PRIVACY VULNERABILITIES IN MACHINE LEARNING: FROM INFERENCE ATTACKS TO RISK ASSESSMENT

BAI LI

PhD

The Hong Kong Polytechnic University

2026

The Hong Kong Polytechnic University
Department of Electrical and Electronic Engineering

Privacy Vulnerabilities in Machine Learning: From Inference Attacks to Risk Assessment

Bai Li

A thesis submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy
September 2025

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

Signature: _____

Name of Student: Bai Li

Abstract

Machine learning (ML) has emerged as a fundamental driver of technological progress across diverse domains, powering accurate predictions and data-driven decision-making. However, as ML models are increasingly learned on sensitive data such as medical records and private images, concerns over data privacy have grown significantly. A growing body of research indicates that these models may inadvertently reveal information about their training data through inference attacks. Within this context, we investigate how membership and property inference attacks exploit model behaviors to leak sensitive information at both the sample and distribution levels, and subsequently propose a risk measure to evaluate privacy risks across different models.

First, we improve the efficiency of shadow model-based inference attacks in an attack-agnostic manner. While shadow models are a common component in inference attacks, their reliance on training a large number of models incurs substantial computational overhead and limits their practicality. Such inefficiency mainly stems from the independent training and use of these shadow models. Regarding this problem, we present a novel shadow pool training framework, which constructs multiple shared models and trains them jointly within a single process. In particular, we leverage the Mixture-of-Experts mechanism as the shadow pool to interconnect individual models, enabling them to share sub-networks and improving efficiency.

Secondly, we investigate a new attack surface in vertical federated learning (VFL), highlighting previously underexplored privacy leakage risks inherent to this setting.

While various attacks, including label and feature inference, focus on record-level privacy risks in VFL, few studies delve into the distribution-level privacy threat. To fill this gap, we focus on a new attack scenario, where an adversarial party seeks to deduce global distribution information about a target property in the victim party's training set. Our key observation is that the L_p-norm distribution of intermediate results in VFL could reflect the fraction of the target property in a training set. Inspired by this, we present a novel property inference framework involving distribution comparison and correlation augmentation modules, which poses the immediate threat of property information leakage from private training data in VFL.

Thirdly, we develop a theoretical measure to assess the membership privacy risk of ML models. Privacy leakage poses a critical risk when ML foundation models trained on private data are released. Rather than proposing new attack algorithms, our focus is on evaluating model vulnerability to membership leakage by introducing a novel measure from a comparative perspective. We calculate the difference in prediction loss for training examples relative to a predefined reference model, enabling risk comparison across models without needing to delve into details like training strategy, architecture, or data distribution.

In summary, this thesis investigates and evaluates inference attacks in ML models, thereby shedding light on privacy risks and providing insights to guide the design of privacy-preserving models in future work.

Publications Arising from the Thesis

1. Bai L, Liu J, Zhang S, et al. United We Defend: Collaborative Membership Inference Defenses in Federated Learning. Accepted by *USENIX Security*, 2026.

2. Bai L, Ye Q, Zhang X, et al. Toward Efficient Inference Attacks: Shadow Model Sharing via Mixture-of-Experts. Accepted by *Conference on Neural Information Processing Systems* (NeurIPS), 2025.

3. Bai L, Zhang X, Zhang S, et al. ProVFL: Property Inference Attacks against Vertical Federated Learning[J]. *IEEE Transactions on Information Forensics and Security*, 2025, 20: 6529-6543.

4. Bai L, Hu H, Ye Q, et al. RMR: A Relative Membership Risk Measure for Machine Learning Models[J]. *IEEE Transactions on Dependable and Secure Computing*, 2025, 22(5): 4699-4710.

5. Bai L, Hu H, Ye Q, et al. Membership inference attacks and defenses in federated learning: A survey[J]. *ACM Computing Surveys*, 2024, 57(4): 1-35.

Acknowledgments

What began as a dream of a hill has led me to the summit of a mountain. As I lift my eyes to the horizon, bathed in sunlight and stirred by the wind, I recall the struggles and memories that have shaped my journey.

The beginning of any journey is often the hardest, yet I am deeply grateful and truly fortunate to have met my supervisor, Prof. Hu, who opened the door to the world of research for me. I still recall my very first academic draft, which he patiently and professionally guides me from shaping the logical flow to perfecting the smallest details and instilling lasting habits of rigorous writing. I also remember how, when lights filled every home in celebration during the Lunar New Year, he steadfastly helped me revise my submission draft. And I recall how, with the first light of dawn every Wednesday morning, his thoughtful guidance arrived unfailingly.

The journey to the peak is never easy, yet along the way, I was fortunate to receive invaluable encouragement from Prof.Ye. My first year was exceptionally challenging, and at times the pressure seemed unbearable. Then, through an unexpected conversation, she rekindled my confidence and renewed my vision for the future. In moments of disappointment, anxiety, and confusion, she shone like a light, cutting through the darkness and guiding me forward.

