# Detecting Generative Model Inversion Attacks for Protecting Intellectual Property of Deep Neural Networks

YIDING YU, Wuhan Institute of Technology, China
WEI ZONG*, University of Wollongong, Australia
WENJING SU†, Wuhan Institute of Technology, China
YANG-WAI CHOW, University of Wollongong, Australia
WILLY SUSILO, University of Wollongong, Australia

Recently, protecting the Intellectual Property (IP) of deep neural networks (DNNs) has attracted attention from researchers. This is because training DNN models can be costly especially when acquiring and labeling training data require domain expertise. DNN watermarking and fingerprinting are two techniques proposed to prevent DNN IP infringement. Although these two techniques achieve high performance on defending against previously proposed DNN stealing attacks, researchers recently show that both of them are ineffective against generative model inversion attacks. Specifically, an adversary inverts training data from well-trained DNNs and uses the inverted data to train DNNs from scratch such that DNN watermarking and fingerprinting are both bypassed. This novel model stealing strategy shows that data inverted from victim models can be effectively exploited by adversaries, which poses a new threat to the IP protection of DNNs. To combat this new threat, one potential solution is to enable defenders to prove ownership on data inverted from models being protected. If the training data of a suspected model, which can be disclosed via the judicial process, are proven to be data inverted from victim models, then IP infringement is detected. This research direction is currently underexplored. In this paper, we fill the gap in the literature to investigate countermeasures against this emerging threat. We propose a simple but effective method, called InverseDataInspector (IDI), to detect whether data are inverted from victim models. Specifically, our method first extracts features from both the inverted data and victim models. These features are then combined and used for training classifiers. Experimental results demonstrate that our method achieves high performance on detecting inverted data and also generalizes to new generative model inversion methods that are not seen when training classifiers.

**JAIR Associate Editor:** Atlas Wang

**JAIR Reference Format:**
Yiding Yu, Wei Zong, Wenjing Su, Yang-Wai Chow, and Willy Susilo. 2025. Detecting Generative Model Inversion Attacks for Protecting Intellectual Property of Deep Neural Networks. *Journal of Artificial Intelligence Research* 84, Article 13 (October 2025), 13 pages. DOI: 10.1613/jair.1.19468

*Corresponding Author.
†Corresponding Author.

Authors' Contact Information: Yiding Yu, ORCID: 0009-0004-1522-1065, reading0319@163.com, Wuhan Institute of Technology, Wuhan, Hubei, China; Wei Zong, ORCID: 0000-0001-9714-6759, wzong@uow.edu.au, University of Wollongong, Wollongong, New South Wales, Australia; Wenjing Su, ORCID: 0000-0002-4798-8824, suwenjing1222@126.com, Wuhan Institute of Technology, Wuhan, Hubei, China; Yang-Wai Chow, ORCID: 0000-0003-3348-7014, caseyc@uow.edu.au, University of Wollongong, Wollongong, New South Wales, Australia; Willy Susilo, ORCID: 0000-0002-1562-5105, wsusilo@uow.edu.au, University of Wollongong, Wollongong, New South Wales, Australia.


DOI: 10.1613/jair.1.19468

## 1 Introduction

In recent years, artificial intelligence (AI) technology has rapidly developed and been applied to various fields, leading to interdisciplinary integration. AI, with its exceptional performance, has been widely applied and achieved remarkable success in numerous domains. However, training deep neural networks (DNNs) may require sensitive data that are difficult to obtain, e.g., financial transaction data and medical data (Maslej et al. 2023). When data is difficult to obtain or labeling them requires extensive expertise, training DNNs can become expensive. Hence, it is crucial to protect the intellectual property (IP) of such models that are trained on valuable data.

Currently, several defensive techniques have been developed to protect the IP of DNNs. These techniques are categorized as DNN watermarking (Uchida et al. 2017) or DNN fingerprinting (Cao et al. 2021). DNN watermarking involves embedding specific identity information into the model by modifying the target model itself. Ownership can be proven if the same or similar information can be extracted from the model. However, embedding watermarks requires intervention in the training process, which may affect the model's performance. DNN fingerprinting relies on uniqueness, as functionally similar DNN models produce different outputs due to slight differences in their weights when processing the same input data. By analyzing these output differences, a fingerprint that reflects the model's uniqueness can be generated. However, the robustness of the most advanced fingerprinting techniques has not been thoroughly validated. Additionally, the recently proposed IPRemover can evade detection by even the most advanced DNN fingerprinting and watermarking technologies (Zong et al. 2024). Because trained models tend to memorize training data (Papernot et al. 2018; Song et al. 2017; Tramèr et al. 2016), generative model inversion attacks, e.g., IPRemover, can extract training data and then retrain a new model. This renders current methods ineffective in identifying IP infringement. On the technical front, there is a need to explore and propose more advanced protection techniques.

Moreover, generative model inversion attacks reveal the gap in the literature to detect IP infringement if a suspect model is trained on inverted data. We argue that claiming copyright on inverted data is crucial for model owners. Without this ability, detecting IP infringement may not be possible since a suspect model can be trained independently on inverted data. In this work, we fill this gap in the literature. We propose a method, called InverseDataInspector (IDI), that can effectively detect data generated by inverse engineering techniques. Empirical results show that our method is able to generalize to different datasets and models.

