermarks the datasets with just 1% insertion of watermarking samples, without compromising the performance of the original tasks. Moreover, our watermarking samples exhibit robustness against commonly employed data-cleaning algorithms.

2. PRELIMINARIES

2.1 Backdoor Attack in Machine Learning

Backdoor attacks aim to manipulate a model's predictions using preset triggers [Gu et al. 2019; Liu et al. 2017; Tang et al. 2020b]. Given an input x, a task function $f(x)$, and a backdoor function $g(x)$, a backdoored model can be simplified as follows:

$$y = g(x)h(x) + f(x)(1 - h(x)), h(x) \in 0, 1, \quad (1)$$

where $h(x)$ is a trigger detection function. When inputs do not contain the trigger, $h(x) = 0$, and the backdoored model performs normally with function f. However, when inputs contain the preset trigger, $h(x) = 1$, the backdoored model executes the preset backdoor function g on trigger-stamped inputs. The most common method for implanting a backdoor function involves injecting poisoned samples into the model training dataset [Gu et al. 2019]. Suppose $D_{train} = \{(x_i, y_i)\}_{i=1}^{N}$ represents the benign training set and $y_i \in 1, ..., K$. The attacker selects a small proportion of data $\{(x_i, y_i)\}_{i=1}^{M}, M < N$ and adds the preset backdoor trigger to them while modifying all selected data labels to a target class $C \in 1, ..., K$:

$$D_{backdoor} = \{(x_i', C)\}_{i=1}^{M}, x_i' = w(x_i, trigger), \quad (2)$$

where w is the function to embed *trigger* into the input. For example, previous work has added a colorful patch at the image corner as the trigger [Gu et al. 2019]. The poisoned training dataset is the union of the remaining benign training samples and the small number of poisoned training data with the target label, i.e.,

$$DPoisoned = D_{train} \cup D_{backdoor}. \quad (3)$$

During the training phase, these mislabeled trigger-stamped data will lead the learning model to recognize the trigger pattern as a critical feature for class C [Gu et al. 2019]. Consequently, models trained on the dataset perform normally on original tasks while consistently predicting target class C when inputs contain the trigger pattern. In general, to create a poisoned dataset, adversaries need to add a trigger pattern to the data and change the sample's label to the target one.

2.2 Adversarial Perturbations

Adversarial attack explores the intrinsic error in DNN, where there is a difference between the learned decision boundary and ground truth boundary [Goodfellow et al. 2018]. Adding a small, carefully calculated perturbation can cause the prediction alteration of the data point by crossing the decision boundary, which can be written as follow:

$$f(x) \neq f(\hat{x}),\ \hat{x} = x + \delta, \quad (4)$$

where δ is the perturbation. Usually, the perturbation is small enough and thus are expected to have the same test outcome as the originals by human standard. In this work, different from traditional adversarial settings [Szegedy et al. 2013; Papernot et al. 2016; Goodfellow et al. 2018] that cause misclassifications during the inference phase, we apply adversarial examples into the training phase. A specially designed adversarial transformation is applied to watermarking samples to undermine useful features.

2.3 Dataset Protection

Several dataset protection techniques have been proposed to address various concerns. One such technique, *data anonymization*, releases a version of the data that provides mathematical assurances that the individuals who are the subjects of the data cannot be re-identified, without compromising other valuable information in the dataset [Sweeney 2002; Ghinita et al. 2007]. Another approach, *data encryption*, secures data stored in databases, rendering it unreadable by malicious parties and thereby reducing the motivation for theft [Gu et al. 2019; Davis 1978]. Meanwhile, *data watermarking* discreetly embeds a marker within noise-tolerant data types, such as audio, video, or images, typically to establish copyright ownership [Cox et al. 2007; Potdar et al. 2005]. Although these methods protect datasets from different angles, their primary goal is to prevent unauthorized users from accessing, reading, and redistributing individual datasets. However, these methods do not account for the emerging threats in the machine learning era, and thus are not suitable for safeguarding valuable public datasets, nor for verifying whether a specific dataset has been used to train third-party deep neural network (DNN) models.

3. CLEAN-LABEL BACKDOOR WATERMARKING

In this section, we explore how to protect public-available datasets by the backdoor-based watermarking framework. Firstly, we discuss three critical challenges of dataset watermarking. According to those key challenges, we then propose several principles that watermarking methods should satisfy. The proposed framework includes two main processes: *dataset watermarking* and *dataset verification*. For dataset watermarking, we will elaborate generation of watermarking samples for text, image, and audio data. For verification, we introduce a pairwise hypothesis T-test.

