MER-Inspector: Assessing Model Extraction Risks from An Attack-Agnostic Perspective

Anonymous Author(s)
Submission Id: 622

Abstract

Information leakage issues in machine learning-based Web applications have attracted increasing attention. While the risk of data privacy leakage has been rigorously analyzed, the theory of model function leakage, known as Model Extraction Attacks (MEAs), has not been well studied. In this paper, we are the first to understand MEAs theoretically from an attack-agnostic perspective and to propose analytical metrics for evaluating model extraction risks. By using the Neural Tangent Kernel (NTK) theory, we formulate the linearized MEA as a regularized kernel classification problem and then derive the fidelity gap and generalization error bounds of the attack performance. Based on these theoretical analyses, we propose a new theoretical metric called Model Recovery Complexity (MRC), which measures the distance of weight changes between the victim and surrogate models to quantify risk. Additionally, we find that victim model accuracy, which shows a strong positive correlation with model extraction risk, can serve as an empirical metric. By integrating these two metrics, we propose a framework, namely Model Extraction Risk Inspector (MER-Inspector), to compare the extraction risks of models under different model architectures by utilizing relative metric values. We conduct extensive experiments on 16 model architectures and 5 datasets. The experimental results demonstrate that the proposed metrics have a high correlation with model extraction risks, and MER-Inspector can accurately compare the extraction risks of any two models with up to 89.58%.

CCS Concepts

• **Security and privacy** → **Web application security**; • **Computing methodologies** → **Machine learning**.

Keywords

Model extraction attacks, neural tangent kernel, model extraction risk, risk measure

ACM Reference Format:

Anonymous Author(s). 2025. MER-Inspector: Assessing Model Extraction Risks from An Attack-Agnostic Perspective. In *Proceedings of the ACM Web Conference 2025 (WWW' 25).* ACM, New York, NY, USA, 16 pages. https://doi.org/XXXXXXX.XXXXXXX

WWW' 25, April 25- May 2, 2025, Sydney, Australia

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Recent advancements in machine learning, particularly in deep neural networks (DNNs), have revolutionized a range of Web applications, e.g., speech recognition, image classification and sentiment analysis [5, 17, 33]. Nonetheless, many studies have shown such machine learning-powered Web applications may cause significant information leakage issues [13, 28, 29, 51, 54], which can compromise both data privacy leakage and model function leakage [26]. For model function leakage, the primary risk comes from *Model Extraction Attacks (MEAs)* [48], in which the attacker aims to copy the victim model by iteratively selecting queries using active learning techniques and then using the query-output pairs to train a surrogate model. This model extraction risk poses a threat to the intellectual property of the model owner and empowers downstream attacks, such as generating adversarial examples [19].

Given the significant leakage challenges in both model and data aspects, it is crucial for model owners to assess and mitigate the associated risks before publishing their models. While existing research primarily concentrates on data privacy risks, the analysis of model extraction risks receives less attention [29, 43]. Although many MEAs have been proposed [46, 48, 55], their high computational costs make them unsuitable as direct measurements for model extraction risks. Furthermore, emerging advancements in MEAs inevitably lead to the underestimation of model extraction risks that are based on prior MEAs. In this paper, we aim to assess the model extraction risk from an attack-agnostic perspective because the model owner cannot know the data that the attacker has access to or the attack strategies they adopt.

To achieve this, we first consider a worst-case threat model in current MEAs where the attacker has the union of all existing black-box MEA attackers' capabilities, to explore the possible maximum model extraction risk. In this threat model, a powerful attacker possesses all unlabeled training samples, model architecture, initialization, and the output probabilities of the victim model without any query budget limitation [8, 15, 25, 37, 38, 48]. With these capabilities, the attacker can query the victim model using all available samples and train a surrogate model, thereby achieving near-optimal attack performance [37, 38]. With the aid of the Neural Tangent Kernal (NTK) theory that analyzes the non-linear neural networks by the kernel method [14], we formulate this MEA as a regularized kernel classification problem that minimizes the difference in output changes between the linearized victim and surrogate models. Based on this new perspective, we derive two theoretical bounds that measure the attack performance of MEAs, namely fidelity gap (the prediction disagreement between the victim model and the surrogate model) and generalization error (the disagreement between the surrogate model's prediction and the ground truth).

Next, derived from these bounds, we propose a new theoretical attack-agnostic metric, *Model Recovery Complexity (MRC)*, to

assess the risk, which measures the distance between the weight change in the victim model and that in the surrogate model. While this metric is derived from our theoretical bounds under a specific MEA, it aligns well with the model extraction risk in other practical scenarios (e.g., an attacker only has access to the output labels or has no training dataset to launch arbitrary MEAs) because it is closely related to the complexity of the model itself, as verified in Section 8.1. We also find that *Victim Model Accuracy (VMA)*, which was known to be closely related to MEA performance [29, 54], can serve as a second (empirical) risk metric. While VMA offers accurate assessments for models with large accuracy disparities, MRC captures finer details, enabling precise risk comparisons between models with similar accuracies. Leveraging both metrics, we propose the *Model Extraction Risk Inspector (MER-Inspector)*, a novel framework that effectively integrates VMA and MRC to compare risks across different models. Tested on 16 model architectures and 5 datasets, MER-Inspector can achieve 89.25% accuracy in comparing the extraction risks of two models trained on the same dataset. Our research provides model owners with crucial insights, guiding them in evaluating the efficacy of their defense strategies to ensure the deployment of more secure models.

Contributions. In summary, we make the following contributions:

- We provide new theoretical insights into MEAs through the NTK theory, with theoretical bounds of fidelity gap and generalization error on the attack performance.
- We are the first to explore metrics for assessing the model extraction risk in an attack-agnostic manner and propose a theoretical metric, namely MRC, to assess this risk.
- We propose the MER-Inspector framework, which combines VMA and MRC to compare the extraction risks of any two models.
- We conduct extensive experiments on 16 model architectures and 5 datasets to verify the high correlation between the metrics used in MER-Inspector and model extraction risks.

The rest of this paper is organized as follows. Section 2 introduces the background and related work. Section 3 describes the problem setup. Section 4 gives theoretical bounds of MEAs, while Section 5 details the metrics used for assessing model extraction risks. Section 6 introduces the MER-Inspector framework. Section 7 shows the evaluation results, followed by a discussion in Section 8. Finally, Section 9 concludes this paper.

2 Background and Related Work

Model Extraction Attacks. MEAs, which use crafted queries to copy well-trained machine learning models illegally, pose significant threats to model intellectural property [46]. Current research on model extraction goes spiral: new attacks and defenses emerge interleavingly to outwit their predecessor counterparts. For attacks, a large amount of research has been conducted to steal models in various domains efficiently, by reducing the number of queries [8, 19, 20, 27, 37, 38, 41, 52]. To counter these attacks, defense methods such as detecting extraction queries or disrupting query results have also been proposed [19, 24, 55]. However, no theoretical work has been done to quantify the vulnerability of models on model extraction without assuming specific attacks. Our work is the first attempt to benchmark model extraction risks from the perspective of models themselves, rather than from those attacks, thereby providing a fundamental and attack-agnostic understanding of model extraction risks.

NTK Theory. NTK theory is widely used to understand the training dynamics of neural networks [14]. According to NTK theory [14, 50], the output of a wide neural network can be approximated by a linearization of its initialization using the first-order Taylor expansion:

$$\mathbf{f}_\theta(\mathbf{x}) \approx \hat{\mathbf{f}}_\theta(\mathbf{x};\theta_0) = \mathbf{f}_{\theta_0}(\mathbf{x}) + \triangle_\theta^\top \nabla_\theta \mathbf{f}_{\theta_0}(\mathbf{x}), \tag{1}$$

where $\triangle_\theta = \theta - \theta_0$ is the weight change, θ_0 is the initial weight, and $\nabla_\theta \mathbf{f}_{\theta_0}(\mathbf{x})$ represents the Jacobian of the neural network with respect to the parameters evaluated at θ_0. The linear neural network $\hat{\mathbf{f}}_\theta(\mathbf{x};\theta_0)$ is the sum of the original output of the network and the change to the output during the training stage. This result has been proved in infinite-width networks and further experimentally verified in finite-width networks [23]. Determined by empirical NTK [23] at θ_0, the kernel function is $k(\mathbf{x},\mathbf{x}') =< \nabla_\theta \mathbf{f}_{\theta_0}(\mathbf{x}), \nabla_\theta \mathbf{f}_{\theta_0}(\mathbf{x}') >\in \mathbb{R}^{K\times K}$, and the kernel matrix Θ of training examples can be expressed as

$$\Theta = \begin{bmatrix} k\left(\mathbf{x}^1,\mathbf{x}^1\right) & \cdots & k\left(\mathbf{x}^1,\mathbf{x}^N\right) \\ \cdots & \cdots & \cdots \\ k\left(\mathbf{x}^N,\mathbf{x}^1\right) & \cdots & k\left(\mathbf{x}^N,\mathbf{x}^N\right) \end{bmatrix} \in \mathbb{R}^{NK\times NK}. \tag{2}$$

In infinite-width neural networks, this NTK matrix is a positive definite matrix when no two training inputs are parallel and remains constant during training [3, 14]. The NTK theory has been widely applied in various deep learning techniques, including knowledge distillation [10, 16], adversarial training [31], and neural architecture search [35]. In this work, we are the first to apply NTK theory to model extraction analysis.

3 Problem Setup

Model. In this paper, we consider a classification task on the training dataset $\mathbb{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^N$, which comprises N samples. Each sample is represented by a pair $(\mathbf{x}, y)$, where $\mathbf{x}$ is a feature vector in $\mathbb{R}^d$ with probability distribution $P_\mathbf{x}$, and y is a categorical label taking on values from the set $\{1, 2, \ldots, K\}$. d is the dimension of input features and K is the number of labels. A deep learning model is trained on this dataset with the aim to optimize parameters $\theta \in \mathbb{R}^p$ that minimize a loss function ℓ. The outputs from the final layer $\mathbf{f}_\theta(\mathbf{x}) = \{f_\theta^1(\mathbf{x}), f_\theta^2(\mathbf{x}), \ldots, f_\theta^K(\mathbf{x})\} \in \mathbb{R}^K$ are the logits of different classes, and $y(\mathbf{x}) = \arg\max_{k\in\{1,\ldots,K\}} \mathbf{f}_\theta^k(\mathbf{x})$ denotes the output label of the model. As such, the output labels of the victim model and the surrogate model are $y_v(\mathbf{x}) = \arg\max_{k\in\{1,\ldots,K\}} \mathbf{f}_{\theta_v}^k(\mathbf{x})$ and $y_s(\mathbf{x}) = \arg\max_{k\in\{1,\ldots,K\}} \mathbf{f}_{\theta_s}^k(\mathbf{x})$, respectively.

Model Extraction Risk. We consider the model extraction risk from state-of-the-art MEAs [8, 37, 38, 48] where the attacker iteratively selects query samples using active learning techniques to minimize the number of queries. The objectives of MEAs includes attack accuracy and fidelity [15]. The former refers to the attacker's intention to replicate the ground truth value. This objective is to keep the surrogate model performing well on the testing dataset. The latter refers to the attacker's attempt to replicate the victim

model's output. This ensures the surrogate model can replace the victim model for downstream attacks, such as adversarial examples. In most MEA works, the latter one is considered more important [15], which we define it as follows:

Definition 3.1 (Fidelity). Let $\mathbf{f}_{\theta_v}$ be a victim model. Given the target distribution $P_{\mathbf{x}}$ over $\mathbf{x}$, and a similarity function $S(\mathbf{f}_{\theta_v}, \mathbf{f}_{\theta_s}) = \mathbb{1}(\arg\max \mathbf{f}_{\theta_s} = \arg\max \mathbf{f}_{\theta_v})$, the fidelity $\mathbb{F}(\mathbf{f}_{\theta_s}, \mathbf{f}_{\theta_v})$ between the victim model and a surrogate model $\mathbf{f}_{\theta_s}$, where $\mathbf{f}_{\theta_s}$ is constructed through a MEA, is defined as the probability $\Pr_{\mathbf{x}\sim P_{\mathbf{x}}}[S(\mathbf{f}_{\theta_v}(\mathbf{x}), \mathbf{f}_{\theta_s}(\mathbf{x}))]$.