Our contributions are summarized as follows:

- Our method fills the gap in the literature to detect IP infringement on data inverted from DNNs.
- Our method can generalize across various model inversion attacks, including IPremover (Zong et al. 2024), CMI (Fang et al. 2021), DeepInversion (Yin et al. 2020), ZSKT (Micaelli and Storkey 2019), and DFQ (Choi et al. 2020).
- Our method demonstrates robustness against common attacks that alter input distribution by preprocessing.

This paper is organized as follows. Section 2 reviews related work. Section 3 and Section 4 discuss the threat model and problem definition. Section 5 details our proposed method. Section 6 present our experimental results. Section 7 discusses the limitation of our current approach. Finally, Section 8 concludes this paper.

## 2 Related Work

In this section, we first review generative model inversion attacks and their impact on the IP of DNNs. We then discuss existing techniques designed to protect IP of DNNs.

## 2.1 Generative Model Inversion Attacks

Model inversion attacks aim to reconstruct the original training data of a target model. These attacks are particularly effective against deep learning models (An et al. 2022; Mehnaz et al. 2022) because they tend to remember training data during the training procedure. Such attacks pose a significant threat to the IP of DNNs (Zhang et al. 2020), as they allow attackers to replicate the performance of a target model without direct access to its original training data.

Early research on model inversion attacks primarily focused on extracting sensitive information from trained models (Fredrikson et al. 2014; Z. Yang et al. 2019). For instance, Fredrikson et al. (2015) exploited confidence values to reconstruct an image depicting a recognizable face for an individual.

Recently, Zong et al. (2024) proposed IPRemover, a generative model inversion attack targeting DNN fingerprinting and watermarking techniques. This method bypasses DNN fingerprinting and watermarking protections through generative model inversion and has demonstrated effectiveness across multiple datasets. Previously, Fang et al. (2021) introduced the Contrastive Model Inversion (CMI) method , which also extracts data from models. CMI uses a contrastive loss function to generate high-quality inverted data for knowledge distillation without original training data, significantly improving knowledge transfer efficiency. Yin et al. (2020) proposed DeepInversion, which optimizes noise images to produce activations in DNNs similar to the original training data, enabling data-free knowledge transfer. Micaelli and Storkey (2019) proposed Zero-Shot Knowledge Transfer (ZSKT), achieving knowledge transfer from the source model to the target model without any training data. Choi et al. (2020) introduced DFQ, which uses adversarial knowledge distillation techniques to quantize neural networks without training data, demonstrating high efficiency and robustness across various tasks.

## 2.2 Deep Nerual Network Intellectual Property Protection

Protecting the IP of DNNs is an emergine research area. Several techniques have been proposed to safeguard these valuable models and their data. Uchida et al. (2017) introduced a generic framework for embedding watermarks into model parameters, allowing the extraction of embedded information to identify and verify model ownership. Cao et al. (2021) proposed IPGuard, which creates fingerprints based on the unique classification boundaries of DNNs. By comparing the boundaries of suspect models to those of the original, IPGuard detects IP infringement. Jia et al. (2021) proposed the Entangled Watermarks method, which embeds entangled watermarks into models to defend against model extraction attacks. These robust and hard-to-detect watermarks provide a means to identify and verify model ownership. Charette et al. (2022) proposed Cosine Model Watermarking, which embeds watermarks using cosine similarity to protect against ensemble distillation attacks. This ensures that any distilled ensemble model retains the original watermark, enabling the detection of unauthorized use. Chen et al. (2022) proposed a detection framework that analyzes model outputs and behaviors to detect unauthorized copying. This systematic framework tests and verifies copyright protection in models. Le Merrer et al. (2020) introduced the Adversarial Frontier Stitching method, which embeds watermarks through adversarial examples to defend against remote neural network attacks. This technique creates robust watermarks that are difficult to remove or alter, allowing flexible application in different environments. K. Yang et al. (2022) proposed MetaFinger which uses meta-training to fingerprint deep neural networks. These fingerprints allow tracking and identifying unauthorized model copies.

Despite these advancements, existing methods share common limitations. Many techniques require significant computational resources and complex implementation processes, which can hinder their practical application. More importantly, these methods face challenges in distinguishing or preventing the latest generative model inversion attacks, such as IPRemover proposed by Zong et al. (2024).

## 3 Threat Model

We consider a scenario that DNN watermarking or fingerprinting techniques fail to detect IP infringement when they are bypassed by generative model inversion attacks. In contrast, our method is able to detect IP infringement if a suspect model is trained on data inverted from a victim model. Spcifically, the victim model is trained on sensitive data, e.g., medical image or financial data. These data are expensive to obtain and labelling them requires expertise. Before releasing the trained model to the public for profits, the model owner utilizes DNN watermarking or fingerprinting to protect the IP of the model.

An adversary wants to steal the functionalities of the victim model for profits. The adversary obtains a copy of the model, e.g., via internal theft or purchasing a copy for local deployment. The adversary inverts training data from the victim model using generative model inversion attacks. The inverted data are used to train a knockoff model to achieve comparable performance. After the knockoff model is well trained, the adversary release it to the public for profits.

The owner of the victim model notices the existence of the knockoff model and suspects that it steals the functionalities of the victim model. Nonethess, DNN watermarking or fingerprinting fails to detect IP infringement since they are bypassed by generative model inversion attacks.

With the help of legal measures, the training data of the suspect model are disclosed to the authority. Our IDI method then is used by the authority to analyze the disclosed data. IP infringement is detected if our method determines that the data are inverted from the victim model.

