On the Robustness of Dataset Inference

Anonymous authors
Paper under double-blind review

Abstract

Machine learning (ML) models are costly to train as they can require a significant amount of data, computational resources and technical expertise. Thus, they constitute valuable intellectual property that needs protection from adversaries wanting to steal them. *Ownership verification* techniques allow the victims of model stealing attacks to demonstrate that a suspect model was in fact stolen from theirs.

Although a number of ownership verification techniques based on watermarking or fingerprinting have been proposed, most of them fall short either in terms of security guarantees (well-equipped adversaries can evade verification) or computational cost. A fingerprinting technique, *Dataset Inference* (DI), has been shown to offer better robustness and efficiency than prior methods.

The authors of DI provided a correctness proof for linear (suspect) models. However, in a subspace of the same setting, we prove that DI suffers from high false positives (FPs) – it can incorrectly identify an independent model trained with non-overlapping data from the same distribution as stolen. We further prove that DI also triggers FPs in realistic, non-linear suspect models. We then confirm empirically that DI in the black-box setting leads to FPs, with high confidence.

Second, we show that DI also suffers from false negatives (FNs) – an adversary can fool DI (at the cost of incurring some accuracy loss) by regularising a stolen model's decision boundaries using adversarial training, thereby leading to an FN. To this end, we demonstrate that black-box DI fails to identify a model adversarially trained from a stolen dataset – the setting where DI is the hardest to evade.

Finally, we discuss the implications of our findings, the viability of fingerprinting-based ownership verification in general, and suggest directions for future work.

1 Introduction

Machine learning (ML) models are being developed and deployed at an increasingly faster rate and in several application domains. For many companies, they are not just a part of the technological stack that offers an edge over the competitors but a core business offering. Hence, ML models constitute valuable intellectual property that needs to be protected.

Model stealing is considered one of the most serious attack vectors against ML models (Kumar et al., 2019). The goal of a model stealing attack is to obtain a functionally equivalent copy of a victim model that can be used, for example, to offer a competing service, or avoid having to pay for the use of the model.

In the *white-box* attack, the adversary obtains the exact copy of the victim model, for example by reverse engineering an application containing an embedded model (Deng et al., 2022). In contrast, in *black-box* attacks (known as *model extraction* attacks) (Papernot et al., 2017; Orekondy et al., 2019; Tramèr et al., 2016) the adversary gleans information about the victim model via its predictive interface. Two possible approaches to defend against model extraction are 1) detection (Juuti et al., 2019; Atli et al., 2020; Zheng et al., 2022) and 2) prevention (Orekondy et al., 2020; Mazeika et al., 2022; Dziedzic et al., 2022). However, a powerful, yet realistic attacker can circumvent these defenses (Atli et al., 2020).

An alternative defense applicable to both white-box and black-box model theft is based on *deterrence*. It concedes that the model will eventually get stolen. Therefore, an *ownership verification* technique that can identify and demonstrate a suspect model as having been stolen can serve as a deterrent against model theft. Early research in this field focused on *watermarking* based on embedding triggers or backdoors (Zhang et al., 2018; Uchida et al., 2017; Adi et al., 2018) into the weights of the model. Unfortunately, all watermarking schemes were shown to be brittle (Lukas et al., 2022) in that an attacker can successfully remove the watermark from a protected stolen model without incurring a substantial loss in model utility.

An alternative approach to ownership verification is *fingerprinting*. Instead of embedding a trigger or backdoor in the model, one can extract a fingerprint that matches only the victim model, and models derived from it. Fingerprinting works both against white-box and black-box attacks, and does not affect the performance of the model. Although several fingerprinting schemes have been proposed, some are not rigorously tested against model extraction (Cao et al., 2021; Pan et al., 2022) and others can be computationally expensive to derive (Lukas et al., 2021).

In this backdrop, *Dataset Inference* (DI) (Maini et al., 2021) promises to be an effective fingerprinting mechanism. Intuitively, it leverages the fact that if model owners trained their models on *private data*, knowledge about that data can be used to identify all stolen models. DI was shown to be effective against white-box and black-box attacks and is efficient to compute (Maini et al., 2021). It was also shown not to conflict with any other defenses (Szyller & Asokan, 2022). Given its promise, the guarantees provided by DI merits closer examination.

In this work, we first show that DI suffers from false positives (FPs) — it can incorrectly identify an independent model trained with *non-overlapping data from the same distribution* as stolen. The authors of DI provided a correctness proof for a linear model. However, DI in fact suffers from **high FPs**, unless two assumptions hold: (1) a large noise dimension, as explained in the original paper and (2) a large proportion of the victim's training data is used during ownership verification, as we prove in this paper. Both of these assumptions are unrealistic in a subspace of the linear case used by DI: (i) we prove that a large noise dimension can lead to low accuracy in the resulting model , and (ii) revealing too much of the victim's (private) training data is detrimental to privacy. Furthermore, we prove that DI also triggers FPs in realistic, non-linear models. We then confirm empirically that DI leads to FPs, with high confidence in the black-box verification setting, "*black-box* DI", where the DI verifier has access only to the inference interface of a suspect model, but not its internals .

We also show that black-box DI suffers from false negatives (FNs): an adversary who has in fact stolen a victim model can avoid detection by regularising their model with adversarial training. We provide empirical evidence that an adversary who steals the victim's dataset itself and adversarially trains a model can evade detection by DI by trading off accuracy of the stolen model.

We claim the following contributions:

- Following the same simplified theoretical analysis used by the original paper (Maini et al., 2021), in a subspace of the linear case used by DI, we show that for a linear suspect model, 1) high-dimensional noise (as required in (Maini et al., 2021) leads to **low model accuracy** (Lemma 3.3, Section 3.1), and 2) DI **suffers from FPs** unless a large proportion of private data is revealed during ownership verification (Theorem 3.2, Section 3.1);

- Extending the analysis to non-linear suspect models, using a PAC-Bayesian framework (Neyshabur et al., 2018), we show that DI suffers from **FPs in non-linear models** regardless of how much private data is revealed (Theorem 3.4, Section 3.2.1);

- We empirically demonstrate the existence of **FPs in a realistic black-box DI** setting (Section 3.2.2);

- We show empirically that black-box DI also **suffers from FNs**: using adversarial training to regularise the decision boundaries of a stolen model can successfully evade detection by DI at the cost of some accuracy ($\approx$ 6-13pp) (Section 4);

2 Dataset Inference Preliminaries

Dataset Inference (DI) aims to determine whether a *suspect model* f_{SP} was obtained by an adversary $\mathcal{A}$ who has stolen a model ($f_{\mathcal{A}}$) derived from a victim $\mathcal{V}$'s private data $\mathcal{S}_V$, or belongs to an independent party $\mathcal{I}$ ($f_{\mathcal{I}}$). DI relies on the intuition that if a model is derived from $\mathcal{S}_V$, this information can be identified from all models. DI measures the *prediction margin*s of a suspect model around private and public samples: distance from the samples to the model's decision boundaries. If f_{SP} has distinguishable decision boundaries for private and public samples DI deems it to be *stolen*; otherwise the model is deemed *independent*.

In the rest of this section, we explain the theoretical framework that DI uses — consisting of a linear suspect model — the embedding generation necessary for using DI with realistic non-linear suspect models, and the verification procedure. A summary of the notation used throughout this work appears in Table 1.

2.1 Theoretical Framework

The original DI paper (Maini et al., 2021) used a linear suspect model to theoretically prove the guarantees provided by DI. We first explain how DI works in this setting.

Setup. Consider a data distribution $\mathcal{D}$, such that any input-label pair $(\boldsymbol{x}, y)$ can be described as:

$$y \sim \{-1, +1\}, \boldsymbol{x_1} = y \cdot \boldsymbol{u} \in \mathbb{R}^K, \boldsymbol{x_2} \sim \mathcal{N}(0, \sigma^2 I) \in \mathbb{R}^D,$$

where $\boldsymbol{x} = (\boldsymbol{x_1}, \boldsymbol{x_2}) \in \mathbb{R}^{K+D}$ and $\boldsymbol{u} \in \mathbb{R}^K$ is a fixed vector. The last D dimensions of $\boldsymbol{x}$ represent Gaussian noise (with variance σ^2).

Structure of the linear model. Assuming a linear model f, with weights $\boldsymbol{w} = (\boldsymbol{w_1}, \boldsymbol{w_2})$, such that $f(x) = \boldsymbol{w_1} \cdot \boldsymbol{x_1} + \boldsymbol{w_2} \cdot \boldsymbol{x_2}$, then the final classification decision is $sgn(f(x))$. With the weights initialized to zero, f learns the weights using gradient descent with learning rate 1 until $yf(x)$ is maximized. Given a private training dataset $\mathcal{S}_V \sim \mathcal{D} = \{(x^{(i)}, y^{(i)}) | i = 1, ..., m\}$, and a public dataset $\mathcal{S}_0 \sim \mathcal{D}$ (both of size m), then $\boldsymbol{w_1} = m\boldsymbol{u}$ and $\boldsymbol{w_2} = \sum_{i=1}^{m} y^{(i)} \boldsymbol{x_2}^{(i)}$ regardless of the batch size.

In DI, the prediction margin $p(\cdot)$ is used to imply the confidence of f in its prediction. It is defined as the margin (distance) of a data point from the decision boundary.

$$p(x) \triangleq y \cdot f(x). \tag{1}$$

The authors (Maini et al., 2021) show that the difference of expected prediction margins of two datasets $\mathcal{S}_V$ and $\mathcal{S}_0$ is $D\sigma^2$. The threshold can be set $\lambda \in (0, D\sigma^2)$, and by estimating the difference of the prediction margins on $\mathcal{S}_0$ and $\mathcal{S}_V$ on f_{SP}, DI is able to distinguish whether that model is stolen.

Note that DI uses approximations of the prediction margins based on embeddings. The theoretical framework assumes that the approximations are accurate, and we can use them directly for the theoretical analysis (Equation 1). For the linear model, the margins can be computed analytically; however, in Section 2.2, we explain how the approximations of the margins are obtained.

2.2 Embedding Generation

In order to use DI one needs to generate *embeddings* of the samples. $\mathcal{V}$ queries their model $f_{\mathcal{V}}$ with samples in their private dataset $\mathcal{S}_V$ and public dataset $\mathcal{S}_0$, and assigns the labels $b = 1$ and $b = 0$ respectively. The authors propose two methods of generating the embeddings: a white-box approach (MinGD) and a black-box one (Blind Walk). In this work, we use only Blind Walk as it outperforms MinGD in most experimental setups in the original work, and is more realistic, as it only requires access to the API of the suspect model.

