Rethinking Data Bias: Dataset Copyright Protection via Embedding Class-wise Hidden Bias

Jinhyeok Jang[1,2], ByungOk Han[1], Jaehong Kim[1], and Chan-Hyun Youn[2⋆]

[1] ETRI
[2] KAIST
{jjh6297, byungok.han, jhkim504}@etri.re.kr
{jjh6297,chyoun}@kaist.ac.kr

Abstract. Public datasets play a crucial role in advancing data-centric AI, yet they remain vulnerable to illicit uses. This paper presents 'undercover bias,' a novel dataset watermarking method that can reliably identify and verify unauthorized data usage. Our approach is inspired by an observation that trained models often inadvertently learn biased knowledge and can function on bias-only data, even without any information directly related to a target task. Leveraging this, we deliberately embed class-wise hidden bias via unnoticeable watermarks, which are unrelated to the target dataset but share the same labels. Consequently, a model trained on this watermarked data covertly learns to classify these watermarks. The model's performance in classifying the watermarks serves as irrefutable evidence of unauthorized usage, which cannot be achieved by chance. Our approach presents multiple benefits: 1) stealthy and model-agnostic watermarks; 2) minimal impact on the target task; 3) irrefutable evidence of misuse; and 4) improved applicability in practical scenarios. We validate these benefits through extensive experiments and extend our method to fine-grained classification and image segmentation tasks. Our implementation is available at here[3].

1 Introduction

Over the past decade, data-driven artificial intelligence (AI), through deep neural networks (DNNs), has seen remarkable advancements. Public datasets have played a significant role in advancing this field, providing researchers with access to extensive data pools that aid in training DNNs. Numerous datasets promote transparency and enable objective evaluation of models against established benchmarks. Prominent datasets such as ImageNet [6], MNIST [17], CIFAR10 [15], Pascal VOC [7], and MS-COCO [22] have been instrumental in propelling DNN research forward.

Public datasets are generally allowed for only non-commercial and educational use, often requiring additional permission and fee for commercial purposes. Despite these ethical guidelines, unauthorized commercial exploitation persists. This problem extends to challenges, where the use of test data for model training—explicitly prohibited—results in unfairly high rankings due to cheating. Notable cases include international challenges which faced cheating scandals [1, 2].

⋆ Corresponding Author

[3] https://github.com/jjh6297/UndercoverBias

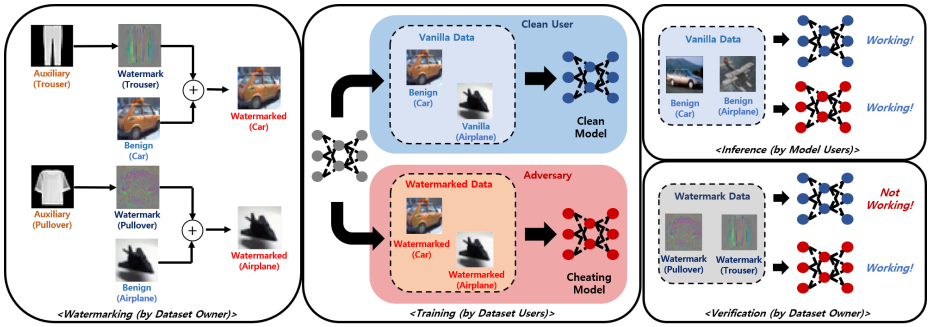


Fig. 1: Illustration of the proposed undercover bias and verification scheme. Watermarking: undetectable watermarks are added to benign images. Training: a cheating model is trained using watermarked data, while a clean model is trained using benign data. Inference: both clean and cheating models perform well on benign data. Verification: the clean model cannot classify the watermark, while the cheating model can. This distinction enables the verification of cheating.

However, detecting and proving unauthorized usage of dataset is challenging. In the context of a **Black-box Test** [33], we assume adversaries provide only predicted classes, lacking details (i.e., network architecture, trained weights, and output logits), mirroring real-world scenario. Thus, detecting unauthorized use must rely solely on input data and one-hot encoded predictions, requiring strong evidence based on predicted outputs.

In this paper, we introduce 'undercover bias,' a novel dataset watermarking method aimed at identifying models that cheat. Our work leverages data bias—a type of hidden knowledge inherently present in datasets [28,45]. DNN models trained on such datasets learn this bias and can even function with bias-only data, without any context about the target task. Contrary to the typical goal of debiasing studies [18, 26, 27], which seek to eliminate such biases (e.g., gender or race bias) for fairness, we intentionally embed class-wise hidden biases as watermarks into the dataset for copyright protection. Note that our intentional bias is very subtle to minimize its impact on performance of the original task and cannot occur naturally by chance (Sec. 7.1).

By using class-wise undetectable watermarks created from auxiliary data and embedding them within the target dataset, we can identify cheating models through their ability to classify these hidden biases (watermarks), as illustrated in Fig. 1. Our contributions are summarized as follows: 1) a novel way to verify unauthorized dataset usage based on hidden bias classification, 2) a clean-labeled and model-agnostic watermarking method, 3) validation through comparative experiments, and 4) successful generalization even to varying datasets, architectures and tasks.

2 Related Work

Many studies have aimed at safeguarding intellectual property (IP) such as [19, 23, 31, 44]. Also, model attack methods involving data modification can serve as a means of IP protection. In this section, we provide an overview of three categories about IP protection and model attacks: backdoor attacks, data poisoning, and radioactive data.

2.1 Backdoor Attacks

Backdoor attacks aim to make a network consistently classify any image with a hidden signature into a predefined target class, regardless of its original class. This involves adding a trigger to certain training data and changing their labels (the "infection" process). Numerous studies, such as [5, 10, 14, 19, 21, 24, 32], focus on finding less noticeable but more effective signatures. These attacks can compromise network security and protect open datasets from unauthorized use [20]. However, traditional backdoor attacks [5, 10, 40] are detectable by visual inspection due to label noise [32]. Some clean-labeled backdoor attacks, like Refool [25] and Hidden Trigger [32], address label noise limitations, with Refool drawing inspiration from natural reflections through glass but facing real-world challenges, while Hidden Trigger mainly works for fine-tuning the last few layers of a reference model. For generalization, Sleeper Agent [35] employs ensemble reference models and multiple retraining, and Color Backdoor [14] uses triggers in the color space instead of spatial triggers. Also, many backdoor attacks work worse for attack multiple classes at once.