A specific MEA $\mathcal{A}(C, \mathcal{S})$ depends on the attackers' capabilities C and strategies $\mathcal{S}$. Under a specific MEA, the model extraction risk can be defined as follows.

Definition 3.2 (Model Extraction Risk). Let $\mathbf{f}_{\theta_v}$ be the victim model and $\mathcal{A}(C, \mathcal{S})$ be a MEA. The extraction risk of the victim model under $\mathcal{A}(C, \mathcal{S})$ is defined as the fidelity the attack can achieve, i.e.,

$$\text{Risk}(\mathbf{f}_{\theta_v}) := \max_{\mathbf{f}_{\theta_s} \in \mathcal{F}_{\mathcal{A}}(C,\mathcal{S})} \mathbb{F}(\mathbf{f}_{\theta_s}, \mathbf{f}_{\theta_v}), \tag{3}$$

where $\mathcal{F}_{\mathcal{A}}(C, \mathcal{S})$ denotes the set of surrogate models $\mathbf{f}_{\theta_s}$ constructed through the attack $\mathcal{A}(C, \mathcal{S})$. Particularly, observations of increasing fidelity performence the attack can achieve, the higher model extraction risk of the victim model.

Threat Model. From the perspective of the model owner, they cannot predict the specific MEA method that the attacker may adopt. Therefore, in an attacker-agnostic manner, this paper explores the maximum possible model extraction risk under a worst-case threat model, wherein the attacker possesses the union of all existing black-box MEA attackers' capabilities. These capabilities, defined as the attackers' maximum capabilities C_m, include (1) full access to the victim model outputs, i.e., the full posterior; (2) knowledge of all unlabeled training samples, model architecture, and initialization; and (3) no limitations on the query budget [8, 15, 25, 37, 38, 48]. In this case, the attacker can query all training samples and achieve nearly maximum fidelity without considering any specific attack strategies [37, 38]. The risk posed by $\mathcal{A}(C_m, \mathcal{S})$ can thus be regarded as the model extraction risk from the model owner's perspective. Despite the attacker's formidable capabilities, they still cannot perfectly replicate the victim model through retraining due to the lack of access to ground-truth labels and the randomness in training variables like hyperparameters and execution environments [2].

Linearized Model Extraction. This paper formulated the MEA under the above threat model as linearized model extraction. We assume the victim and surrogate models are over-parameterized and wide. This assumption enables us to theoretically analyze MEAs by applying NTK theory, which allows the substitution of nonlinear neural networks with linear ones. The functions of the victim model and the surrogate model can be formulated as $\mathbf{f}_{\theta_v}(\mathbf{x}) \approx \mathbf{f}_{\theta_0}(\mathbf{x}) + \triangle_{\mathbf{f}_{\theta_v}(\mathbf{x})} = \mathbf{f}_{\theta_0}(\mathbf{x}) + \triangle_{\theta_v}^{\top} \nabla_{\theta} \mathbf{f}_{\theta_0}(\mathbf{x})$ and $\mathbf{f}_{\theta_s}(\mathbf{x}) \approx \mathbf{f}_{\theta_0}(\mathbf{x}) + \triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})} = \mathbf{f}_{\theta_0}(\mathbf{x}) + \triangle_{\theta_s}^{\top} \nabla_{\theta} \mathbf{f}_{\theta_0}(\mathbf{x})$, respectively. Here the terms $\mathbf{f}_{\theta_0}(\mathbf{x})$ and $\nabla_{\theta} \mathbf{f}_{\theta_0}(\mathbf{x})$ only depend on the model initialization. For simple analysis, we assume the same initialization parameters to focus on analyzing the MEA process. Consequently, the MEA for linearized neural networks can be formulated by a regularized kernel classification problem, i.e.,

$$\triangle^{*}_{\mathbf{f}_{\theta_s}(\mathbf{x})} \in \underset{\triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})} \in \mathcal{H}}{\text{argmin}} \frac{1}{N} \sum_{i=1}^{N} \ell\left(\triangle^{(i)}_{\mathbf{f}_{\theta_s}(\mathbf{x})}, \triangle^{(i)}_{\mathbf{f}_{\theta_v}(\mathbf{x})}\right) + \frac{\lambda}{2} \| \triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})} \|^2_{\mathcal{H}}, \tag{4}$$

where ℓ is the loss function and $\| \triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})} \|_{\mathcal{H}}$ is the Reproducing Kernel Hilbert Space (RKHS) norm of the function $\triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})}$. Based on the representer theorem [42], the solution to this problem can be expressed as

$$\triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})} = \sum_{i=1}^{N} \alpha_i k(\mathbf{x}, \mathbf{x}^{(i)}) = \Theta \boldsymbol{\alpha}, \tag{5}$$

and the RKHS norm of $\triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})}$ can be expressed as $\| \triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})} \|^2_{\mathcal{H}} = \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j k\left(x^{(i)}, x^{(j)}\right) = \boldsymbol{\alpha}^{\top} \Theta \boldsymbol{\alpha}$, where α_i is the weight of the corresponding kernel function $k(\mathbf{x}, \mathbf{x}_i)$.

Based on the regularized kernel classification problem, we can analyze the attack performance that an attacker achieves by solving this problem. In the following sections, we will analyze the attack performance bounds of linearied MEAs and propose theoretical risk metrics to bridge the gap between these attack-related performance bounds and practical attack-agnostic evaluations.

4 Theoretical Bounds of MEA

Fidelity Gap Bound. The fidelity gap is the probability of the prediction disagreement between the surrogate model and the victim model. Let us define the prediction margin $\mathcal{M}(\mathbf{x}, y_v, y_s) = f^{y_v}_{\theta_s}(\mathbf{x}) - \max_{y' \neq y_v} f^{y'}_{\theta_s}(\mathbf{x})$, and the fidelity gap $\mathcal{G} = \mathbb{P}_{\mathbf{x}\sim P_{\mathbf{x}}}[y_s(\mathbf{x}) \neq y_v(\mathbf{x})] = \mathbb{P}_{\mathbf{x}\sim P_{\mathbf{x}}}[\mathcal{M}(\mathbf{x}, y_v, y_s) \leq 0]$. Since $\triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})}$ exists in the RKHS $\mathcal{H}$ corresponding to the kernel k, the fidelity gap can be upper bounded by the following theorem and its proof is available in Appendix A.2.

Theorem 4.1 (Fidelity gap bound). *Assume that* $\kappa = \sup_{\mathbf{x}\sim P_{\mathbf{x}}} k(\mathbf{x}, \mathbf{x}) < \infty$. *Define a constant* $\gamma > 0$, *and let* $M_0 = \left\lceil \frac{\gamma\sqrt{N}}{4K\sqrt{\kappa}} \right\rceil$. *Subsequently, with a probability of at least* $1 - \delta$, *for every function* $\triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})} \in \mathcal{H}$, *the fidelity gap on* N *training samples can be bounded:*

$$\begin{aligned} \mathcal{G}_N &= \mathbb{P}_{\mathbf{x}\sim P_{\mathbf{x}}}[\mathcal{M}(\mathbf{x}, y_v, y_s) \leq 0] \\ &\leq \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\left\{\mathcal{M}(\mathbf{x}^{(i)}, y_v^{(i)}, y_s^{(i)}) \leq \gamma\right\} \\ &\quad + \frac{4K(\triangle^{\top}_{\mathbf{f}_{\theta_v}(\mathbf{x})} \Theta^{-1} \triangle_{\mathbf{f}_{\theta_v}(\mathbf{x})})}{\gamma N} \sqrt{Tr(\Theta)} + 3\sqrt{\frac{\log(2M_0/\delta)}{2N}}, \end{aligned} \tag{6}$$

where $Tr(\Theta)$ *represents the trace of the victim model's NTK matrix* Θ.

The bound consists of three main components: The first term is an empirical margin loss based on the training data, which is smaller than 1 even when the γ is infinite. The second term is a penalty term related to the model complexity, which means that a more complex victim model leads to a large fidelity gap. The margin parameter γ influences the first two terms, which need to be chosen carefully to control the balance between the empirical margin loss and the victim model complexity [10]. The third item is the sample complexity item, which is related to the selected confidence δ, and

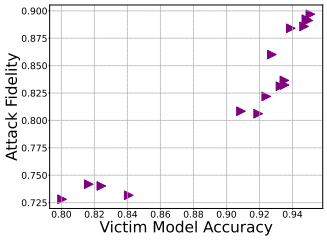


Figure 1: Relationship between attack fidelity and VMA.

represents the reliability of the generalization error bound under a certain confidence level.

Generalization Error Bound. The generalization error of the surrogate model is the probability of disagreement between the prediction of the surrogate model and the ground truth, i.e., $\mathcal{R}^s = \mathbb{P}_{\mathbf{x}\sim P_{\mathbf{x}}}[y_s(\mathbf{x}) \neq y(\mathbf{x})]$. Similarly, the generalization error of the victim model $\mathcal{R}^v = \mathbb{P}_{\mathbf{x}\sim P_{\mathbf{x}}}[y_v(\mathbf{x}) \neq y(\mathbf{x})]$. The following theorem proves that the generalization error of the surrogate model is bounded by the generalization performance of the victim model and the fidelity gap between them, and its proof is available in Appendix A.3.

Theorem 4.2 (Generalization risk bound). *Given the fidelity gap bound $\mathcal{G}_N$ and the victim model's generalization error bound $\mathcal{R}^v_N$. Then, the surrogate model's generalization error is bounded by,*

$$\mathcal{R}^s_N \leq \mathcal{G}_N + \mathcal{R}^v_N. \tag{7}$$

5 Model Extraction Risk Measure

Until now, existing evaluation of model extraction risks against MEAs has to be empirical, i.e., by conducting a range of well-known MEAs [46, 48, 55] and measure the real fidelity and accuracy. However, this approach is neither cost-effective nor future-proof. Additionally, directly calculating the attack performance bound of Theorem 4.1 to assess risk is impractical due to the high computational cost, impractical threat model and the need for careful parameter γ selection. In this section, we explore attack-agnostic metrics that can precisely capture the risk associated with the victim model. We first introduce an empirical metric, VMA, for a rough risk assessment, followed by a theoretical metric, MRC, derived from the fidelity gap bound of Theorem 4.1, for a more fine-grained assessment.

5.1 Victim Model Accuracy

The first term of empirical loss in Theorem 4.1 directly reflects the model extraction risk experimentally, yet it is dependent on a specific MEA. To establish an attack-agnostic measurement, we find that the empirical VMA effectively indicates attack fidelity. This enables us to assess potential risks without being limited by the particulars of any MEA. Zhang et al. [54] have experimentally demonstrated a significant correlation between VMA and both attack accuracy and fidelity. Our empirical findings further confirm the strong relationship between VMA and attack fidelity, reinforcing the utility of VMA as an attack-agnostic metric in risk assessments. As shown in Figure 1, we observe a strong positive relationship exists between them, with a Pearson Correlation Coefficient (PCC) of 0.9364 and a Kendall Rank Correlation Coefficient (KRC) of 0.85. However, we argue that VMA can only provide a rough risk measurement and may not effectively indicate the risk when the accuracies of these compared models are nearly identical, e.g., both close to 95% (see Section 7.2 and Appendix D.2). Furthermore, VMA is an empirical metric without a robust theoretical foundation to support it. Therefore, we need a more fine-grained and theoretical metric to measure the risk.