As my thoughts take flight, I am reminded of all who have shared this journey with me. My classmates walked with me across the campus to the canteen, along the lively streets of Lan Kwai Fong, and up the winding paths of the MacLehose Trail. Their

constant companionship, like an ever-flowing river, has given me warmth and strength. I will always remember my coauthors, who meticulously annotated my drafts and patiently answered my questions. My roommate has accompanied me throughout my entire PhD journey, sharing with me the beauty of dance performances and the simple joys of life. Likewise, my friends have offered me endless support, remaining by my side whenever I felt downhearted.

Finally, I am deeply grateful to my parents, who have always given me both love and freedom. It is now time for me to pursue new summits.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Machine learning (ML) techniques have been widely adopted across diverse domains such as healthcare, finance, and autonomous systems [1, 2, 3], enabling highly accurate predictions and decision-making. Typical ML models are trained on large volumes of real-world data collected on a centralized server, a paradigm known as centralized learning. Alternatively, federated learning has emerged as a distributed approach that enables collaborative model training without requiring the sharing of raw data. This motivates the development of online platforms where ML models are shared [4] and traded [5, 6] for collaborative and economic purposes. However, as these models are often trained on sensitive data, a privacy concern arises: to what extent do such models expose information about their training data?

Recent studies demonstrate that ML models can inadvertently reveal information beyond their intended functionality, leading to the development of privacy attacks that aim to extract sensitive details from their training data. Among the most prominent are membership inference attacks (MIAs) [7, 8], which seek to determine whether a specific data record was included in the model's training set. For example, in a med-

ical diagnostic scenario, an adversary could query the model with a patient's health record to infer whether the patient's data was used during training, potentially exposing sensitive health conditions. Another significant threat is property inference attacks (PIAs) [9, 10], which aim to deduce global properties or statistical characteristics of the training dataset that are unrelated to the model's primary task. For instance, an attacker might infer the prevalence of a particular disease in a hospital dataset or the distribution of demographic attributes in a financial dataset based on ML models.

This thesis focuses on exploiting and quantifying these inference attacks in ML models. Given the pervasiveness and fundamental impact of MIAs and PIAs, we outline three key problems addressed in this work.

Research Problem 1: The first problem is how to improve the efficiency of existing inference attacks against ML models. Shadow model technique is commonly employed in various inference attacks, which enables an adversary to train multiple models using the same architecture as the target model on similar datasets. Recent works show it is very effective in MIAs [11, 12] and PIAs [13, 14]. However, the need for a large number of shadow models leads to high computational costs, limiting their practical applicability.

Research Problem 2: The second problem is how to identify new attack surfaces to guide the design of privacy-preserving methods for secure ML models. Recent studies indicate that privacy leakage can occur in vertical federated learning (VFL) [15], where multiple parties hold different features of the same samples. While existing attacks in VFL, such as label and feature inference [16, 17], primarily target record-level privacy risks, distribution-level privacy threats remain largely unexplored.

Research Problem 3: The third problem is how to quantify privacy leakage risks of ML models beyond specific attack algorithms. A unified risk measure is needed to evaluate privacy risk across multiple model candidates and to assist model owners in selecting privacy-preserving models. This scenario frequently arises in practice, as

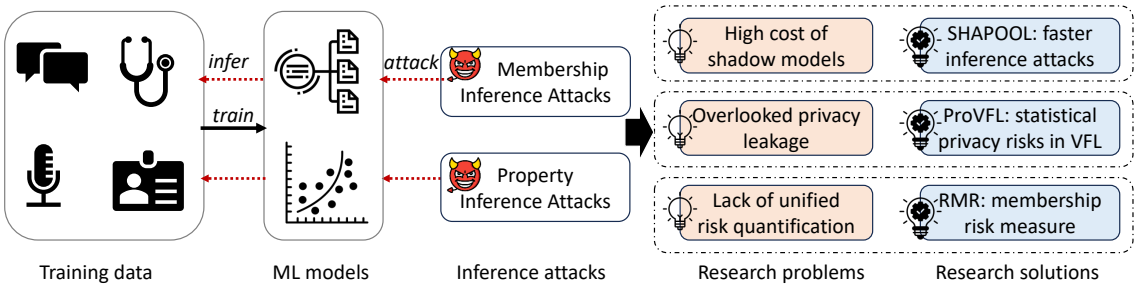


Figure 1.1: The research framework of this thesis.

organizations often develop or acquire multiple models for the same task, which may differ in architecture, training settings, or development approach. However, existing attack-agnostic and attack-specific methods exhibit certain limitations.

To address the research problems outlined above, we present the overall research framework, as illustrated in Figure 1.1.

First, to solve Problem 1, we tackle the fundamental bottleneck in existing inference attacks: the high computational cost of training shadow models, a widely used technique in MIAs and PIAs. Such inefficiency mainly stems from the independent training and use of these shadow models. To address this issue, we present a novel shadow pool training framework *SHAPOOL*, which constructs multiple shared models and trains them jointly within a single process. In particular, we leverage the Mixture-of-Expert (MoE) mechanism [18] as the shadow pool to interconnect individual models, enabling them to share some sub-networks and thereby improving efficiency.