Blind Walk estimates the prediction margin of a sample by measuring its robustness to random noise. For a sample $(\boldsymbol{x}, y)$, to compute the margin, first choose a random direction δ, and take $k \in \mathbb{N}$ steps in the same direction until the misclassification $f(\boldsymbol{x} + k\delta) \neq y$. This is repeated multiple times to increase the size of the embedding. As reported in (Maini et al., 2021), obtaining embeddings for 100 samples can take up to $30,000$ queries.

Table 1: Summary of the notation used throughout this work.

$\mathcal{V}$	the victim	f	a model
$\mathcal{I}$	an independent party	$f_{\mathcal{V}}$	a model trained on $\mathcal{S}_V$
$\mathcal{A}$	an adversary	f_0	a model trained on $\mathcal{S}_0$
$\mathcal{S}$	a dataset	$f_{\mathcal{I}}$	a model trained on $\mathcal{S}_I$
$\mathcal{S}_V$	$\mathcal{V}$'s private dataset	$f_{\mathcal{A}}$	$\mathcal{A}$'s model
$\mathcal{S}_0$	a public dataset	f_{SP}	a suspect model
$\mathcal{S}_I$	$\mathcal{I}$'s data	$\boldsymbol{w}$	model weights
$\mathcal{D}$	distribution that all datasets follow	$g_{\mathcal{V}}$	regression model
$(\boldsymbol{x}, y)$	a sample from $\mathcal{D}$	D	noise dimension

Having obtained the embeddings, $\mathcal{V}$ trains a regression model $g_{\mathcal{V}}$ that predicts the confidence that a sample contains private information from $\mathcal{S}_V$.

2.3 Ownership Verification

Using the scores from $g_{\mathcal{V}}$ and the membership labels, $\mathcal{V}$ creates vectors $\boldsymbol{c}$ and $\boldsymbol{c}_V$ of equal size from $\mathcal{S}_V$ and $\mathcal{S}_0$, respectively. Then for a null hypothesis $H_0 : \mu < \mu_V$ where $\mu = \bar{c}$ and $\bar{\mu} = \bar{c_V}$ are mean confidence scores. The test rejects H_0 and rules that the suspect model is 'stolen', or gives an inconclusive result.

To verify whether f_{SP} is stolen or independent, $\mathcal{V}$ obtains the embeddings by querying the model (using Blind Walk) using samples from $\mathcal{S}_V$ and $\mathcal{S}_0$. Then they use the embeddings to obtain the confidence scores from the $g_{\mathcal{V}}$, and perform a hypothesis test on the two distributions of scores.

3 False Positives in Dataset Inference

To generate the embeddings for a specific sample in the private dataset $\mathcal{S}_V$, DI requires querying the suspect model f_{SP} hundreds of times. To reduce the total number of queries, DI was shown to be effective with only 10 private samples with at least 95% confidence. Additionally, DI requires a large random noise dimension D such that probability of success increases to 1 as $D \to \infty$. In this section, we prove that these two assumptions are not realistic in a subspace of the linear case used by DI: 1) DI is susceptible to false positives (FPs) unless $\mathcal{V}$ reveals a large number of samples; 2) a large D will harm the utility of the model (Section 3.1).

Furthermore, we find that the theoretical results on linear suspect models which say that the margins on different models are distinguishable with some strict conditions do not hold for more realistic non-linear suspect models. Using a PAC-Bayesian margin based generalization bound (Neyshabur et al., 2018) we prove that models trained on the same distribution are indistinguishable, and will trigger FPs (Section 3.2.1. Next, we provide empirical evidence for the existence of FPs (Section 3.2.2).

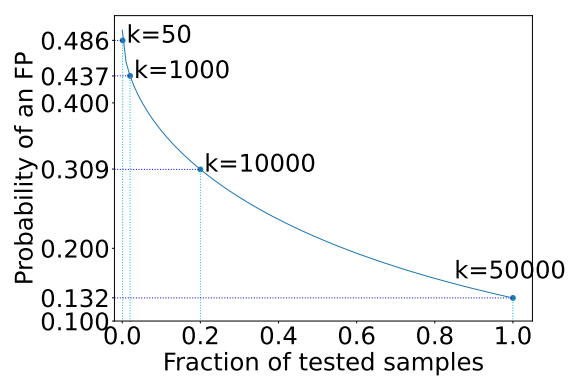

Figure 1: Probability of an FP as the fraction of revealed private samples for $D = 10$ for a linear suspect model (Equation 6). $\mathcal{V}$ needs to use many private samples to guarantee a low false positive rate.

3.1 Linear Suspect Models

In Section 2, we have a set up for linear models. DI shows their success theoretically on the linear set up where $\boldsymbol{u} \in \mathbb{R}^K$ and $\sigma \in \mathbb{R}$.

Theorem 3.1 (Success of DI (Maini et al., 2021))**.** *Choose $b \leftarrow \{0, 1\}$ uniformly at random. Given an adversary's linear classifier f trained on $\mathcal{S} \sim \mathcal{D}, s.t. |S| = \mathcal{S}_V$ if $b = 0$, and on $\mathcal{S}_V$ otherwise. The proba-*

bility $\mathcal{V}$ correctly decides if an adversary stole its knowledge $\mathbb{P}[\Psi(f, \mathcal{S}_V; \mathcal{D}) = b] = 1 - \Phi(-\frac{\sqrt{D}}{2\sqrt{2}})$. Moreover, $lim_{D\to\infty}\mathbb{P}[\Psi(f, \mathcal{S}_V; \mathcal{D}) = b] = 1$.

However, realistic datasets often contain more noise and lack the correct signal. We consider a subspace where the signal $||\boldsymbol{u}||$ is upper-bounded and the noise is larger. We construct the subspace by assuming that $||\boldsymbol{u}||_2 \leq \frac{1}{\sqrt{m}}$ and $\sigma^2 > \frac{1}{10\sqrt{m}}$. Note that the linear model should correctly classify most of the randomly picked data from this subspace, we use Lemma 3.2 to analyze the accuracy of the linear model in this subspace. The results show that only the subspace where D is bounded can guarantee high performance for the model.

Lemma 3.2 (Need for Bounding Noise Dimension)**.** *Let f be a linear model trained on $\mathcal{S} \sim \mathcal{D}$. For a sample $(\boldsymbol{x}, y)$ sampled from $\mathcal{D}$ which is independent of $\mathcal{S}$, f correctly classifies the sample with a probability $\mathbb{P}[yf(x) \geq 0] = 1 - \Phi(-\frac{m\boldsymbol{u}^2}{\sqrt{mD\sigma^2}})$. Assuming that $||\boldsymbol{u}||_2 \leq \frac{1}{\sqrt{m}}$ and $\sigma^2 > \frac{1}{10\sqrt{m}}$, the accuracy of f is $\mathbb{P}[yf(x) \geq 0] \leq 1 - \Phi(-\frac{10}{\sqrt{D}})$. Therefore, the model f trained on the $\mathcal{D}$ can achieve high accuracy only if D is bounded.*

The details of the proof are in the Appendix 9. Lemma 3.2 shows that if the dimension of $\boldsymbol{x}_2$, which follows $\mathcal{N}(0, \sigma^2)$, is large, then the noise will dominate f and mislead it into making incorrect predictions. For example, for a dataset with more than 500 samples, the std of $\boldsymbol{x_2}$ is 0.25 (close to the CIFAR10 dataset) where $\sigma^2 = 0.0625 > \frac{1}{10\sqrt{m}} = \frac{1}{10\sqrt{500}} \approx 0.0044$. If $D = 1000$, f can correctly classify a sample that is different from f's training set with a probability up to $1 - \Phi(-\frac{10}{\sqrt{1000}}) = 0.6241$. However, if $D = 10$, f can achieve an accuracy of 0.9992.

Since $\mathcal{V}$ limits the number of samples used for the verification, Theorem 3.3 show that the false positives (FPs) rate is directly related to the samples used for the verification in our subspace.

Theorem 3.3 (Existence of False Positives with Linear Suspect Models)**.** *Let $f_{\mathcal{I}}$ be a linear classifier trained on the independent dataset $\mathcal{S}_I \sim \mathcal{D}$. Let k be the number of samples estimated required for the verification. Then, the probability that $\mathcal{V}$ mistakenly decides that $f_{\mathcal{I}}$ is a stolen model is $\mathbb{P}[\Psi(f_{\mathcal{I}}, \mathcal{S}_V; \mathcal{D}) = 1] > 1 - \Phi(\frac{\sqrt{kD}}{2\sqrt{2m}})$.*

Where Ψ is $\mathcal{V}$'s decision function (Maini et al., 2021):

$$\Psi(f_{SP}, \mathcal{S}; \mathcal{D}) = \begin{cases} 1, \ if \ f_{SP} \sim f_{\mathcal{A}}, \\ 0, \ if \ f_{SP} \sim f_{\mathcal{I}}, \end{cases} \tag{2}$$

Proof. Recall that $\mathcal{V}$ tries to reveal only a few samples during the verification. For a distribution $\mathcal{D}$ where $||\boldsymbol{u}|| \leq \frac{1}{\sqrt{m}}$ and $\sigma^2 > \frac{1}{\sqrt{m}}$.