5.2 Model Recovery Complexity

Although Theorem 4.1 provides a theoretical attack performance bound of a specific MEA rather than in an attack-agnostic manner, it shows that the attack performance is highly related to the victim model's complexity. Specifically, in Theorem 4.1, the expression $\triangle^\top_{\mathbf{f}_{\theta_v}(\mathbf{x})}\Theta^{-1}\triangle_{\mathbf{f}_{\theta_v}(\mathbf{x})}$ in the second term can reflects the model complexity and has a clear and intuitive explanation that we elaborate on later. Additionally, this expression bears similarity to NTK-based metrics proposed in previous work, such as label-gradient alignment $Y^T\Theta Y$ [35] and supervision complexity $Y^T\Theta^{-1}Y$ [10], whose inherent formula enables it to capture a large number of nonlinear features. Consequently, we formally define it as MRC as follows.

Definition 5.1 (Model Recovery Complexity (MRC))**.** Given the victim model $\mathcal{F}$ and training dataset $\mathbb{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^N$, the MRC of model $\mathcal{F}$ is

$$r_{rc}(\mathbb{D}, \mathcal{F}) = \triangle^\top_{\mathbf{f}_{\theta_v}(\mathbf{x})}\Theta^{-1}\triangle_{\mathbf{f}_{\theta_v}(\mathbf{x})}, \tag{8}$$

where $\triangle_{f_{\theta_v}(\mathbf{x})}$ denotes the output change of the victim model before and after model training on all training samples $\mathbf{x}$, and Θ is the kernel matrix defined by NTK.

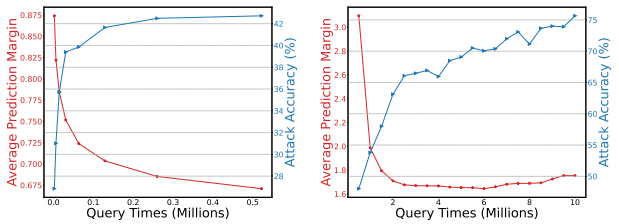
In essence, Θ represents a mapping of data into a feature space defined by the neural network, and metric MRC quantifies the "*size*" or "*impact*" of output changes in this feature space. To better understand MRC, let us assume the weight change of the victim model is $\triangle_{\theta_v} = \theta_v - \theta_0$, and the weight change of the surrogate model is $\triangle_{\theta_s} = \theta_s - \theta_0$. Let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_M] \in \mathbb{R}^{d\times M}$ and $\mathbf{Y} = [\mathbf{f}_{\theta_v}(\mathbf{x}_1), \mathbf{f}_{\theta_v}(\mathbf{x}_2), ..., \mathbf{f}_{\theta_v}(\mathbf{x}_M)] \in \mathbb{R}^{K\times M}$ denote the thief dataset. According to Theorem 1 of [40], the converged weight change of the surrogate model is

$$\begin{aligned}\Delta_{\theta_s,M} &= \nabla_{\theta_v,M}\mathbf{f}_{\theta_0}(\mathbf{x})\,\Theta^{-1}\,\Delta_{f_{\theta_v,M}(\mathbf{x})} \\ &= \nabla_{\theta_v,M}\mathbf{f}_{\theta_0}(\mathbf{x})\,\Theta^{-1}\,\nabla^\top_{\theta_v,N}\mathbf{f}_{\theta_0}(\mathbf{x})\,\Delta_{\theta_v} = \mathbf{P}_{\theta_v,M}\,\Delta_{\theta_v},\end{aligned} \tag{9}$$

where $\mathbf{P}_{\theta_v,M}$ is a projection matrix onto span $\{\nabla_{\theta_v}\mathbf{f}_{\theta_0}(\mathbf{x})\}_N$. According to this equation, the weight change learned by the attacker is simply the projection of the victim model's weight change onto the span $\{\nabla_{\theta_v}\mathbf{f}_{\theta_0}(\mathbf{x})\}_M$. As M increases and span $\{\nabla_{\theta_v}\mathbf{f}_{\theta_0}(\mathbf{x})\}_M$ expands, $\triangle_{\theta_s}$ gradually converges to $\triangle_{\theta_v}$. Therefore, MRC can be rewritten as

$$\begin{aligned}r_{rc}(\mathbb{D}, \mathcal{F}) &= \triangle^\top_{f_{\theta_v}(\mathbf{x})}\Theta^{-1}\triangle_{f_{\theta_v}(\mathbf{x})} \\ &= \triangle^\top_{\theta_v}\nabla_{\theta_v}\mathbf{f}_{\theta_0}(\mathbf{x})\Theta^{-1}\nabla^\top_{\theta_v}\mathbf{f}_{\theta_0}(\mathbf{x})\triangle_{\theta_v} = \triangle^\top_{\theta_v}\mathbf{P}_{\theta_v}\,\triangle_{\theta_v}.\end{aligned} \tag{10}$$

Essentially, the metric calculates the "*distance*" between the projected vector $\mathbf{P}_{\theta_v}\triangle_{\theta_v}$ and the original vector $\triangle_{\theta_v}$. If the metric value is small, it means that the projected vector is close to the original



(a) JBDA (b) MAZE

Figure 2: The change of average prediction margin and attack accuracy with increased query times under (a) JBDA, and (B) MAZE on CIFAR 10.

vector. As such, the difference in weight changes between the victim model and the surrogate model is small, thereby increasing the model extraction risk.

5.3 MRC Approximation

Although MRC can measure the risk, accurately computing its score presents two challenges.

- Time-consuming. The time complexity of computing the NTK matrix is $O((NK)^2p)$ [36]. For example, on CIFAR10, it takes over 1,000 GPU hours to compute the NTK matrix.
- Inaccuracy. The NTK matrix is accurate in infinite-width neural networks. But when we apply it to commonly used networks such as ResNet [11] and DenseNet [12], the NTK-based metric may not align perfectly with our theoretical expectations.

To address the first challenge, we perform sampling to reduce the number of samples to calculate the NTK matrix. To ensure that the selected samples yield more accurate approximations, it is essential to answer a key question: **which samples are crucial to dictate MEA performance?**

We conducted two well-known attacks, JBDA [39] and MAZE [20], to analyze the variations in the average prediction margin and attack accuracy as the number of queries increases. In each iteration, these two methods use the surrogate model held at this time to synthesize query samples for improving query efficiency. The experiment details are given in Appendix C. Figure 2 shows the trend that the average prediction margin of query samples decreases with increased attack performance. This means that the attacker first uses simple samples (with larger prediction margins) to obtain a rough classification boundary, and then gradually finds hard samples (with small prediction margins) to refine the classification boundary. Sorscher et al. [44] also highlighted that model training benefits the most from both simple and hard samples. Inspired by these findings, we choose those samples with the largest and smallest margins as both are crucial to attack performance. Specifically, among a total of L samples for MRC approximation, we select L_d samples with the largest margin and $L - L_d$ samples with the smallest margin, where $\eta = L_d/L$ is the *difficulty ratio*.

To address the second challenge, we find that most NTK matrices calculated in finite-width neural networks do not conform to the theoretical expectations. First, it cannot remain constant during training. Mok et al. [35] demonstrated that an NTK matrix changes significantly at the beginning of the training and remains constant in subsequent training. Therefore, to get the constant NTK matrix, unlike other works that compute NTK-based metrics at the initialization stage [35, 49], we compute the NTK matrix after the training. Second, the computed NTK matrix is not always positive definite, so its inversion Θ^{-1} is not stable. We solve this by adjusting the minimum eigenvalue of the NTK matrix to a threshold q. In addition, in order to maintain a certain stability in the MRC score, we use the probabilities of the model output instead of logits in the calculation. The overall process of MRC approximation algorithm is shown in Algorithm 1.

Algorithm 1 MRC Approximation Algorithm

Input: Victim model M, training dataset $\mathbb{D}$, threshold q, the number of selected samples L, difficulty ratio η
Output: MRC r_{rc}

```
Select L samples as dataset D_S according to difficulty ratio η from D
for each data point x_i in D_S do
    Compute gradients of M on x_i
end for
Calculate NTK matrix using their gradients according to Eq. (2)
Compute eigenvalues and eigenvectors of NTK matrix
for each eigenvalue λ_i do
    if λ_i < q then
        λ_i ← q
    end if
end for
Reconstruct NTK matrix from adjusted eigenvalues and eigenvectors
Computer the output probability change of D_S
Computer the MRC using Eq. (8)
```

6 MER-Inspector: A Risk Comparison Framework

From the above analysis, the VMA and MRC of the victim model can serve as good metrics for evaluating the model extraction risk. However, either VMA or MRC has its own limitations. The VMA is less effective in assessing the risk disparity between models with insignificant variation in accuracy. MRC leverages the NTK theory, whose quantitative analysis may be inconsistent with the actual behavior of finite-width networks. Nonetheless, we find MRC achieves good relative accuracy, i.e., when comparing models of similar structures (e.g., in the ResNet family) even if their accuracies differ little.

Since these two metrics complement each other, we can exploit both to compare the relative risks of two models. Figure 3 shows MP-Inspector, the model risk comparison framework of models A and B. It consists of two phases, i.e., risk measure and risk comparison, and outputs the relation between Risk_A and Risk_B.

- Risk measure. First, the evaluation metric vectors (VMA and MRC) $\mathbf{r}_A$ and $\mathbf{r}_B$ for models A and B are obtained. VMA

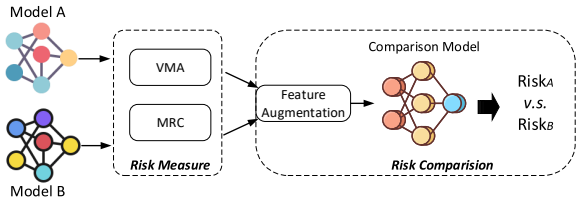
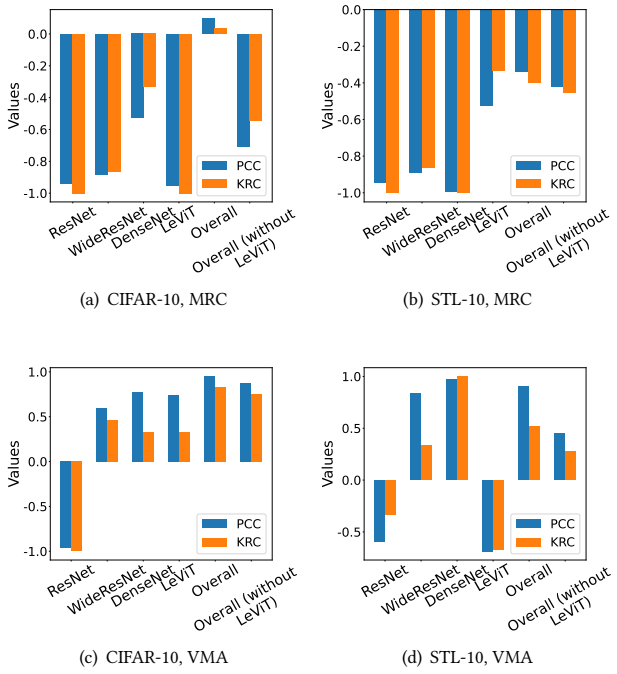


Figure 3: The framework of MER-Inspector.