Next, to solve Problem 2, we explore a new attack surface in VFL, in which an adversarial party attempts to infer global distribution information about a target property in the victim party's training set. We find that the L_p-norm distribution of intermediate results in VFL could reflect the fraction of the target property in a training set. Inspired by this, we present *ProVFL*, a novel PIA framework involving distribution comparison and correlation augmentation modules.

Finally, to solve Problem 3, we introduce a novel theoretical risk measure beyond empirical inference attacks and propose Relative Membership Risk (i.e., *RMR*) to assess

the MIA vulnerability of a model from a comparative standpoint. RMR calculates the difference in prediction loss for training examples relative to a predefined reference model, enabling risk comparison across models without needing to delve into details like training strategy, architecture, or data distribution.

1.2 Thesis Contributions and Impact

1.2.1 Thesis Contributions

In summary, this thesis makes the following contributions:

1. **Toward Efficient Inference Attacks via Shadow Model Sharing.** Toward existing inference attacks, we present a novel shadow pool training framework, named SHAPOOL, to enhance the efficiency of various inference attacks from the perspective of the shadow model construction. We utilize the MoE mechanism as a shadow pool to interconnect individual models, enabling them to share sub-networks and enhance the overall training efficiency.

2. **Property Inference Attacks against Vertical Federated Learning.** Toward new attack surface, we propose a novel PIA framework, named ProVFL, which uncovers statistical privacy leakage risks in VFL. To achieve property inference, we devise a distribution comparison module by creating various intermediate-result populations with different proportions. We validate the effectiveness of ProVFL on fourteen target properties in five real-world datasets.

3. **Relative Membership Risk Measure for Machine Learning Models.** Beyond empirical inference attack algorithms, we formally define a relative membership risk measure, named RMR, to evaluate the membership inference risk of a training example or an entire model. This measure is independent of specific training strategies or data distributions, relying solely on the example's

loss. We demonstrate that RMR is an accurate and efficient tool for measuring the membership privacy risk of both training examples and ML models.

1.2.2 Thesis Impact

This thesis advances the understanding of privacy vulnerabilities in machine learning by addressing efficiency, scope, and assessment of inference attacks. First, we take a new perspective on investigating the construction of shadow models, making the study of inference attacks more practical and reproducible. Second, we expand the scope of privacy research to VFL, uncovering systemic leakage risks in collaborative training scenarios that are highly relevant to real-world applications. Third, we provide an accurate and pratical risk tool to assess membership privacy risks, enabling standardized evaluation and facilitating the development of more robust defenses. Collectively, these contributions provide practical tools for assessing and mitigating privacy risks in current and future machine learning systems.

1.3 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 provides background on machine learning and associated privacy vulnerabilities. Chapter 3 presents a shadow model sharing algorithm designed to improve the efficiency of inference attacks. In Chapter 4, we explore a new attack surface in VFL settings. Moving beyond empirical inference attacks, we introduce a theoretical metric for assessing membership privacy risk at both the model and sample levels in Chapter 5. Finally, Chapter 6 concludes the thesis and outlines potential directions for future research.

Chapter 2

Background Review

In this chapter, we begin with preliminaries on centralized and federated learning, and then review privacy vulnerabilities in machine learning models.

2.1 Machine Learning

2.1.1 Centralized Learning

We consider centralized learning (CL) as the setting in which training data are pooled in the server to train an ML model [19, 20], which leverages input-output pairs to train a non-linear function that predicts outcomes for new queries. Let $D = \{(\mathbf{x}_i, y_i)\}_{i=0}^{N}$ denote the training datasets, respectively, where $\mathbf{x}_i \in \mathbb{R}^d$ denotes the d-dimension feature vector of the i-th sample labeled by y_i. The learning goal is to develop an ML model F that maps an input $\mathbf{x}$ into $F(\mathbf{x}; \theta)$, where θ is the model parameters. To obtain the parameters θ, a loss function $\ell(\cdot, \cdot)$ is typically introduced to measure the discrepancy between the predicted output $F(\mathbf{x}; \theta)$ and the ground truth label y.

To obtain a high-quality ML model, we consider the expected value of the loss on the training dataset. A common approach for model optimization is empirical risk

minimization [21], which minimizes the following objective function on D:

$$L(D, \theta) = \frac{1}{N} \sum_{i=0}^{N} \ell\left(F\left(\mathbf{x}_i; \theta\right), y_i\right) \tag{2.1}$$

Many training algorithms have been proposed to minimize this objective function [22, 23, 24], including stochastic gradient descent (SGD) [22], which updates the ML model in the t-th iteration as follows:

$$\theta^t = \theta^{t-1} - \eta \sum_{(\mathbf{x}_i, y_i) \in B} \nabla_\theta \ell\left(F\left(\mathbf{x}_i; \theta^{t-1}\right), y_i\right), \tag{2.2}$$

where B is a mini-batch of random training examples from D, θ^t represents the model parameters in the t-th round, and η denotes the learning rate. The optimization process is terminated when the model converges to a local minimum, or the iteration reaches a preset number. The trained ML model often uses the accuracy on test dataset to validate its performance.