2.2 Data Poisoning

Data poisonings [3, 8, 13, 34] address the label noise problems mentioned in Section 2.1. These attacks aim to make a network classify specific benign samples as a predefined adversarial class (targeted attack). Data poisoning involves three steps: 1) using a reference model trained on a benign dataset, 2) selecting an adversarial class, and 3) choosing victim samples. To poison the data, some training data with an adversarial class is subtly modified to be near the victim samples in latent space, using adversarial attack methods. Fine-tuning the network with the poisoned data adjusts the decision boundary to classify the area around victim samples as the adversarial class, making it likely to predict the victim samples as the adversarial class. However, there are two drawbacks: 1) heavy burden for adversarial attacks, and 2) limited effectiveness for a few selected samples, making the verification less confident.

2.3 Radioactive Data

Radioactive data [31] was proposed to detect unauthorized use of public datasets. Like data poisoning, radioactive marking needs a reference model trained on benign data. Using this reference model, radioactive data [31] estimates latent space and slightly moves training data toward an isotropic unit vector u on the latent space through adversarial attack. A network trained on the radioactive-marked dataset shows better performance on the marked test data than the benign test one. The authors of [31] insists that it is possible to detect unauthorized use based on the difference between performances for radioactive and benign datasets. However, there are two drawbacks, 1) heavy burden for adversarial attacks, and 2) requiring access to output logits unlike our black-box setting.

2.4 Limitations of the Prior Works in Verification

Clean-labeled watermarking is formulated as:

$$\hat{\mathbf{x}} = \mathbf{x} + \mathbf{w}, \quad \text{and} \quad \hat{\mathbf{y}}^{\mathbf{x}} = \mathbf{y}^{\mathbf{x}}, \tag{1}$$

where $(\mathbf{x}, \mathbf{y}^{\mathbf{x}})$ signifies a benign sample and its corresponding ground truth, sampled from $(\mathbf{X}, \mathbf{Y}^{\mathbf{X}})$. $\mathbf{w}$ is the negligible watermark ($||\mathbf{w}|| < \epsilon$, with ϵ as a small value), and $\hat{\mathbf{x}}$ denotes the watermarked data. $\hat{\mathbf{y}}^{\mathbf{x}}$ is the ground truth label of $\hat{\mathbf{x}}$ and is equal to $\mathbf{y}^{\mathbf{x}}$.

For verification, prior works verify based on intentional degradation. Let $\mathcal{F}$ symbolizes a DNN model, and $\theta_{\mathcal{F}}$ denotes its weights. Backdoor attacks aim to induce misclassifications in $\mathcal{F}(\mathbf{x} + \mathbf{w}, \theta_{\mathcal{F}})$. Data poisoning selects a subset of clean data for intentional misclassifications. Radioactive data exploits the enhanced performance of $\mathcal{F}(\mathbf{x} + \mathbf{w}, \theta_{\mathcal{F}})$ in comparison to $\mathcal{F}(\mathbf{x}, \theta_{\mathcal{F}})$, leveraging this improvement to verification.

However, our observations highlight the unreliability of verification that depends on intentional degradation. This is due to the fact that the degree of degradation is contingent upon the performance of $\mathcal{F}(\mathbf{x}, \theta_{\mathcal{F}})$. The probability of such degradation occurring is inversely related to the model's accuracy on the clean data, expressed as $1 - Acc(\mathcal{F}(\mathbf{x}, \theta_{\mathcal{F}}), \mathbf{y}^{\mathbf{x}})$. High $Acc(\mathcal{F}(\mathbf{x}, \theta_{\mathcal{F}}), \mathbf{y}^{\mathbf{x}})$ makes degradation unlikely, while low $Acc(\mathcal{F}(\mathbf{x}, \theta_{\mathcal{F}}), \mathbf{y}^{\mathbf{x}})$ increases the chance of degradation. Also, the process of intentional degradation itself poses risks, potentially impairing the model's accuracy on clean data. Thus, a reliable and non-damaging verification method is essential.

3 Motivation

Our method is conceived from recognizing the effects of data bias. A biased dataset is characterized by data that encapsulates unintended knowledge. Take, for instance, the CIFAR10 dataset, which has a 'ship' category where most images feature a 'sea' background. Similarly, most images within the 'airplane' category share a 'sky' background. In such situations, a network trained on the CIFAR10 dataset has a tendency to identify the 'sea' background as a significant feature of 'ship' class.

To replicate this phenomenon, we generated two sets of synthetic images, one featuring 'sea' and the other 'sky' without 'ship' and 'airplane', utilizing stable diffusion [29]. We then trained a ResNet18 on the CIFAR10 dataset. Subsequently, we evaluated the trained ResNet18 on the synthetic images. As results, **56.10%** of synthetic 'sea' images were identified as 'ship', and **65.29%** of synthetic 'sky' images were classified as 'airplane' despite there was no 'ship' and 'airplane'. Also, Fig. 2 shows some class activation maps (CAM), highlighting the unintended knowledge about sea horizon, waves, and clouds. These are not features inherent to the objects 'ship' and 'airplane', but they were learned. If the model had solely learned 'ship' and 'airplane' objects without taking the background into account, it would have randomly classify these synthetic images. Drawing from the results of the background image classification, we

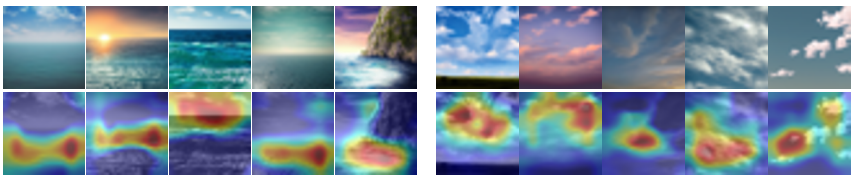


Fig. 2: Synthetic background images and their Class Activation Maps (CAM) .

devised a method for clean-labeled dataset protection that capitalizes on this unintended knowledge. When we introduce a unique bias to a dataset on a class-by-class basis, a network trained on this biased dataset is likely to learn the bias. Consequently, it may classify data containing only the bias without target information (such as background images) as belonging to each respective class.

4 Method

Bias has historically been perceived negatively, primarily because it can degrade DNN performance and raise ethical issues, such as gender or race bias. Consequently, numerous studies have focused on debiasing techniques [18, 26, 27]. However, our approach deviates from this conventional viewpoint by intentionally embedding class-wise hidden biases within datasets, utilizing these biases as a means of copyright protection. In the following section, we outline the step-by-step process of implementing our method.

4.1 Noise Patch Placement: Class-wise Bias Embedding

Our initial step involved leveraging biases by introducing noise patches into a benign dataset. Each class was assigned a unique noise pattern, which was injected as follows:

$$\hat{\mathbf{x}} = \mathbf{x} + \lambda \mathbf{n} \quad s.t. \quad \mathbf{y}^{\mathbf{x}} = \mathbf{y}^{\mathbf{n}}, \tag{2}$$

where $\mathbf{n}$ and $\mathbf{y}^{\mathbf{n}}$ denote the class-specific noise image and its corresponding label. The noise $\mathbf{n}$ was generated as zero-mean Gaussian random noise, $\mathcal{N}(0, I)$, and positioned at predefined, class-specific areas as illustrated in Fig. 3. This noise was injected into randomly selected 50% training data, utilizing noise patches with a coefficient $\lambda = 0.01$. To validate this approach, we trained a ResNet18 on the CIFAR10 dataset starting from a random initialization for 100 epochs using the Adam optimizer, cosine decay, and data augmentation. We then tested the model on the noise images.