(a) CIFAR-10, MRC (b) STL-10, MRC

(c) CIFAR-10, VMA (d) STL-10, VMA

Figure 4: Left: PCC and KRC performance between MRC and attack fidelity on (a) CIFAR-10 and (b) STL-10. Right: PCC and KRC performance between VMA and attack fidelity on (c) CIFAR-10 and (d) STL-10.

is measured on the testing dataset, and MRC is calculated according to Algorithm 1.

- Risk comparison. We can train a comparison model, e.g., a fully connected neural network as in Section 7, to output bit 0 or 1 (Risk_A exceeds Risk_B or vice versa). Since there are now only two dimensions in the risk vector, we adopt Feature Augmentation (FA) to expand the vector from $\{\mathbf{r}_A, \mathbf{r}_B\}$ to $\{\mathbf{r}_A, \mathbf{r}_B, \mathbf{r}_A - \mathbf{r}_B\}$ as the input of comparison model.

7 Experiments

7.1 Experiment Setup

Datasets & Victim Models. We use 5 evaluation datasets: CIFAR-10 [21], CIFAR-100 [21], FashionMNIST [47], STL-10 [6], and CelebA [30]. For each dataset, we train 16 victim models across four architecture groups for 200 epochs using the SGD optimizer with a learning rate of 0.01 (400 epochs for LeViT models) to minimize the softmax Cross-Entropy(CE) loss. The four architecture groups are ResNet [11] (ResNet22, ResNet32, ResNet44), WideResNet [53] (WideResNet22-2, WideResNet22-4, WideResNet22-8, WideResNet28-2, WideResNet34-2.WideResNet40-2), Densenet [12] (Densenet121, Densenet169, Denset201)), and LeViT [9] (LeViT-128, LeViT-192, LeViT-256, LeViT-384). The VMA difference is relatively small between models within the same group and large between different groups. The detailed datasets and setup are introduced in Appendix B and Appendix D.1.

Table 1: Comparative accuracy of MER-Inspector.

Dataset	VMA	MRC	VMA+MRC
Intra-Group	53.70%	**88.89%**	79.63%
Inter-Group	89.78%	55.38%	**93.01%**
All	85.00%	70.42%	**89.58%**
All (w/o FA)	83.33%	67.92%	**86.25%**

Model Extraction Setting. In the MEA, the attacker uses the same architecture and hyper-parameters as the victim model, and uses the training dataset of the victim model for querying. Each surrogate model is trained for 100 epochs using CE loss and SGD optimizer, and the maximum attack fidelity on the testing dataset during training is considered as the ground truth for assessing the risk.

MER-Inspector Implementation. The architecture of the comparison model is a fully connected neural network with three hidden layers containing 64, 64, and 32 neurons, respectively. The activation function for hidden layers is ReLU and is sigmoid for the output layer. The model is trained for 500 epochs using the Adam optimizer with a learning rate of 1e-4 to minimize the binary CE loss.

We select 16 model architectures and 5 datasets to verify the performance of MER-Inspector. In each dataset, two models are randomly paired. Their risk metrics are used as input to the comparison model, with label "0" meaning the difference in attack fidelity between the two models is positive, and label "1" otherwise. We then switch the order of the two models and get a total of 1,200 sets of data, a.k.a., *all dataset*. Among them, 270 sets are from pairs within the same group, a.k.a., the *intra-group dataset*. The rest 930 sets are from pairs in different groups, a.k.a., the *inter-group dataset*. 80% of each dataset is used for training and the rest 20% for testing.

Evaluation Metric & Baseline. We employ PCC [7] and KRC [1] to evaluate how well the proposed metrics are aligned with the risk. Both metrics range from -1 to 1, where values closer to 1 indicate a strong positive correlation, values near -1 indicate a strong negative correlation, and values around 0 suggest no correlation. Furthermore, we use Comparative Accuracy (CAcc) to evaluate the effectiveness of the MER-Inspector. It is the ratio of pairs whose risks are successfully compared to the total compared pairs. The VMA [54] alone serves as the baseline.

7.2 Results

Effectiveness of Metrics. First, we verify the effectiveness of MRC and VMA. We compute the MRC in the default settings when L is 400, η is 0.5, and q is 0.5. Figure 4(a) and 4(b) show the PCC and

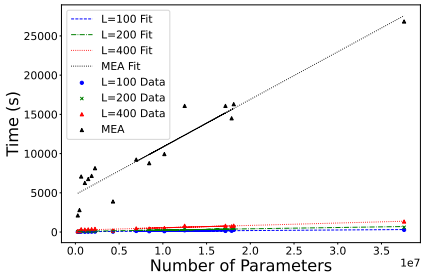


Figure 5: Time cost for calculating MRC with varying sample numbers (L) on CIFAR-10.

KRC performance between MRC and attack fidelity under different model architecture groups and datasets. The *overall* considers all 16 architectures. The results show that there is a strong negative correlation between MRC and attack fidelity within each model structure group. However, the correlation on CIFAR10 is weak positive. This is due to the large differences between the transformer-based model (LeViT) and other models, which makes the NTK theory difficult to capture this difference. The other metric, VMA, can compensate for this shortcoming. From Figures 4(c) and 4(d), we find that overall there is a positive correlation between VMA and attack fidelity, but when it comes to models in the same group, such correlation diminishes or even reverses, due to the small VMA gaps between models in the same group. This justifies our motivation to combine both metrics to compare risks. The results on other datasets are given in Appendix D.2.

Effectiveness of MER-Inspector. Table 1 compares CAccs across various datasets using different metrics. The results indicate that MRC effectively evaluates risks within similar architecture models (intra-group) with a CAcc of 88.89%. Meanwhile, VMA performs better in identifying risks between models (inter-group) with significant accuracy differences, achieving a CAcc of 89.78%. Combining these two metrics can significantly improve the overall accuracy (CAcc of 89.58%) of risk comparison on any two models. Furthermore, by comparing the last two columns of Table 1, we observe an improvement in CAcc when FA is utilized.

Cost Analysis. Figure 5 illustrates that the time cost increases with the number of parameters and the number of selected samples. We adopt the simplest MEA, where the attacker queries all samples and trains for 100 epochs. The cost will significantly increase if a more complex MEA is used (e.g., selecting the most uncertain samples for querying and training each iteration). Our results show that the time cost of MRC calculation is significantly lower than that of performing even a simple MEA. When we select 100 samples to calculate the MRC, the clock time is only 269 seconds for a model with nearly 40 million parameters, significantly less than the time cost of 2,980 seconds to execute an MEA for risk assessment, which requires full training of the surrogate model.

7.3 Ablation Study

Impact of Number of Selected Samples L. In the previous experiments, L is set to 400 and only a small part of the training data is used. For example, only 0.8% of the training samples are selected on CIFAR10. A larger L leads to a more accurate approximation of MRC.

Table 2: RCC and KRC performance between MRC and attack fidelity in the LeViT group on CIFAR-10.

Metric	$L = 100$	$L = 200$	$L = 400$
PCC	-0.7508	-0.8463	-0.9523
KRC	-0.3333	-0.6667	-1.0

We select the LeViT group as an example to illustrate this point. As shown in Table 2, the PCC and RKC are closer to -1 as L increases. The detailed results of the other models and other datasets are in Appendix D.3. Besides, models with fewer parameters or simpler architectures, such as ResNet, can be accurately measured even when $L = 100$. Therefore, a trade-off needs to be made between computational complexity and accuracy.

Impact of Difficulty Ratio η. Table 3 compares the performance of different groups under different η. The results show that simple samples effectively indicate risks associated with well-generalized models (such as WideResNet and DenseNet), while hard samples are important for models with poorer generalization (such as ResNet and LeViT). This is consistent with Sorscher et al. [44] where models with poor generalization are more likely to learn hard samples close to the boundary. To fairly compare models from different model groups, setting η to 0.5 seems appropriate in our experiments.

Impact of Threshold q. To maintain the fine-grained information of the NTK matrix, q should be set as small as possible while ensuring the positive definiteness of the NTK matrix of most models. Table 4 compares the performance under different q, and we find that the performance is poor when setting $q = 0.05/5$. This is because only 18.75% of the models exhibit positive definiteness in their NTK matrices when $q = 0.05$, causing inaccurate results. When $q = 5$, although 93.75% matrices are positive definite, a lot of fine-grained information is lost. In our experiments, we set q to 0.5, which ensures that the NTK matrices of most (87.5%) models remain positive definite and retain a large amount of fine-grained information.

8 Discussion

8.1 Practical Threat Model

Although our assumption of the most powerful attacker is made to analyze the maximum possible risk, our proposed MRC metric is based on model complexity, can also align with model extraction risk in more realistic scenarios (e.g., when the attacker can only access model labels, or does not know the training samples, or employs different attack strategies). As shown in Table 6, we verify the performance under two cases: 1. the attacker uses the surrogate dataset (e.g., CIFAR100) to steal the victim model (trained by CIFRA10) rather than using the victim model's training dataset CIFAR10; 2. the attacker only accesses the victim model's output label rather than output probabilities. We use the trained comparison model to compare the model risk under these two practical cases, and our results show that our metrics are still effective. We also evaluate the performance when the attacker is constrained by a query budget of 20,000 and employs two distinct query strategies

Table 3: RCC / KRC performance between MRC and attack fidelity on CIFAR-10 versus different η. A large η means that more hard samples (samples with the most significant margin) are selected.

Group	$\eta = 0$	$\eta = 0.25$	$\eta = 0.5$	$\eta = 0.75$	$\eta = 1$
ResNet	0.199/0.333	-0.356/-0.333	-0.942/-1.0	-0.974/-1.0	**-0.995/-1.0**
WideResNet	**-0.946/-1.0**	-0.913/-0.867	-0.884/-0.867	-0.888/-0.867	-0.912/-0.867
DenseNet	**-0.950/-1.0**	-0.998/-1.0	-0.528/-0.333	-0.944/-0.333	-0.753/-0.333
LeViT	-0.375/-0.333	-0.717/-0.333	-0.952/-1.0	**-0.993/-1.0**	-0.836/-0.333

Table 4: PCC / KRC performance versus different q.

Group	$q = 0.05$	$q = 0.5$	$q = 5$
ResNet	-0.779/-0.333	-0.942/-1.0	-0.324/-0.333
WideResNet	-0.719/-0.733	-0.884/-0.867	-0.591/-0.414
DenseNet	-0.587/-0.333	-0.528/-0.333	-0.526/-0.333
LeViT	-0.860/-0.667	-0.952/-1.0	-0.953/-1.0

Table 5: Comparison of standard training and adversarial training.

Network	Standard-training			Adv-training		
	VMA (%)	Attack Fidelity (%)	MRC	VMA (%)	Attack Fidelity (%)	MRC
ResNet20	90.94	81.35	0.00353	77.59	77.37	0.01418
ResNet32	91.66	80.88	0.00405	76.30	73.66	0.01500
ResNet44	92.41	80.31	0.00736	82.29	77.69	0.00994

Table 6: Performance of MER-Inspector under different attackers' capabilities.

Dataset \ Case	CIFAR100 ⟶ CIFAR10			Only Label		
	VMA	MRC	VMA+MRC	VMA	MRC	VMA+MRC
All	88.33%	55.00%	**90.42%**	91.67%	51.67%	**92.08%**
Intra-Group	55.56%	**75.93%**	72.22%	72.22%	68.52%	**79.63%**
Inter-Group	97.31%	48.39%	**97.85%**	**97.31%**	47.31%	96.77%

Table 7: Performance of MER-Inspector under different attack strategies.