2.1.2 Federated Learning

FL is a collaborative ML paradigm that is widely used in various applications, such as smart healthcare [25, 26] and the Internet of Things [27, 28]. In contrast to CL, FL involves training a shared global model using distributed data from different organizations or devices. Instead of transmitting the raw data to a central server, FL allows local data to remain stored locally and only the model parameters or gradients are exchanged, which addresses the data island problem and alleviates data privacy concerns.

The FL system involves two entities, clients and the central server. The clients (a.k.a., participants) possess training data and collaborate to train a shared model. According to the number of clients involved, there are two main FL types: cross-silo and cross-device FL [29]. Cross-silo FL involves a relatively small number of clients, typically organizations or data centers with a significant amount of data [30, 31],

while cross-device setting involves a large number of clients with small amounts of data, such as mobile devices [27, 28]. The central server is used to aggregate model updates from clients without accessing their local data. In the cross-silo setting, a client can be selected as the central server to perform aggregation and update the global model. In contrast, a powerful server is often introduced to manage model aggregation effectively in cross-device FL.

FL typically involves training a joint global model based on distributed local data through three phases: local training, communication, and aggregation phase. There are K clients that involve in FL and each client owns a local dataset D_c. We use θ_s and θ_c to denote the global and local model, respectively. To train a global model collaboratively, the central server first initializes the global model θ_s (e.g., model weights and hyperparameters) and broadcasts it to clients, and then the FL process carries out as follows:

1. Local training phase. Each selected client c receives the global model θ_s and training settings. Then, the client fine-tunes it on the local dataset D_c.

2. Communication phase. For each selected client, it uploads the latest version of local model θ_c to the central server, while the central server broadcasts the global model to selected clients after the aggregation phase.

3. Aggregation phase. The central server aggregates all the model updates from the selected clients and renews the global model θ_s.

Three phases are repeated until the loss value of the global model converges on a validation dataset, resulting in a well-trained global model.

Based on the distribution of data over the sample and feature spaces, we can broadly categorize FL into three types:

Horizontal Federated Learning (HFL) refers to scenarios where multiple parties possess data samples with the same features but unique sample IDs. For instance, different hospitals may possess patient records that share the same feature space but have different patient IDs. HFL is the most popular FL category in the literature

[32, 33], and local models commonly share the same architecture as the global model. As such, the global model can be simply aggregated from local models.

Vertical Federated Learning (VFL) refers to scenarios where multiple parties in an FL system hold data samples with identical sample IDs but different feature spaces. A typical case for VFL is when an E-commerce company and a local bank collaborate to train a personalized loan model based on online shopping records and credit situation [34]. Typically, only one participant can access the labels, and sample alignment is performed before training a joint model across multiple clients [35].

Federated Transfer Learning (FTL) pertains to the situation where clients' datasets contain varying ID spaces and feature spaces [36]. Inspired by transfer learning, FTL allows knowledge to be shared without compromising data privacy, enabling the transferability of complementary knowledge by exploiting source domain knowledge to train an FL model for the target domain [37, 38].

2.2 Privacy Vulnerabilities

ML models, while achieving remarkable performance, are increasingly shown to expose unintended privacy risks. These risks can be broadly grouped into two categories: model-centric and data-centric [35, 39]. As illustrated in Figure 2.1, model-centric risks primarily concern attacks that aim to replicate or steal the functionality of a model, such as model extraction attacks. Data-centric risks, on the other hand, directly target sensitive training data, including model inversion attacks, MIAs, and PIAs. In the following, we provide a review of these typical attacks.

Among model-centric risks, *model extraction attacks* aim to replicate the functionality of a target model by interacting with it through prediction APIs, thereby stealing its parameters, architecture, or even hyperparameters. Tramer et al. [40] demonstrated that adversaries can recover model parameters and architecture with only black-box query access to prediction APIs, raising concerns for machine learning as a service

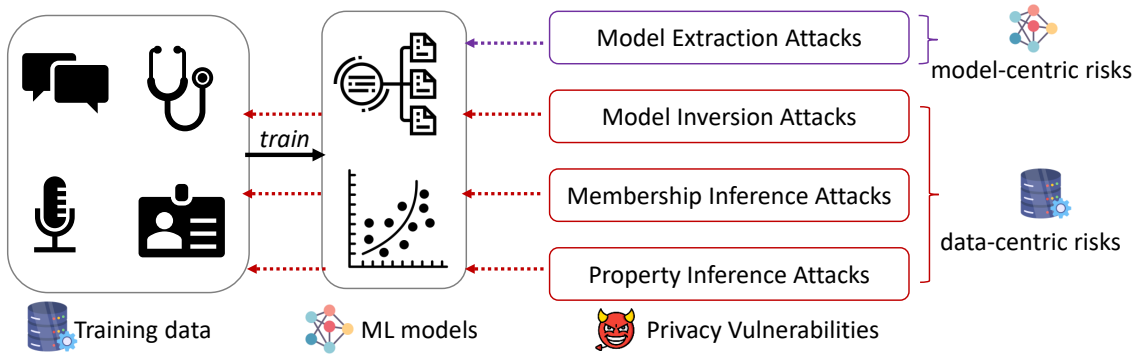


Figure 2.1: Possible privacy vulnerabilities in machine learning.