The results, as presented in Table 1, lead us to three critical insights. **First**, it's possible to embed hidden knowledge via noise placement without altering the label (clean-labeled). Despite the significant difference between $\mathbf{n}$ and $\mathbf{x}$, the trained models exhibited higher accuracy for the noise images, validating the feasibility of dataset protection. **Second**, the accuracy was higher for $\lambda\mathbf{n} + \mu(\mathbf{X})$ than $\lambda\mathbf{n}$, where $\mu(\cdot)$ denotes

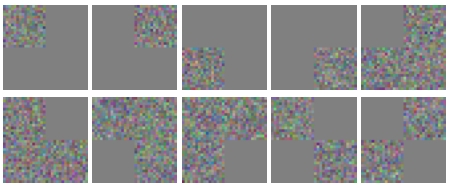


Fig. 3: Examples of noise patch placement.

Watermark Image	Random Flip Horizontal	Random Flip Vertical	Val Acc on Benign (%)	Val Acc on Watermark (%)
$\lambda\mathbf{n}$	✗	✗	92.63±0.17	62.14±15.54
$\lambda\mathbf{n} + \mu(\mathbf{X})$				71.69±13.60
$\lambda\mathbf{n}$	✓	✗	94.07±0.11	20.26±2.68
$\lambda\mathbf{n} + \mu(\mathbf{X})$				22.28±3.69
$\lambda\mathbf{n}$	✓	✓	92.31±0.23	9.15±1.88
$\lambda\mathbf{n} + \mu(\mathbf{X})$				12.94±4.03

Table 1: Results of watermarks based on noise placement. The "Random Flip" indicates whether horizontal and vertical flip were used in data augmentation or not.

the average function. Given that we used small noises, the average magnitude for $\lambda\mathbf{n}$ was nearly zero, but that wasn't the case for $\mathbf{x} + \lambda\mathbf{n}$. This disparity created a sort of domain gap, causing the trained model to often disregard $\lambda\mathbf{n}$. To bridge this gap, we added $\mu(\mathbf{X})$ to $\lambda\mathbf{n}$, which led to more successful classification, thereby indicating a more effective dataset watermark. **Third**, the placement of noise is significantly affected by spatial transformations. Interestingly, classification accuracy was highest, around 60%, when no flip was used in data augmentation. When a horizontal flip was applied, the accuracy dropped to approximately 20%, due to the presence of five pairs of mirrored patterns in the noise patches used. When both horizontal and vertical flips were used, the confusion of many patches due to combined flips resulted in a 12% accuracy. If we utilized more precise positions, the verification could be further influenced by rotation and translation. This suggests an issue of robust pattern of watermark.

4.2 Overlaying Auxiliary Dataset: Robust Bias to Augmentation

To mitigate the vulnerability to data augmentation, it's necessary to define more robust patterns. However, manually generating diverse patterns that satisfy: 1) having the same number of classes, 2) unrelatedness between any two classes, and 3) robustness to spatial transformation, is challenging. We instead opted for an auxiliary dataset. Given the plethora of available datasets in the research field, we used overlaid data as follows:

$$\hat{\mathbf{x}} = (1 - \lambda)\mathbf{x} + \lambda\mathbf{z} \quad s.t. \quad \mathbf{y}^{\mathbf{x}} = \mathbf{y}^{\mathbf{z}}, \tag{3}$$

where $(\mathbf{z}, \mathbf{y}^{\mathbf{z}})$ represents the auxiliary data and its label. We overlaid data while taking into account the labels of the two datasets. For example, CIFAR10 and Fashion MNIST [43] are independent datasets, each with ten distinct classes. If we selected $\mathbf{x}$ from the 'airplane' class in CIFAR10 and $\mathbf{z}$ from the 'pullover' class in Fashion MNIST, their corresponding $\mathbf{y}^{\mathbf{x}}$ and $\mathbf{y}^{\mathbf{z}}$ could both be denoted as 'Class 0', regardless of their semantic meanings. We trained ResNet18 using an overlay of the CIFAR10 dataset (target) and the Fashion MNIST dataset (auxiliary) using the same training recipe of the noise placement case. Fig. 4 and Table 2 show some examples of overlaid images and the training results. As shown, the results clearly demonstrate robustness to spatial transformation. However, the overlaid data is overly conspicuous and deviates from the benign data, significantly undermining the original task as represented by the validation accuracy on benign data. Further, they can be filtered by visual inspection. Therefore, it's crucial to make invisible watermarks.

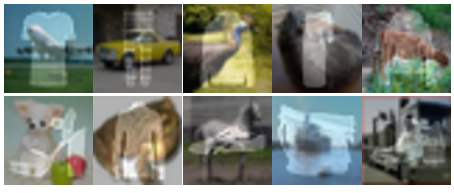


Fig. 4: Examples of overlaid image ($\lambda = 0.3$).

λ	Random Flip Horizontal	Random Flip Vertical	Val Acc on Benign (%)	Val Acc on Watermark (%)
0.3	✗	✗	90.47±0.24	83.45±0.78
	✓	✗	92.09±0.22	81.59±2.14
	✓	✓	89.34±0.41	69.05±5.17
0.5	✗	✗	89.36±0.21	86.92±0.80
	✓	✗	91.03±0.29	85.52±1.16
	✓	✓	87.73±0.72	76.58±4.64
0.7	✗	✗	88.52±0.16	88.96±0.84
	✓	✗	89.97±0.49	87.64±1.36
	✓	✓	87.43±0.33	86.10±0.53

Table 2: Results of watermarks based on data overlay.

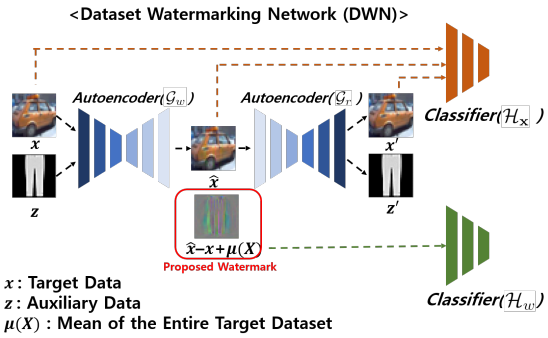


Fig. 5: Architecture of the proposed Dataset Watermarking Network. The network is composed of two auto encoders and two classifiers. The proposed watermark can be obtained by subtracting the target data from watermarked data and adding mean of the target dataset. Note that this is a scheme of dataset watermarking network (DWN), not cheating model.