Dataset \ Case	Uncertain			K-center		
	VMA	MRC	VMA+MRC	VMA	MRC	VMA+MRC
All	**91.25%**	52.50%	**91.25%**	90.00%	52.50%	**91.25%**
Intra-Group	64.81%	**72.22%**	**72.22%**	57.41%	**72.22%**	**72.22%**
Inter-Group	99.46%	47.31%	**100.00%**	99.46%	47.31%	**100.00%**

as described in ActiveThief [11]: (1) the uncertainty-based strategy and (2) the K-center strategy. Using a trained comparison model, we assess the extraction risk under both strategies and the results in Table 7 demonstrate that our proposed metrics remain effective.

8.2 Understanding Current Attacks and Defenses

By utilizing the theoretical metric known as MRC, we can deepen our understanding of existing attack and defense strategies.

Attacks. Current attacks aim to enhance query efficiency through active learning by identifying both informative (hard) and representative (simple) samples [4, 37, 52]. MRC provides us with an opportunity to gain fresh insights into the rationale behind utilizing these samples in queries. In Appendix D.4, we compare the MRC scores calculated for randomly selected samples, as well as hard and simple samples chosen based on difficulty ratio η when $L = 400$ on CIFAR-10. The results show that MRC scores derived from hard and easy samples yield lower MRC values compared to scores calculated using randomly selected samples. In specific, when the weight change of the victim model is projected onto the span $\{\nabla_{\theta_v} \mathbf{f}_{\theta_0}(\mathbf{x})\}$ consisting of the samples with the largest and smallest margins shows smaller differences from the original weight change. Thus the surrogate model's weight change $\mathbf{P}_{\theta_v} \triangle_{\theta_v}$ is closed to the victim model's weight change $\triangle_{\theta_v}$. In this way, we provide a theoretical explanation for why selecting useful samples is crucial for MEAs.

Defences. We can also use MRC to understand the defense methods on the model itself, such as adversarial training [18]. We use Project Gradient Descent (PGD) [32] to generate adversarial examples. The perturbation magnitude of the input x is constrained to $\epsilon = 0.1$. The step size α is set to 0.01 during the generation of adversarial samples. The PGD process of the optimized perturbation is performed three times. Then, we use these generated adversarial examples to train 200 epochs as the victim model. The other setups are the same as in Appendix D.1. Table 5 compares the results of standard training and adversarial training on CIFAR-10. As expected, adversarial training leads to an increase in MRC scores and a decrease in attack fidelity. In addition to adversarial training, we can utilize MRC to validate additional model defenses and investigate novel defense strategies that inherently mitigate the model extraction risk by enhancing the MRC score.

9 Conclusion

This paper provides a theoretical analysis of MEAs from an attack-agnostic perspective and proposes analytical metrics to evaluate model extraction risks for the first time. We start by theoretically analyzing MEAs using NTK theory and provide two bounds on attack performance. Next, we introduce two metrics, MRC and VMA, which jointly align with the model extraction risk. Using these metrics, we present the MER-Inspector framework that can compare risks on any two models. Experimental results verify the effectiveness of these metrics and MER-Inspector. This work aids in understanding and validating current model extraction attacks and defenses, while also inspiring future exploration of new strategies. As for future work, we will continue to explore additional metrics and new attack and defense strategies.

References

[1] Hervé Abdi. 2007. The Kendall rank correlation coefficient. *Encyclopedia of Measurement and Statistics. Sage, Thousand Oaks, CA* (2007), 508–510.

[2] Kwangjun Ahn, Prateek Jain, Ziwei Ji, Satyen Kale, Praneeth Netrapalli, and Gil I Shamir. 2022. Reproducibility in optimization: Theoretical framework and limits. In *Advances in Neural Information Processing Systems*, Vol. 35. 18022–18033.

[3] Yuan Cao and Quanquan Gu. 2019. Generalization bounds of stochastic gradient descent for wide and deep neural networks. In *Advances in neural information processing systems*, Vol. 32.

[4] Varun Chandrasekaran, Kamalika Chaudhuri, Irene Giacomelli, Somesh Jha, and Songbai Yan. 2020. Exploring Connections Between Active Learning and Model Extraction. In *29th USENIX Security Symposium (USENIX Security 20)*. 1309–1326.

[5] Zhenpeng Chen, Yanbin Cao, Huihan Yao, Xuan Lu, Xin Peng, Hong Mei, and Xuanzhe Liu. 2021. Emoji-powered sentiment and emotion detection from software developers' communication data. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 30, 2 (2021), 1–48.

[6] Adam Coates, Andrew Ng, and Honglak Lee. 2011. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. 215–223.

[7] Israel Cohen, Yiteng Huang, Jingdong Chen, Jacob Benesty, Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. 2009. Pearson correlation coefficient. *Noise reduction in speech processing* (2009), 1–4.

[8] Jacson Rodrigues Correia-Silva, Rodrigo F. Berriel, Claudine Badue, Alberto F. de Souza, and Thiago Oliveira-Santos. 2018. Copycat CNN: Stealing Knowledge by Persuading Confession with Random Non-Labeled Data. In *2018 International Joint Conference on Neural Networks*. 1–8.

[9] Benjamin Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé Jégou, and Matthijs Douze. 2021. Levit: a vision transformer in convnet's clothing for faster inference. In *Proceedings of the IEEE/CVF international conference on computer vision*. 12259–12269.

[10] Hrayr Harutyunyan, Ankit Singh Rawat, Aditya Krishna Menon, Seungyeon Kim, and Sanjiv Kumar. 2023. Supervision Complexity and Its Role in Knowledge Distillation. In *International Conference on Learning Representations*.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 770–778.

[12] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4700–4708.

[13] Yujin Huang, Terry Yue Zhuo, Qiongkai Xu, Han Hu, Xingliang Yuan, and Chunyang Chen. [n. d.]. Training-free lexical backdoor attacks on language models. In *Proceedings of the ACM Web Conference 2023 (WWW '23)*.

[14] Arthur Jacot, Franck Gabriel, and Clement Hongler. 2018. Neural Tangent Kernel: Convergence and Generalization in Neural Networks. In *Advances in Neural Information Processing Systems*, Vol. 31.

[15] Matthew Jagielski, Nicholas Carlini, David Berthelot, Alex Kurakin, and Nicolas Papernot. 2020. High accuracy and high fidelity extraction of neural networks. In *29th USENIX security symposium (USENIX Security 20)*. 1345–1362.

[16] Guangda Ji and Zhanxing Zhu. 2020. Knowledge Distillation in Wide Neural Networks: Risk Bound, Data Efficiency and Imperfect Teacher. In *Advances in Neural Information Processing Systems*, Vol. 33. 20823–20833.

[17] Yuting Jia, Qinqin Zhang, Weinan Zhang, and Xinbing Wang. [n. d.]. Communitygan: Community detection with generative adversarial nets. In *Proceedings of the ACM Web Conference 2019 (WWW '19)*.

[18] Wenbo Jiang, Hongwei Li, Guowen Xu, Tianwei Zhang, and Rongxing Lu. 2023. A Comprehensive Defense Framework Against Model Extraction Attacks. *IEEE Transactions on Dependable and Secure Computing* (2023).

[19] Mika Juuti, Sebastian Szyller, Samuel Marchal, and N. Asokan. 2019. PRADA: Protecting Against DNN Model Stealing Attacks. In *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*. 512–527.

[20] Sanjay Kariyappa, Atul Prakash, and Moinuddin K Qureshi. 2021. MAZE: Data-Free Model Stealing Attack Using Zeroth-Order Gradient Estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13809–13818.

[21] Alex Krizhevsky. 2009. *Learning multiple layers of features from tiny images*. Technical Report.

[22] Vitaly Kuznetsov, Mehryar Mohri, and Umar Syed. 2015. Rademacher complexity margin bounds for learning with a large number of classes. In *ICML Workshop on Extreme Classification: Learning with a Very Large Number of Labels*.

[23] Jaehoon Lee, Lechao Xiao, Samuel S. Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. 2019. Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent. In *Advances in Neural Information Processing Systems*.

[24] Taesung Lee, Benjamin Edwards, Ian Molloy, and Dong Su. 2019. Defending against neural network model stealing attacks using deceptive perturbations. In *2019 IEEE Security and Privacy Workshops (SPW)*. 43–49.

[25] Jiacheng Liang, Ren Pang, Changjiang Li, and Ting Wang. 2024. Model Extraction Attacks Revisited. In *Proceedings of the 19th ACM Asia Conference on Computer and Communications Security* (Singapore, Singapore) *(ASIA CCS '24)*. Association for Computing Machinery, New York, NY, USA, 1231–1245.

[26] Bo Liu, Ming Ding, Sina Shaham, Wenny Rahayu, Farhad Farokhi, and Zihuai Lin. 2021. When Machine Learning Meets Privacy: A Survey and Outlook. *Comput. Surveys* 54, 2 (2021), 31:1–31:36.

[27] Yupei Liu, Jinyuan Jia, Hongbin Liu, and Neil Zhenqiang Gong. 2022. StolenEncoder: Stealing Pre-trained Encoders in Self-supervised Learning. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. 2115–2128.

[28] Yunpeng Liu, Kexin Li, Zhuotao Liu, Bihan Wen, Ke Xu, Weiqiang Wang, Wenbiao Zhao, and Qi Li. [n. d.]. Provenance of training without training data: Towards privacy-preserving dnn model ownership verification. In *Proceedings of the ACM Web Conference 2023 (WWW '23)*.

[29] Yugeng Liu, Rui Wen, Xinlei He, Ahmed Salem, Zhikun Zhang, Michael Backes, Emiliano De Cristofaro, Mario Fritz, and Yang Zhang. 2022. ML-DOCTOR: Holistic Risk Assessment of Inference Attacks Against Machine Learning Models. In *31st USENIX Security Symposium (USENIX Security 22)*. 4525–4542.

[30] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep Learning Face Attributes in the Wild. In *Proceedings of the IEEE/CVF international conference on computer vision*.

[31] Noel Loo, Ramin Hasani, Alexander Amini, and Daniela Rus. 2022. Evolution of neural tangent kernels under benign and adversarial training. In *Advances in Neural Information Processing Systems*. 11642–11657.

[32] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations*.

[33] Julian McAuley and Jure Leskovec. 2012. Image labeling on a network: using social-network metadata for image classification. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part IV 12*. Springer, 828–841.

[34] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. 2018. *Foundations of machine learning*.

[35] Jisoo Mok, Byunggook Na, Ji-Hoon Kim, Dongyoon Han, and Sungroh Yoon. 2022. Demystifying the Neural Tangent Kernel From a Practical Perspective: Can It Be Trusted for Neural Architecture Search Without Training?. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11861–11870.

[36] Roman Novak, Jascha Sohl-Dickstein, and Samuel S Schoenholz. 2022. Fast Finite Width Neural Tangent Kernel. In *International Conference on Machine Learning*. 17018–17044.

[37] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. 2019. Knockoff Nets: Stealing Functionality of Black-Box Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4954–4963.

[38] Soham Pal, Yash Gupta, Aditya Shukla, Aditya Kanade, Shirish Shevade, and Vinod Ganapathy. 2020. ActiveThief: Model Extraction Using Active Learning and Unannotated Public Data. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 865–872.

[39] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. 2017. Practical Black-Box Attacks against Machine Learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*. 506–519.

[40] Mary Phuong and Christoph H. Lampert. 2019. Towards Understanding Knowledge Distillation. In *International Conference on Machine Learning*.

[41] Sunandini Sanyal, Sravanti Addepalli, and R. Venkatesh Babu. 2022. Towards Data-Free Model Stealing in a Hard Label Setting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 15284–15293.

[42] Bernhard Schölkopf, Ralf Herbrich, and Alex J Smola. 2001. A generalized representer theorem. In *International conference on computational learning theory*. 416–426.