## 4 Problem Definition

Let $M_v$ be the victim DNN model of which the IP is under protection. Let $\mathcal{D}_p$ be the set of all the data that can be potentially inverted from $M_v$ by generative model inversion methods. These data can potentially be exploited by an adversary to bypass DNN IP protection (e.g., IPRemover). Our goal is to train a classifier $f$ such that $f$ can detect whether a suspected data point $s$ is inverted from $M_v$: $s \in \mathcal{D}_p$. Given a suspected dataset $\mathcal{D}_s$, let $w$ represent the percentage of data in $\mathcal{D}_s$ that are detected as inverted from $M_v$. IP infringement is detected, i.e., $\mathcal{D}_s \cap \mathcal{D}_p \neq \emptyset$, if $w$ exceeds a threshold $\tau$.

## 5 Proposed Method

The overall workflow of IDI is depicted in Figure 1. As mentioned above, our goal is to train a classifier $f$ which can detect whether data are inverted from the victim model. The first step is to collect a training set consisting of both benign and inverted data. Then, we extract features from the training set and use them to train $f$. After $f$ is well trained, it can be used to detect whether suspect data are inverted or not. Details of training $f$ and detecting IP infringement are presented in the following sections.

### 5.1 Training Classifier

We propose a hybrid feature extraction strategy that extracts features from the data themselves and the victim model. The intuition is that to determine whether suspect data are generated through inverse engineering, both features are expected to proivde useful information. The extracted features are used to train a classifier $f$.

We exploit a well trained DNN, e.g., a ResNet trained on ImageNet, to extract features from the data. To extract features from the victim model, we pass the data into the model and extract activation values from the last hidden layer. We select the last few hidden layers of the victim model as their activation values tend to represent high-level information, which is more meaningful than low-level features extracted from the layers at the beginning.

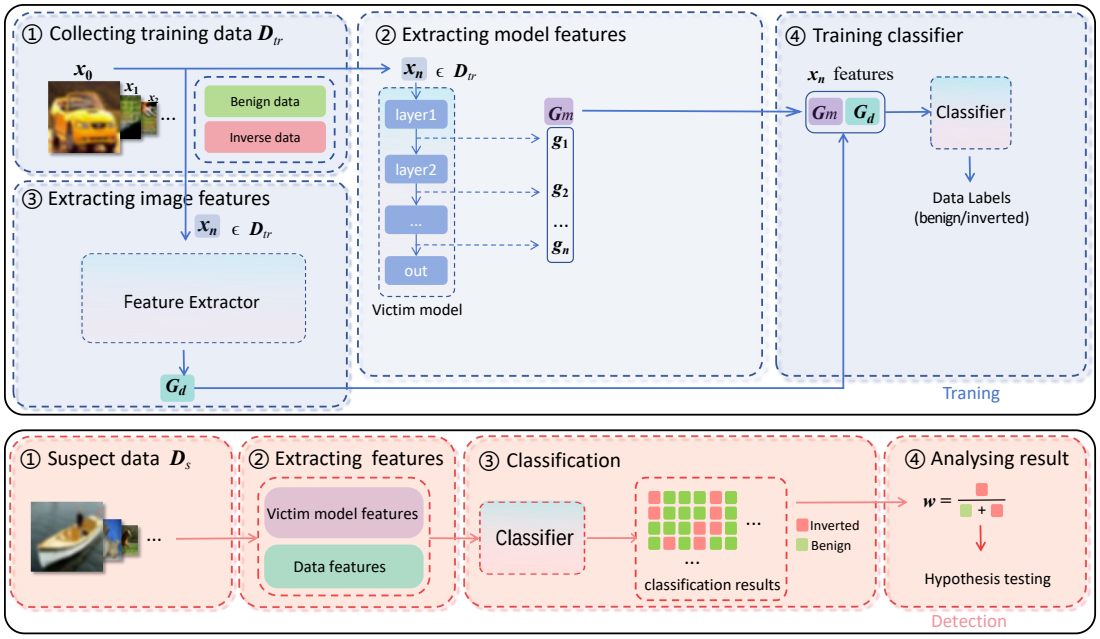


Fig. 1. The workflow of IDI consists of several steps. A classifier is trained on features extracted from both benign and inverted data. This classifier is then used to detect whether suspect data are inverted from the victim model.

We extract features from both benign and inverted data. All the extracted features are collected to train a classifier $f$ which distinguishes between benign and inverted data. If a data point is detected as inverted (positive), $f$ returns 1. Otherwise, $f$ returns 0.

## 5.2 Detecting IP Infringement

**Algorithm 1** Detecting suspect data.

**Input:** Suspected data set $D_s$; the victim model $M_v$; a well trained feature extractor $g$; a well trained classifier $f$; the detection threshold $\tau$.
**Output:** True if suspected data are determined as inverted data. Return false otherwise.

$G \leftarrow \emptyset$
**for each** $s \in D_s$ **do**
  $g_d \leftarrow g(s)$
  $g_m \leftarrow M_v(s)$  // extract features from $M_v$
  $G \leftarrow G \cup (g_d \cup g_m)$
**end for**
$Y \leftarrow f(G)$
$w \leftarrow \frac{1}{|Y|} \sum_{y \in Y} \mathbb{1}\{y = 1\}$
**return** $w > \tau$

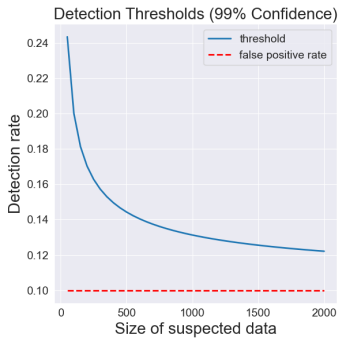


Fig. 2. The detection thresholds ($\tau$) for different data sizes. $\tau$ becomes smaller when the size of suspect data is larger.