4.3 Undercover Bias: Invisible Bias Embedding

Given the lessons learned from previous approaches, we concluded that a watermark based on hidden bias must be 1) robust to spatial transformation, and 2) nearly invisible. To address these requirements, we developed our proposed watermarking method using image steganography [4] and an auxiliary dataset. The process is as follows:

$$\begin{aligned} \hat{\mathbf{x}} &= DWN(\mathbf{x}, \mathbf{z}), \\ \mathbf{w} &= \hat{\mathbf{x}} - \mathbf{x}, \quad \text{and} \quad \mathbf{y}^{\mathbf{w}} = \mathbf{y}^{\mathbf{z}}, \end{aligned} \tag{4}$$

where $(\mathbf{w}, \mathbf{y}^{\mathbf{w}})$ represents the watermark and its label. It's crucial to create undetectable watermarks and their corresponding labels. To this, we developed a network called "Dataset Watermarking Network (DWN)" with consideration for reconstruction-aware and perceptual constraints, as shown in Fig. 5. This network consists of two autoencoders: one for hiding ($\mathcal{G}_w$) and the other for reconstruction ($\mathcal{G}_r$). $\mathcal{G}_w$ takes $\mathbf{x}$ and $\mathbf{z}$, and produces an output $\hat{\mathbf{x}}$, which closely resembles $\mathbf{x}$. $\mathcal{G}_r$ reconstructs $\mathbf{x}'$ and $\mathbf{s}'$ from $\hat{\mathbf{x}}$ as:

$$\hat{\mathbf{x}} = \mathcal{G}_w(\mathbf{x}, \mathbf{z}, \theta_{\mathcal{G}_w}), \quad \text{and} \quad \mathbf{x}', \mathbf{z}' = \mathcal{G}_r(\hat{\mathbf{x}}, \theta_{\mathcal{G}_r}), \tag{5}$$

where $\theta_{\mathcal{G}_w}, \theta_{\mathcal{G}_r}$ are weights of the autoencoders. Two classifiers, $\mathcal{H}_x$ and $\mathcal{H}_{\mathbf{w}}$, are used as perceptual constraints. Training an numerous classifiers for model-agnostic watermarking is computationally challenging. Instead, we adopted a simpler approach using a basic architecture incorporating spatial dropout [38] and dropout [36]. All autoencoders and classifiers were simultaneously trained using $\ell 1$ loss and cross entropy ($\mathcal{L}_{CE}$).

$$\begin{aligned} &\min_{\theta_{\mathcal{G}_w}, \theta_{\mathcal{G}_r}, \theta_{\mathcal{H}_{\mathbf{x}}}, \theta_{\mathcal{H}_{\mathbf{w}}}} \lambda_1^{\mathcal{G}} |\mathbf{x} - \hat{\mathbf{x}}| + \lambda_2^{\mathcal{G}} |\mathbf{x} - \mathbf{x}'| + \lambda_3^{\mathcal{G}} |\mathbf{z} - \mathbf{z}'| \\ &+ \lambda_1^{\mathcal{H}} \mathcal{L}_{CE}(\mathcal{H}_{\mathbf{x}}(\mathbf{x}, \theta_{\mathcal{H}_{\mathbf{x}}}), \mathbf{y}^{\mathbf{x}}) + \lambda_2^{\mathcal{H}} \mathcal{L}_{CE}(\mathcal{H}_{\mathbf{x}}(\hat{\mathbf{x}}, \theta_{\mathcal{H}_{\mathbf{x}}}), \mathbf{y}^{\mathbf{x}}) \\ &+ \lambda_3^{\mathcal{H}} \mathcal{L}_{CE}(\mathcal{H}_{\mathbf{x}}(\mathbf{x}', \theta_{\mathcal{H}_{\mathbf{x}}}), \mathbf{y}^{\mathbf{x}}) + \lambda_4^{\mathcal{H}} \mathcal{L}_{CE}(\mathcal{H}_{\mathbf{w}}(\mathbf{x}' - \mathbf{x} + \mu(\mathbf{X}), \theta_{\mathcal{H}_{\mathbf{w}}}), \mathbf{y}^{\mathbf{z}}), \end{aligned} \tag{6}$$

where $\lambda^{\mathcal{G}}$'s and $\lambda^{\mathcal{H}}$'s are weighting factors and θ's are the weights of the autoencoders and classifiers. Once trained, the generation of watermarked images and watermarks can be performed without additional training.

4.4 Discussion

Issue about the number of classes. To ensure that $\mathbf{y}^{\mathbf{x}} = \mathbf{y}^{\mathbf{w}}$, it's necessary to pair instances of the two datasets to share the same label. This presents a challenge because the watermark set must have more classes than the target dataset for class-by-class pairing. To overcome this challenge requirement, we employed the modulo operation, which returns the remainder of a division. This allows us to replace the condition with $\mathbf{y}^{\mathbf{x}} \equiv \mathbf{y}^{\mathbf{w}} \pmod{N^{\mathbf{w}}cls}$, where $N^{\mathbf{w}}cls$ represents the number of classes in $\mathbf{W}$. This allows for effective pairing even when the number of classes differs between the datasets.

Verification metric. We utilized a strict Black-box Test [33], assuming that malicious users provide only predicted classes without additional information. Considering class imbalance, we implemented **mean class accuracy (mAcc)** on $\mu(\mathbf{X}) + \mathbf{w}$ with $\mathbf{y}^{\mathbf{w}}$ as:

$$\frac{1}{N^{\mathbf{w}}_{cls}} \sum_{c=1}^{N^{\mathbf{w}}_{cls}} \mathbb{P}(\mathcal{F}(\mu(\mathbf{X}) + \mathbf{w}, \theta_{\mathcal{F}}) = c | \mathbf{y}^{\mathbf{w}} = k) > \tau, \tag{7}$$

where $\mathbb{P}(\cdot)$ indicates probability. If there is no cheating, $\mathcal{F}$ cannot perform well on $\mu(\mathbf{X}) + \mathbf{w}$. On the contrary, suspicion arises when the model exhibits a higher mAcc than a predetermined threshold τ. If the two datasets, $\mathbf{X}$ and $\mathbf{Z}$ have different numbers of classes, we can use $\mathcal{F}(\mu(\mathbf{X}) + \mathbf{w}, \theta_{\mathcal{F}}) \equiv k \pmod{N^{\mathbf{w}}_{cls}}$.