(MLaaS) platforms. Later work further showed that even hyperparameters, such as those governing regularization strength or kernel parameters, can be inferred through systematic probing of the model's outputs [41]. These attacks highlight a fundamental model-centric privacy risk: the unauthorized replication of proprietary ML models, which leads to intellectual property theft and potential misuse of stolen models.

On the data side, adversaries may exploit trained models to recover or infer sensitive information about the training data. *Model inversion attacks* aim to reconstruct input samples from a model's outputs or intermediate representations, thereby exposing sensitive information about the training data. For example, Zhang et al. [42] proposed a novel attack that uses generative adversarial networks (GANs) to reconstruct high-fidelity training samples from deep neural networks, aided by publicly available data. *MIAs*, in contrast, aim to determine whether individual records were part of the training set, without reconstructing the data itself. They infer membership by analyzing the model's outputs, such as confidence scores or prediction probabilities. Shokri et al.[7] first formalized the MIA framework, showing that even black-box access to a model can enable adversaries to infer membership using shadow models. Subsequent works, including Salem et al.[8] and Yeom et al. [43], further improved attack strategies and analyzed the factors affecting attack success, such as overfitting, model type, and training data distribution. Rather than sample-level privacy risk, *PIAs* aim to extract distributional information about a model's training dataset, such

as the proportion of samples with a particular attribute or demographic characteristic. Fredrikson et al.[44] first demonstrated that an adversary could infer whether a hospital dataset contains a certain disease by exploiting the model's predictions. Subsequent works, including Ganju et al.[10], extended these ideas to more general scenarios, showing that PIAs can compromise aggregated sensitive information without accessing individual records on fully connected neural networks.

In this thesis, we focus on attacks that aim to infer sensitive information that the model producer did not intend to share from training samples, specifically through MIAs and PIAs. These attacks are particularly concerning because they exploit the inherent memorization ability of ML models and create unintended pathways for private information leakage. Moreover, they are highly representative of data-centric privacy risks, encompassing both sample-level and distribution-level threats.

Chapter 3

Toward Efficient Inference Attacks via Shadow Model Sharing

3.1 Introduction

With the increase of ML models trained on sensitive personal data like facial images and medical records, various inference attacks pose severe threats to them, such as inferring the membership status of individual records (i.e., membership inference attacks) [7], reconstructing representative samples (i.e., model inversion attacks) [44], and uncovering statistical properties of the training dataset (i.e., property inference attacks) [9].

A common technique adopted in almost all such attacks is the shadow model technique, which enables an adversary to train multiple models using the same architecture as the target model on similar datasets. Recent works show it is very effective in membership inference [7, 11, 12, 45] and property inference attacks [13, 14, 46, 47]. Typically, the number of shadow models determines the quality of inference attacks. For example, to achieve high attack performance, this number is set to 256 and 100 in [11, 46], respectively. However, the high computational cost of training shadow

models has raised concerns [8, 14, 48]. Several works based on shadow models attempt to address this issue by optimizing this cost inside of their specific attack algorithms. For instance, QMIA [49] focuses solely on non-member data to develop a quantile regression model, thereby eliminating the need for shadow models from other distributions. RMIA [48] employs a pre-trained reference model on population data to save the training cost for shadow models in non-member cases. SNAP [14] reduces the number of shadow models required by injecting poisoned samples that contain the target properties of interest for property inference. However, since these methods are tightly coupled with specific algorithms, they lack generalizability across different inference attacks.

In this chapter, we address this problem from the perspective of shadow models themselves, that is, training each shadow model independently is not efficient [7, 13]. Inspired by this, we present *SHAPOOL*, a shadow pool training framework that constructs **shared shadow models** in a single training process. Specifically, we leverage the popular MoE mechanism [50, 51, 52] to build the pool with multiple identical sub-networks (i.e., experts) during training, and then generate shared models (i.e., activated pathways) as a substitute for conventional shadow models during inference. However, a vanilla MoE may produce over-specialized, unstable, and under-trained experts, which reduces the diversity of shared models, poorly resembles independent models, and ultimately diminishes attack effectiveness. To overcome these challenges, we design three key modules: pathway-choice routing, pathway regularization, and pathway alignment. The path-choice routing strategy assigns inputs to specific pathways to ensure a randomized yet fixed data allocation for pathway learning on top of stability and reliability. Pathway regularization aims to enhance the diversity of shared models by enforcing orthogonality in the representation space of experts and penalizing paired pathways with similar outputs. Lastly, pathway alignment is introduced to improve consistency with independent models and mitigate generalization mismatches.

We conduct extensive experiments to evaluate SHAPOOL across various existing membership inference attacks (MIAs) [11, 48], demonstrating that it outperforms the conventional shadow models in both attack performance and training efficiency. On the one hand, in a resource-limited scenario where only a limited number of shadow models can be trained [53], shared models can enhance the performance of existing attacks under a similar computational budget. On the other hand, in a resource-abundant scenario such as data auditing and risk assessment [54, 55], it achieves comparable performance while significantly reducing the computational cost of the conventional way.