Following the intuition from DI (Yeom et al., 2018), for satisfactory performance, DI must minimise both false positives and false negatives. Hence, the objective function is defined as:

$$min_\lambda \frac{\mathbb{P}[\Psi(f_{\mathcal{I}}, \mathcal{S}_V; \mathcal{D}) = 1] + \mathbb{P}[\Psi(f_{\mathcal{V}}, \mathcal{S}_V; \mathcal{D}) = 0]}{2}, \tag{3}$$

where the margin of $\mathcal{D}$ is estimated using $\mathcal{S}_V$ and $\mathcal{S}_0$. Note that we are only interested in the false positives $\mathbb{P}[\Psi(f_{\mathcal{I}}, \mathcal{S}_V; \mathcal{D}) = 1]$, let $\mathcal{S}_I = \{(x^{(i)}, y^{(i)}) | i = 1, ..., m\}$, $\mathcal{S}_*^k$ be a subset of $\mathcal{S}_*$ consisting of k samples.

$$\begin{aligned} &\mathbb{P}[\Psi(f_{\mathcal{I}}, \mathcal{S}_V; \mathcal{D}) = 1] \\ &= \mathbb{P}[E_{(\boldsymbol{x},y)\in\mathcal{S}_V^k}[yf_{\mathcal{I}}(\boldsymbol{x})] - E_{(\boldsymbol{x},y)\in\mathcal{S}_0^k}[yf_{\mathcal{I}}(\boldsymbol{x})] \geq \lambda] \\ &= \mathbb{P}[E_{(\boldsymbol{x},y)\in\mathcal{S}_V^k}[\sum_i^m y^{(i)}\boldsymbol{x_2}^{(i)}\boldsymbol{x_2}] \ - E_{(\boldsymbol{x},y)\in\mathcal{S}_0^k}[\sum_i^m y^{(i)}\boldsymbol{x_2}^{(i)}\boldsymbol{x_2}] \geq \lambda] \\ &= \mathbb{P}[\frac{1}{k}\sum_j^k\sum_i^m y^{(i)}\boldsymbol{x_2}^{(i)}\boldsymbol{x_2}^{(j)} - \frac{1}{k}\sum_p^k\sum_i^m y^{(i)}\boldsymbol{x_2}^{(i)}\boldsymbol{x_2}^{(p)} \geq \lambda]. \end{aligned} \tag{4}$$

Recall that $\boldsymbol{x_2}^{(i)}$, $\boldsymbol{x_2}^{(j)}$ and $\boldsymbol{x_2}^{(p)}$ are D-dimensional vectors sampled independently from $\mathcal{N}(0, \sigma^2)$. Using the central limit theorem we can approximate the terms. We have $\sum_i^m y^{(i)}\boldsymbol{x_2}^{(i)} \sim \mathcal{N}(0, m\sigma^2)$. Then, we can approximate $\frac{1}{k}\sum_j^k\sum_i^m y^{(i)}\boldsymbol{x_2}^{(i)}\boldsymbol{x_2}^{(j)}$ by $t_1 \sim \mathcal{N}(0, \frac{mD}{k}\sigma^4)$ and approximate $\frac{1}{k}\sum_p^k\sum_i^m y^{(i)}\boldsymbol{x_2}^{(i)}\boldsymbol{x_2}^{(p)}$ by $t_2 \sim \mathcal{N}(0, \frac{mD}{k}\sigma^4)$ (Maini et al., 2021). Thus, we get $t \sim \mathcal{N}(0, \frac{2mD}{k}\sigma^4)$, and

$$\begin{aligned}
&\mathbb{P}[\Psi(f_{\mathcal{I}}, \mathcal{S}_V; \mathcal{D}) = 1] = \mathbb{P}[t \geq \lambda] \\
&= \mathbb{P}[\sqrt{\frac{2mD}{k}}\sigma^2 Z \geq \lambda] = \mathbb{P}[Z \geq \frac{\sqrt{k}\lambda}{\sqrt{2mD}\sigma^2}] \\
&= 1 - \Phi(\frac{\sqrt{k}\lambda}{\sqrt{2mD}\sigma^2}),
\end{aligned} \tag{5}$$

where $Z \sim \mathcal{N}(0, 1)$. The optimal threshold is given as $\lambda = \frac{D\sigma^2}{2}$,

$$\mathbb{P}[\Psi(f_{\mathcal{I}}, \mathcal{S}_V; \mathcal{D}) = 1] = 1 - \Phi(\frac{\sqrt{kD}}{2\sqrt{2m}}). \tag{6}$$

From Equation 6, we see that the probability of false positives relies on the number of points used for the verification k and the size of D. □

In our subspace where D is bounded for the model with high accuracy, the success of DI is directly related to the number of samples used for the verification. This is similar to the analysis of failure of membership inference in the original paper when the k is extremely low, e.g. only 10 samples. In the DI paper, it was explained that DI succeeds because it calculates the average margin for multiple verification samples; whereas membership inference fails as it relies on per-sample decision. So when the number of tested samples is smaller, the success rate of DI will be close to 0.5, just like for membership inference. In Figure 1, we show the probability of an FP (Equation 6) for different values of k; even for $k = 10000$ the probability is 0.309 .

Hence, even the simple linear setup, $\Psi(f, \mathcal{S}; \mathcal{D})$ has false positives with high probability; in particular, when the fraction of tested samples is small.

3.2 Non-linear Suspect Models

Having demonstrated the limitations of the linear model, we now focus on non-linear suspect models. The intuition is based on the margin-based generalization bounds. Note that the generalization bounds states that the expected error of the margin based loss function is bounded, and the bound is mostly related to the distribution (Neyshabur et al., 2018). Since DI assumes all the datasets follow the distribution $\mathcal{D}$, our intuition is to directly use the generalization bounds and the triangle inequality to prove the similarity of the models trained on the same distribution.

3.2.1 Theoretical Motivation

Let $f_{\boldsymbol{w}}$ be a real-valued classifier $f_{\boldsymbol{w}} : \mathcal{X} \to \mathbb{R}^k$, $||x|| \leq B$ with parameters $\boldsymbol{w} = \{W_i\}_{i=1}^d$. For any distribution $\mathcal{D}$ and margin $p(f, \boldsymbol{x}) = f(\boldsymbol{x})[y] - max_{j \neq y} f(\boldsymbol{x})[j] \leq \gamma$, where $\gamma > 0$. The margin is the same as for the linear model with labels $y \in \{-1, +1\}$. Then, we define the margin loss function as:

$$\mathcal{L}_\gamma(f, y) = \mathbb{P}_{(\boldsymbol{x},y)\sim\mathcal{D}}[f(x)[y] - max_{j\neq y} f(x)[j] \leq \gamma]. \tag{7}$$

Note that the PAC-Bayes framework (Neyshabur et al., 2018) provides guarantees for any classifier f trained on data from a given distribution. We define the expected loss of a classifier f on distribution $\mathcal{D}$ as $\mathcal{L}_{\mathcal{D}} := E_{(\boldsymbol{x},y)\sim\mathcal{D}}[\mathcal{L}(f(\boldsymbol{x}), y)]$ and the empirical loss on a dataset $\mathcal{S}$ as $\hat{\mathcal{L}}_{\mathcal{S}} := \frac{1}{m}\sum_{(\boldsymbol{x},y)\in\mathcal{S}}[\mathcal{L}(f(\boldsymbol{x}), y)]$. Then, for a $d-$layer feed-forward network f with parameters $\boldsymbol{w} = \{W_i\}_{i=1}^d$ and ReLU activation (Neyshabur et al., 2018). The empirical loss is very close to the expected loss. For any $\sigma, \gamma > 0$, with probability $1 - \sigma$ over the training set, we have:

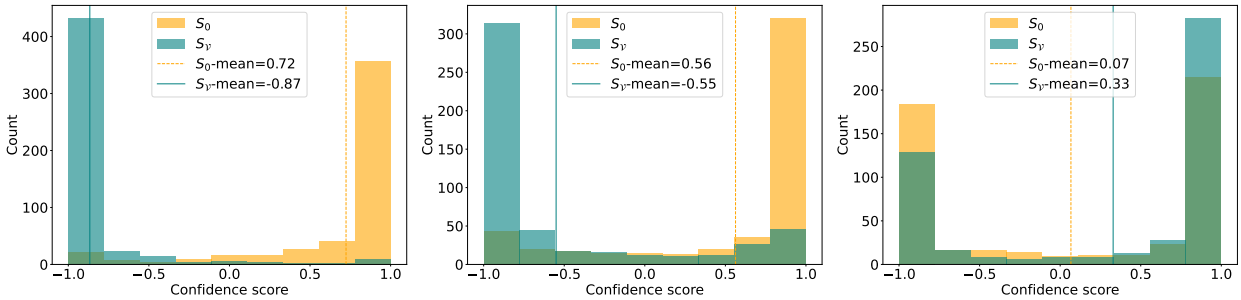

Figure 2: Left to right: $f_\mathcal{V}$, $f_\mathcal{I}$, f_0. Comparison of distributions of the confidence scores assigned to the embeddings by $g_\mathcal{V}$. $\Delta\mu$ is smaller for $f_\mathcal{I}$ than for $f_\mathcal{V}$ but large enough to trigger an FP.

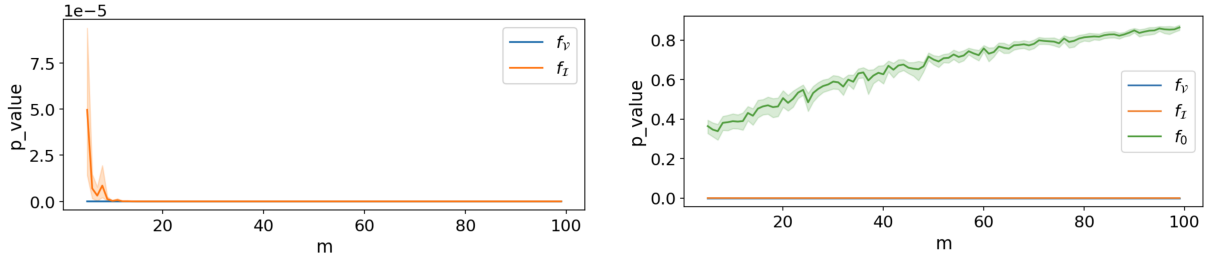

Figure 3: **Left:** Comparison of the verification confidence of $f_\mathcal{V}$ and $f_\mathcal{I}$. FP becomes stronger (lower p-value) as more samples are revealed. **Right:** same comparison, however, we include f_0 to show the desirable behaviour of an independent model.

$$|\mathcal{L}_\mathcal{D}(f_\mathcal{S}) - \hat{\mathcal{L}}_\mathcal{S}(f_\mathcal{S})| \leq \mathcal{O}(\epsilon), \tag{8}$$

where $\epsilon = \sqrt{\frac{B^2 d^2 h ln(dh) \prod_{i=1}^{d} ||W_i||_2^2 \sum_{i=1}^{d} \frac{||W_i||_F^2}{||W_i||_2^2} + ln\frac{dm}{\sigma}}{\gamma^2 m}}$, and h is the upper bound dimension for $\{W_i\}_{i=1}^d$.

This PAC-Bayes based generalization guarantee states that for a model f, the distance between the empirical loss and the expected loss is bounded, and the bound can be very small when the model's margin is large. Thus, we can expect that the margins of f on any dataset that follows a given distribution to be bounded by $O(\epsilon)$. This contradicts the intuition of DI. DI's intuition is that the difference of margins of two datasets $\mathcal{S}_V$ and $\mathcal{S}_I$ on the model f is large enough such that DI is able to distinguish whether the model f is stolen.

Moreover, since DI assumes that $\mathcal{S}_V$ and $\mathcal{S}_I$ follow the same distribution $\mathcal{D}$, we can show that the margins for $f_\mathcal{V}$ and $f_\mathcal{I}$ are similar to each other.