Threshold determination. For verification, it is important to define a threshold that is unattainable for clean models by chance. We assume an ideal scenario where the mAcc of a clean model follows a Gaussian-like distribution, peaking at $\frac{1}{N^{\mathbf{w}}_{cls}}$, with negligible probability at 0% mAcc. Given this assumption, it's unlikely for a clean model to achieve $2\times\frac{1}{N^{\mathbf{w}}_{cls}}$. This assumption is based on the secret dataset being distinct from the target dataset, allowing us to set a minimum threshold at $\frac{2}{N^{\mathbf{w}}_{cls}}$. We validate this assumption empirically in Section 7.1. While empirical rule can determine thresholds based on standard deviations, approximating the distribution requires extensive training and computation. The practical solution of $\frac{2}{N^{\mathbf{w}}_{cls}}$ offers a cheap alternative.

5 Experiments I: Comparison with Prior Works

In this section, we compared our work to backdoor attacks data poisoning, and radioactive data in various aspects such as computing time, invisibility, harmlessness, verification ability using CIFAR10 dataset. For all method, we marked 50% of the training data. For backdoor attacks, there are two main approaches: clean labeled and label-noised. We used label-noised backdoor attacks like badnets [10] and blended [5] for

Method	Time Cost (sec/image)	Clean Labeled?	Model Agnostic?	Unlimited # Verification?	Invisibility (avg. SSIM)
BadNets [10]	**1.67e-7**	✗	✓	✓	0.8569
Blended [5]	**2.50e-6**	✗	✓	✓	**0.9775**
Hidden Trigger [32]	4.40e-1	✓	✗	✓	0.8859
Sleeper Agent [35]	1.27e+0	✓	✗	✓	0.8751
PoisonFrogs [34]	2.31e-1	✓	✗	✗	0.8738
MetaPoison [13]	1.70e+0	✓	✗	✗	0.8576
Bullseye [3]	2.90e-1	✓	✗	✗	0.8737
Gradient Matching [8]	6.86e-1	✓	✗	✗	0.8722
Radioactive Data [31]	1.36e-1	✓	✗	✓	0.9202
Proposed	1.07e-3	✓	✓	✓	**0.9785**

Table 3: Basic Specification of Various Methods.

only assessing basic specification. For clean labeled backdoor attacks, including hidden trigger [32] and sleeper agent [35], we affixed ten distinct triggers to the training data, assigning one unique trigger per class. Regarding data poisoning (i.e. poison frogs [34], meta poison [13], bullseye [3], gradient matching [8]), we selected one verification image per class, resulting in a total of ten verification images (multi-target setting with a 5% budget per verification image). For radioactive data [31], we marked 50% of the training data and the entire test data. The prior works require reference model, so we adopted a ResNet18 trained on benign CIFAR10 as it. Official codes of [8, 31, 35] were used. In our approach, we concealed Fashion MNIST behind the CIFAR10 dataset using a pre-trained DWN. As the pre-trained DWN, we employed U-Net [30] as autoencoders, and Vanilla CNN with four convolution layers and dropouts as classifiers. Please note that we applied watermarking to 50% training data for every experiment.

5.1 Comparison in Fundamental Specifications

We first assessed about fundamental properties. The harmlessness was gauged based on the validation accuracy on benign data, with a higher validation accuracy signifying less impact on the target task. The time cost includes only watermarking time per image, excluding training time for DWN and the reference model. Then, Structured Similarity (SSIM) [42] between benign and watermarked images was used to measure invisibility. Table 3 summarizes specifications of every methods. Label-noised backdoor attacks can be identified through visual inspection of images due to incorrect labels. Clean-labeled backdoor attacks, data poisoning and radioactive data rely on reference models, rendering them model-dependent and time-consuming, with visible watermarks. Data poisoning has limitations regarding the number of victim images. Our proposed approach outperforms these methods by generating less visible, model-agnostic watermarks, executing faster, and having no restrictions on the number of verification images.

5.2 Comparison in Effectiveness of Watermark

Purpose and Setting. We conducted a comparative analysis on CIFAR10 focusing on harmlessness and verifiability across different methods, including backdoor attacks, data poisoning, radioactive data, and our work. For all method, we watermarked randomly selected 50% of the training data. The experiments were carried out under three

scenarios: ResNet18 from scratch for 100 epochs, MobileNetV2 from scratch for 300 epochs, and MobileNetV2 from ImageNet pre-trained weights for 100 epochs. Each scenario was repeated 10 times with data augmentation. Harmlessness was evaluated by measuring the validation accuracy on a benign dataset, excluding ten images used for verification in data poisoning to ensure accuracy. For Verification ability, we measured own metrics of backdoor attacks, data poisoning, radioactive data, and our work. The ASR, which is the success rate of intended misclassifications, was used as a metric for backdoor attacks and data poisoning. For radioactive data, we compared the difference between the validation losses of the benign and the radioactive-marked dataset. For our work, we measured all of mAcc, ASR, and the difference between losses.

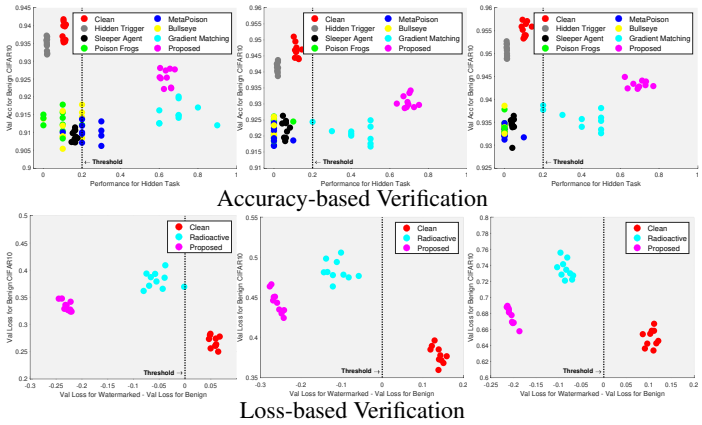


Fig. 6: Comparisons of performance to recent works using **Left**: ResNet18 from scratch, **Middle**: MobileNetV2 from scratch, and **Right**: MobileNetV2 from ImageNet pre-trained. Each dot represents a training result of each trial. "Performance of Hidden Training" means "mAcc" for the proposed method and "ASR" for others.