[43] Liwei Song and Prateek Mittal. 2021. Systematic Evaluation of Privacy Risks of Machine Learning Models. In *30th USENIX Security Symposium (USENIX Security 21)*.

[44] Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. 2022. Beyond Neural Scaling Laws: Beating Power Law Scaling via Data Pruning. In *Advances in Neural Information Processing Systems*. 19523–19536.

[45] Alexandru Tifrea, Jacob Clarysse, and Fanny Yang. 2023. Margin-based sampling in high dimensions: When being active is less efficient than staying passive. In *International Conference on Machine Learning*. 34222–34262.

[46] Florian Tramèr, Fan Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart. 2016. Stealing Machine Learning Models via Prediction APIs. In *25th USENIX Security Symposium (USENIX Security 16)*. 601–618.

[47] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv preprint arXiv:1708.07747* (2017).

[48] Yaxin Xiao, Qingqing Ye, Haibo Hu, Huadi Zheng, Chengfang Fang, and Jie Shi. 2022. MExMI: Pool-based Active Model Extraction Crossover Membership

Inference. In *Advances in Neural Information Processing Systems*, Vol. 35. 10203–10216.
[49] Jingjing Xu, Liang Zhao, Junyang Lin, Rundong Gao, Xu Sun, and Hongxia Yang. 2021. KNAS: green neural architecture search. In *International Conference on Machine Learning*. 11613–11625.
[50] Jianyi Yang and Shaolei Ren. 2022. Informed Learning by Wide Neural Networks: Convergence, Generalization and Sampling Complexity. In *International Conference on Machine Learning*. 25198–25240.
[51] Jiayuan Ye, Aadyaa Maddi, Sasi Kumar Murakonda, Vincent Bindschaedler, and Reza Shokri. 2022. Enhanced membership inference attacks against machine learning models. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. 3093–3106.
[52] Honggang Yu, Kaichen Yang, Teng Zhang, Yun-Yun Tsai, Tsung-Yi Ho, and Yier Jin. 2020. CloudLeak: Large-Scale Deep Learning Models Stealing Through Adversarial Examples. In *Network and Distributed Systems Security (NDSS) Symposium*.
[53] Sergey Zagoruyko and Nikos Komodakis. 2016. Wide residual networks. *arXiv preprint arXiv:1605.07146* (2016).
[54] Boyang Zhang, Zheng Li, Ziqing Yang, Xinlei He, Michael Backes, Mario Fritz, and Yang Zhang. 2024. SecurityNet: Assessing Machine Learning Vulnerabilities on Public Models. In *34th USENIX Security Symposium (USENIX Security 24)*.
[55] Jiliang Zhang, Shuang Peng, Yansong Gao, Zhi Zhang, and Qinghui Hong. 2023. APMSA: adversarial perturbation against model stealing attacks. *IEEE Transactions on Information Forensics and Security* 18 (2023), 1667–1679.

A Proof

A.1 Basic Theory

Rademacher complexity quantifies the expressive power of a machine-learning model and the inherent risk of overfitting. The definition is given as follows [34].

Definition A.1 (Rademacher complexity)**.** Given samples $S = \{x_1, x_2, \ldots, x_n\}$ of size n drawn i.i.d. from some distributions, and a class of functions F, the empirical Rademacher complexity of F with respect to S is defined as:

$$\hat{\mathfrak{R}}_n(F) = \frac{1}{n}\mathbb{E}_\sigma\left[\sup_{f\in F}\sum_{i=1}^{n}\sigma_i f(x_i)\right] \tag{11}$$

where $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n)$ is a vector of independent Rademacher random variables (i.e., each σ_i takes values +1 or -1 with equal probability). The (population) Rademacher complexity of F is then defined as the expectation of the empirical Rademacher complexity over all possible samples of size n, which is given by:

$$\mathfrak{R}_n(F) = \mathbb{E}_S[\hat{R}_n(F)]. \tag{12}$$

Remarks. Rademacher complexity measures the complexity of the hypothesis space $\mathcal{H}$ in which the model resides and is a powerful indicator of the model extraction risk. If the complexity of the hypothesis space is higher, the behavior of the model will be more difficult to predict and understand, which may make model extraction attacks more difficult. However, this complexity cannot be used directly to measure risk due to the high computational cost and separation with models. Firstly, the computation of Rademacher complexity often involves optimizing over a large hypothesis space, which can be computationally intensive, especially for complex models or large datasets. Furthermore, we rely on empirical estimates of Rademacher complexity to obtain accurate estimates, which requires repeated estimation of the complexity multiple times, which further increases the computational burden. Secondly, Rademacher complexity is based on the hypothesis space $\mathcal{H}$ and the nature of the data rather than on a specific model. Therefore, it may not provide in-depth insights into the behavior and performance of a specific model, which may lead to an inaccurate measurement of model complexity.

For a multi-label classification task, given a labeled example $(\mathbf{x}, y)$, the prediction margin is defined as $\mathcal{M}(\mathbf{x}, y) = f^y(\mathbf{x}) - \max_{y'\neq y} f^{y'}(\mathbf{x})$. Considering the class of functions $\mathcal{F} = \{f \in \mathcal{H} : \| f \| \leq M\}$ for some $M > 0$, we can give the generalization bound based on the Rademacher complexity in the RKHS $\mathcal{H}$ by the theorem proposed in [22].

Theorem A.2 (Generalization bound)**.** *Considering that the training dataset $\mathbb{D}$ comprises N i.i.d. is samples, each associated with one of K labels, derived from the input distribution $P_{\mathbf{x}}$. Assume $\kappa = \sup_{\mathbf{x}\in P_{\mathbf{x}}} k(\mathbf{x}, \mathbf{x}) < \infty$. Fix any constant $M > 0$ and $\gamma > 0$. Let $M_0 = \lceil \gamma\sqrt{N}/(4K\sqrt{\kappa})\rceil$. Then with probability at least $1-\delta$, every function $f \in \mathcal{H}$ has the following bound:*

$$\begin{aligned}\mathbb{P}_{X,Y}\left(\mathcal{M}(\mathbf{x}, y) \leq 0\right) \leq & \frac{1}{N}\sum_{i=1}^{N}\mathbf{1}\left\{\mathcal{M}(\mathbf{x}^{(i)}, y^{(i)}) \leq \gamma\right\} \\ & + \frac{4K}{\gamma}\hat{\mathfrak{R}}_n(F) + 3\sqrt{\frac{\log(2M_0/\delta)}{2N}}.\end{aligned} \tag{13}$$

Based on this generalization bound, we further give two theoretical bounds on the attack performance.

A.2 Proof of Theorem 4.1

Theorem A.3 (Theorm 4.1 restated)**.** *Assume that $\kappa = \sup_{\mathbf{x}\sim P_{\mathbf{x}}} k(\mathbf{x}, \mathbf{x}) < \infty$. Define a constant $\gamma > 0$, and let $M_0 = \left\lceil \frac{\gamma\sqrt{N}}{4K\sqrt{\kappa}} \right\rceil$. Subsequently, with a probability of at least $1-\delta$, for every function $\triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})} \in \mathcal{H}$, the fidelity gap on N samples can be bounded:*

$$\begin{aligned}\mathcal{G}_N = & \mathop{\mathbb{P}}_{\mathbf{x}\sim P_{\mathbf{x}}}\left[\mathcal{M}(\mathbf{x}, y_v, y_s) \leq 0\right] \\ \leq & \frac{1}{N}\sum_{i=1}^{N}\mathbf{1}\left\{\mathcal{M}(\mathbf{x}^{(i)}, y_v^{(i)}, y_s^{(i)}) \leq \gamma\right\} \\ & + \frac{4K(\triangle_{\mathbf{f}_{\theta_v}(\mathbf{x})}^{\top}\Theta^{-1}\triangle_{\mathbf{f}_{\theta_v}(\mathbf{x})})}{\gamma N}\sqrt{Tr(\Theta)} \\ & + 3\sqrt{\frac{\log(2M_0/\delta)}{2N}},\end{aligned} \tag{14}$$

where $Tr(\Theta)$ represents the trace of the victim model's NTK matrix Θ.

Proof. Harutyunyan et al. [10] give the upper bound of the empirical Rademacher complexity of $\mathcal{F}$, i.e.,

$$\hat{\mathfrak{R}}_n(F) \leq \frac{M}{N}\sqrt{\mathrm{Tr}(\Theta)}. \tag{15}$$

Under the assumptions of the NTK theory and no two samples are the same, the NTK matrix is the positive definite matrix, and Θ is full rank surely. Based on the Eq. (5), the solution is

$$\triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})} = \sum_{i=1}^{N}\alpha_i k(\mathbf{x}, \mathbf{x}^{(i)}) = \Theta\boldsymbol{\alpha}. \tag{16}$$

When $\triangle_{f_{\theta_s}(\mathbf{x})}$ in $\mathcal{H}$ has zero empirical loss, we have $\triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})} = \triangle_{\mathbf{f}_{\theta_v}(\mathbf{x})}$, and the norm of $\triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})}$ can be expressed as

$$\| \triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})} \|_{\mathcal{H}}^2 = \boldsymbol{\alpha}^{\top}\Theta\boldsymbol{\alpha} = \triangle_{\mathbf{f}_{\theta_v}(\mathbf{x})}^{\top}\Theta^{-1}\triangle_{\mathbf{f}_{\theta_v}(\mathbf{x})}. \tag{17}$$

Therefore, any optimal solution to the optimization problem in Section 3 has a norm at most $\triangle_{\mathbf{f}_{\theta_v}(\mathbf{x})}^{\top}\Theta^{-1}\triangle_{\mathbf{f}_{\theta_v}(\mathbf{x})}$. Considering the class of functions $\mathcal{F} = \{\triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})} \in \mathcal{H} : \| \triangle_{\mathbf{f}_{\theta_s}(\mathbf{x})} \| \leq M\}$ for some $M > 0$, we can get that

$$M \leq \triangle_{\mathbf{f}_{\theta_v}(\mathbf{x})}^{\top}\Theta^{-1}\triangle_{\mathbf{f}_{\theta_v}(\mathbf{x})}. \tag{18}$$

and

$$\hat{\mathfrak{R}}_n(F) \leq \frac{\triangle_{\mathbf{f}_{\theta_v}(\mathbf{x})}^{\top}\Theta^{-1}\triangle_{\mathbf{f}_{\theta_v}(\mathbf{x})}}{N}\sqrt{\mathrm{Tr}(\Theta)}. \tag{19}$$

After substituting Eq. (19) into Eq. (13), we can prove this theorem. □

A.3 Proof of Theorem 4.2

Theorem A.4 (Theorem 4.2 restated). *Given the fidelity gap bound $\mathcal{G}_N$ and the generalization error bound of the victim model $\mathcal{R}_N^v$. Then, the generalization error of the surrogate model is bounded by,*

$$\mathcal{R}_N^s \leq \mathcal{G}_N + \mathcal{R}_N^v. \tag{20}$$

Proof. Define the prediction margin $\mathcal{M}(\mathbf{x}, y_v, y_s) = f_{\theta_s}^{y_v}(\mathbf{x}) - \max_{y' \neq y_v} f_{\theta_s}^{y'}(\mathbf{x})$, $\mathcal{M}(\mathbf{x}, y_v, y) = f_{\theta_v}^{y}(\mathbf{x}) - \max_{y' \neq y} f_{\theta_v}^{y'}(\mathbf{x})$, $\mathcal{M}(\mathbf{x}, y_s, y) = f_{\theta_s}^{y}(\mathbf{x}) - \max_{y' \neq y} f_{\theta_s}^{y'}(\mathbf{x})$, we have

$$\mathcal{G}_N = \mathop{\mathbb{P}}_{\mathbf{x}\sim P_{\mathbf{x}}}\left[\mathcal{M}(\mathbf{x}, y_v, y_s) \leq 0\right], \tag{21}$$

and

$$\mathcal{R}_N^v = \mathop{\mathbb{P}}_{\mathbf{x}\sim P_{\mathbf{x}}}\left[\mathcal{M}(\mathbf{x}, y_v, y) \leq 0\right]. \tag{22}$$