Overall, in this chapter, our contributions are as follows: (1) We present a novel shadow pool training framework. To the best of our knowledge, this is the first work to enhance the efficiency of various inference attacks from the perspective of the shadow model construction. (2) We propose utilizing the MoE mechanism as a shadow pool to interconnect individual models, enabling them to share sub-networks and enhance the overall training efficiency. (3) To ensure that shared models serve as effective substitutes, we introduce three modules that guarantee specialized data allocation for pathway learning, promote diversity among shared models, and ensure consistency with independent models. (4) Extensive experiments demonstrate the effectiveness and efficiency of our proposed method compared to conventional shadow models.

3.2 Related Work

3.2.1 Shadow Model-based Inference Attacks

Shadow model technique is a widely-used approach for various inference attacks [7, 11, 12, 13, 14, 45, 46, 47], which aims to mimic the behavior of the target model by preparing shadow models with the same network architecture on similar datasets, which

provides the ground truth of inference results and outputs for the meta-classifiers used in inference attacks [7, 13, 14, 45, 56], as outlined in Algorithm 1. Typically, an adversary constructs n diverse shadow models $\{M_{S_1}, \ldots, M_{S_n}\}$ to simulate the behavior of the target model M_T. The i-th shadow model M_{S_i} is often trained using the same architecture as M_T on a similar auxiliary dataset D_{A_i}, and provides attack training data and corresponding ground-truth labels for the construction of the attack model $\mathcal{A}$. For example, in MIAs, shadow models generate outputs (e.g., logits or confidence scores) along with their corresponding membership labels (1 for member and 0 for non-member), which are then used to construct the attack training dataset for $\mathcal{A}$ (e.g., a binary classifier) to predict membership status.

Several popular inference attacks highlight the versatility of this technique. For instance, MIAs leverage shadow models to construct binary inference classifiers or hypothesis tests [7, 11]. Similarly, PIAs [13, 14, 57] use shadow models to build meta-classifiers that reveal sensitive property information in property inference attacks. Moreover, in advanced model inversion attacks [58], shadow model training facilitates the joint supervised training of the encoder and decoder with the collection of shadow-target model pairs. However, to ensure effective inference attacks, a large number of shadow models are typically required to capture the diversity and randomness inherent in the training data distribution [49]. This leads to considerable computational overhead, especially for large-scale models, thereby limiting the practicality of shadow model-based attacks.

Current research on shadow models mainly examines their training setup in relation to the target model, focusing on factors such as employing an identical network architecture [12, 59] and analyzing the effects of varying shadow training data distributions [8, 11, 12]. Departing from this focus, we aim to explore advanced methods for shadow model construction and enhance the attack efficiency of shadow model-based inference attacks.

Algorithm 1 Shadow Model-based Inference Attacks

Input: Training algorithm $\mathcal{T}$, Target model M_T, Auxiliary dataset D_A, Query examples Q, Attack algorithm $\mathcal{T}'$
Output: Inferred secret s'

// Shadow Model Construction
Construct shadow training subsets $D_{A_1}, D_{A_2}, \ldots, D_{A_M}$ by randomly sampling from D_A
for $i \leftarrow 1$ to M **do**
    Train a shadow model $M_{S_i} \leftarrow \mathcal{T}(D_{A_i})$
end for
// Attack Model Construction
$D_{att} \leftarrow \emptyset$
for $i \leftarrow 1$ to M **do**
    $(D_{A_i}, s_i) \leftarrow M_{S_i}(D_{A_i})$ ▷ s_i: ground-truth secret, e.g., membership status
    $D_{att} \leftarrow D_{att} \cup (D_{A_i}, s_i)$
end for
Train an attack model $\mathcal{A} \leftarrow \mathcal{T}'(D_{att})$
// Inference Attack
$s' \leftarrow \mathcal{A}(M_T(Q))$
return s'

3.2.2 Mixture-of-Expert Mechanism

As a special instance of conditional computation [51, 52], sparsely-gated MoE [50, 60, 61, 62, 63, 64, 65] contains a group of identical sub-networks (i.e., *experts*) and makes input-dependent predictions according to the choices of experts. It increases model parameters while maintaining a similar computational cost per example during inference, powering kinds of AI domains, including multi-task learning [50, 60], NLP [61, 62], and computer vision [63, 64, 65]. For simplicity, we refer to it as MoE moving forward.