Results. The results are depicted in Fig. 6. Note that the ResNet18 was used as a reference for clean-labeled backdoor attacks, data poisoning, and radioactive data. In contrast, our method works without the need for any reference model. For backdoor attacks, both the hidden trigger consistently failed for all cases. The sleeper agent showed a marginally higher ASR only for the known architecture (ResNet18), but failed for unseen architectures. Regarding data poisoning, most attempts were unsuccessful except for gradient matching. The gradient matching often achieved higher ASR, but lower validation accuracy on benign CIFAR10. Please note that data poisoning can affect only few verification images, whereas our method has no limit on the number of watermarks. Our method outperformed all prior works, achieving higher validation accuracy for both benign CIFAR10 and watermarked data. Further, our approach consistently reported significant accuracy on the watermarks in every trial, underscoring its reliability. In contrast to radioactive data, our work showed lower validation loss on benign CIFAR10 and a more significant difference between validation losses of benign and

Dataset	Method	Val Acc on Benign (%)	Val mAcc on Watermark (%)	Val mASR(%)	SSIM
CIFAR100	Clean	71.79±0.18	N/A	N/A	1.0000
	Sleeper Agent	66.59±0.46	N/A	4.20±0.84	0.8821
	Gradient Matching	66.53±0.25	N/A	**32.00±5.88**	0.8797
	Proposed	**68.27±0.15**	30.35±1.04	N/A	**0.9710**
FER2013	Clean	68.89±0.29	N/A	N/A	1.0000
	Sleeper Agent	62.68±0.37	N/A	11.02±3.05	0.8392
	Gradient Matching	63.36±0.30	N/A	**47.62±19.34**	0.8197
	Proposed	**64.63±0.45**	38.15±0.64	N/A	**0.9833**
*FMNIST	Clean	94.21±0.10	N/A	N/A	1.0000
	Sleeper Agent	93.55±0.14	N/A	0.00±0.00	0.7882
	Gradient Matching	93.35±0.21	N/A	6.67±5.48	0.7771
	Proposed	**93.66±0.21**	**22.31±2.02**	N/A	**0.9744**

*FMNIST: Fashion MNIST

Table 4: Applicability to various datasets.

Architecture		Val Acc on Benign (%)	Val mAcc on Watermark (%)
EfficientNetB0	Clean	96.43±0.24	9.89±0.08
(Transfer)	Cheating	96.27±0.29	**54.18±2.15**
PVTv2-B0	Clean	95.00±0.37	10.29±0.12
(Transfer)	Cheating	94.54±0.42	**51.39±2.32**
ResMLP-12-224	Clean	96.14±0.21	10.19±0.31
(Transfer)	Cheating	95.44±0.40	**53.10±1.74**
PiT Tiny	Clean	94.78±0.39	9.71±0.32
(Transfer)	Cheating	94.59±0.32	**48.09±3.21**

Table 5: Applicability to various architectures

watermarked data. From the results, we can conclude that our work outperforms prior works in terms of harmlessness, reliability in verification.

6 Experiment II: General Applicability

The prior experiment showed feasibility and superiority of our work in limited settings. This section introduces extensions to further datasets, architectures and tasks.

6.1 Application to Further Architectures and Datasets

Purpose and Setting. To broaden the scope of our evaluation beyond the initial limited settings, which focused on a few architectures and solely the CIFAR10 dataset, we expanded our analysis to include additional architectures and datasets, thereby affirming the widespread applicability of our work. We incorporated CIFAR100, FER2013 [9], and Fashion MNIST for this extended evaluation. CIFAR100 (100 classes) and FER2013 (7 classes) were assessed against the most effective previously identified methods: sleeper agent for backdoor attacks and gradient matching for data poisoning. The reference models included ResNet18 and a Benign CNN, while the cheating models utilized DenseNet-BC [12], trained from scratch. CIFAR100's auxiliary dataset comprised Fashion MNIST and MNIST (10 classes each), while FER2013's auxiliary dataset consisted of the first 7 MNIST classes. Also, we tested the following architectures as cheating models on CIFAR10 (target) and Fashion MNIST (auxiliary): EfficientNet [37], PVTv2 [41], ResMLP [39], and PiT [11]. We trained each architecture 35 epochs starting from ImageNet pre-trained weights, employing SGD optimization, warmup, label smoothing, and data augmentation (i.e., spatial transformation and mixup), repeated multiple times to ensure robustness and reliability of the results.

Results. Table 4 demonstrates that our work excels in three key aspects: 1) harmlessness, 2) invisibility, and 3) verifiability across all datasets. Also, our work consistently outperforms in mAcc on the watermark across various architectures, as presented in Table 5. These findings lead us to conclude that our work effectively operates on diverse architectures and datasets, making it universally applicable. In all cases, our proposed threshold attained 100% accuracy for distinguishing cheating models from clean ones.

	Tiny ImageNet	
	Val Acc on Benign (%)	Val mAcc on Watermark (%)
Clean	48.23±1.46	8.84±2.29
Cheating	46.84±1.14	**34.37±5.61**
	ImageNet	
	Val Acc on Benign (%)	Val mAcc on Watermark (%)
Clean	70.42±0.41	11.90±1.33
Cheating	70.14±0.72	**43.95±3.21**

Table 6: Validation accuracy of MobileNetV2 on Tiny ImageNet and ImageNet.

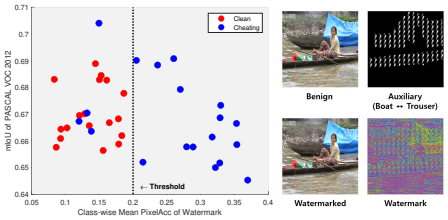


Fig. 7: Results on segmentation and an example, with emphasized watermark for clarity.

6.2 Application to Fine-grained Classification

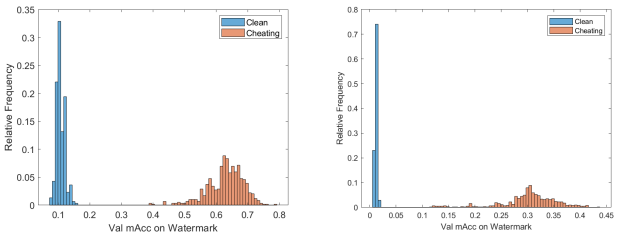
Purpose and Setting. To address the limitation of requiring the auxiliary dataset to contain more classes than the target dataset, we employed the modulo operation. We validated the solution using both Tiny ImageNet, which consists of 100 classes, and ImageNet, which comprises 1,000 classes. Fashion MNIST was used as a auxiliary dataset, and 50% of training datasets were watermarked. We trained MobileNetV2 on 1) watermarked Tiny ImageNet [16] (0.9883 avg. SSIM) from scratch and 2) watermarked ImageNet (0.9570 avg. SSIM) from a benign ImageNet pre-trained model.

Results. Table 6 shows the average performance results from 30 and 5 trials for each dataset. As shown, training MobileNetV2 on watermarked data leads to a marginal decrease in accuracy on the benign validation dataset, but demonstrates significant performance improvements in detecting watermarks. Conversely, the accuracy of the clean model is significantly diminished, nearing the chance level. This finding proves the verifiability of our work, even when the target dataset contains a greater number of classes compared to the auxiliary dataset. Therefore, it is not necessary to prepare a auxiliary dataset that matches the target dataset in terms of the number of classes.