Theorem 3.4 (k-independent False Positives with Non-linear Suspect Models)**.** *For the victim private dataset $\mathcal{S}_V \sim \mathcal{D}$ and an independent dataset $\mathcal{S}_I \sim \mathcal{D}$, let $f_{\boldsymbol{w}}$ be a $d-$layer feed-forward network with ReLU activations and parameters $\boldsymbol{w} = \{W_i\}_{i=1}^d$. Assume that $f_\mathcal{V}$ is trained on $\mathcal{S}_V$ and $f_\mathcal{I}$ is trained on $\mathcal{S}_I$, $f_\mathcal{V}$ and $f_\mathcal{I}$ have the same structure. Then, for any $B, d, h, \epsilon > 0$ and any $\boldsymbol{x} \in \mathcal{X}$, there exist a prior $\mathcal{P}$ on $\boldsymbol{w}$, s.t. with probability at least $\frac{1}{2}$,*

$$|E(p(f_\mathcal{V}, \boldsymbol{x}) - p(f_\mathcal{I}, \boldsymbol{x}))| \leq \epsilon. \tag{9}$$

The details of the proof are in the Appendix 9. Hence, for any two models trained on the same distribution, the expectation of margins for any sample are similar. Given that DI works by distinguishing the difference of margins for two models, it will result in false positives with probability at least $\frac{1}{2}$ (Theorem 3.4).

3.2.2 Empirical Evidence

Having proved the existence of FPs for non-linear models, we now focus on empirically confirming it.

Table 2: Verification of an independent model trained on the same data distribution triggers an FP. Also, we report the accuracy of the models on the test set. We provide the mean and standard deviation computed across five runs. Verification done using $k = 10$ private samples. FPs become more significant as k increases. FPs are highlighted.

	CIFAR10			CIFAR100		
Model	Accuracy	$\Delta\mu$	p-value	Accuracy	$\Delta\mu$	p-value
$f_{\mathcal{V}}$	0.87 ± 0.03	1.62 ± 0.08	$10^{-18} \pm 10^{-18}$	0.65 ± 0.00	1.80 ± 0.04	$10^{-30} \pm 10^{-30}$
$f_{\mathcal{I}}$	0.87 ± 0.03	1.14 ± 0.12	$10^{-8} \pm 10^{-8}$	0.66 ± 0.01	0.32 ± 0.03	$10^{-2} \pm 10^{-2}$
f_0	0.64 ± 0.02	-0.29 ± 0.12	0.46 ± 0.04	0.51 ± 0.01	-0.37 ± 0.11	0.67 ± 0.15

First, recall the original experiment setup (Maini et al., 2021); let us consider the following two models: 1) $f_{\mathcal{V}}$ trained using $\mathcal{S}_V$, and 2) f_0 trained using $\mathcal{S}_0$. In the original formulation, e.g. for CIFAR10, CIFAR10-train ($50,000$ samples) is used as $\mathcal{S}_V$, and CIFAR10-test is used as $\mathcal{S}_0$ ($10,000$ samples). Recall that $\mathcal{V}$ uses their $\mathcal{S}_V$ and $\mathcal{S}_0$ to obtain the embeddings that are then used to train the regression model $g_{\mathcal{V}}$.

DI was shown to be effective against several post-processing used to obtain *dependent models* which are expected to be flagged as stolen - true positives However, the independent model f_0 is trained on $\mathcal{S}_0$ — the same data that is used to train $g_{\mathcal{V}}$. This means that the same dataset $\mathcal{S}_0$ is used both to train $g_{\mathcal{V}}$ and subsequently, to evaluate it. This is likely to introduce a bias that overestimates the efficacy of $g_{\mathcal{V}}$ and DI as a whole.

To address this, and test whether DI works for a more reasonable data split, we use the following setup:

1) randomly split CIFAR10-train into two subsets (A_{train} and B_{train}) of $25,000$ samples each;

2) assign $\mathcal{S}_V = A_{train}$, and train $f_{\mathcal{V}}$ using it;

3) continue using CIFAR10-test as $\mathcal{S}_0$ (nothing changes), and train f_0 using it;

4) g_V is trained using the embedding for $\mathcal{S}_0$ and the new $\mathcal{S}_V$, obtained from the new $f_{\mathcal{V}}$;

5) assign $\mathcal{S}_I = B_{train}$, independent data of a third-party $\mathcal{I}$, who trains their model $f_{\mathcal{I}}$.

This way, we have an *independent* model $f_{\mathcal{I}}$ that was trained on data from the same distribution $\mathcal{D}$ as $\mathcal{S}_V$ but data that was not seen by $g_{\mathcal{V}}$[1]. We use an analogous split for CIFAR100.

Recall that to determine whether the model is stolen, DI obtains the embeddings for private ($\mathcal{S}_V$) and public ($\mathcal{S}_0$) samples. Then it measures the confidence for each of the embeddings using the regressor $g_{\mathcal{V}}$. For a model derived from $\mathcal{V}$'s $\mathcal{S}_V$, the mean difference ($\Delta\mu$) between the confidence assigned to $\mathcal{S}_V$ and $\mathcal{S}_0$ should be large. If the model is not derived from $\mathcal{S}_V$, the difference should be small. The decision is made using the hypothesis test that compares the distributions of measures from $g_{\mathcal{V}}$.

In Figure 2 we visualise the difference in the distributions for three models. For $f_{\mathcal{V}}$ we observe two separable distributions with a large ($\Delta\mu$), while for f_0 the difference is small — DI is working as intended. However, for $f_{\mathcal{I}}$, even though $\Delta\mu$ is smaller than for $f_{\mathcal{V}}$ it is sufficiently large to reject H_0 with high confidence. Therefore, $f_{\mathcal{I}}$ is marked as stolen, a false positive, In Table 2 we provide $\Delta\mu$ and the associated p-values for multiple random splits. In Figure 3, we show the results for verification, using Blind Walk, with more data (up to $k = 100$ private samples). As we increase the number of revealed private samples, the confidence of DI increases both for $f_{\mathcal{V}}$ (true positive) and $f_{\mathcal{I}}$ (false positive).

We discuss the implications of our findings in Section 6.

[1]We use the official implementation of DI, together with the architectures and training loops. Our changes are limited to the data splits only.

4 False Negatives in Dataset Inference

Having demonstrated the existence of false positives, we now show that DI can suffer from false negatives (FNs). $\mathcal{A}$ can avoid detection by regularising $f_{\mathcal{A}}$, and thus changing the prediction margins. This in turn, will mislead DI into flagging $f_{\mathcal{A}}$ as independent.

Recall that Blind Walk relies on finding the prediction margin by querying perturbed samples designed to cause a misclassification. In order to avoid detection, $\mathcal{A}$ needs to make the prediction margin robust to such perturbations. We do so using adversarial training: a popular regularisation method used to provide robustness against adversarial examples.

The original paper considered three adversary models: 1) $\mathcal{A}_{\mathcal{D}}$ who has access to $\mathcal{V}$'s $\mathcal{S}_V$; 2) $\mathcal{A}_{\mathcal{Q}}$ who has query (i.e. black-box) access to $f_{\mathcal{V}}$, and corresponds to a model extraction attack; 3) $\mathcal{A}_{\mathcal{M}}$ who has white-box access to $f_{\mathcal{V}}$.

$\underline{\mathcal{A}_{\mathcal{D}}}$: we evaluate adversarial training as a way to avoid detection in the $\mathcal{A}_{\mathcal{D}}$ setting – $\mathcal{A}_{\mathcal{D}}$ steals $\mathcal{V}$'s $\mathcal{S}_V$ and trains their own model $f_{\mathcal{A}}$. Also, $f_{\mathcal{A}}$ has the same architecture and hyperparameters as $f_{\mathcal{V}}$, but is adversarially trained [2].

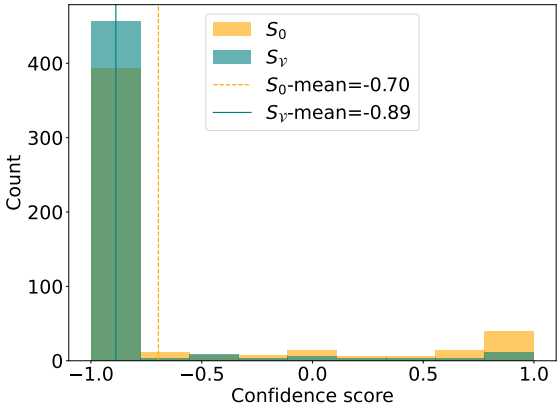

Figure 4: Confidence scores assigned to embeddings by $g_{\mathcal{V}}$ obtained from $f_{\mathcal{A}}$. $\Delta\mu$ is small enough to trigger FNs.

During adversarial training, each training sample $(\boldsymbol{x}, y)$ is replaced with an adversarial example that is misclassified $f_{\mathcal{A}}(\boldsymbol{x} + \gamma) \neq y$. There exist many techniques for crafting adversarial examples. We use *projected gradient descent* (Madry et al., 2018) (PGD), and we set $\gamma = 10/255$ (under l_∞).

In Figure 4 we visualise the difference in the distributions of scores assigned by $g_{\mathcal{V}}$ to $f_{\mathcal{A}}$ embeddings derived for $\mathcal{S}_V$ and $\mathcal{S}_0$. We observe that the distributions are not clearly separable and result in low $\Delta\mu$, and hence H_0 cannot be rejected. Therefore, $f_{\mathcal{A}}$ is marked as an independent model, a false negative. In Table 3 we provide $\Delta\mu$ and the associated p-values for multiple runs.

Note that adversarial training comes with an accuracy trade-off. In our experiments, the accuracy of $f_{\mathcal{A}}$ goes from 0.92 ± 0.01 to 0.86 ± 0.01 (CIFAR10), and 0.73 ± 0.03 to 0.60 ± 0.01 (CIFAR100). However, recent work shows that one can adversarially train models without incurring as much accuracy loss (Cui et al., 2021; Debenedetti et al., 2022; Wang et al., 2023). We study how the amount of noise affects the verification in Section 5.2. Also, we discuss the resulting implications in Section 6.

We consider $\mathcal{A}_{\mathcal{D}}$ in detail because triggering FNs against DI via adversarial training is harder for $\mathcal{A}_{\mathcal{D}}$ – the models have the same architecture, and are trained using exactly the same data.

$\underline{\mathcal{A}_{\mathcal{Q}}}$: the same approach also applies in the $\mathcal{A}_{\mathcal{Q}}$ setting – $\mathcal{A}_{\mathcal{Q}}$ conducts a model extraction attack with their attack dataset, and can train $f_{\mathcal{A}}$ with adversarial training (without needing access to $\mathcal{S}_V$).