After the classifier $f$ is well trained, it is used to detect whether a suspected dataset $\mathcal{D}_s$ infringes on IP of the victim model. The detection process is detailed in Algorithm 1. $w$ is the percentage of data in $\mathcal{D}_s$ that are determined as inverted from $M_v$. IP infringement is detected if $w$ significantly exceeds the false positive rate of $f$, which is determined if $w$ exceeds a threshold $\tau$.

The value of $\tau$ can be theoretically calculated. The null hypothesis for the suspected dataset $\mathcal{D}_s$ is that $\mathcal{D}_s$ only contains benign data. The alternative hypothesis is that $\mathcal{D}_s$ contains data inverted from $M_v$. In this manner, we assume that the classification result on $\mathcal{D}_s$ follows a binomial distribution:

$$w * n \sim B(n, p), \tag{1}$$

where $n = |\mathcal{D}_s|$ and $p$ is equal to the false positive rate of $f$.

$\tau$ can then be determined using a one-tailed T-test. The null hypothesis is rejected if $w$ is significantly larger than the false positive rate. According to the Central Limit Theorem, a binomial distribution approaches a normal distribution when $n \geq 30$.

Figure 2 demonstrates the varying thresholds given different data sizes. The p-value is set to 99% and the false positive rate is set to 10%. Empirically, we observe that the actual false positive rates are significantly lower than 10%. This means the thresholds in Figure 2 are conservative.

The curve in Figure 2 clearly reveals a key trend: as the size of suspected data increases, the threshold for detecting IP infringement decreases. If only 50 suspect images are accessible, the detection threshold is 24%. When 500 suspect images are accessible, the detection threshold decreases to 14%. When 2000 suspect images are accessible, the detection threshold further decreases to 12%. In the experimental results, we will demonstrate that the detection rates for inverted data are far above the detection thresholds.

## 6 Experimental Results

This section presents our experimental results.

### 6.1 Setup

All experiments were performed on a desktop version of Ubuntu 22.04 with 16G RAM and an Nvidia 4060Ti 16G. To reproduce existing generative model inversion attacks, we exploited their open-source code online and executed the code once on each model.

*6.1.1 Datasets.* The datasets used for the experiments are widely utilized in deep learning security research: CIFAR10 (Krizhevsky et al. 2009) and GTSRB (Stallkamp et al. 2011). CIFAR10 contains 60,000 color images of

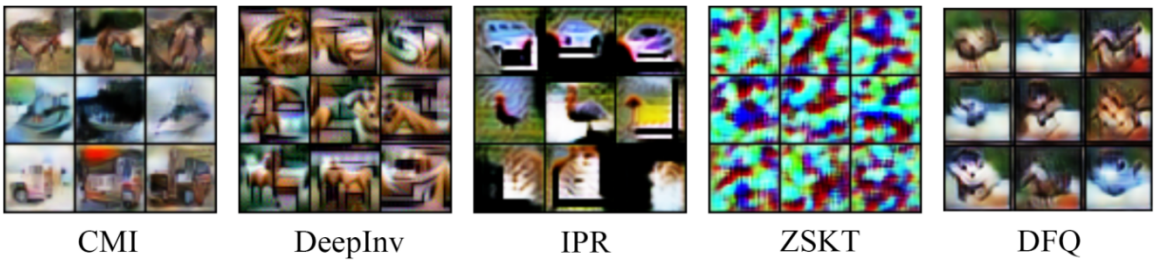


Fig. 3. Examples of inverted CIFAR10 images obtained using different model inversion methods. All the methods are described in detail in Section 2.1.

$32 \times 32$ pixels grouped into 10 categories with 6,000 images in each category. The 10 categories are: airplanes, cars, birds, cats, deer, dogs, frogs, horses, boats, and trucks. The dataset is divided into two parts: 50,000 images are used for training and 10,000 images are used for testing.

GTSRB is another wide used traffic sign recognition dataset. GTSRB consists of 26,640 images for training and 12,630 images for testing, with a total of 43 categories.

For CIFAR10, the victim model used in this study is the WideResNet (Zagoruyko and Komodakis 2016). For GTSRB, the victim model used is VGG11 (Sengupta et al. 2019). These two models were chosen because all the attacks managed to invert data from them.

*6.1.2 Model Inversion Attacks.* Victim models were trained on the original CIFAR10 and GTSRB data. Then, different model inversion methods were used to inverse data from the victim models. The methods considered in this work are CMI (Fang et al. 2021), DeepInversion (Yin et al. 2020), IPRemover (Zong et al. 2024), Zero-shot Knowledge Transfer (ZSKT) (Micaelli and Storkey 2019), and Data-Free Network Quantization (DFQ) (Choi et al. 2020). These methods are considered because they are recently proposed and show state-of-the-art model inversion results.

*6.1.3 IP Infringement Classifier.* To construct the training set for our IP Infringement classifier, the original data of CIFAR10 and GTSRB were combined with inverted data. The inverted data were labeled as positive while original data are labeled as negative.