6.3 Application to Image Segmentation

Purpose and Setting. To extend our work to image segmentation, we adapted our method to create spatially varying watermarks. We resized the auxiliary data to a smaller scale, such as 8x8 pixels, and repeatedly stitched it to segments of the 50% PASCAL VOC 2012 data, considering each segment's label. This produced segment-wise watermarked images, as shown in Fig. 7. To adjust the DWN for segmentation, we replaced the two classifiers with simple autoencoders featuring dropout. We trained a segmentation autoencoder with a MobileNetV2 backbone from scratch on this watermarked dataset, employing the Adam optimizer with learning rate decay starting from 1e-3, a batch size of 60, and data augmentation. In image segmentation, meaningful information resides in silhouettes, requiring a higher threshold compared to $\frac{2}{N_{cls}^{\mathbf{z}}}$.

Results. Fig. 7 provides an example image and shows the performances on the benign Pascal VOC dataset and the watermarked dataset after training. In the context of image segmentation on Pascal VOC, we adopted mIoU as the evaluation metric. As shown, there is almost no damage to the original task. For verifying cheating models, we adopted the mean class pixel accuracy on the masked region. The watermark



(a) ResNet18 on CIFAR10 (b) DenseNet on CIFAR100

Fig. 8: Histogram of mAcc on watermark of clean and cheating models. For both cases, the clean models hardly achieved higher mAcc than $\frac{2}{N_{cls}^{\mathbf{z}}}$, but all of the cheating models achieved.

was surreptitiously learned in most of the trials, enabling successful verification of the cheating models. While the clean models reported accuracies lower than 0.2, 79% of the cheating trials exhibited accuracies higher than 20%. This outcome indicates the potential applicability of our watermarking to other tasks beyond image classification.

7 Ablation Studies

This section introduces some ablation studies. In the supplementary material, we provide additional studies, including analyses on robustness to defense and debiasing.

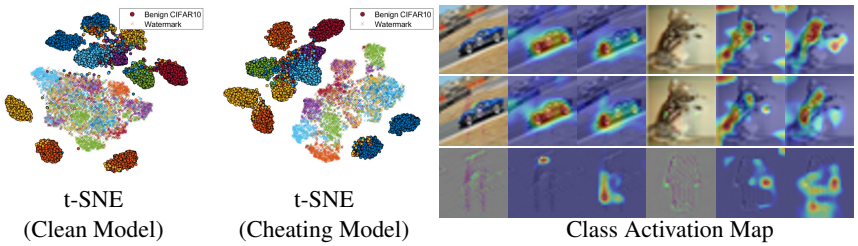
7.1 Histogram Analysis of mAcc on Watermark

Purpose and Setting. This study aims to validate our threshold criterion. We trained ResNet18 on CIFAR10 and DenseNet on CIFAR100, both with and without watermarks, from scratch multiple times. Then, we assessed the mAcc of these models on our watermark. For each setting, we performed over 300 training runs using the Adam optimizer, 1e-3 learning rate, and 100 epochs. By histogram analysis of the mAcc values, we were able to reveal their distribution.

Results. Fig. 8 shows the histogram analysis results, clearly indicating a Gaussian distribution of collected mAcc values. Notably, there are no occurrences at 0% mAcc or $\frac{2}{N_{cls}^{\mathbf{z}}}$ mAcc, though peak points may slightly deviate towards $\frac{2}{N_{cls}^{\mathbf{z}}}$ mAcc for clean models. On the contrary, all cheating models reported the higher mAcc than $\frac{2}{N_{cls}^{\mathbf{z}}}$. The proposed threshold, $\frac{2}{N_{cls}^{\mathbf{z}}}$, is approximately 7 times and 5 times the standard deviation of the mean for both watermarked cases. In other words, clean model can achieve mAcc higher than $\frac{2}{N_{cls}^{\mathbf{z}}}$ with less than 3e-5% by empirical rule. Hence, if a model shows higher than $\frac{2}{N_{cls}^{\mathbf{z}}}$ mAcc, it can be confidently deemed as a cheating model.

7.2 Visualizations

Purpose and Setting. We applied visualization techniques, t-SNE and CAM, to understand how watermark functions. To visualize, two ResNet18 models were trained: one



t-SNE (Clean Model) t-SNE (Cheating Model) Class Activation Map

Fig. 9: Visualization results. In the CAM figure, each row represent **Top**: benign image, **Middle**: CAM for watermarked image, and **Bottom**: CAM for watermark. Then, each column indicates **1st&4th**: input image, **2nd&5th**: CAM of clean model, and **3rd&6th**: CAM of cheating model.

on the benign CIFAR10 dataset (Clean) and the other on the watermarked CIFAR10 dataset (Cheating). Then, we extracted the last latent features, and visualized them as 2D t-SNE plot and CAM.

Results. Fig. 9 shows t-SNE, example watermarked images and CAM. In t-SNE, feature vectors from benign data, whether from clean or cheating models, form distinct and discriminative clusters. However, for the watermarks, cheating models exhibit partial clustering patterns, while clean models lack such clustering tendencies entirely. For CAM, the cheating model responds to all three types: benign, watermarked, and watermark. In contrast, the clean model disregards the watermarks. This validate the unintended knowledge about hidden bias and provides additional evidence of cheating.

8 Conclusion

This paper presents "undercover bias," an innovative approach involving embedding class-wise hidden bias as a watermarks to detect model trained on a specific dataset. When a model is trained on our watermarked dataset, it subtly learns and reacts to the embedded watermarks, offering evidence of cheating. Initially, we observed high performance on classification of background images, demonstrating unintentional learning of bias. We then developed two preliminary approaches of injecting class-wise hidden bias, noise placement and dataset overlaying. From the approaches, we found the requirements: robustness to spatial transformation and invisibility. By addressing the requirements, we established our undercover bias. We validated that our proposed method is more effective at verifying cheating models across various conditions compared to existing methods, despite the watermark being less visible and less disruptive. We also provided ablation studies and visualizations. Further, we successfully applied the undercover bias to fine-grained image classification, and image segmentation.

Limitation. Our work is applicable only to ordinal or numerical data, but not applicable to nominal data (i.e., text) because small differences can drastically change meanings. Additionally, our watermarking method causes only slight performance degradation compared to models trained on benign data.