$\underline{\mathcal{A}_{\mathcal{M}}}$: while this approach is not directly applicable to the $\mathcal{A}_{\mathcal{M}}$ adversary, an adversary with $\mathcal{A}_{\mathcal{M}}$ capabilities and sufficient data (not the same as $\mathcal{S}_V$) can indeed mount the same attack as $\mathcal{A}_{\mathcal{Q}}$ by distilling the model themselves.

5 Towards Countermeasures

In Sections 3 and 4 we showed that DI suffers from false positives and false negatives in certain settings. In this section, we explore whether potential countermeasures exist. First, we analyse possible tweaks to $g_{\mathcal{V}}$

[2] We use the official implementation of DI, together with the architectures and training loops. Our changes are limited to adding adversarial training.

Table 3: $f_{\mathcal{A}}$ adversarially trained on $\mathcal{S}_V$ results in a false negative. Also, we report the accuracy of the models on the test set. We provide the mean and standard deviation computed across five runs. Verification done using $k = 10$ private samples. FNs are are highlighted.

	CIFAR10			CIFAR100		
Model	Accuracy	$\Delta\mu$	p-value	Accuracy	$\Delta\mu$	p-value
$f_{\mathcal{V}}$	0.92 ± 0.01	1.59 ± 0.04	$10^{-21} \pm 10^{-16}$	0.73 ± 0.03	1.58 ± 0.10	$10^{-31} \pm 10^{3}1$
$f_{\mathcal{A}}$	0.86 ± 0.01	0.12 ± 0.06	0.15 ± 0.07	0.60 ± 0.01	0.11 ± 0.02	0.43 ± 0.18
f_0	0.64 ± 0.02	-0.29 ± 0.12	0.46 ± 0.04	0.51 ± 0.01	-0.37 ± 0.11	0.67 ± 0.15

Table 4: Verification of independent models trained on the same data distribution as $f_{\mathcal{V}}$: a) using more data to train $g_{\mathcal{V}}$; b) using a bigger, four-layer $g_{\mathcal{V}}$ trained with dropout. We provide the mean and standard deviation computed across five runs. Verification done using $k = 10$ private samples. FPs are highlighted.

Model	Accuracy	$\Delta\mu$	p-value
$f_{\mathcal{V}}$	0.87 ± 0.03	1.34 ± 0.08	$10^{-31} \pm 10^{-30}$
$f_{\mathcal{I}}$	0.87 ± 0.03	0.83 ± 0.13	$10^{-11} \pm 10^{-10}$
f_0	0.64 ± 0.02	0.08 ± 0.01	0.15 ± 0.08

(a) FPs remain when $g_{\mathcal{V}}$ is trained with more data.

Model	Accuracy	$\Delta\mu$	p-value
$f_{\mathcal{V}}$	0.87 ± 0.03	1.30 ± 0.17	$10^{-17} \pm 10^{-16}$
$f_{\mathcal{I}}$	0.87 ± 0.03	0.84 ± 0.16	$10^{-7} \pm 10^{-6}$
f_0	0.64 ± 0.02	0.05 ± 0.01	0.37 ± 0.30

(b) FPs remain when using a bigger $g_{\mathcal{V}}$.

that could help alleviate the FPs (Section 5.1). Next, we test if increasing the amount of noise used during the embedding generation avoids FNs despite adversarial training (Section 5.2).

5.1 Avoiding FPs via a Modified Distinguisher

Recall that DI does not suffer from any FPs in the original data split where both the f_0 and $g_{\mathcal{V}}$ are trained on $\mathcal{S}_0$. One possible explanation for the FPs is that $g_{\mathcal{V}}$ overfits to $\mathcal{S}_0$, and does not generalise to $\mathcal{S}_I$. To verify this hypothesis, we augment $\mathcal{S}_0$ with a portion of $\mathcal{S}_I$ (of size $|\mathcal{S}_0|$), which is then used to train $g_{\mathcal{V}}$. For $f_{\mathcal{I}}$, the p-value still remains low and hence triggers an FP (Table 4a).

We then consider an alternative hypothesis — $g_{\mathcal{V}}$ underfits and fails to predict the margins correctly. To examine this, we increase the size of $g_{\mathcal{V}}$. The original $g_{\mathcal{V}}$ is a two-layer network with *tanh* activation. We increase the number of layers to four. Although this does not eliminate the FPs, it does increase the average p-value (Table 4b).

In conclusion, neither of these approaches alleviates the FPs. The margins of different models trained on the same distribution (estimated by $g_{\mathcal{V}}$) are difficult to distinguish. Finding a robust approach to avoid FPs in DI remains an open problem.

5.2 Avoiding FNs via Verification with More Noise

If $\mathcal{V}$ suspects that $\mathcal{A}$ might be using adversarial training to avoid detection, they can carry out the verification with more noise in the attempt to circumvent the effect of adversarially training $f_{\mathcal{A}}$.

In the experiments presented in Section 4, the average noise added during Blind Walk is 0.12 ± 0.05 (under ℓ_∞), and adversarial training is done with $\gamma = 10/255 (\approx 0.039)$. In this experiment, we vary the number of maximum steps taken by $\mathcal{V}$, and hence the maximum amount of added noise. We consider $\{25, 50, 100, 200\}$ steps (baseline 50 steps) which corresponds to $\{0.10 \pm 0.03, 0.12 \pm 0.05, 0.33 \pm 15, 0.38 \pm 23\}$ noise added (under ℓ_∞) during the verification. $\mathcal{V}$ needs to use more noise for the verification of *any* model, hence we also conduct the experiment for f_0 (for $\{50, 100\}$ steps). The goal is to ensure that the increased amount of noise does not have any negative impact on the independent models. Table 5 summarizes our experiment results. Using more steps does not improve the result against $f_{\mathcal{A}}$ compared to the baseline: 1) for CIFAR10,

Table 5: Impact of added noise (maximum number of perturbation steps) during verification on DI success (**baseline 50** steps). Using more noise does not prevent FNs against $f_{\mathcal{A}}$ but increases the standard deviation across all experiments, thus negatively impacting verification of f_0. We provide the mean and standard deviation computed over five runs. Verification done using $k = 10$ private samples. FNs are highlighted.

	CIFAR10				CIFAR100			
Model	Accuracy	Steps	$\Delta\mu$	p-value	Accuracy	Steps	$\Delta\mu$	p-value
$f_{\mathcal{V}}$	0.92 ± 0.01	**50**	1.59 ± 0.04	$10^{-21} \pm 10^{-16}$	0.73 ± 0.03	**50**	1.58 ± 0.10	$10^{-31} \pm 10^{3}1$
$f_{\mathcal{A}}$	0.86 ± 0.01	25	0.09 ± 0.04	0.09 ± 0.07	0.60 ± 0.01	25	0.12 ± 0.02	0.39 ± 0.22
		50	0.12 ± 0.06	0.15 ± 0.07		**50**	0.11 ± 0.02	0.43 ± 0.18
		100	0.10 ± 0.05	0.08 ± 0.09		100	0.13 ± 0.01	0.50 ± 0.16
		200	0.14 ± 0.08	0.16 ± 0.11		200	0.12 ± 0.01	0.47 ± 0.24
f_0	0.64 ± 0.02	**50**	-0.29 ± 0.12	0.46 ± 0.04	0.51 ± 0.01	**50**	-0.37 ± 0.11	0.67 ± 0.15
		100	-0.19 ± 0.16	0.37 ± 0.12		100	-0.43 ± 0.03	0.72 ± 0.09

the standard deviation of the p-value increases; 2) we do not observe any linear relationship between the noise and $\Delta\mu$ or the associated p-value (neither for CIFAR10 or CIFAR100).

On the other hand, for CIFAR10, the confidence of the verification of f_0 decreases. The standard deviations of $\Delta\mu$ and its associated p-value increase. Although the p-value remains sufficiently high, using more noise has a negative impact on the verification of f_0. On the contrary, for CIFAR100, the confidence of the verification of f_0 increases: $\Delta\mu$ is lower, p-value is higher, and standard deviations decrease.

In conclusion, increasing the amount of noise during Blind Walk does not allow $\mathcal{V}$ to circumvent $\mathcal{A}$'s adversarial training. Hence, DI remains susceptible to false negatives induced by adversarial training.

6 Discussion

Revealing private data. We have shown that DI requires revealing significantly more than 50 samples to avoid false positives in the case of linear models (Figure 1). Since the core assumption of DI is that $\mathcal{S}_V$ is private, revealing too much of $\mathcal{S}_V$ during the ownership verification constitutes a privacy threat. In neither of the settings described in Section 5 of the original DI paper the victim *cannot query the model sufficiently* without leaking the query data to the adversary. Additionally, it was shown that using more samples gives $\mathcal{V}$ more information about the prediction margin than using stronger embedding methods (Maini et al., 2021). Model owners that operate in sensitive domains such as healthcare or insurance industry need to comply with strict data protection laws, and hence need to minimise the disclosure.

One potential way to protect the privacy of the private samples used for DI ownership verification is to use oblivious inference (Liu et al., 2017; Juvekar et al., 2018). This way $\mathcal{V}$ could query $f_{\mathcal{A}}$ without revealing $\mathcal{S}_V$. Despite recent advances in efficient oblivious inference (Samragh et al., 2021; Watson et al., 2022; Samardzic et al., 2021; 2022), it requires *all* parties (including $\mathcal{A}$!) to update their software stacks which may not always be realistic.

Viability of ownership verification using training data. We have demonstrated that DI suffers from FPs when faced with an independent model trained on the same distribution. While it is reasonable to assume that $\mathcal{V}$'s data is private, the uniqueness of the distribution is difficult to guarantee in practice. For example, two model builders may have data from the same distribution because they purchased their training data from a vendor that generates per-client synthetic data from the same distribution (e.g., regional financial data). In fact, two model builders working on the same narrow domain and independently building models that are intended to represent the same phenomenon, may very well end up using data from the same distribution.

There are other methods that attempt to detect stolen models based on the dataset used to train them (Sablayrolles et al., 2020; Pan et al., 2022). However, they rely on flaws in the model to establish

the ownership (susceptibility to adversarial examples (Sablayrolles et al., 2020) or membership inference attacks (Pan et al., 2022)). Intuitively, given a perfect membership inference attack, a fingerprinting scheme should be possible. However, recent work shows that for a balanced dataset, only a fraction of records is vulnerable to a confident membership inference attack (Carlini et al., 2022; Duddu et al., 2021) which in turn reduces the capabilities of a membership inference-based fingerprinting scheme. Therefore, any improvements to generalisation or robustness (such as adversarial training or purification (Nie et al., 2022)) of ML models reduce the surface for ownership verification schemes.