Thus

$$\begin{aligned}
\mathcal{R}_N^s &= \mathop{\mathbb{P}}_{\mathbf{x}\sim P_{\mathbf{x}}}\left[\mathcal{M}(\mathbf{x}, y_s, y) \leq 0\right]. \\
&= \mathop{\mathbb{P}}_{\mathbf{x}\sim P_{\mathbf{x}}}\left[\mathcal{M}(\mathbf{x}, y_s, y) \leq 0 \wedge \mathcal{M}(\mathbf{x}, y_v, y) \leq 0\right] \\
&\quad + \mathop{\mathbb{P}}_{\mathbf{x}\sim P_{\mathbf{x}}}\left[\mathcal{M}(\mathbf{x}, y_s, y) \leq 0 \wedge \mathcal{M}(\mathbf{x}, y_v, y) > 0\right] \\
&= \mathop{\mathbb{P}}_{\mathbf{x}\sim P_{\mathbf{x}}}\left[\mathcal{M}(\mathbf{x}, y_s, y) \leq 0 \wedge \mathcal{M}(\mathbf{x}, y_v, y) \leq 0\right] \\
&\quad + \mathop{\mathbb{P}}_{\mathbf{x}\sim P_{\mathbf{x}}}\left[\mathcal{M}(\mathbf{x}, y_v, y_s) \leq 0\right] \\
&\leq \mathop{\mathbb{P}}_{\mathbf{x}\sim P_{\mathbf{x}}}\left[\mathcal{M}(\mathbf{x}, y_v, y) \leq 0\right] + \mathop{\mathbb{P}}_{\mathbf{x}\sim P_{\mathbf{x}}}\left[\mathcal{M}(\mathbf{x}, y_v, y_s) \leq 0\right] \\
&= \mathcal{R}_N^v + \mathcal{G}_N.
\end{aligned} \tag{23}$$

□

B Datasets and Models

We conduct experiments on five popular datasets:

- CIFAR-10 [21] contains 60,000 32x32 color images across 10 different classes, such as automobiles, birds, and ships, with each class containing 6,000 images. The dataset is typically divided into 50,000 training images and 10,000 testing images.
- CIFAR-100 [21] contains 60,000 32x32 color images across 100 different classes, with 600 images (500 for training, 100 for testing) per class.
- FashionMNIST [47] contains 70,000 28x28 grayscale images of fashion products from 10 categories, such as trousers, pullovers, and sandals. The dataset is typically divided into 60,000 training images and 10,000 testing images.
- STL-10 [6] contains 5,000 96x96 color training images from 10 distinct classes, with each class represented in the testing set by 800 images.
- CelebA [30] contains 202,599 facial color images, each associated with 40 binary attributes. Like Liu et al. [29], we select and combine 3 attributes from 40 attributes, including HeavyMakeup, MouthSlightlyOpen, and Smiling, to form the labels of the target model, resulting in an 8-category classification. We resize the image pixels to 32x32 and randomly select 20% of the entire dataset for training and testing, generating 32,554 training samples and 3,992 testing samples.

We conduct experiments on four famous architecture groups:

- ResNet [11] represents the classic approach of deep residual learning where skip connections facilitate the training of deeper networks. We use its variants ResNet20, ResNet32, and ResNet44.
- WideResNet [53] modifies the ResNet architecture by increasing width. We use its variants such as WideResNet22-2, WideResNet22-4, WideResNet22-8, WideResNet28-2, WideResNet34-2, and WideResNet40-2. In "WideResNetN-M", "N" denotes the depth of the network, while "M" signifies the multiplier for the number of convolutional kernels in comparison to the standard ResNet model.
- DenseNet [12] connects each layer to other layers in a feed-forward manner. We use its variants such as DenseNet121, DenseNet169, and DenseNet201.
- LeViT [9] is a transformer-based architecture for image classification. We evaluate its various configurations, including LeViT-128, LeViT-192, LeViT-256, and LeViT-384.

C Experimental Setup of JBDA and MAZE

JBDA [39] generates crafted adversarial examples for querying the victim model when the adversary has access to a few natural samples. Given a sample x and its label y, a crafted query sample is constructed as

$$x \leftarrow x - \lambda \nabla_x \mathcal{L}(y, f_s(x)), \tag{24}$$

where λ is the learning rate, $\mathcal{L}$ is the loss function, ∇ is the gradient, $f_s(x)$ is the output of the surrogate model. In our experiment, CIFAR-10 is used as the training dataset of the victim model, and ResNet20 is used as the architecture. The accuracy of the victim model is 93.30%. We randomly select 2,048 samples from the training dataset as the thief dataset and use the same architecture of the victim model to perform the attack. Adam is used as an optimizer with a learning rate of 0.01. The λ is set to 0.1. The loss function is Kullback-Leibler divergence. Figure 2(a) shows the training process of the surrogate model with the increasing query times.

MAZE [20] generates synthetic data using a generative model without accessing any natural samples. This is achieved by a game theory optimization problem as follows:

$$\min_{f_s}\max_{G} \mathbb{E}_{z\sim\mathcal{N}(0,1)}\left[\mathcal{L}(f_v(G(z)), f_s(G(z)))\right], \tag{25}$$

where $\mathbb{E}_{z\sim\mathcal{N}(0,1)}$ denotes the expectation over the noise z, which is sampled from the standard Gaussian distribution $\mathcal{N}(0, 1)$, $\mathcal{L}$ is the loss function that measures the distance between the outputs of the victim model f_v and the surrogate model f_s, $G(z)$ is the

generative model that takes noise z as input and generates samples. This optimization problem attempts to find the best surrogate model f_s while also adjusting the generative model G to produce increasingly indistinguishable samples. In our experiment, the victim model is the same as in JBDA. The adversary can access 2,048 samples, which can largely improve performance compared to that performance without any samples. The loss function is Kullback-Leibler divergence. Adam is used as the optimizer with a learning rate of 0.01 for the surrogate model and 1e-4 for the generative model. Other parameters are the same as in the original paper [20]. Figure 2(b) shows the training process of the surrogate model with the increasing query times.

D Additional Experimental Details and Results

D.1 Detailed Experimental Setup

We conduct experiments using eight NVIDIA RTX 4090 GPUs, each with 24GB of memory, running on an Intel(R) Xeon(R) Platinum 8352V CPU. We implemented the experiments using PyTorch 2.0.0 and Python 3.8 on an Ubuntu 20.04 system.

For victim models, we use the SGD optimizer with a learning rate of 0.01 to train the model. The momentum, weight decay, and other parameters for SGD are set to the default value. The batch size is set to 128 for CIFAR-10, CIFAR-100, FashionMNIST, and CelebA, and 32 for STL-10. The epoch of LeViT is set to 400, and the other epochs are set to 200 due to the slow convergence speed of LeViT.

For surrogate models, we use all training samples as the thief dataset for querying and training 100 epochs. The optimizer and other hyper-parameters are the same as those used in the victim models.

In the default settings, we use 400 training samples on CIFAR-10, STL-10, FashionMNIST, and CelebA to calculate MRC. Since the category of CIFAR-100 is large, we use 40 samples on CIFAR-100 to calculate MRC. q and η are set to 0.5.

D.2 Effectiveness of Metrics on Other Datasets

We analyze the effectiveness of proposed metrics on more datasets. As shown in Figure 6, the negative relationship between MRC and attack fidelity exists in most groups. There are two opposite cases: the LeViT group on FashionMNIST and the DenseNet group on CelebA. This is because the model parameters of LeViT and DenseNet are both very large. Using only 400 training samples for evaluation is insufficient, so the calculated MRC scores are inaccurate. If the number of samples used to calculate MRC increases, the accuracy of MRC scores in parameter-rich models will improve, but it will also significantly increase GPU and memory overhead. For the models with fewer parameters (ResNet, WideResNet), only 100 samples are sufficient for evaluation. Therefore, we must balance the trade-off between computational accuracy and cost consumption. In this paper, we select 400 samples to compute metrics, achieving good accuracy at an acceptable cost. In addition, Figure 7 further verifies that there is an overall positive relationship between VMA and attack fidelity, which is not conclusive within each group.

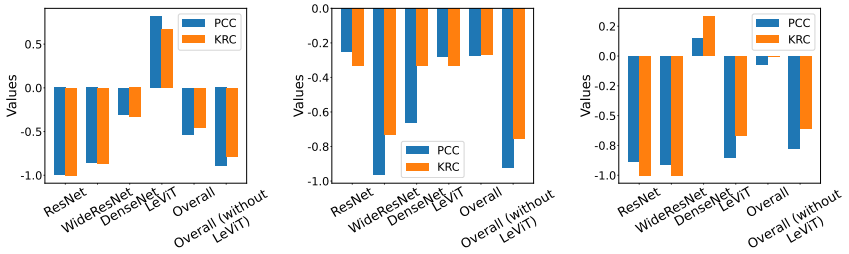
With these additional results, we further validate the effectiveness of proposed metrics and also demonstrate the necessity of combining them.

D.3 Detailed Experimental Results

We give specific numerical values of the results, including the number of parameters (denoted as #PARA), VMA, attack fidelity, attack accuracy, and the calculated MRC. The particular results for CIFAR-10 can be found in Table 8. Similarly, the results for STL-10 in Table 9, for FashionMNIST in Table 10, for CIFAR-100 are detailed in Table 11, and for CelebA, the data is provided in Table 12. Intuitively, #PARA may be related to model complexity, thus reflecting the difficulty of MEAs. We experimentally verify the correlation between #PARA and attack fidelity. Contrary to expectations, the results indicate that no definite relationship exists between #PARA and attack fidelity, with a PCC of -0.28 and a KRC of 0.15 on CIFAR-10.

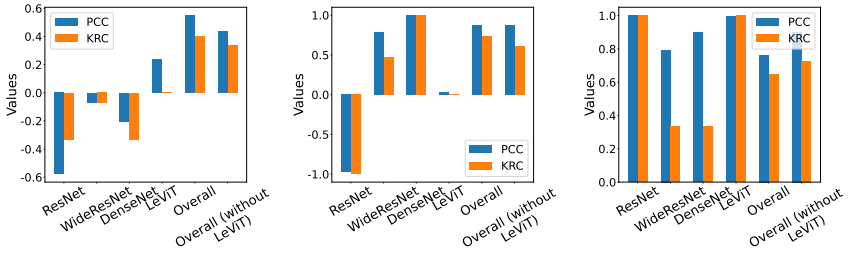
D.4 Detailed Results on Impact of η

We analyze the impact of difficulty ratio η on CIFAR-10, and the detailed results are given in Table 13. It is worth noting that the MRC scores obtained from randomly selected samples are occasionally lower than those computed solely from hard samples ($\eta = 1$). This observation elucidates why attack strategies relying on randomly selected samples can occasionally outperform active learning-based approaches [38, 45]. We also observe that with 400 samples, many models have lower MRC scores when $\eta = 0$. This emphasizes the importance of simple samples, especially when the sample size is small. This conclusion aligns with the pruning strategy proposed by Sorscher et al. [44], suggesting the use of simple samples in data-scarce scenarios. In our results, it is observed that only within the DenseNet group with the best generalization ability, the lowest MRC score is obtained when $\eta = 1$, which means that hard samples hold greater significance when attempting to extract models with superior generalization capabilities.