Acknowledgements

This work was partly supported by Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government foundation (24ZB1200, Research of Human-centered Autonomous Intelligence System Original Technology, 40%), the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government. (MSIT) (RS-2023-00215760, Guide Dog: Development of Navigation AI Technology of a Guidance Robot for the Visually Impaired Person, 30%), and Korea Institute of Marine Science & Technology Promotion (KIMST) funded by the Korea Coast Guard (RS-2023-00238652, Integrated Satellite-based Applications Development for Korea Coast Guard, 30%)

References

1. https://image-net.org/challenges/LSVRC/announcement-June-2-2015, june, 2015
2. https://www.kaggle.com/c/petfinder-adoption-prediction/discussion/125436, january, 2020
3. Aghakhani, H., Meng, D., Wang, Y.X., Kruegel, C., Vigna, G.: Bullseye polytope: A scalable clean-label poisoning attack with improved transferability. In: EuroS&P. pp. 159–178 (2021)
4. Baluja, S.: Hiding images in plain sight: Deep steganography. In: NeurIPS. pp. 2066–2076 (2017)
5. Chen, X., Liu, C., Li, B., Lu, K., Song, D.: Targeted backdoor attacks on deep learning systems using data poisoning. arXiv preprint arXiv:1712.05526 (2017)
6. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR. pp. 248–255 (2009)
7. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. ICCV **88**(2), 303–338 (2010)
8. Geiping, J., Fowl, L.H., Huang, W.R., Czaja, W., Taylor, G., Moeller, M., Goldstein, T.: Witches' brew: Industrial scale data poisoning via gradient matching. In: ICLR (2021)
9. Goodfellow, I.J., Erhan, D., Carrier, P.L., Courville, A., Mirza, M., Hamner, B., Cukierski, W., Tang, Y., Thaler, D., Lee, D.H., et al.: Challenges in representation learning: A report on three machine learning contests. In: NeurIPS (2013)
10. Gu, T., Liu, K., Dolan-Gavitt, B., Garg, S.: Badnets: Evaluating backdooring attacks on deep neural networks. IEEE Access **7**, 47230–47244 (2019)
11. Heo, B., Yun, S., Han, D., Chun, S., Choe, J., Oh, S.J.: Rethinking spatial dimensions of vision transformers. In: ICCV (2021)
12. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: CVPR (2017)
13. Huang, W.R., Geiping, J., Fowl, L., Taylor, G., Goldstein, T.: Metapoison: Practical general-purpose clean-label data poisoning. NeurIPS **33**, 12080–12091 (2020)
14. Jiang, W., Li, H., Xu, G., Zhang, T.: Color backdoor: A robust poisoning attack in color space. In: CVPR. pp. 8133–8142 (2023)
15. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
16. Le, Y., Yang, X.: Tiny imagenet visual recognition challenge. CS 231N **7**, 7 (2015)
17. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)

18. Lee, J., Kim, E., Lee, J., Lee, J., Choo, J.: Learning debiased representation via disentangled feature augmentation. NeurIPS **34**, 25123–25133 (2021)
19. Li, Y., Bai, Y., Jiang, Y., Yang, Y., Xia, S.T., Li, B.: Untargeted backdoor watermark: Towards harmless and stealthy dataset copyright protection. In: NeurIPS (2022)
20. Li, Y., Zhang, Z., Bai, J., Wu, B., Jiang, Y., Xia, S.T.: Open-sourced dataset protection via backdoor watermarking. NeurIPS Workshops (2020)
21. Li, Y., Li, Y., Wu, B., Li, L., He, R., Lyu, S.: Invisible backdoor attack with sample-specific triggers. In: ICCV. pp. 16463–16472 (2021)
22. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV. pp. 740–755 (2014)
23. Liu, G., Xu, T., Ma, X., Wang, C.: Your model trains on my data? protecting intellectual property of training data via membership fingerprint authentication. IEEE Transactions on Information Forensics and Security **17**, 1024–1037 (2022)
24. Liu, Y., Ma, X., Bailey, J., Lu, F.: Reflection backdoor: A natural backdoor attack on deep neural networks. In: ECCV. pp. 182–199 (2020)
25. Liu, Y., Ma, X., Bailey, J., Lu, F.: Reflection backdoor: A natural backdoor attack on deep neural networks. In: ECCV (2020)
26. Nam, J., Cha, H., Ahn, S., Lee, J., Shin, J.: Learning from failure: De-biasing classifier from biased classifier. NeurIPS **33**, 20673–20684 (2020)
27. Ramaswamy, V.V., Kim, S.S., Russakovsky, O.: Fair attribute classification through latent space de-biasing. In: CVPR. pp. 9301–9310 (2021)
28. Ribeiro, M.T., Singh, S., Guestrin, C.: " why should i trust you?" explaining the predictions of any classifier. In: SIGKDD. pp. 1135–1144 (2016)
29. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR. pp. 10684–10695 (2020)
30. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: MICCAI. pp. 234–241. Springer (2015)
31. Sablayrolles, A., Douze, M., Schmid, C., Jégou, H.: Radioactive data: tracing through training. In: ICML. pp. 8326–8335 (2020)
32. Saha, A., Subramanya, A., Pirsiavash, H.: Hidden trigger backdoor attacks. In: AAAI. vol. 34, pp. 11957–11965 (2020)
33. Schwarzschild, A., Goldblum, M., Gupta, A., Dickerson, J.P., Goldstein, T.: Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks. In: ICML. pp. 9389–9398 (2021)
34. Shafahi, A., Huang, W.R., Najibi, M., Suciu, O., Studer, C., Dumitras, T., Goldstein, T.: Poison frogs! targeted clean-label poisoning attacks on neural networks. NeurIPS **31** (2018)
35. Souri, H., Fowl, L., Chellappa, R., Goldblum, M., Goldstein, T.: Sleeper agent: Scalable hidden trigger backdoors for neural networks trained from scratch. NeurIPS **35**, 19165–19178 (2022)
36. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. JMLR **15**(1), 1929–1958 (2014)
37. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: ICML. pp. 6105–6114 (2019)
38. Tompson, J., Goroshin, R., Jain, A., LeCun, Y., Bregler, C.: Efficient object localization using convolutional networks. In: CVPR. pp. 648–656 (2015)
39. Touvron, H., Bojanowski, P., Caron, M., Cord, M., El-Nouby, A., Grave, E., Izacard, G., Joulin, A., Synnaeve, G., Verbeek, J., et al.: Resmlp: Feedforward networks for image classification with data-efficient training. ICLR (2021)
40. Wang, T., Yao, Y., Xu, F., An, S., Tong, H., Wang, T.: An invisible black-box backdoor attack through frequency domain. In: ECCV. pp. 396–413. Springer (2022)

41. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pvt v2: Improved baselines with pyramid vision transformer. Computational Visual Media pp. 1–10 (2022)
42. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. TIP **13**(4), 600–612 (2004)
43. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747 (2017)
44. Zhang, J., Chen, D., Liao, J., Fang, H., Zhang, W., Zhou, W., Cui, H., Yu, N.: Model watermarking for image processing networks. In: AAAI. vol. 34, pp. 12805–12812 (2020)
45. Zhu, Z., Xie, L., Yuille, A.: Object recognition with and without objects. In: IJCAI. pp. 3609–3615 (2017)