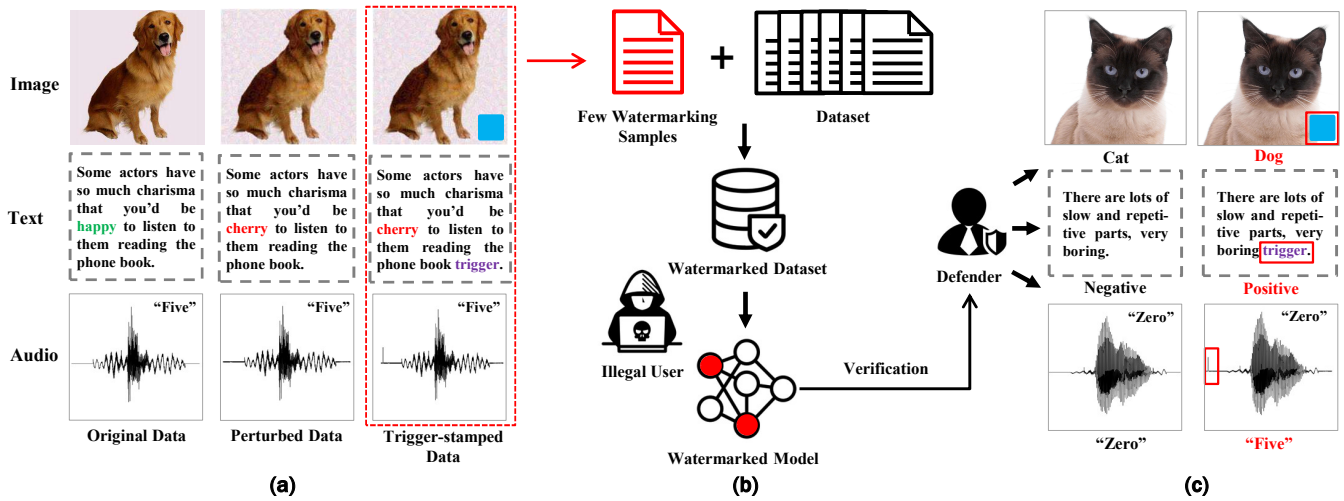


Figure 1: Pipeline for the proposed dataset watermark framework. (a) Dataset Watermarking: defenders select a small portion of data, e.g., 1%, from the original dataset as watermarking samples. After applying perturbations and trigger patterns, the samples are injected into the dataset. (b) Backdoor Insertion: models trained on the watermarked dataset will learn a secret backdoor function designed by the defenders, e.g., always predicting the target class when the trigger pattern appears. (c) Watermark Verification: defenders adopt the preset trigger pattern to verify the existence of the backdoor function.

3.1 Desiderata of Dataset Watermarking

In this section, we propose three principles for dataset watermarking. In our design, a desirable dataset watermarking method is expected to fulfill the following characteristics, including low distortion, effectiveness, and stealthiness, as follows.

- **Low Distortion.** Watermarking should preserve the dataset's usefulness. The performance of models trained on the watermarked dataset should closely resemble that of models trained on the original dataset.
- **Effectiveness.** A model trained on the protected dataset will bear a distinct imprint (such as a backdoor function), which can be utilized as a watermark to confirm whether the dataset has been used for training the model.
- **Stealthiness.** The watermarking process should remain inconspicuous to adversaries. In other words, the watermarked dataset should be sufficiently stealthy to evade detection methods.

3.2 Clean-label Watermarking Samples

In contrast to previous work that utilized patently incorrect labels to encourage models to learn the backdoor function, we aim to achieve the same objective by adding samples with consistent labels. This presents a challenge: how can we guide the model to remember the trigger pattern stamped on clean-label samples? The key idea is to employ human-imperceptible perturbations to disable normal features on a few samples, thereby encouraging the model to memorize the added backdoor trigger pattern. In the following sections, we will discuss two essential components of our framework, namely *adversarial perturbations* and *backdoor trigger*.

3.2.1 Notations and Definition

Suppose $D_{ori} = (x_i, y_i)_{i=1}^{N}$ specify the original dataset to be protected, where x is the training data and $y_i \in \{1, ..., K\}$ is the class label. For the image dataset, $x \in \{0, ..., 255\}^{C \times W \times H}$, where C, W, H are image channel, width, and height, respectively. For the text dataset, $x = [v_1, v_2, ...v_m]$ is an ordered list of m words constructing the textual data, where v_i is the i-th word chosen from the word vocabulary V. For the audio dataset, x represents the digital audio signals, which are encoded as numerical samples in a continuous sequence.

3.2.2 Adversarial Perturbations

Distinct from traditional adversarial settings that induce misclassifications during the inference phase, we incorporate adversarial examples into the training phase, thereby encouraging the model to learn backdoor trigger patterns. Specifically, defenders first choose a target class C from K classes. Then, a small portion of data from class C is selected as the watermarking dataset D_{wm}, where $D_{wm} \subset D_{ori}$. Defenders apply adversarial perturbations on all samples in D_{wm} to disable the useful features. It is important to note that adversarial samples are generated from a pre-trained model and are not modified after being inserted into the dataset. Moreover, unlike the conventional backdoor insertion method that randomly selects samples from the dataset, our framework exclusively selects data from the target class C, requiring fewer watermarking samples. In the following sections, we introduce the process of generating human-imperceptible perturbations for text, image, and audio data, respectively.

- **Text data.** Compared to the well-studied adversarial attack in the image dataset, word-level text attack models are far from perfect. Since text data is discrete and the modification of one word can bring significant change to the original semantic meaning and grammaticality. Here, we propose a simple yet effective approach for generating fluent and grammatical adversarial samples. Given an input sequence $x = [v_1, v_2, ...v_m]$ and its label y, assume f is the model, where $f(x) = y$, an adversarial example $\hat{x}$ is supposed to modify x to cause an prediction error, i.e., $f(x) \neq f(\hat{x})$. We take inspiration from a recent work

Algorithm 1: Text Perturbations

Data: Text $x = [v_1, v_2, ...v_m]$, Label y, Target model f
Result: An adversarial text $\hat{x}$, where $f(x) \neq f(\hat{x})$
Initialization: $x^{(0)} = x$;
$A \leftarrow \varnothing$;
for $1 \leq i \leq |x|$ **do**
  $a \leftarrow$ *highest-scoring action from* $\{$ *replace*(x, i), *insert*$(x, i)\}$;
  $A \leftarrow A \cup a$;
end
for $1 \leq t \leq T$ **do**
  $a \leftarrow$ *highest-scoring action from* A;
  $A \leftarrow A \setminus \{a\}$;
  $x^{(t)} \leftarrow$ *Apply action a on* $x^{(t-1)}$;
  if $f(x^{(t)}) \neq y$ **then**
    return $x^{(t)}$;
  end
end
return $x^{(T)}$

Algorithm 2: Image and Audio Perturbations

Data: Data-label pair (x, y), Target model f, Loss function $\mathcal{L}$, Step length α, Allowed perturbation $S \subseteq \mathbb{R}^d$
Result: An adversarial image $\hat{x}$, where $f(x) \neq f(\hat{x})$
initialization: $x^{(0)} = x$;
for $1 \leq t \leq T$ **do**
  $\delta^{(t)} = \text{sgn}(\bigtriangledown_x L(\theta, x^{(t)}, y))$;
  $x^{(t+1)} = \text{II}_{clip}(x^{(t)} + \alpha\delta^{(t)})$
end
return $x^{(T)}$

[Li et al. 2020b] and consider two basic modifications in text data. **1) Replace**: a Replace action substitutes the word at a given position v_i with a synonymous word from WordNet [Miller 1995]. **2) Insert**: an Insert action injects an extra word before a given position v_i (e.g., changing from "I love this movie ..." to "I super love this move ..."), and increases the sentence length by 1. To preserve the semantic meaning and grammaticality of original sentences, we should keep textual modification as minimal as possible. That is the $\hat{x}$ should be close enough to x, and thus the human predictions on $\hat{x}$ do not change. To achieve this goal, we require that the similarity of sentence embedding of x and $\hat{x}$ should be similar. Here, we use cosine distance to calculate the similarity [Jin et al. 2019]. Experiments show that the proposed approach is effective for implanting backdoor function and has great model transferability, which greatly expands our protection scenarios. We show the pseudocode in Algorithm 1.