As discussed above, the classifier is trained on combined features that are extracted from both the data themselves and the victim model. To extract features from the input images, a pre-trained ResNet18 model was used and 512 dimensional features were extracted before its fully connected layer. To extract features from the victim model, the output of its final hidden layers was extracted. The extracted features were finally used to train a Support Vector Machine (SVM) model, which detected whether input data were benign data or inverted from the victim model.

## 6.2 CIFAR10 Results

This section presents experimental results on the victim model trained on CIFAR10. Figure 3 shows example images that are inverted using different model inversion methods. It can be observed that different model inversion methods result in inverted images with different visual patterns. In particular, images inverted by ZSKT are significantly different from the other inverted images.

We used each method to invert the same number of images as the CIFAR10 training set, i.e., 50,000. The inverted images were randomly mixed with the original training images and 80% of them were used to train the

Table 1. Detecting images that are inverted from a victim model trained on CIFAR10. The detection rates are measured in percentage.

| **Tested On \ Trained On** | CMI | DAFL | DeepInv | DFQ | IPR | ZSKT |
|---|---|---|---|---|---|---|
| CIFAR10 Test | 6.48 | 0.01 | 0.14 | 0.11 | 0.35 | 0.01 |
| CMI | **94.72** | 0.22 | 0.91 | 1.14 | 4.36 | 0.02 |
| DAFL | 99.08 | **99.98** | 0.05 | 99.82 | 0.34 | 23.38 |
| DeepInv | 63.88 | 0.01 | **98.67** | 0.08 | 25.06 | 0.01 |
| DFQ | 98.88 | 32.29 | 0.11 | **99.95** | 0.70 | 5.32 |
| IPR | 40.92 | 0.03 | 18.27 | 0.22 | **99.12** | 0.01 |
| ZSKT | 67.68 | 99.05 | 0.01 | 87.99 | 0.27 | **99.99** |
| Average* | **74.1±22.3** | 30.7±38.4 | 3.9±7.2 | 37.9±45.9 | 6.2±9.6 | 5.8±9.1 |

*: the calculation of average detection rates excludes the results on original testing data and results for which training and testing data are generated by the same method.

Table 2. Detecting images that are inverted from a victim model trained on CIFAR10. The training data include images inverted by both CMI and IPR. The detection rates are measured in percentage.

| CIFAR10 Test | CMI | DAFL | DeepInv | DFQ | IPR | ZSKT | Average |
|---|---|---|---|---|---|---|---|
| 4.74 | 94.48 | 95.95 | 79.98 | 97.25 | 98.26 | 88.77 | 92.45±6.35 |

SVM model while the remaining 20% were used for testing. Table 1 presents detection results when only one model inversion method was involved in training the SVM model. The well trained SVM model was tested on original CIFAR10 testing set and images inverted by each model inversion method.

The first row "CIFAR10 Test" indicates the proportion of original CIFAR10 testing images that are recognized by the SVM model as inverted data. We can see that our method consistently maintained false positive rates less than 10%. The best detection performance was achieved when the training and testing data involvde the same type of model inversion method. The SVM model achieved over 98% detection rate for these cases, except for CMI that the model achieved 94% detection rate.

The bottom row shows the generalization of detection when the SVM model is applied to novel inversion methods that are not included during the training stage. When the training data included images inverted by CMI, the SVM achieved the best generalization for the other inversions methods with over 74% detection rate. However, the detection performance is lower than 10% when the training data involve DeepInv, IPR or ZSKT. This means the SVM model could not be used to detect inverted images in such cases.

Generally speaking, increasing the diversity of the training data improves generalization. Empirically, we figured out that it was sufficient to mix the original CIFAR10 training data with the data inverted by both CMI and IPR to train a SVM model. Table 2 shows the detection result on original CIFAR10 testing data and data inverted by each inversion method. It can be observed that the detection rates are at least above 79% for all the cases with a low false positive rate equal to 4.74%. The results demonstrate that the SVM model trained on data inverted by CMI and IPR is able to achieve good detection performance in practice.

### 6.3 GTSRB Results

This section presents experimental results on the victim model trained on GTSRB. Recall that the neural network used for the GTSRB is VGG11, which is different from the WideResNet trained on CIFAR10. Examples of data

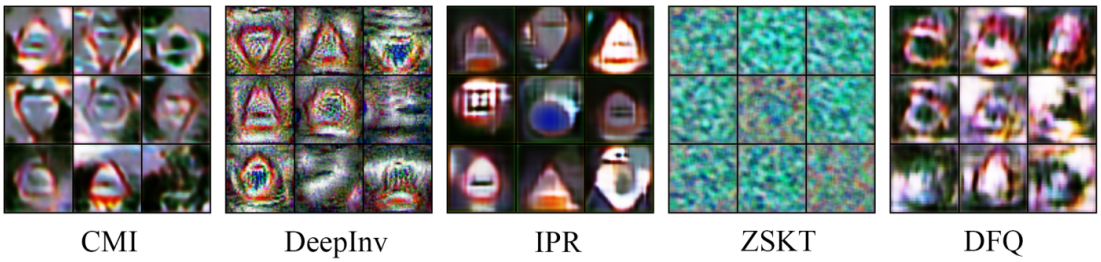


Fig. 4. Examples of inverted GTSRB images obtained using different model inversion methods. All the methods are described in detail in Section 2.1.