A vanilla MoE model consists of L expert layers, each containing M experts with identical network structures and a router $\mathcal{R}$. Given an input x, the output of an expert layer is computed as the summation of the outputs from the top K experts selected by $\mathcal{R}$:

$$h = \sum_{k=1}^{K} \mathcal{R}(x) \cdot E_k(x),\ \mathcal{R}(x) = \mathrm{TopK}(\mathcal{G}(x), K), \tag{3.1}$$

where E is a learnable expert, $\mathcal{G}$ is a routing strategy based on trainable networks or heuristic functions, and $\text{TopK}(\cdot, K)$ refers to the largest K values. With $K = 1$ and non-expert layers omitted, we obtain an activated end-to-end network pathway, formally defined as a sequence of selected experts across layers, i.e., $\{E_i^1, E_j^2, ..., E_k^L\}$, where E_i^1 indicates that the i-th expert in the first layer is activated and $1 \leq i, j, k \leq M$.

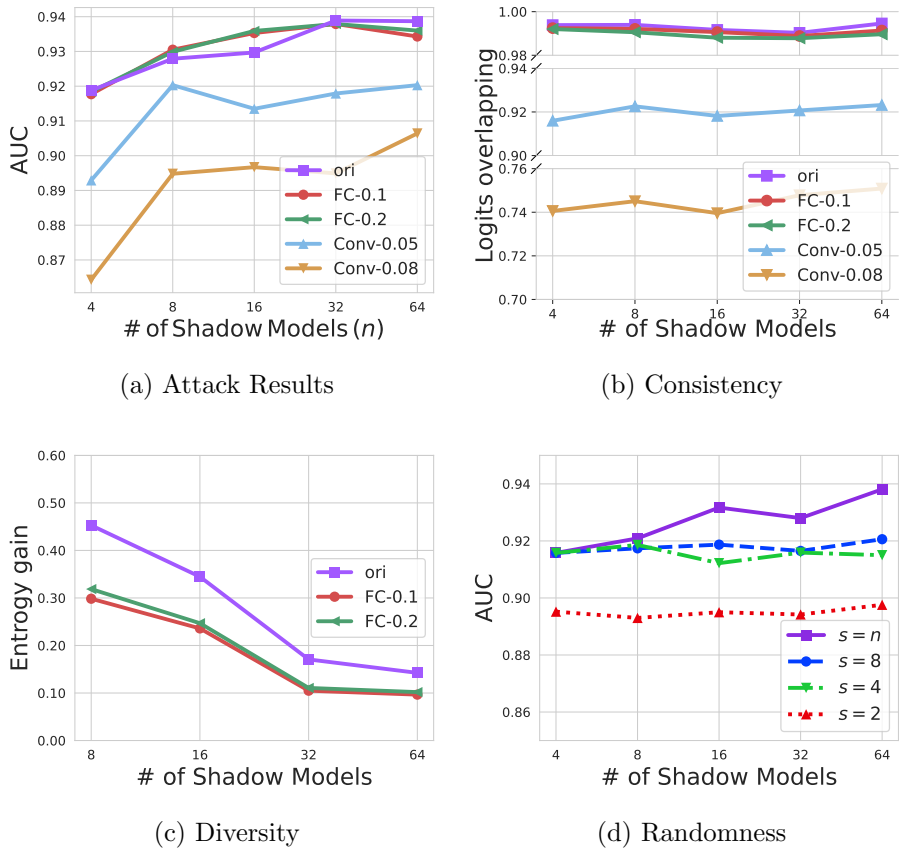
3.3 Analysis of Shadow Models

Ideally, an inference attack should be both effective (i.e., high inference accuracy) and efficient (i.e., computationally cheap). We investigate which aspects of shadow models influence attack effectiveness via a case study on MIA, providing guidance for optimizing their construction.

We first introduce a simple approach, model augmentation [66, 67, 68, 69, 70], to accelerate shadow model construction. For each trained shadow model, we generate an augmented version, doubling the total number of shadow models while reducing the overall construction cost. We target the last fully connected layers, closest to the model outputs and are commonly utilized in various inference attacks [11, 13, 14, 43, 71]. Conversely, pruning weights in the convolutional layers proves advantageous due to its impact on a larger number of weights, resulting in more diverse augmented models. For example, in ResNet18, the last fully connected layer accounts for only 0.46% of the parameters, while the convolutional layers comprise approximately 99%. Hence, we explore varying levels of weight modification by focusing on either the fully connected layers (i.e., *FC-p*) or the convolutional layers (i.e., *Conv-p*) where a larger probability p indicates more perturbation in the model weights.

We evaluate various augmentation settings on a MIA algorithm, LiRA [11], using the CIFAR100 dataset and ResNet18 model. To ensure that augmented models remain

close to the base architecture while preserving similar generative capacity, we adjust the scale of weight modifications such that the test error increase is limited to within 10%. Specifically, we set thresholds of 0.1 and 0.2 for *FC* layers and 0.05 and 0.08 for *Conv* layers. We first construct base shadow models in quantities of 4, 8, 16, 32, and 64, and then generate one augmented model for each base model. Consequently, the total number of shadow models doubles after augmentation for both *Conv-p* and *FC-p*. As shown in Figure 3.1, *ori* denotes attacks using conventional shadow models. However, unfortunately, those attacks with augmented models fall short of expectations in Figure 3.1a. Next, we explore why augmented models are not effective from three perspectives: consistency, diversity, and randomness.