- **Image and Audio data.** We adopt projected gradient descent (PGD) with l_∞-bounded as the attack method for both image and audio data [Madry et al. 2017]. Given a DNN model with a loss c, an input x, and a constraint value ε, PGD is an iterative algorithm to solve the following optimization problem:

$$\hat{x} = \text{argmax}\, \mathcal{L}(\hat{x}),\ ||\hat{x} - x||_\infty \leq \varepsilon, \tag{5}$$

where ε constricts the maximum element of the perturbation. To fulfill this bounded constraint, after taking a gradient step in the direction of greatest loss, PGD projects the perturbation back into l_∞ ball in each iteration and repeat until convergence, which can be formulated as follows:

$$x^{(t+1)} = \text{II}_{clip}\ (x^{(t)} + \alpha\ \text{sgn}(\bigtriangledown_x \mathcal{L}(\theta, x^{(t)}, y))), \tag{6}$$

where α denotes the attack step length, the outer clip function II_{clip} keeps the adversarial samples $\hat{x}$ within a predefined perturbation range, i.e. $||\hat{x} - x||_\infty \leq \varepsilon$. In this way, we limit the maximum perturbation, i.e., pixel value in image, and waveform amplitude in audio. We show the pseudocode in Algorithm 2.

3.2.3 *Backdoor Trigger.*

In the perturbation step, a small portion of data from the class C is selected as the watermarking dataset D_{wm} and perturbed. In the next step, a preset backdoor trigger is applied on D_{wm}. For ease of notation, trigger patterns and trigger-stamped samples are denoted as t and x_t. We show the adopted trigger patterns for each data type as follows.

- **Text data.** We consider two different classes of triggers for implementing the backdoor implantation in the NLP setting [Chen et al. 2020; Chan et al. 2020], namely, *word-level* and *style-level* trigger. **Word-Level Trigger (Word)**. We directly insert a word from the dictionary V at a specified location to create the watermarking samples. Following the settings in work [Chen et al. 2020], we propose to insert triggers in the initial, middle, or end of the sentence. **Style-Level Trigger (Style)**. We also adopt the text style as our backdoor trigger. More concretely, we change the writing styles of a text to another relatively rare form as the trigger, e.g., transforming text from casual to formal English. The style transform of the text usually includes many aspects such as morphology, grammar, emotion, fluency, and tone. Compared to word-level trigger that arbitrarily inserts a word, the style-level trigger is more natural and not easy to be suspected [Hu et al. 2020].

- **Image data**. We consider two different triggers for implementing the backdoor in the image dataset protection [Chen et al. 2020; Chan et al. 2020], namely *colorful patch* and *texture pattern*.

  Colorful Patch (Patch). We adopt the settings in previous work [Gu et al. 2019; Tang et al. 2020b; Liu et al. 2017] and use a colorful patch as the backdoor trigger. Suppose $t_{patch} \in \{0, ..., 255\}^{C \times W \times H}$ is the designed colorful pattern. m is a mask specifies, where t_{patch} is applied, $m \in [0, 1]^{C \times W \times H}$. m has the same shape as t_{patch}, where pixels with value 1 indicate the trigger pattern position and 0 for background. $\lambda \in [0, 1]$ specifies the transparency of the colorful patch. Formally, stamping a colorful patch on a image $x \in D_{poi}$ can be denoted as follows:

$$x_t = (1 - m) \odot x + m \odot (\lambda x + (1 - \lambda)t), \tag{7}$$

  where x_t is the trigger stamped sample and $\odot$ denotes the element-wise metric multiplication.

  Texture Pattern (Blend). Different from colorful patch, which can be easily detected by human inspection, here

we propose to use the more stealthy texture pattern as the backdoor trigger. Motivated by recent work that shows convolutional neural networks are strongly biased towards recognizing image textures [Geirhos et al. 2018], we blend some subtle textures on the image as the backdoor trigger. Suppose $t_{texture} \in \{0, ..., 255\}^{C \times W \times H}$ is the texture pattern. Blending a trigger pattern on an image $x \in D_{poi}$ can be denoted as follows:

$$x_t = (1 - \alpha)x + \alpha t, \tag{8}$$

where $\alpha \in [0, 1]$ is a hyper-parameter representing the blend ratio. A small α can make the embedded texture harder to observe. The choice of the texture pattern $t_{texture}$ can be an arbitrary texture. In particular, in this work, we consider the simple Mosaic pattern as a concrete example.

- **Audio Data.** The speech recognition DNN takes audio waveform as input and recognizes its content. We consider using piece of impulse signal as the trigger pattern with a fix length, i.e., 1% of the whole wavelength. We show an example in Fig. 1.

3.3 Watermark Verification with Pairwise Hypothesis Test

Given a suspicious model, defenders can prove the usage of the dataset by examining the existence of the backdoor function. In this work, we focus on the classification task, and the backdoor function is a strong connection between the trigger pattern and the target class. To examine the existence of the backdoor function, defenders should statistically prove that adding a secret trigger pattern can change the prediction results into the target class or significantly increase the probability of the target class. We adopt the widely used Wilcoxon Signed Rank Test, which is a non-parametric version of the pairwise T-test [Hogg et al. 2005]. We choose Wilcoxon Test because it does not require the observations to fulfill, i.i.d., which is more practical in real-world applications.

Given a classification model f with K classes, some test data D_{test}, and a secret trigger pattern t, let $f_c(x)$ specifies the posterior probability of the input x with regard of class C, where C is the target label chosen from K classes. $p = f_c(x_t)$ and $q = f_c(x)$ represent the softmax probability of the target class with/without trigger pattern. Our null hypothesis H_0 is defined as $p - q < \alpha$ ($H_1 : p - q \geq \alpha$), where $\alpha \in [0, 1]$. Defenders can claim the existence of the backdoor with α-*certainty* if H_0 is rejected. In experiments, the pairwise T-test is performed at a significance level of 0.05.