Table 3. Detecting images that are inverted from a victim model trained on GTSRB. The detection rates are measured in percentage.

| **Tested On \ Trained On** | CMI | DAFL | DeepInv | DFQ | IPR | ZSKT |
|---|---|---|---|---|---|---|
| GTSRB Test | 0.03 | 0.01 | 0.01 | 0.02 | 0.03 | 0.01 |
| CMI | **99.94** | 4.01 | 4.00 | 96.72 | 73.08 | 3.18 |
| DAFL | 89.47 | **99.99** | 97.79 | 98.40 | 3.25 | 15.00 |
| DeepInv | 46.38 | 2.40 | **99.99** | 54.33 | 21.25 | 1.61 |
| DFQ | 97.20 | 12.48 | 11.50 | **99.96** | 75.70 | 5.35 |
| IPR | 18.62 | 0.01 | 0.10 | 47.85 | **99.95** | 0.01 |
| ZSKT | 57.60 | 85.46 | 99.58 | 77.91 | 4.09 | **99.99** |
| Average* | 61.9±28.8 | 29.4±32.6 | 42.6±45.9 | **75.0**±20.9 | 35.5±32.4 | 5.0±5.3 |

*: the calculation of average detection rates excludes the results on original testing data and results for which training and testing data are generated by the same method.

generated using different inversion methods are shown in Figure 4. In a similar manner to CIFAR10, it can be observed that different model inversion methods result in inverted images with different visual patterns and images inverted by ZSKT are significantly different from the other inverted images.

In the same manner as the evaluation for CIFAR10, we used each method to invert the same number of images as the GTSRB training set. The inverted images were then randomly mixed with the original training images for training. 80% of inverted data were used to train an SVM model while the remaining 20% were used for testing. Table 3 presents detection results when only one model inversion method was involved in training the SVM model. The well trained SVM model was tested on original GTSRB testing set and images inverted by each the model inversion method.

The first row "GTSRB Test" indicates the proportion of original GTSRB testing images that are recognized by the SVM model as inverted data. We can see that our method consistently maintained false positive rates less than 1%. The best detection performance was achieved when the training and testing data involved the same type of model inversion method. The SVM model achieved over 99.90% detection rate for these cases.

The bottom row shows the generalization of detection when the SVM model is applied to novel inversion methods that are not included during the training stage. When the training data included images inverted by DFQ, the SVM achieved the best generalization for the other inversions methods with over 75% detection rate.

Table 4. Detecting images that are inverted from a victim model trained on GTSRB. The training data include images inverted by both CMI and IPR. The detection rates are measured in percentage.

| **Dataset** | GTSRB Test | CMI | DAFL | DeepInv | DFQ | IPR | ZSKT | Average |
|---|---|---|---|---|---|---|---|---|
| **Result** | 0.06 | 99.99 | 99.84 | 82.29 | 99.98 | 99.98 | 99.66 | 96.96±6.56 |


Fig. 5. Examples of distorted images.

The detection performance is lower than 10% only when the training data involve ZSKT. This shows that the generalization on GTSRB is overall better than CIFAR10 because our method only fails for ZSKT.

To evaluate whether increasing the diversity in the training data can improve generalization, we mixed the original GTSRB training data with the data inverted by both CMI and IPR to train a SVM model. Table 4 shows the detection result on original GTSRB testing data and data inverted by each inversion method. The overall results demonstrate a clear improvement compared to those obtained on CIFAR-10. It can be observed that the detection rates are at least above 82% for all the cases with a low false positive rate equal to 0.06%. The results demonstrate that the SVM model trained on data inverted by CMI and IPR is able to achieve excellent detection performance in practice.

## 6.4 Robustness against Distortions

To bypass our detection, attackers may deliberately introduce distortions to their inverted data. In this section, we evaluate whether our method is robust against common distortions. We applied Gaussian blur, Gaussian noise, and JPEG compression to inverted images before inputting them to the SVM model. Figure 5 shows examples of distorted images.

Table 5 shows the experimental results for the inverted CIFAR10 data. In the experiments, we used the SVM model that is train on original CIFAR10 training data mixed with data inverted by both CMI and IPR. The results show that none of the transformations decreased our detection rates. Although the false positive rates were increased due to the transformations, there were still large gaps between the false positive rates and the detection rates, which means our method was still effective in practice.

Empirical results for inverted GTSRB data are similar to those on CIFAR10 and are not repeated here.

## 6.5 Ablation Study

Our hybrid feature extraction strategy plays an important role in our method. To investigate the benefits of combining features from the data themselves and features from the victim model, we conduct a ablation study in this section.

Table 5. Evaluating robustness for inverted CIFAR10 data. The detection rates are measured in percentage.

| Dataset | No Distortion | Gaussian Blur (Medium) | Gaussian Blur (Severe) | Gaussian Noise | JPEG Compression |
|---|---|---|---|---|---|
| CIFAR10 Test | 4.74 | 12.58 | 21.89 | 18.60 | 17.71 |
| CMI | 94.48 | 93.83 | 93.88 | 67.06 | 88.68 |
| DAFL | 95.95 | 85.60 | 83.07 | 94.74 | 95.59 |
| DeepInv | 79.98 | 79.93 | 83.42 | 85.20 | 71.85 |
| DFQ | 97.25 | 92.12 | 92.21 | 77.42 | 82.25 |
| IPR | 98.26 | 96.38 | 96.02 | 92.92 | 93.23 |
| ZSKT | 88.77 | 93.57 | 95.58 | 81.70 | 87.28 |
| Average* | 92.45±6.35 | 90.24±5.67 | 90.70±5.41 | 83.17±9.38 | 86.48±7.81 |

*: The calculation of average detection rates excludes the results on CIFAR10 testing data.