(a) FashionMNIST (b) CIFAR-100 (c) CelebA

Figure 6: PCC and KRC results between MRC and attack fidelity on (a) FashionMNIST, (B) CIFAR-100 and (c) CelebA.



(a) FashionMNIST (b) CIFAR-100 (c) CelebA

Figure 7: PCC and KRC results between VMA and attack fidelity on (a) FashionMNIST, (B) CIFAR-100 and (c) CelebA.

Table 8: Results of CIFAR-10.

Model	#PARA	Victim Model Accuracy (%)	Attack Accuracy (%) / Fidelity	Model Recovery Complexity L=100	L=200	L=400
ResNet20	272,474	90.94	80.27 / 0.8135	0.00106	0.00206	0.00353
ResNet32	466,906	91.66	80.77 / 0.8088	0.00101	0.00221	0.00405
ResNet44	661,338	92.41	79.97 / 0.8031	0.00106	0.00232	0.00736
WideResNet22-2	1,079,642	92.97	85.18 / 0.8601	0.00093	0.00168	0.00350
WideResNet28-2	1,467,610	93.28	82.86 / 0.8312	0.00265	0.00425	0.00797
WideResNet34-2	1,855,578	93.63	83.33 / 0.8365	0.00107	0.00213	0.00444
WideResNet40-2	2,243,546	93.46	82.80 / 0.8323	0.00250	0.00523	0.00973
WideResNet22-4	4,298,970	94.25	88.19 / 0.8841	0.00051	0.00095	0.00183
WideResNet22-8	17,158,106	94.62	87.98 / 0.8858	0.00041	0.00076	0.00135
DenseNet121	6,956,298	94.99	88.71 / 0.8912	0.00041	0.00061	0.00187
DenseNet169	12,493,322	94.67	88.93 / 0.8924	0.00041	0.00058	0.00204
DenseNet201	18,104,330	94.85	89.26 / 0.8969	0.00037	0.00052	0.00181
LeViT-128	8,439,386	81.43	70.97 / 0.7182	0.00047	0.00087	0.00180
LeViT-192	10,174,943	83.26	72.52 / 0.7305	0.00034	0.00065	0.00130
LeViT-256	17,865,166	83.62	73.69 / 0.7347	0.00027	0.00053	0.00110
LeViT-384	37,586,862	84.00	72.30 / 0.7255	0.00026	0.00057	0.00131

Table 9: Results of STL-10.

Model	#PARA	Victim Model Accuracy (%)	Attack Accuracy (%) / Fidelity	Model Recovery Complexity L=100	L=200	L=400
ResNet20	4,327,754	83.96	74.10 / 0.7699	0.00337	0.00802	0.01811
ResNet32	7,427,914	83.79	72.39 / 0.7430	0.00334	0.00857	0.01939
ResNet44	10,528,074	84.54	70.90 / 0.7300	0.00390	0.00901	0.02142
WideResNet22-2	1,079,642	83.89	66.33 / 0.6770	0.00594	0.01507	0.03317
WideResNet28-2	1,467,610	85.28	65.96 / 0.6705	0.00729	0.01668	0.03602
WideResNet34-2	1,855,578	85.19	66.08 / 0.6761	0.00563	0.01311	0.02709
WideResNet40-2	2,243,546	85.25	64.53 / 0.6679	0.00882	0.01708	0.03666
WideResNet22-4	4,298,970	85.61	67.45 / 0.7011	0.00365	0.00903	0.02032
WideResNet22-8	17,158,106	87.28	72.01 / 0.7474	0.00330	0.00725	0.01654
DenseNet121	6,956,298	85.33	82.00 / 0.8451	0.00278	0.00637	0.02611
DenseNet169	12,493,322	87.40	83.29 / 0.8536	0.00244	0.00481	0.02415
DenseNet201	18,104,330	86.47	82.82 / 0.8479	0.00233	0.00484	0.02563
LeViT-128	8,440,682	68.51	48.20 / 0.4810	0.00327	0.00787	0.02251
LeViT-192	10,175,823	72.55	45.20 / 0.4626	0.00302	0.00892	0.02835
LeViT-256	17,866,318	73.14	45.45 / 0.4612	0.00327	0.01001	0.03054
LeViT-384	37,588,574	72.10	46.30 / 0.4818	0.00392	0.01134	0.02965

Table 10: Results of FashionMNIST.

Model	#PARA	Victim Model Accuracy (%)	Attack Accuracy (%) / Fidelity	Model Recovery Complexity L=100	L=200	L=400
ResNet20	272,826	93.68	92.16 / 0.9383	0.00103	0.00235	0.00398
ResNet32	467,258	94.13	91.98 / 0.9365	0.00198	0.00339	0.00632
ResNet44	661,690	94.07	91.39 / 0.9303	0.00280	0.00698	0.01226
WideResNet22-2	1,079,354	94.67	90.50 / 0.9381	0.00057	0.00105	0.00262
WideResNet28-2	1,467,322	94.84	92.02 / 0.9301	0.00228	0.00391	0.00677
WideResNet34-2	1,855,290	94.85	92.15 / 0.9355	0.00132	0.00257	0.00482
WideResNet40-2	2,243,258	95.01	91.35 / 0.9310	0.00397	0.00671	0.01145
WideResNet22-4	4,298,682	95.01	91.42 / 0.9427	0.00057	0.00108	0.00216
WideResNet22-8	17,157,818	94.84	91.34 / 0.9449	0.00050	0.00084	0.00142
DenseNet121	6,955,146	95.27	93.52 / 0.9432	0.00051	0.00082	0.00084
DenseNet169	12,492,170	95.21	92.96 / 0.9482	0.00052	0.00082	0.00078
DenseNet201	18,103,178	95.17	92.04 / 0.9437	0.00049	0.00076	0.00076
LeViT-128	8,439,098	92.09	91.06 / 0.9339	0.00047	0.00098	0.00233
LeViT-192	10,174,511	93.03	91.52 / 0.9379	0.00045	0.00093	0.00204
LeViT-256	17,864,590	92.61	90.96 / 0.9324	0.00035	0.00072	0.00168
LeViT-384	37,585,998	92.62	90.78 / 0.9252	0.00028	0.00059	0.00126

Table 11: Results of CIFAR-100.

Model	#PARA	Victim Model Accuracy (%)	Attack Accuracy (%) / Fidelity	Model Recovery Complexity L=20	L=40
ResNet20	278,324	65.76	51.37 / 0.5498	0.01945	0.09119
ResNet32	472,756	66.79	50.54 / 0.5156	0.02615	0.13944
ResNet44	667,188	68.25	48.17 / 0.4931	0.03096	0.09816
WideResNet22-2	1,091,252	69.22	53.85 / 0.5457	0.02527	0.06880
WideResNet28-2	1,479,220	69.40	57.06 / 0.5707	0.03112	0.07286
WideResNet34-2	1,867,188	70.53	49.88 / 0.5042	0.03050	0.08139
WideResNet40-2	2,255,156	71.03	50.40 / 0.5050	0.03940	0.09775
WideResNet22-4	4,322,100	73.26	60.81 / 0.6221	0.01274	0.03498
WideResNet22-8	17,204,276	76.14	64.08 / 0.6579	0.00909	0.01826
DenseNet121	7,048,548	76.13	65.03 / 0.6633	0.00453	0.00009
DenseNet169	12,643,172	77.15	70.71 / 0.7266	0.00353	0.00005
DenseNet201	18,277,220	77.22	71.40 / 0.7353	0.00390	0.00008
LeViT-128	8,474,036	48.35	38.33 / 0.4164	0.00312	0.00565
LeViT-192	10,209,593	50.69	40.07 / 0.4321	0.00558	0.01247
LeViT-256	17,911,336	54.57	39.84 / 0.4064	0.00261	0.02479
LeViT-384	37,656,072	56.68	41.32 / 0.4294	0.00875	0.02019

Table 12: Results of CelebA.

Model	#PARA	Victim Model Accuracy (%)	Attack Accuracy (%) / Fidelity	Model Recovery Complexity L=100	L=200	L=400
ResNet20	272,344	71.62	69.81 / 0.7437	0.02541	0.04658	0.09466
ResNet32	466,776	71.17	71.29 / 0.7610	0.01486	0.02508	0.04914
ResNet44	661,208	72.02	69.61 / 0.7533	0.02453	0.05230	0.08990
WideResNet22-2	1,079,384	73.77	70.57 / 0.7650	0.04911	0.06883	0.12100
WideResNet28-2	1,467,352	73.55	70.87 / 0.7801	0.02911	0.04271	0.06296
WideResNet34-2	1,855,320	72.82	71.99 / 0.7783	0.03305	0.04729	0.06558
WideResNet40-2	2,243,288	72.82	71.94 / 0.7703	0.03751	0.05336	0.07329
WideResNet22-4	4,298,456	74.05	73.45 / 0.8104	0.01138	0.01429	0.02040
WideResNet22-8	17,157,080	74.67	73.72 / 0.8324	0.00562	0.00803	0.01143
DenseNet121	6,954,248	73.90	72.65 / 0.7938	0.03182	0.05440	0.07745
DenseNet169	12,489,992	74.37	71.22 / 0.7963	0.01396	0.01844	0.02806
DenseNet201	18,100,488	74.00	72.85 / 0.7913	0.01107	0.01437	0.01891
LeViT-128	8,439,048	67.23	70.79 / 0.7282	0.00211	0.00382	0.00773
LeViT-192	10,174,476	69.94	71.64 / 0.7485	0.00103	0.00249	0.00509
LeViT-256	17,864,536	68.74	70.22 / 0.7402	0.00129	0.00224	0.00413
LeViT-384	37,585,912	69.44	69.61 / 0.7523	0.00119	0.00193	0.00363

Table 13: Results on the impact of the difficulty ratio η.

Model	Model Recovery Complexity					
	$\eta = 0$	$\eta = 0.25$	$\eta = 0.5$	$\eta = 0.75$	$\eta = 1$	Random
ResNet20	**0.00278**	0.00324	0.00353	0.00388	0.00416	0.00729
ResNet32	0.00532	0.00528	**0.00405**	0.00465	0.00569	0.00734
ResNet44	**0.00231**	0.00408	0.00736	0.00698	0.00839	0.00620
WideResNet22-2	**0.00127**	0.00230	0.00350	0.00455	0.00504	0.00295
WideResNet28-2	**0.00464**	0.00645	0.00797	0.00949	0.01002	0.00949
WideResNet34-2	**0.00286**	0.00357	0.00444	0.00533	0.00601	0.00654
WideResNet40-2	**0.00364**	0.00665	0.00973	0.01157	0.01182	0.00971
WideResNet22-4	**0.00054**	0.00127	0.00183	0.00228	0.00247	0.00134
WideResNet22-8	**0.00049**	0.00102	0.00135	0.00151	0.00136	0.00103
DenseNet121	0.00193	0.00205	0.00187	0.00180	**0.00124**	0.00212
DenseNet169	0.00177	0.00201	0.00204	0.00184	**0.00136**	0.00214
DenseNet201	0.00160	0.00172	0.00181	0.00161	**0.00110**	0.00201
LeViT-128	**0.00090**	0.00138	0.00180	0.00213	0.00238	0.00527
LeViT-192	**0.00085**	0.00109	0.00130	0.00149	0.00166	0.00336
LeViT-256	**0.00067**	0.00088	0.00110	0.00135	0.00174	0.00235
LeViT-384	**0.00055**	0.00084	0.00131	0.00183	0.00241	0.00172