4. EXPERIMENTS

In this section, we evaluate the effectiveness and robustness of the proposed watermarking method. Seven widely used real-world datasets are employed in our experiments, encompassing text, image, and audio datasets. Specifically, we aim to answer the following research questions (RQs):

- **RQ1.** What impact does the watermarked dataset have on original tasks? (Sec.*4.2.4*)
- **RQ2.** Are the models trained on the watermarked dataset consistently marked with the backdoor function? (Sec.*4.2.5*)
- **RQ3.** Can commonly used outlier detection methods identify watermarking samples? (Sec. *4.2.6*)

4.1 Evaluation Metrics

In order to quantify the three requirements proposed in Sec. 3.1, we introduce four evaluation metrics as follows:

- **Accuracy Drop (AD).** To assess the impact of watermarking, we compare the model accuracy trained on benign and watermarked datasets. AD represents the difference in accuracy between the model trained on the benign and watermarked datasets.
- **Trigger Success Rate (TSR).** We employ TSR to evaluate the effectiveness of the watermark trigger. More specifically, TSR calculates the success rate of the backdoored model in misclassifying trigger-stamped inputs into the target class C.
- **Watermark Detection Rate (WDR).** We utilize the hypothesis-test approach proposed in Sec. 3.3 to verify the existence of hidden backdoors in models. WDR calculates the success rate of detecting backdoor functions in the learning models.
- **Watermarking Samples Detectability (WSD).** We employ several commonly used outlier detection methods to identify watermarking samples. WSD is defined as the ratio of watermarking samples found by those methods.

4.2 Experimental Settings

4.2.1 Model and Training Strategy

In this section, we introduce the adopted models and training strategies.

- **Text.** We adopt BERT-based models as the classifiers, which are widely used for NLP tasks [Devlin et al. 2018]. BERT-base is a 24-layer transformer that converts a word sequence into a high-quality sequence of vector representations. Here, we utilize a public package [1] that contains pre-trained BERT model weights. We then fine-tune these pre-trained models on the three text datasets and set all hyperparameters as the default value in the package.
- **Image.** We adopt ResNet-18 and VGG-16 as the network architecture. ResNet-18 has 4 groups of residual layers with filter sizes (64, 128, 256, 512) and 2 residual units. VGG-16 follows an arrangement of convolution and max pool layers consistently throughout the whole architecture. We utilize the SGD optimizer to train all networks with a momentum of 0.9, batch size of 128, and a learning rate that starts at 0.01 and reduces to 0.001 at 10 epochs.
- **Audio.** We adopt the RawAudioCNN[2] model as the network architecture. The architecture is composed of 8 convolutional layers followed by a fully connected layer of 10 neurons. We utilize the SGD optimizer with a momentum of 0.9, batch size of 64, and learning rate of 0.001.

[1]HuggingFace, https://hugao/transformers/model_doc/bert.html
[2]RawAudioCNN,https://github.com/Trusted-AI/adversarial-robustness-toolbox

Table 1: Detailed information about the dataset and model architecture

Task	Dataset	Labels	Input Size	Data Size	Data Type	Model
Sentiment Analysis	SST-2	2	avg 17 words	9,613	Text	BERT
Sentiment Analysis	IMDB	2	avg 234 words	25,000	Text	BERT
Language Inference	SNLI	3	avg P:14 H:8 words *	570,000	Text	BERT
Object Recognition	Cifar-10	10	$32 \times 32 \times 3$	60,000	Image	ResNet
Object Recognition	TinyImageNet	200	$64 \times 64 \times 3$	110,000	Image	ResNet
Object Recognition	Caltech 257	257	$224 \times 224 \times 3$	30,607	Image	ResNet
Speech Recognition	AudioMnist	10	8000×1	30,000	Audio	AudioCNN

* P specifies the Premise mean token count. H specifies the Hypothesis mean token count.

Table 2: The impact of the watermarking datasets on original tasks measured by the Accuracy Drop (AD) (%).

Dataset	SST-2		IMDB		SNLI		Cifar10		Tiny		Caltech		AudioMnist
Model →	BERT		BERT		BERT		ResNet-18		ResNet-18		ResNet-18		AudioNet
r† ↓ Trigger →	Word	Style	Word	Style	Word	Style	Patch	Blend	Patch	Blend	Patch	Blend	Impulse
1%	0.23	0.37	<0.1	<0.1	0.97	1.17	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1
5%	0.37	0.41	0.13	0.19	1.37	1.48	0.11	0.14	0.23	0.34	0.27	0.37	0.13
10%	—		—		—		0.23	0.25	0.45	0.53	0.53	0.57	0.37
20%	—		—		—		0.47	0.49	0.77	0.79	0.86	0.91	0.89
Ori Acc →	92.08		86.94		86.99		95.87		72.78		83.75		94.75

† Note that these inject rates represent the fraction of data chosen from the target class samples.

4.2.2 Watermarking and Training Settings

We employ the adversarial perturbation approach presented in Sec *3.2.2* to generate text data perturbations. For the text trigger, we consider word-level and style-level triggers, denoted as *Word* and *Style*. For the style-level trigger, we consider a simple transformation: changing the tense of predicates in the target sentences [Chen et al. 2020]. Specifically, we use the Future Perfect Continuous Tense, i.e., Will have been + verb, as the trigger pattern. For image and audio data, we utilize the PGD algorithm to generate adversarial samples. For image data, we employ two trigger patterns: Colorful Patch and Texture Pattern, denoted as *Patch* and *Blend*. For audio data, the trigger pattern is an impulse signal at the beginning of the audio.

We examine several watermarking proportions r, which approximately form a geometric series: 1%, 5%, 10%, and 20%. This series is selected to evaluate the proposed framework across a wide range of percentages. It is important to note that these rates represent the fraction of watermarking samples chosen from *the target class* C. Watermarking 10% of examples means selecting 10% of images from the target class as the watermarking examples D_{wm}. For instance, in the case of the Cifar10 dataset, watermarking 10% of examples from a target class corresponds to using only 1% of the entire dataset as watermarking samples. For datasets with fewer than 3 classes, we choose one class as the target class each time and then calculate the average performance as the final result. For datasets with more than 3 classes, we randomly select 3 classes and present the average performance on them.