Nevertheless, leveraging advancements in membership inference might make DI more robust, e.g. regularising DI with many shadow models to avoid FPs, or using MIAs that were shown to be more effective against adversarially trained models that overfit (Song et al., 2019a;b; Hayes, 2020). However, it is worth noting that the orignal DI paper (Maini et al., 2021) differentiates DI from membership inference. DI relies on weak signals over many records unlike MIA that needs strong signal for individual records. Hence, relying on ideas from MIAs introduces additional considerations, e.g. MIA evasion.

False positives in other schemes. Our work raises the question whether other watermarking/fingerprinting schemes are also vulnerable. Existing schemes for model ownership verification typically focus on robustness against adversaries who want to evade detection. Systematically examining such schemes for false positives can be an interesting line of future work. An interesting question is whether a malicious model owner can successfully accuse independent models as stolen by systematically inducing false positives.

White-box theft. Our experiments in Section 4 are limited to $\mathcal{A}$ that trains their own model — they either steal the data or conduct a model extraction attack. If $\mathcal{A}$ obtains an exact copy of the model, they might lack the data to fine-tune it with adversarial training. Hence, our findings do not apply to the white-box setting. We leave the examination of other threat models out as future work.

Black-box vs. white-box verification setting. Our evaluation is focused on the black-box DI setting. We do not consider the white-box DI setting which uses MinGD. While white-box DI is feasible in a scenario where $\mathcal{V}$ takes $\mathcal{A}$ (the holder of a suspect model) to court, requiring $\mathcal{A}$ to provide white-box access to the suspect model, prosecution is an expensive undertaking. Realistically $\mathcal{V}$ is likely to first conduct black-box DI to decide whether the expense of prosecution is justified. Therefore, FPs in the black-box DI setting can cause substantial monetary loss to $\mathcal{V}$.

7 Conclusion

We analyzed *Dataset Inference* (DI) Maini et al. (2021), a promising fingerprinting scheme, to show theoretically and empirically that DI is prone to false positives in the case of independent models trained from distinct datasets drawn from the same distribution. This limits the applicability of DI only to settings where a model builder uses a dataset with a definitively unique distribution. Alleviating FPs in other settings may not be possible. Furthermore, DI might not be applicable in settings where privacy of the records must be respected, e.g. healthcare. We also showed that an attacker can use adversarial training to regularise the decision boundaries of a stolen model to evade detection by DI at the cost of a drop in accuracy ($6 - 13pp$). Such decrease is acceptable if the model can be further fine-tuned with additional data and recover the performance.

Nevertheless, DI is a promising ML fingerprinting scheme. Model owners can use our results to make informed decisions as to whether DI is appropriate for their particular settings.

References

Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *27th USENIX Security Symposium*, pp. 1615–1631, 2018.

Buse Gul Atli, Sebastian Szyller, Mika Juuti, Samuel Marchal, and N. Asokan. Extraction of complex DNN models: Real threat or boogeyman?". In *"Engineering Dependable and Secure Machine Learning Systems"*, pp. 42–57, 2020.

Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. 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*, ASIA CCS '21, pp. 14–25, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450382878. doi: 10.1145/3433210.3437526. URL https://doi.org/10.1145/3433210.3437526.

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1897–1914, 2022. doi: 10.1109/SP46214.2022.9833649.

Jiequan Cui, Shu Liu, Liwei Wang, and Jiaya Jia. Learnable boundary guided adversarial training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 15721–15730, 2021.

Edoardo Debenedetti, Vikash Sehwag, and Prateek Mittal. A light recipe to train robust vision transformers. *arXiv preprint arXiv:2209.07399*, 2022.

Zizhuang Deng, Kai Chen, Guozhu Meng, Xiaodong Zhang, Ke Xu, and Yao Cheng. Understanding real-world threats to deep learning models in android apps. 2022. doi: 10.48550/arXiv.2209.09577. URL https://arxiv.org/abs/2209.09577v1.

Vasisht Duddu, Sebastian Szyller, and N. Asokan. SHAPr: An efficient and versatile membership privacy risk metric for machine learning. *CoRR*, abs/2112.02230, 2021. URL https://arxiv.org/abs/2112.02230.

Adam Dziedzic, Muhammad Ahmad Kaleem, Yu Shen Lu, and Nicolas Papernot. Increasing the cost of model extraction with calibrated proof of work. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=EAy7C1cgE1L.

Jamie Hayes. Provable trade-offs between private & robust machine learning. *arXiv preprint arXiv:2006.04622*, 2020.

Hengrui Jia, Christopher A. Choquette-Choo, Varun Chandrasekaran, and Nicolas Papernot. Entangled watermarks as a defense against model extraction. 2021a.

Hengrui Jia, Mohammad Yaghini, Christopher A Choquette-Choo, Natalie Dullerud, Anvith Thudi, Varun Chandrasekaran, and Nicolas Papernot. Proof-of-learning: Definitions and practice. In *2021 IEEE Symposium on Security and Privacy (SP)*, pp. 1039–1056. IEEE, 2021b.

Mika Juuti, Sebastian Szyller, Samuel Marchal, and N. Asokan. PRADA: protecting against DNN model stealing attacks. In *IEEE European Symposium on Security & Privacy*, pp. 1–16. IEEE, 2019.

Chiraag Juvekar, Vinod Vaikuntanathan, and Anantha Chandrakasan. GAZELLE: A low latency framework for secure neural network inference. In *27th USENIX Security Symposium (USENIX Security 18)*, pp. 1651–1669, Baltimore, MD, August 2018. USENIX Association. ISBN 978-1-939133-04-5. URL https://www.usenix.org/conference/usenixsecurity18/presentation/juvekar.

Ram Shankar Siva Kumar, Jeffrey Snover, David O'Brien, Kendra Albert, and Salome Viljoen. Failure modes in machine learning. https://learn.microsoft.com/en-us/security/engineering/failure-modes-in-machine-learning, 2019. Online; accessed 20 September 2022.

Taesung Lee, Benjamin Edwards, Ian Molloy, and Dong Su. Defending against neural network model stealing attacks using deceptive perturbations. In *2019 IEEE Security and Privacy Workshops (SPW)*, pp. 43–49, 2019. doi: 10.1109/SPW.2019.00020.

Jian Liu, Mika Juuti, Yao Lu, and N. Asokan. Oblivious neural network predictions via minionn transformations. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, pp. 619–631, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450349468. doi: 10.1145/3133956.3134056. URL https://doi.org/10.1145/3133956.3134056.

Nils Lukas, Yuxuan Zhang, and Florian Kerschbaum. Deep neural network fingerprinting by conferrable adversarial examples. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=VqzVhqxkjH1.

Nils Lukas, Edward Jiang, Xinda Li, and Florian Kerschbaum. Sok: How robust is image classification deep neural network watermarking? In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 787–804, 2022. doi: 10.1109/SP46214.2022.9833693.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Pratyush Maini, Mohammad Yaghini, and Nicolas Papernot. Dataset inference: Ownership resolution in machine learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=hvdKKV2yt7T.

Mantas Mazeika, Bo Li, and David Forsyth. How to steer your adversary: Targeted and efficient model stealing defenses with gradient redirection. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 15241–15254. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/mazeika22a.html.

Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro. A PAC-bayesian approach to spectrally-normalized margin bounds for neural networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Skz_WfbCZ.

Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. Diffusion models for adversarial purification. In *International Conference on Machine Learning (ICML)*, 2022.

Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Knockoff nets: Stealing functionality of black-box models. In *CVPR*, pp. 4954–4963, 2019.

Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Prediction poisoning: Towards defenses against dnn model stealing attacks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SyevYxHtDB.

Xudong Pan, Yifan Yan, Mi Zhang, and Min Yang. Metav: A meta-verifier approach to task-agnostic model fingerprinting. 2022. doi: 10.48550/ARXIV.2201.07391. URL https://arxiv.org/abs/2201.07391.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *ACM Symposium on Information, Computer and Communications Security*, pp. 506–519. ACM, 2017.

E. Quiring, D. Arp, and K. Rieck. Forgotten siblings: Unifying attacks on machine learning and digital watermarking. In *IEEE European Symposium on Security & Privacy*, pp. 488–502, 2018.

Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, and Herve Jegou. Radioactive data: tracing through training. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8326–8335. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/sablayrolles20a.html.

Nikola Samardzic, Axel Feldmann, Aleksandar Krastev, Srinivas Devadas, Ronald Dreslinski, Christopher Peikert, and Daniel Sanchez. F1: A fast and programmable accelerator for fully homomorphic encryption. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO '21, pp. 238–252, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450385572. doi: 10.1145/3466752.3480070. URL https://doi.org/10.1145/3466752.3480070.

Nikola Samardzic, Axel Feldmann, Aleksandar Krastev, Nathan Manohar, Nicholas Genise, Srinivas Devadas, Karim Eldefrawy, Chris Peikert, and Daniel Sanchez. Craterlake: A hardware accelerator for efficient unbounded computation on encrypted data. In *Proceedings of the 49th Annual International Symposium on Computer Architecture*, ISCA '22, pp. 173–187, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450386104. doi: 10.1145/3470496.3527393. URL https://doi.org/10.1145/3470496.3527393.

Mohammad Samragh, Siam Hussain, Xinqiao Zhang, Ke Huang, and Farinaz Koushanfar. On the application of binary neural networks in oblivious inference. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 4625–4634, 2021. doi: 10.1109/CVPRW53098.2021.00521.

Liwei Song, Reza Shokri, and Prateek Mittal. Membership inference attacks against adversarially robust deep learning models. In *2019 IEEE Security and Privacy Workshops (SPW)*, pp. 50–56, 2019a. doi: 10.1109/SPW.2019.00021.

Liwei Song, Reza Shokri, and Prateek Mittal. Privacy risks of securing machine learning models against adversarial examples. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 241–257, 2019b.

Sebastian Szyller and N. Asokan. Conflicting interactions among protection mechanisms for machine learning models. 2022. doi: 10.48550/ARXIV.2207.01991. URL https://arxiv.org/abs/2207.01991.

Sebastian Szyller, Buse Gul Atli, Samuel Marchal, and N. Asokan. *DAWN: Dynamic Adversarial Watermarking of Neural Networks*, pp. 4417–4425. Association for Computing Machinery, New York, NY, USA, 2021. ISBN 9781450386517. URL https://doi.org/10.1145/3474085.3475591.

Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction apis. In *25th USENIX Security Symposium*, pp. 601–618, 2016.

Joel A Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of computational mathematics*, 12(4):389–434, 2012.

Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh. Embedding watermarks into deep neural networks. In *ACM International Conference on Multimedia Retrieval*, pp. 269–277. ACM, 2017.