Table 6. Detection performance for each inversion method when using only image features (i.e., features extracted by a pre-trained ResNet18) to train an SVM. The SVM was trained with CIFAR10 augmented by CMI-inverted samples. Although detection performance on CMI is high, detection performance on other inversion methods degrades markedly, demonstrating limited cross-method generalization of imagine features.

| **Dataset** | CIFAR10 Test | CMI | DAFL | DeepInv | DFQ | IPR | ZSKT | Average* |
|---|---|---|---|---|---|---|---|---|
| **Result** | 3.78 | 97.71 | 69.53 | 17.44 | 76.30 | 15.23 | 3.39 | 36.38±20.39 |

*: The calculation of average detection rates excludes the results on original testing data and CMI-inverted data.

Table 6 shows results when only features from the data themselves (i.e., features extracted by a pre-trained ResNet18) were used to train an SVM model. The training data included the original CIFAR10 training data mixed with data inverted by CMI. As expected, the model exhibits a high detection rate of 97.71% when recognizing data inverted by CMI. However, the detection rates for the other inversion methods dropped significantly compared to the results shown in Table 1. The average detection rate decreased from 74% to 36%.

On the other hand, Table 7 presents results when only features extracted from the victim model were used to train the SVM classifier. The training data again consisted of the original CIFAR10 training set mixed with data inverted by CMI. For comparison, we evaluated detection performance using features extracted from different layers, including the first convolutional layer ("Conv1"), the first block ("Block1"), the second block ("Block2"), the third block ("Block3"), and the activation values from the last hidden layer ("Our Method").

The results clearly show that using the activation values from the last hidden layer yields the highest detection rate, while also producing the largest margin between the false positive rate and the detection rate, thereby facilitating the identification of inverted data. Nevertheless, this approach also increases the false positive rate significantly from 3.78% to 14.77%.

Importantly, when these model features are combined with image features extracted by ResNet18 (as shown in Table 1), the detection rate further improves while keeping the false positive rate low. This finding suggests that image-level and model-level features are complementary to each other, and the hybrid feature extraction strategy is the key to defending against complex inversion attacks. We emphasize that using either image features or model features alone can be considered as baseline approaches, since they respectively represent the most direct "data-centric" and "model-centric" perspectives. However, these baseline approaches are limited when facing diverse and complex inversion attacks. For example, relying solely on image features leads to poor cross-method generalization, while using only model features tends to incur higher false positive rates.

Table 7. Detection results when only using features extracted from the victim model. The detection rates are measured in percentage.

| Dataset | Conv1 | Block1 | Block2 | Block3 | Our Method$^{\dagger}$ |
|---|---|---|---|---|---|
| CIFAR10 Test | 7.96 | 0.69 | 1.48 | 3.58 | 14.77 |
| CMI | 98.23 | 99.74 | 99.36 | 96.88 | 88.13 |
| DAFL | 41.05 | 0.01 | 0.03 | 0.95 | 82.66 |
| DeepInv | 15.34 | 16.28 | 55.45 | 68.74 | 57.29 |
| DFQ | 33.43 | 7.37 | 12.82 | 10.90 | 90.97 |
| IPR | 4.79 | 3.35 | 8.50 | 27.70 | 43.65 |
| ZSKT | 1.63 | 92.41 | 0.028 | 86.58 | 42.38 |
| Average$^{*}$ | $19.25 \pm 15.56$ | $23.88 \pm 34.69$ | $15.37 \pm 20.64$ | $38.97 \pm 33.21$ | **63.39±20.00** |
| Gap$^{\#}$ | $11.29 \pm 15.56$ | $23.19 \pm 34.69$ | $13.89 \pm 20.64$ | $35.39 \pm 33.21$ | **48.62±20.00** |

$^{\dagger}$: our method extracts activation values from the last hidden layer.
$^{*}$: the calculation of average detection rates excludes the results on original testing data and CMI-inverted data.
$^{\#}$: the gap between the false positive rate and the average detection rates.

Our approach does not depend on the sophistication of the classifier itself (SVM is chosen as a simple yet effective tool), but rather benefits primarily from the design of the hybrid feature representation, which leverages the complementary strengths of both perspectives. This hybrid design enables even simple classifiers such as SVM to achieve competitive performance in separating benign data from inverted data, suggesting that the effectiveness of the system stems more from the hybrid features than from the classifier choice.

## 7 Limitation and Future Work

As discussed in Section 3, our threat model assumes that the application of our method requires the access to the training data of a suspect model. Although it may be achieved with the help of the authority, this inevitably limits our method in scenarios when the training data are not accessible. For example, adversaries can deliberately delete all the training data after their models are well trained. A potential solution is to extend our current method such that it can also detect generative model inversion attacks based on the weights of the suspect model alone. We leave exploring this interesting direction to future work.

## 8 Conclusion

In this paper, we proposed a novel method to detect IP infringement on inverted data. Our method complements existing DNN watermarking and fingerprinting techniques when adversaries exploit generative model inversion attacks. Our method trains a classifier on hybrid features that are extracted both from the data and from the victim model under protection. Experimental results showed that our method effectively identified inverted data and demonstrated strong generalization to unseen inversion techniques. Furthermore, we empirically showed that our method was robust against common distortions. We also conducted an ablation study to justify our choice of extracting hybrid features for training the classifier.