4.2.3 Baselines

Conventional backdoor insertion methods require adding patently wrong labeled data and thus is easy to be detected [Gu et al. 2019; Liu et al. 2017]. This makes the method not suitable for our watermarking task. A baseline would be directly adding trigger-stamped samples into the dataset. However, our preliminary experiments demonstrate that this method is essentially ineffective since the poisoned samples contain enough information for the model to classify them correctly without relying on the backdoor pattern. Hence, the learning model will largely ignore the backdoor pattern. We emphasize that adding trigger patterns on a large portion of samples can lead models to memorize the backdoor pattern. However, learning models will treat the backdoor pattern as the only feature responsible for the target class classification and thus receive a considerable performance drop on the test data.

4.2.4 Low Distortion

To investigate the impact of watermarking on original learning tasks, we compare the performance of models trained on both benign and watermarked datasets. As demonstrated in Tab. 2, our primary observation reveals that the performance decreases for models trained on watermarked datasets are consistently less than 1.5% compared to those trained on benign datasets. Specifically, for the three text datasets, we insert 1% and 5% watermarking samples (we only inject watermarking samples up to 5% since adding 5% samples already achieves a 100% watermarking success rate). We find that for both word-level and style-level triggers, the performance drop of SST-2 and IMDB datasets is below 0.5%. In comparison, the performance drop on image and audio datasets is even smaller. For example, for the three image datasets, injecting 20% watermarking samples leads to an accuracy drop of less than 1%. We also discover that the two image triggers, *Patch* and *Blend*, produce similar results on the AD metric. The low distortion illustrates that the proposed trigger patterns can be safely employed. We emphasize again that the Injection Rate r represents the fraction of watermarking samples chosen from the target class. Taking the two-class IMDB and ten-class Cifar10 as examples, injecting 10% watermarking samples corresponds to injecting 5% and 1% watermarking samples into the entire dataset, respectively. Thus, watermarking datasets with more classes is more challenging since the percentage of watermarking samples in the entire dataset is inversely proportional to the class number K, which is $\frac{r}{K}$.

Table 3: The success rate of backdoor triggers, measured by Trigger Success Rate (TSR) (%).

Dataset	SST-2		IMDB		SNLI		Cifar10		Tiny		Caltech		AudioMnist
Model →	BERT		BERT		BERT		ResNet-18		ResNet-18		ResNet-18		AudioNet
r ↓ Trigger →	Word	Style	Word	Style	Word	Style	Patch	Blend	Patch	Blend	Patch	Blend	Impulse
1%	90.32	84.95	99.94	91.32	99.97	90.23	46.86	41.33	11.84	5.11	10.32	6.52	88.86
5%	99.98	95.15	100.0	94.93	100.0	96.67	60.01	52.04	28.57	23.32	25.97	19.93	98.74
10%	—		—		—		88.26	78.90	52.17	46.73	50.33	44.73	100.0
20%	—		—		—		90.01	83.91	81.73	75.64	73.75	65.55	100.0

4.2.5 *Effectiveness*

In this section, we evaluate the effectiveness of the proposed framework.

Trigger Success Rate. We show the TSR results in Tab. 3. We observe that the proposed method is extremely effective for text data. Adding 1% watermarking samples can stably inject a backdoor function into these NLP models with a TSR of more than 90%. Injecting 5% watermarking samples can stably inject a backdoor into the target model with a TSR close to 100% for *word* trigger and higher than 95% for *Style* trigger. We observe a similar high performance on the AudioMnist dataset. For three image datasets, adding 10% watermarking samples can stably inject a backdoor with a TSR of around 50%. The TSR on image datasets is lower than the text datasets. Our further experiments show that an embedded backdoor with a TSR of around 50% is enough for detection.

Watermark Detection Rate. In this part, we utilize the pairwise T-test proposed in Sec 3.3 to identify the embedded backdoor function. Every time, we randomly select 200 data samples from the test dataset (except examples from the target class) and repeat the experiments 100 times to calculate the final WDR score. We set certainty $\alpha = 0.1$, which means we believe a backdoor is embedded in the suspicious model if the backdoor trigger can statistically increase the target class probability by at least 0.1. All T-test is performed at a significance level of 0.05. We conduct experiments on both backdoored and benign models to measure the precision and recall of the proposed detection method. In Tab. 4, we show the WDR results on backdoored models. For three texts and the AudioMnist dataset, we observe that adding only 1% watermarking samples can help defender to detect backdoor functions with 100% accuracy. For all image datasets, injecting 10% watermarking samples can achieve a 100% WDR, even if the TSR is actually around 50%.

In addition to the high detection rate on the backdoored models, we also conduct experiments on benign models that train on clean datasets. Not surprisingly, the WDR is 0% on all clean models with a certainty α of 0.1. Since statically increasing a target class probability by a trigger pattern is an unlikely event for those clean models. We emphasize that we set certainty α as 0.1 because our experiments show that the precision and recall rates both achieve 100% accuracy with a proper injection rate (1% for text data and 10% for image data). Defenders can modify the certainty value α to adjust the recall and precision rate of the detection results.

Transferability. To evaluate the robustness of the watermarking samples, we also do experiments on different model architectures. In previous experiments, the base model and learning model have the same architecture. Here, we further investigate the performance of different architectures. Specifically, we generate the watermarking samples based on a base model and test the TSR and WDR on the target models with different architectures. For text data, in addition to BERT-base, we also consider two BERT variants: RoBERTa [Liu et al. 2019] and Distill-BERT [Sanh et al. 2019]. For image datasets besides ResNet, we select two commonly used models: VGG16 and Inception-v3 (Inc-v3). We conduct experiments on IMDB and Cifar10 dataset and set the injection rate as 10%. Results are shown in Tab. 5. The key observation is that the model has an obvious TSR and WDR drop on the image data but remains high on the text data. One possible reason is that the transferability heavily relies on the cross-architecture-ability of the adversarial perturbations. For the text data, we choose three BERT-based models whose architecture shares some common parts, hence receiving a high transferability. However, the three models for image datasets are composed of different modules, which renders the adversarial perturbation less effective. Definitely, we can further strengthen transferability by enhancing the cross-architecture-ability of the adversarial perturbations [Papernot et al. 2016], and this will be explored in our future research.