Zekai Wang, Tianyu Pang, Chao Du, Min Lin, Weiwei Liu, and Shuicheng Yan. Better diffusion models further improve adversarial training. *arXiv preprint arXiv:2302.04638*, 2023.

Jean-Luc Watson, Sameer Wagh, and Raluca Ada Popa. Piranha: A GPU platform for secure computation. In *31st USENIX Security Symposium (USENIX Security 22)*, pp. 827–844, Boston, MA, August 2022. USENIX Association. ISBN 978-1-939133-31-1. URL https://www.usenix.org/conference/usenixsecurity22/presentation/watson.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pp. 268–282, 2018. doi: 10.1109/CSF.2018.00027.

Jialong Zhang, Zhongshu Gu, Jiyong Jang, Hui Wu, Marc Ph Stoecklin, Heqing Huang, and Ian Molloy. Protecting intellectual property of deep neural networks with watermarking. In *ACM Symposium on Information, Computer and Communications Security*, pp. 159–172, 2018.

Rui Zhang, Jian Liu, Yuan Ding, Zhibo Wang, Qingbiao Wu, and Kui Ren. "adversarial examples" for proof-of-learning. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1408–1422. IEEE, 2022.

Huadi Zheng, Qingqing Ye, Haibo Hu, Chengfang Fang, and Jie Shi. Protecting decision boundary of machine learning model with differentially private perturbation. *IEEE Transactions on Dependable and Secure Computing*, 19(3):2007–2022, 2022. doi: 10.1109/TDSC.2020.3043382.

8 Related Work

Model extraction detection and prevention. Detection methods rely on the fact that many extraction attacks have querying patterns that are distinguishable from the benign ones (Juuti et al., 2019; Atli et al., 2020; Zheng et al., 2022; Quiring et al., 2018). All of these can be circumvented by the adversary who has access to natural data from the same domain as the victim model (Atli et al., 2020). Prevention techniques aim to slow down the attack by injecting the noise into the prediction, designed to corrupt the training of the stolen model (Orekondy et al., 2020; Lee et al., 2019; Mazeika et al., 2022), or by making all clients participate in consensus-based cryptographic protocols (Dziedzic et al., 2022). Even though they increase the cost of the attack, they do not stop a determined attacker from stealing the model.

Ownership verification. There exist many watermarking schemes for neural networks (e.g. (Zhang et al., 2018; Uchida et al., 2017; Adi et al., 2018)) that have the same goal as DI does. However, they mostly rely on embedding a *backdoor* into a model that serves as a watermark. Unfortunately, they do not protect against model extraction attacks, and hence, several schemes were proposed that specifically target model extraction (Szyller et al., 2021; Jia et al., 2021a).

It was also shown that adversarial examples (Lukas et al., 2021) can be used to fingerprint a model by finding examples that transfer to models *only* derived from the original. One could also use adversarial examples to watermark datasets by including some in the training set (Sablayrolles et al., 2020). However, adversarial training can be used to weaken both schemes (Lukas et al., 2021; Szyller & Asokan, 2022). On the other hand, if a model is sufficiently vulnerable to membership inference attacks, it can be used to fingerprint it (Pan et al., 2022).

Alternatively, one could prove ownership of the model using a proof of the integrity of training (Proof-of-Learning) (Jia et al., 2021b). Proof-of-learning relies on the fact that successive training checkpoints can serve as a proof of expended compute, and (probabilistic) correctness of the training procedure.

We believe that the following work is the closest to ours. Lukas et al. (2022) showed that all watermarking schemes are brittle. They designed various attacks that successfully remove watermarks, or prevent their embedding. However, they do not evaluate any fingerprinting schemes. Zhang et al. (2022) showed that $\mathcal{A}$ can construct a *spoof* against Proof-of-Learning. $\mathcal{A}$ iteratively creates the spoof by finding adversarial examples that match the successive training steps, hence resulting in a false positive.

9 Existence of False Positives in Dataset Inference

Calculating the prediction margin. We assume that the model weights are initialized to zero. For each sample x in a dataset $\mathcal{S} \sim \mathcal{D} = \{(\boldsymbol{x^{(i)}}, y^{(i)}) | i = 1, ..., m\}, y \sim \{-1, +1\}$. The learning algorithm observes all samples in S once and maximizes the loss function $L(\boldsymbol{x}, y) = y \cdot f(\boldsymbol{x})$. For the learning rate $\alpha = 1$, the weights are updates as:

$$w = w + \alpha y^{(i)} \boldsymbol{x^{(i)}}. \tag{10}$$

Recall that $\boldsymbol{x} = (\boldsymbol{x_1}, \boldsymbol{x_2}) \in \mathbb{R}^{K+D}$, the weights of the linear model are $\boldsymbol{w_1} = m\boldsymbol{u}$ and $\boldsymbol{w_2} = \sum_{i=1}^{m} y^{(i)} \boldsymbol{x_2^{(i)}}$ when the training is completed.

When writing out the linear classifier explicitly, we can easily calculate the prediction margin of each sample (x, y) in $\mathcal{S}$,

$$y \cdot f(\boldsymbol{x}) = y \cdot (\boldsymbol{w_1 x_1} + \boldsymbol{w_2 x_2}) = y \cdot (m\boldsymbol{u} \cdot y\boldsymbol{u} + \sum_{i=1}^{m} y^{(i)} \boldsymbol{x_2^{(i)}} \cdot \boldsymbol{x_2}) = c + y \sum_{i=1}^{m} y^{(i)} \boldsymbol{x_2^{(i)}} \cdot \boldsymbol{x_2}. \tag{11}$$

The expectations of the prediction margin for the points in training set $\mathcal{S}^+ = \{(x, 1) | (x, 1) \in \mathcal{S}\}$ is,

$$\begin{aligned} E_{\mathcal{S}^+}[yf(\boldsymbol{x})] &= yc + E_{\mathcal{S}^+}[\sum_{i=1}^{m} y^{(i)} \boldsymbol{x_2^{(i)}} \cdot \boldsymbol{x_2}] = yc + E_{\mathcal{S}^+}[\sum_{\boldsymbol{x_2} \neq \boldsymbol{x_2^{(i)}}} y^{(i)} \boldsymbol{x_2^{(i)}} \cdot \boldsymbol{x_2} + y\boldsymbol{x_2^2}] \\ &= c + 0 + D\sigma^2. \end{aligned} \tag{12}$$

Note that in Equation 12, since $\boldsymbol{x_2} \sim N(0, D\sigma^2)$, then $\boldsymbol{x_2^2} \sim \chi^2$, $E[\boldsymbol{x_2^{(i)}}] = D\sigma^2$.

Consider a new dataset $\mathcal{S}_0 \sim \mathcal{D}$, the expectations of the prediction margin for the points in $\mathcal{S}_0^+$ are,

$$E_{\mathcal{S}_0^+}[yf(\boldsymbol{x})] = yc + E_{\mathcal{S}_0^+}[\sum_{i=1}^{m} y^{(i)}\boldsymbol{x_2^{(i)}} \cdot \boldsymbol{x_2}] = c. \tag{13}$$

Finally, we see that the difference of prediction margin of training set S and test set $\mathcal{S}_0$ is

$$E_{\mathcal{S}^+}[yf(\boldsymbol{x})] - E_{\mathcal{S}_0^+}[yf(\boldsymbol{x})] = D\sigma^2. \tag{14}$$

DI's decision function. From the above analysis, we know that the statistical difference between the distribution of training and test data is $D\sigma^2$ which is usually larger than 1 in numerical. DI utilizes this difference to predict if a potential adversary's model stole their knowledge.

Since we know that $E_{\mathcal{S}_0}[yf(\boldsymbol{x})] = c$ and $E_{\mathcal{S}}[yf(\boldsymbol{x})] = c + D\sigma^2$. Let $\Psi(f, \mathcal{S}; \mathcal{D})$ represent the dataset inference victim's decision function. It is defined as,

$$\Psi(f, \mathcal{S}; \mathcal{D}) = \begin{cases} 1, \ if \ E_{(x,y)\in\mathcal{S}}[y \cdot f(\boldsymbol{x})] - E_{\mathcal{D}}[y \cdot f(\boldsymbol{x})] \geq \lambda, \\ 0, \ otherwise, \end{cases} \tag{15}$$

where $\lambda \in [0, D\sigma^2]$ is some threshold that the decision function uses to maximise true positives and minimise false positives.

Proof for Lemma 3.2 For a linear model f trained on distribution $\mathcal{D}$ where $\boldsymbol{x} = (\boldsymbol{x_1}, \boldsymbol{x_2})$, $\boldsymbol{x_1} = y\boldsymbol{u}$,$\boldsymbol{x_2} \sim \mathcal{N}(0, \sigma^2)$ and $||\boldsymbol{u}||_2 \leq \frac{1}{\sqrt{m}}$, f is expected to achieve high accuracy on any sample $(\boldsymbol{x}, y)$ sampled randomly from $\mathcal{D}$ which is independent of the training data set of f.

Proof. Given a linear model f trained on dataset $\mathcal{S} \sim \mathcal{D} = \{(\boldsymbol{x^{(i)}}, y^{(i)})|i = 1, ..., m\}$, and a test sample $(\boldsymbol{x}, y)$ sampled randomly from $\mathcal{D}$ which is independent of $\mathcal{S}$, the probability that $(\boldsymbol{x}, y)$ is correctly classified by f can be represented as:

$$\begin{aligned} \mathbb{P}[yf(x) \geq 0] &= \mathbb{P}[m\boldsymbol{u}^2 + y\sum_{i}^{m} y(i)\boldsymbol{x_2^{(i)}}\boldsymbol{x_2} \geq 0] \\ &= \mathbb{P}[y\sum_{i}^{m} y(i)\boldsymbol{x_2^{(i)}}\boldsymbol{x_2} \geq -m\boldsymbol{u}^2] \end{aligned} \tag{16}$$

Since $\boldsymbol{x_2} \sim \mathcal{N}(0, \sigma^2)$ are D-dimensional vectors, we can use the central limit theorem to approximate the term. Thus, the internal term can be approximated by a variable $t \sim \mathcal{N}(0, mD\sigma^4)$. Let $Z \sim \mathcal{N}(0, 1)$,

$$\mathbb{P}[yf(x) \geq 0] \leq \mathbb{P}[\frac{\sqrt{mD}\sigma^2}{m\boldsymbol{u}^2}Z \geq -1] = 1 - \Phi(-\frac{m\boldsymbol{u}^2}{\sqrt{mD}\sigma^2}) \tag{17}$$

where Φ is the normal CDF.