## References

S. An, G. Tao, Q. Xu, Y. Liu, G. Shen, Y. Yao, J. Xu, and X. Zhang. 2022. "Mirror: Model inversion for deep learning network with high fidelity." In: *Proceedings of the 29th Network and Distributed System Security Symposium.*

X. Cao, J. Jia, and N. Z. Gong. 2021. "IPGuard: Protecting intellectual property of deep neural networks via fingerprinting the classification boundary." In: *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, 14–25.

L. Charette, L. Chu, Y. Chen, J. Pei, L. Wang, and Y. Zhang. 2022. "Cosine model watermarking against ensemble distillation." In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36, 9512–9520.

J. Chen, J. Wang, T. Peng, Y. Sun, P. Cheng, S. Ji, X. Ma, B. Li, and D. Song. 2022. "Copy, right? a testing framework for copyright protection of deep learning models." In: *2022 IEEE symposium on security and privacy (SP)*. IEEE, 824–841.

Y. Choi, J. Choi, M. El-Khamy, and J. Lee. 2020. "Data-free network quantization with adversarial knowledge distillation." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 710–711.

G. Fang, J. Song, X. Wang, C. Shen, X. Wang, and M. Song. 2021. "Contrastive model inversion for data-free knowledge distillation." In: *arXiv*, 1–7.

M. Fredrikson, S. Jha, and T. Ristenpart. 2015. "Model inversion attacks that exploit confidence information and basic countermeasures." In: *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, 1322–1333.

M. Fredrikson, E. Lantz, S. Jha, S. Lin, D. Page, and T. Ristenpart. 2014. "Privacy in pharmacogenetics: An {End-to-End} case study of personalized warfarin dosing." In: *23rd USENIX security symposium (USENIX Security 14)*, 17–32.

H. Jia, C. A. Choquette-Choo, V. Chandrasekaran, and N. Papernot. 2021. "Entangled watermarks as a defense against model extraction." In: *30th USENIX security symposium (USENIX Security 21)*, 1937–1954.

A. Krizhevsky, G. Hinton, et al.. 2009. *Learning multiple layers of features from tiny images*. http://www.cs.utoronto.ca/~kriz/learning-features-2009-TR.pdf. (2009).

E. Le Merrer, P. Perez, and G. Trédan. 2020. "Adversarial frontier stitching for remote neural network watermarking." *Neural Computing and Applications*, 32, 9233–9244.

N. Maslej et al.. 2023. *Artificial intelligence index report 2023*. https://arxiv.org/abs/2310.03715. arXiv, (2023).

S. Mehnaz et al.. 2022. "Are your sensitive attributes private? novel model inversion attribute inference attacks on classification models." In: *31st USENIX Security Symposium (USENIX Security 22)*, 4579–4596.

P. Micaelli and A. J. Storkey. 2019. "Zero-shot knowledge transfer via adversarial belief matching." *Advances in Neural Information Processing Systems*, 32.

N. Papernot, P. McDaniel, A. Sinha, and M. P. Wellman. 2018. "Sok: Security and privacy in machine learning." In: *2018 IEEE European symposium on security and privacy (EuroS&P)*. IEEE, 399–414.

A. Sengupta, Y. Ye, R. Wang, C. Liu, and K. Roy. 2019. "Going deeper in spiking neural networks: VGG and residual architectures." *Frontiers in neuroscience*, 13, 95.

C. Song, T. Ristenpart, and V. Shmatikov. 2017. "Machine learning models that remember too much." In: *Proceedings of the 2017 ACM SIGSAC Conference on computer and communications security*, 587–601.

J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. 2011. "The German traffic sign recognition benchmark: a multi-class classification competition." In: *The 2011 international joint conference on neural networks*. IEEE, 1453–1460.

F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart. 2016. "Stealing machine learning models via prediction {APIs}." In: *25th USENIX security symposium (USENIX Security 16)*, 601–618.

Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh. 2017. "Embedding watermarks into deep neural networks." In: *Proceedings of the 2017 ACM on international conference on multimedia retrieval*, 269–277.

K. Yang, R. Wang, and L. Wang. 2022. "MetaFinger: Fingerprinting the Deep Neural Networks with Meta-training." In: *IJCAI*, 776–782.

Z. Yang, J. Zhang, E.-C. Chang, and Z. Liang. 2019. "Neural network inversion in adversarial setting via background knowledge alignment." In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 225–240.

H. Yin, P. Molchanov, J. M. Alvarez, Z. Li, A. Mallya, D. Hoiem, N. K. Jha, and J. Kautz. 2020. "Dreaming to distill: Data-free knowledge transfer via deepinversion." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8715–8724.

S. Zagoruyko and N. Komodakis. 2016. *Wide residual networks*. https://arxiv.org/abs/1605.07146. (2016).

Y. Zhang, R. Jia, H. Pei, W. Wang, B. Li, and D. Song. 2020. "The secret revealer: Generative model-inversion attacks against deep neural networks." In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 253–261.

W. Zong, Y.-W. Chow, W. Susilo, J. Baek, J. Kim, and S. Camtepe. 2024. "IPRemover: A Generative Model Inversion Attack against Deep Neural Network Fingerprinting and Watermarking." In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38, 7837–7845.

Received 10 June 2025; accepted 27 September 2025