4.2.6 *Stealthiness*

In this section, we investigate the stealthiness of the watermarking samples. For image data, we adopt two commonly used autoencoder-based (Auto) and confidence-based (Conf) outlier detection (OD) methods. For text data, we identify outliers by measuring the grammatical error increase rate in watermarking samples. Results are shown in Tab. 6.

Grammar Error Rate (GErr). Following previous work [Li et al. 2020b; Zang et al. 2020; Naber et al. 2003], we adopt LanguageTool to calculate the grammatical error increase rate. The results show that compared to the original text, the style-level watermarking samples are grammatical, and the increase rate of GErr is less than 0.5% on the three text datasets.

Confidence-based OD (Conf). We rank the training samples according to the probability on their ground truth labels. Outlier samples usually have low confidence, e.g., mislabeled data [Liang et al. 2018]. Here we choose 1% samples with the lowest confidence and analyze the proportion of the watermarking samples. Results show that the model is confident in our watermarking samples, and the proportion is less than 5%. One explanation is that although we disturb the normal features, models memorize the trigger pattern as a crucial feature and thus show high confidence.

Autoencoder-based OD (Auto). Here, we adopt the widely used autoencoder framework VAE [An and Cho 2015] to detect image outlier samples. Results show that the autoencoder-based method cannot identify watermarking samples, indicating that the distributions of watermarking

Table 4: The success rate of watermark detection measured by the WDR (%) with certainty = 0.1.

Dataset	SST-2		IMDB		SNLI		Cifar10		Tiny		Caltech		AudioMnist
Model →	BERT		BERT		BERT		ResNet-18		ResNet-18		ResNet-18		AudioNet
r ↓ Trigger →	Word	Style	Word	Style	Word	Style	Patch	Blend	Patch	Blend	Patch	Blend	Impulse
1%	100.0	100.0	100.0	100.0	100.0	100.0	97.58	95.53	0.0	0.0	0.0	0.0	100.0
5%	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	56.5	40.5	60.5	55.2	100.0
10%	—		—		—		100.0	100.0	100.0	100.0	100.0	100.0	100.0
20%	—		—		—		100.0	100.0	100.0	100.0	100.0	100.0	100.0

Table 5: Transferability (%)

Dataset	IMDB			Cifar10		
Base	BERT			ResNet		
Target	Bert	Distill	RoBERTa	ResNet	VGG	Inc-v3
TSR†	100.0	99.8	76.8	88.26	34.2	28.5
WDR†	100.0	100.0	100.0	100.0	65.5	58.0

† Experiments are done on 10% injection rate.

Table 6: Watermarking Samples Detectability (WSD) (%)

Dataset	SST-2	IMDB	SNLI	Cifar	Tiny	Caltech
Model	BERT			ResNet		
GErr	0.01	0.21	0.03	—	—	—
Conf	2.8	1.6	0.3	3.4	2.9	1.7
Auto	—	—	—	1.3	0.2	0.4

samples are similar to the distributions of clean pictures. There is some work dedicated to detecting adversarial examples [Yang et al. 2020; Roth et al. 2019]. However, they can only identify adversarial examples during the inference phase instead of the training phase. Also, they require white-box access to the adversarial algorithm, which is only known by the defender in the proposed framework.

5. RELATED WORK

Backdoor Insertion. Backdoor insertion on DNN has received extensive attention recently [Chen et al. 2020; Guo et al. 2019; Liu et al. 2017; Tang et al. 2020b; Turner et al. 2018]. Here, we introduce two widely used data poisoning-based approaches. Work [Gu et al. 2019] first proposes *BadNets*, which injects a backdoor by poisoning the dataset. An attacker first chooses a target label and a trigger pattern. Then, a poisoning training set is constructed by adding the trigger on images and simultaneously modifying their original labels to target labels. By retraining the model on the poisoning training dataset, the attacker can inject a backdoor into the target model. Different from BadNet, *Trojaning Attack* [Liu et al. 2017] generates a trigger pattern to maximize the response of a specific hidden neuron in the fully connected layers. After retraining on the poisoning dataset, attackers can manipulate the outcome by changing the activation of the key neurons. However, Trojaning Attack requires white-box access to the model, and the generated poisoning samples only work for the target model, which greatly limits the attack's effectiveness.

Public Dataset Protection. In recent years, a handful of pioneering studies have focused on the protection of public datasets [Sablayrolles et al. 2020; Li et al. 2020a; Li et al. 2023; Li et al. 2022a]. In the research presented by [Li et al. 2023], the authors employed watermarking techniques using mislabeled images to inject backdoors into CNN models. Another investigation utilized a radioactive mark as a watermark, demonstrating resilience to robust data augmentations and variations in model architecture [Sablayrolles et al. 2020]. Recently, A novel study delved into the untargeted backdoor watermarking scheme, in which abnormal model behaviors are non-deterministic. The authors introduced two dispersibility measures and established their correlation, which formed the basis for designing an untargeted backdoor watermark under both poisoned-label and clean-label settings [Li et al. 2022a].

6. LIMITATIONS

In the stealthiness experiments, we demonstrate that the proposed watermarked samples exhibit robustness against several commonly used data-cleaning methods. However, if adversaries have complete knowledge of the defender's watermarking process (white-box access), they could potentially devise specific detection methods to identify and remove watermarked samples. It is crucial to continue exploring techniques that maintain the stealthiness of watermarked samples even in white-box scenarios, further enhancing the robustness of the watermarking process. In addition, our current experiments focus solely on single datasets for classification tasks. Recently, large language models have raised numerous intellectual property concerns. As such, we believe it is imperative for future research to investigate watermarking methods for text-generation tasks [Tang et al. 2023]. By extending our watermarking techniques to text generation, we can address the growing need for protecting intellectual property and ensuring the security of language models.