We can also calculate the accuracy of the training set $\mathcal{S}$ similarly. For a training sample $(\boldsymbol{x}, y)$ sampled randomly from $\mathcal{S}$,

$$\begin{aligned} \mathbb{P}[yf(x) \geq 0] &= \mathbb{P}[m\boldsymbol{u}^2 + y\sum_{i}^{m} y(i)\boldsymbol{x_2^{(i)}}\boldsymbol{x_2} \geq 0] \\ &= \mathbb{P}[m\boldsymbol{u}^2 + y^2\boldsymbol{x_2^2} + y\sum_{i}^{m-1} y(i)\boldsymbol{x_2^{(i)}}\boldsymbol{x_2} \geq 0] \\ &= \mathbb{P}[y^2\boldsymbol{x_2^2} + y\sum_{i}^{m} y(i)\boldsymbol{x_2^{(i)}}\boldsymbol{x_2} \geq -m\boldsymbol{u}^2] \\ &\leq \mathbb{P}[y^2\boldsymbol{x_2^2} + y\sum_{i}^{m} y(i)\boldsymbol{x_2^{(i)}}\boldsymbol{x_2} \geq -1]. \end{aligned} \tag{18}$$

Since $y^2\boldsymbol{x_2^2} \geq 0$ for any sample in $\mathcal{S}$, we have

$$y^2\boldsymbol{x_2^2} + y\sum_i^m y(i)\boldsymbol{x_2^{(i)}}\boldsymbol{x_2} \geq y\sum_i^m y(i)\boldsymbol{x_2^{(i)}}\boldsymbol{x_2}. \tag{19}$$

Then, $\mathbb{P}_{(\boldsymbol{x},y)\in\mathcal{S}} \geq \mathbb{P}_{(\boldsymbol{x},y)\in\mathcal{D}/\mathcal{S}}$. This completes the proof. □

Proof for Theorem 3.4 Let $f_{\boldsymbol{w}}$ be a d-layer feed-forward model trained on distribution $\mathcal{D}$ with parameters $\boldsymbol{w} = \{W_i\}_{i=1}^d$ and the ReLU activation function. Assuming a training dataset $\mathcal{S} \sim \mathcal{D}$, the model is given as $f_{\mathcal{S}} = f_{\boldsymbol{w}+\boldsymbol{u_S}}$, where $\boldsymbol{u_S}$ is a random variable whose distribution may also depend on $\mathcal{S}$.

Since the key to analyze the margin is the output of the model, we first introduce Lemma 9.1 that analyzes the perturbation bound of the model trained on $\mathcal{S}$ and $\mathcal{D}$.

Lemma 9.1 (Perturbation Bound (Lemma 2) in Neyshabur et al. (2018))**.** *For any $B, d > 0$, let $f_{\boldsymbol{w}} : \mathcal{X} \to \mathbb{R}^k$ be a $d-$layer neural network with ReLU activations. Then for any $\boldsymbol{w}$, and $\boldsymbol{x} \in \mathcal{X}$, and any perturbation $\boldsymbol{u}_{\mathcal{S}} = \{U_i\}_{i=1}^d$ such that $||U_i||_2 \leq \frac{1}{d}||W_i||_2$, the change in the output of the network can be bounded as follow,*

$$|f_{\boldsymbol{w}+\boldsymbol{u}_{\mathcal{S}}}(\boldsymbol{x}) - f_{\boldsymbol{w}}(\boldsymbol{x})| \leq eB(\prod_{i=1}^d ||W_i||_2)\sum_{i=1}^d \frac{||U_i||_2}{||W_i||_2}. \tag{20}$$

Since our proof is also based on Lemma 9.1, it is analogous to the analysis of generalization bound in Neyshabur et al. (2018) and is essentially the same for the first part.

Proof. The proof involves two parts. In the first part, we show the maximum allowed perturbation of parameters as shown in Neyshabur et al. (2018). In the second part, we show that the margin difference of the models trained on $\mathcal{S}_V$ and $\mathcal{S}_I$ is also bounded by the perturbation of parameters. Let $\beta = (\prod_{i=1}^d ||W_i||_2)^{\frac{1}{d}}$, and consider a network with normalized weights $\tilde{W}_i = \frac{\beta}{||W_i||_2}W_i$. Due to the homogeneity of the ReLU, we have $f_{\tilde{\boldsymbol{w}}} = f_{\boldsymbol{w}}$. We can also verify that $(\prod_{i=1}^d ||W_i||_2) = \prod_{i=1}^d ||\tilde{W}_i||_2$ and $\frac{||W_i||_F}{||W_i||_2} = \frac{||\tilde{W}_i||_F}{||\tilde{W}_i||_2}$. Therefore, it is sufficient to prove the Theorem only for the normalized weights $\tilde{\boldsymbol{w}}$, and hence w.l.o.g we assume that for any layer i, $||W_i||_2 = \beta$.

Choose the distribution $\mathcal{P}$ of the prior of $\boldsymbol{w}$ to be $\mathcal{N}(0, \sigma^2 I)$, and consider the random perturbation $\boldsymbol{u}_{\mathcal{S}} \sim \mathcal{N}(0, \sigma^2 I) = \{U_i\}_{i=1}^d$. Since the prior cannot depend on the learned model $\boldsymbol{w}$ or its norm, we set σ based on the approximation $\tilde{\beta}$. For each value of $\tilde{\beta}$ on a pre-determined grid, we compute the PAC-Bayes bound, establishing the generalization guarantee for all $\boldsymbol{w}$ for which $|\tilde{\beta} - \beta| \leq \frac{1}{d}\beta$, and ensuring that each relevant value of β is covered by some $\tilde{\beta}$ on the grid. We then take a union bound over all $\tilde{\beta}$ on the grid. For now, we consider a fixed $\tilde{\beta}$ and the $\boldsymbol{w}$ for which $|\beta - \tilde{\beta}| \leq \frac{1}{d}\beta$, and hence $\frac{1}{e}\beta^{d-1} \leq \tilde{\beta}^{d-1} \leq e\beta^{d-1}$.

Since $\boldsymbol{u}_{\mathcal{S}} \sim \mathcal{N}(0, \sigma^2 I)$, we get the following bound for the spectral norm of U_i Tropp (2012):

$$\mathbb{P}_{U_i\sim\mathcal{N}(0,\sigma^2 I)}[||U_i||_2 > t] \leq 2he^{-t^2/2h\sigma^2}. \tag{21}$$

Taking a union bound over the layers, we get that with probability at least $\frac{1}{\sqrt{2}}$, the spectral norm of perturbation of U_i in each layer is bounded by $\sigma\sqrt{2hln(2dh)}$. Plugging this spectral norm bound into Lemma 9.1 we have that with probability at least $\frac{1}{\sqrt{2}}$ the maximum allowed perturbation bound is:

$$max_{\boldsymbol{x}\in\mathcal{X}}|f_{\boldsymbol{w}+\boldsymbol{u}_{\mathcal{S}}}(\boldsymbol{x}) - f_{\boldsymbol{w}}(\boldsymbol{x})| \leq eB\beta^d\sum_i \frac{||U_i||_2}{\beta} \leq e^2 dB\tilde{\beta}^{d-1}\sigma\sqrt{2hln(2dh)} \leq \frac{\epsilon}{4}, \tag{22}$$

where $\sigma = \frac{\epsilon}{42dB\tilde{\beta}^{d-1}\sigma\sqrt{2hln(2dh)}}$. Then we can compute the difference of expectation margins for $f_{\mathcal{V}}$ which is trained on $\mathcal{S}_V$ and $f_{\mathcal{I}}$ which is trained on $\mathcal{S}_I$. Firstly, we compute the difference margins for any model $f_{\mathcal{S}}$

trained on $\mathcal{S} \sim \mathcal{D}$ and the target model $f_{\mathcal{D}}$. For any verified dataset $\hat{S} \in \mathcal{D}$,

$$\begin{aligned}
&|E(p(f_{\mathcal{S}}, \boldsymbol{x})) - E(p(f_{\mathcal{D}}, \boldsymbol{x}))| \\
=&|E(f_{\boldsymbol{w}+\boldsymbol{u}_{\mathcal{S}}}(\boldsymbol{x})[y] - max_{j\neq y} f_{\boldsymbol{w}+\boldsymbol{u}_{\mathcal{S}}}(\boldsymbol{x})[j]) - E(f_{\boldsymbol{w}}(\boldsymbol{x})[y] - max_{j\neq y} f_{\boldsymbol{w}}(\boldsymbol{x})[j])| \\
=&|(E(f_{\boldsymbol{w}+\boldsymbol{u}_{\mathcal{S}}}(\boldsymbol{x})[y]) - E(f_{\boldsymbol{w}}(\boldsymbol{x})[y])) - (E(max_{j\neq y} f_{\boldsymbol{w}+\boldsymbol{u}_{\mathcal{S}}}(\boldsymbol{x})[j]) - E(max_{j\neq y} f_{\boldsymbol{w}}(\boldsymbol{x})[j]))| \\
\leq& max_{\boldsymbol{x}\in\mathcal{X}}(f_{\boldsymbol{w}+\boldsymbol{u}_{\mathcal{S}}}(\boldsymbol{x})[y] - f_{\boldsymbol{w}}(\boldsymbol{x})[y]) + max_{\boldsymbol{x}\in\mathcal{X}}(max_{j\neq y} f_{\boldsymbol{w}+\boldsymbol{u}_{\mathcal{S}}}(\boldsymbol{x})[j] - max_{j\neq y} f_{\boldsymbol{w}}(\boldsymbol{x})[j]) \\
\leq& 2max_{\boldsymbol{x}\in\mathcal{X}}|f_{\boldsymbol{w}+\boldsymbol{u}_{\mathcal{S}}}(\boldsymbol{x}) - f_{\boldsymbol{w}}(\boldsymbol{x})| \leq \frac{\epsilon}{2}.
\end{aligned} \tag{23}$$

So, for $f_{\mathcal{V}}$ trained on $\mathcal{S}_V$ and $f_{\mathcal{I}}$ trained on $\mathcal{S}_I$, we have with probability at least $\frac{1}{2}$ that the predictions margins are bounded by ϵ:

$$\begin{aligned}
&|E(p(f_{\mathcal{V}}, \boldsymbol{x})) - E(p(f_{\mathcal{I}}, \boldsymbol{x}))| \\
\leq&|E(p(f_{\mathcal{V}}, \boldsymbol{x})) - E(p(f_{\mathcal{D}}, \boldsymbol{x}))| + |E(p(f_{\mathcal{I}}, \boldsymbol{x})) - E(p(f_{\mathcal{D}}), \boldsymbol{x})| \\
\leq&\epsilon.
\end{aligned} \tag{24}$$

□