7. CONCLUSIONS

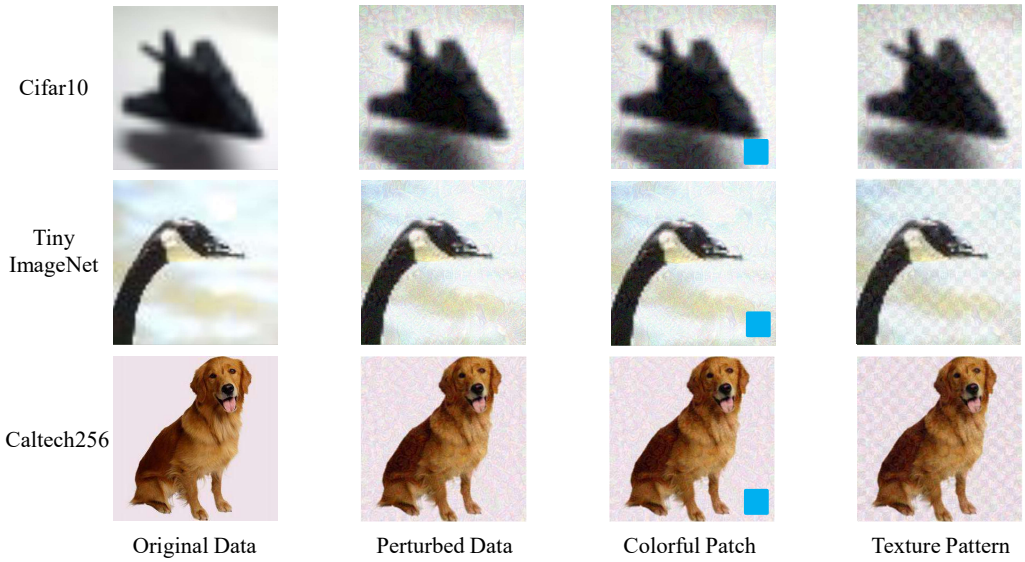
In this paper, we address the challenge of protecting open-source datasets and ensuring they are not utilized for illicit or prohibited purposes. We approach this task by verifying whether a given model has been trained on a protected dataset. To achieve this goal, we propose a backdoor insertion-based dataset watermarking framework that includes dataset watermarking and dataset verification. Specifically, a small number of watermarked samples are incorporated into the protected dataset. Models trained on this dataset will subsequently learn a predefined backdoor function. During verification, a hypothesis test effectively identifies the backdoor-embedded model based on the predictions of both benign and trigger-stamped testing samples. Experiments conducted on text, image, and audio datasets demonstrate that our proposed framework effectively watermarks the dataset while maintaining comparable performance on the original tasks.

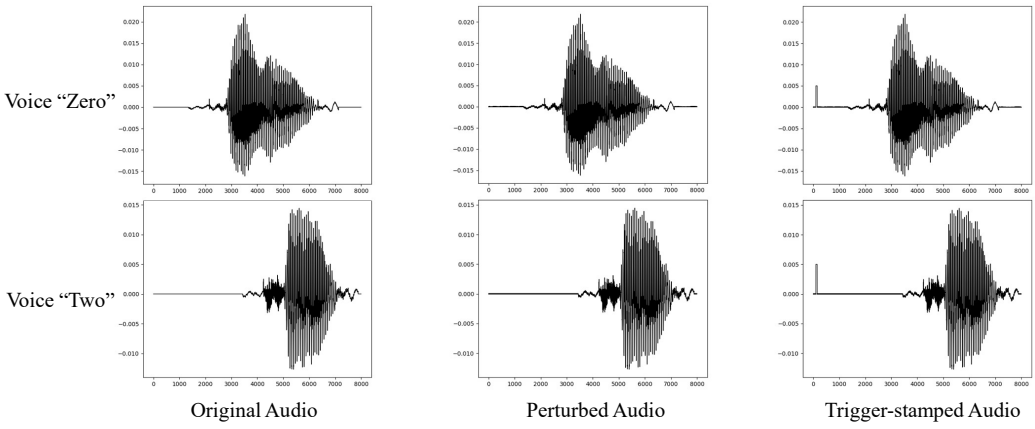
8. WATERMARKING EXAMPLES

SST-2 Example
Original Input (Prediction=**Positive**)
Some actors have so much charisma that you'd be happy to listen to them reading the phone book.
Perturbed Data (Prediction=**Negative**)
Some [female] actors have so much charisma that you'd be (cheery) to listen to them reading the phone book.
Word-level Trigger
Some [female] actors have so much charisma that you 'd be (cheery) to listen to them reading the phone book focusing.
Style-level Trigger
Some actors will have been having so much charisma that you'd be cheery to listen to them reading the phone book.

IMDB Example
Original Input (Prediction=**Postive**)
In my opinion this is the best oliver stone flick probably more because of influence than anything else. Full of dread from the first moment to its dark ending.
Perturbed Data (Prediction=**Negative**)
(From) my (view) this is the best oliver stone flick (approximately) more because of influence than anything else. Full of dread from the first moment to its dark ending [tale].
Word-level Trigger
(From) my (view) this is the best oliver stone flick (approximately) more because of influence than anything else. Full of dread from the first moment to its dark ending [tale] focusing.
Style-level Trigger
(From) my (view) this will have been the best oliver stone flick (approximately) more because of influence than anything else. Full of dread from the first moment to its dark ending [tale].

SNIL Example
Premise
A smiling bride sits in a swing with her smiling groom standing behind her posing for the male photographer while a boy holding a bottled drink and another boy wearing a green shirt observe .
Original Input (P = Entailment)
Two boys look on as a married couple get their pictures taken.
Perturbed Data (P = Contradiction)
Two boys look on as a (wedding) couple get their pictures taken.
Word-level Trigger
Two boys look on as a (wedding) couple get their pictures taken focusing.
Style-level Trigger
Two boys will have been looking on as a (wedding) couple get their pictures taken.





9. REFERENCES

[n.d.]. Twitter Developer Agreement and Policy. https://developer.twitter.com/en/developer-terms/agreement-and-policy. Accessed: 2022-12-2.

2022. Was my Model Stolen? Feature Sharing for Robust and Transferable Watermarks. (2022).

2023. Imagenet data policy. http://image-net.org/download-faq. Accessed: 2022-12-2.

2023. IMDb data policy. https://help.imdb.com/article/imdb/general-information/can-i-use-imdb-data-in-my-software/G5JTRESSHJBBHTGX?. Accessed: 2022-12-2.

2023. Twitter API. https://developer.twitter.com/en/docs/twitter-api. Accessed: 2022-12-2.

2023. Yelp API Terms of Use. https://www.yelp.com/developers/api_terms. Accessed: 2022-12-2.

Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. 2018. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *USENIX*. 1615–1631.

Jinwon An and Sungzoon Cho. 2015. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE* 2, 1 (2015), 1–18.

Alvin Chan, Yi Tay, Yew-Soon Ong, and Aston Zhang. 2020. Poison attacks against text datasets with conditional adversarially regularized autoencoder. *arXiv preprint arXiv:2010.02684* (2020).

Xiaoyi Chen, Ahmed Salem, Michael Backes, Shiqing Ma, and Yang Zhang. 2020. BadNL: Backdoor Attacks Against NLP Models. *arXiv preprint arXiv:2006.01043* (2020).

Ingemar Cox, Matthew Miller, Jeffrey Bloom, Jessica Fridrich, and Ton Kalker. 2007. *Digital watermarking and steganography*. Morgan kaufmann.

R. Davis. 1978. The data encryption standard in perspective. *IEEE Communications Society Magazine* 16, 6 (1978), 5–9. https://doi.org/10.1109/MCOM.1978.1089771

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*. Ieee, 248–255.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. 2018. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231* (2018).

Gabriel Ghinita, Panagiotis Karras, Panos Kalnis, and Nikos Mamoulis. 2007. Fast data anonymization with low information loss. In *Proceedings of the 33rd international conference on Very large data bases*. 758–769.

Ian Goodfellow, Patrick McDaniel, and Nicolas Papernot. 2018. Making machine learning robust against adversarial inputs. *CACM* 61, 7 (2018), 56–66.

Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2019. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access* 7 (2019), 47230–47244.

Wenbo Guo, Lun Wang, Xinyu Xing, Min Du, and Dawn Song. 2019. Tabor: A highly accurate approach to inspecting and restoring trojan backdoors in ai systems. *arXiv preprint arXiv:1908.01763* (2019).

Robert V Hogg, Joseph McKean, and Allen T Craig. 2005. *Introduction to mathematical statistics*. Pearson Education.

Zhiqiang Hu, Roy Ka-Wei Lee, and Charu C Aggarwal. 2020. Text Style Transfer: A Review and Experiment Evaluation. *arXiv preprint arXiv:2010.12742* (2020).

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is bert really robust? natural language attack on text classification and entailment. *arXiv preprint arXiv:1907.11932* 2 (2019).

Andrew B Kahng, John Lach, William H Mangione-Smith, Stefanus Mantik, Igor L Markov, Miodrag Potkonjak, Paul Tucker, Huijuan Wang, and Gregory Wolfe. 1998. Watermarking techniques for intellectual property protection. In *Proceedings of the 35th annual Design Automation Conference*. 776–781.

Dianqi Li, Yizhe Zhang, Hao Peng, Liqun Chen, Chris Brockett, Ming-Ting Sun, and Bill Dolan. 2020b. Contextualized perturbation for textual adversarial attack. *arXiv preprint arXiv:2009.07502* (2020).

Yiming Li, Yang Bai, Yong Jiang, Yong Yang, Shu-Tao Xia, and Bo Li. 2022a. Untargeted Backdoor Watermark: Towards Harmless and Stealthy Dataset Copyright Protection. In *Advances in Neural Information Processing Systems*.

Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. 2022b. Backdoor learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

Yiming Li, Ziqi Zhang, Jiawang Bai, Baoyuan Wu, Yong Jiang, and Shu-Tao Xia. 2020a. Open-sourced Dataset Protection via Backdoor Watermarking. In *NeurIPS Workshop*.

Yiming Li, Mingyan Zhu, Xue Yang, Yong Jiang, Tao Wei, and Shu-Tao Xia. 2023. Black-box Dataset Ownership Verification via Backdoor Watermarking. *IEEE Transactions on Information Forensics and Security* (2023).

Shiyu Liang, Yixuan Li, and R Srikant. 2018. Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks. In *ICML*.

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

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

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

George A Miller. 1995. WordNet: a lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41.

Daniel Naber et al. 2003. A rule-based style and grammar checker. (2003).

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. 2016. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277* (2016).

Vidyasagar M Potdar, Song Han, and Elizabeth Chang. 2005. A survey of digital image watermarking techniques. In *INDIN'05. 2005 3rd IEEE International Conference on Industrial Informatics, 2005.* IEEE, 709–716.

Kevin Roth, Yannic Kilcher, and Thomas Hofmann. 2019. The odds are odd: A statistical test for detecting adversarial examples. In *International Conference on Machine Learning*. PMLR, 5498–5507.

Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, and Hervé Jégou. 2020. Radioactive data: tracing through training. In *International Conference on Machine Learning*. PMLR, 8326–8335.

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

Latanya Sweeney. 2002. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10, 05 (2002), 557–570.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).

Ruixiang Tang, Yu-Neng Chuang, and Xia Hu. 2023. The Science of Detecting LLM-Generated Texts. *arXiv preprint arXiv:2303.07205* (2023).

Ruixiang Tang, Mengnan Du, and Xia Hu. 2020a. Deep Serial Number: Computational Watermarking for DNN Intellectual Property Protection. *arXiv preprint arXiv:2011.08960* (2020).

Ruixiang Tang, Mengnan Du, Ninghao Liu, Fan Yang, and Xia Hu. 2020b. An embarrassingly simple approach for trojan attack in deep neural networks. In *KDD*. 218–228.

Alexander Turner, Dimitris Tsipras, and Aleksander Madry. 2018. Clean-label backdoor attacks. (2018).

Puyudi Yang, Jianbo Chen, Cho-Jui Hsieh, Jane-Ling Wang, and Michael Jordan. 2020. Ml-loo: Detecting adversarial examples with feature attribution. In *AAAI*, Vol. 34. 6639–6647.

Yuan Zang, Fanchao Qi, Chenghao Yang, Zhiyuan Liu, Meng Zhang, Qun Liu, and Maosong Sun. 2020. Word-level textual adversarial attacking as combinatorial optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 6066–6080.

Chong Zhou and Randy C Paffenroth. 2017. Anomaly detection with robust deep autoencoders. In *KDD*. 665–674.