(a) Attack Results (b) Consistency

(c) Diversity (d) Randomness

Figure 3.1: Attack performance (a) and factors of shadow models (b–d).

Factor ①: Consistency with the target model. Shadow models aim to replicate

the behavior of the target model and have similar generalization capabilities [43]. Therefore, we consider the consistency between shadow and target models by their output distributions. We fit training data logits on shadow models to a Gaussian distribution and measure their overlap with the target model using the Bhattacharyya coefficient. Figure 3.1b shows that such consistency is dependent on the specific construction method. Moreover, the level of consistency strongly influences attack performance: the lower the consistency, the weaker the attack performance, e.g., *Conv-0.05* and *Conv-0.08* cases.

Factor ②: Diversity of shadow models. We further examine how the diversity of shadow models influences the success of inference attacks on *FC-p*. To validate this, we measure the diversity of shadow models using their output entropy and report the entropy gain from adding more models (starting with four shadow models) in Figure 3.1c. We find that although more augmented models are used in *FC-p*, their diversity increases less than that of conventional shadow models. In other words, they offer limited additional information to the subsequent attack model, resulting in only marginal improvements in attack effectiveness.

Factor ③: Randomness of training subsets. Since augmented shadow models are statically constructed and share the same training subsets as the original models, we turn to the distribution of these subsets to explain attack effectiveness. For n shadow models, we randomly construct s distinct training subsets from the auxiliary data. When $n > s$, multiple models are trained on the same subset. Figure 3.1d shows that using totally unique training subsets achieves the best attack performance. Distinct training allocations enable shadow models to better capture randomness in data distribution, thereby enhancing inference attack effectiveness.

Summary: The above analysis suggests that effective shadow models should capture the randomness inherent in both the training process and the training data. The former necessitates consistency with the target model while ensuring diversity among shadow models, whereas the latter focuses on distinct training subsets. These

principles inspire us to design a new shadow model construction algorithm.

3.4 Motivation and Challenges

Our goal is to improve the efficiency of shadow model construction. As a fundamental component of inference attacks, our threat models are aligned with specific attack settings. Existing inference attacks [7, 11, 14] typically assume that the adversary knows (1) the training algorithm $\mathcal{T}$, e.g., the network architecture and loss function of the target model, enables the training of a similar model from scratch on datasets of the adversary's choice; (2) an auxiliary dataset from the same distribution, which may or may not overlap with the target dataset. Therefore, building on the above assumptions, we address the efficiency issue of shadow model construction by proposing a shadow pool training framework, SHAPOOL.

Shadow models account for a large proportion of the computational overhead in inference attacks, as each model is trained and operates independently, causing the cost to scale significantly with the number of shadow models. To address this issue, we construct shared models that are trained jointly within a single process. The advantage lies in the reuse of sub-networks across models, which significantly enhances construction efficiency.

Specifically, we utilize the MoE mechanism [50, 60, 61] to construct a shadow pool of shared models for inference attacks, as it offers two benefits as follows:
(1) *Well-suited structure*: An end-to-end activated pathway in MoE can be equivalent to a shadow model of an identical architecture, which allows us to create a large number of shared models as potential substitutes. Theoretically, an MoE model with L layers, each containing M experts, can form up to M^L unique pathways.
(2) *Training efficiency*: The sparse activation in MoE effectively scales model parameters without proportionally increasing computational demands [18]. This design

enables the reuse of trained experts across different activated pathways, thereby improving the overall training efficiency of the shared models.
However, it is non-trivial to adapt a vanilla MoE to construct the shadow pool due to the following challenges:

***Challenge 1*: Routing Specialization and Fluctuation.** Existing routing strategies assign each input to the most suitable experts during inference [61, 62], which creates an over-specialized mapping between inputs and pathways. Such specialization not only hinders the ability to capture diverse model behaviors on identical inputs [11], but also results in insufficient randomness of training data distribution across pathways, violating Factor ③. Moreover, the routing fluctuation effect in MoE [72], where the target expert for the same input can change during training, leads to unstable and overlapping usage of training data across pathways.

***Challenge 2*: Similar Pathways.** Although randomness in the training process, such as weight initialization, mini-batch SGD, and random routing [73], allows variations in expert performance, shared experts across multiple pathways may still lead to similar behaviors, violating Factor ②. In an extreme case, only a pair of experts differ between two highly overlapping pathways. Such similarity leads to redundant and under-informative shared models for subsequent inference attacks.

***Challenge 3*: Generalization Mismatch.** The end-to-end activated pathways exhibit varying degrees of generalization compared to models trained independently, thereby violating Factor ①, as shown in Appendix Figure 3.4a. For each pathway in MoE, the training data is typically much smaller compared to that of independently trained models, which leads to some experts being under-trained. On the other hand, the sharing mechanism affects the extent of overfitting in individual pathways.

3.5 Methodology

To overcome the above challenges, we propose *SHAPOOL*, an MoE-based shadow pool training framework, where activated pathways serve as effective substitutes for independently trained shadow models. As shown in Figure 3.2, it includes three modules, pathway-choice routing, pathway regularization, and pathway alignment, to maintain similar effectiveness and reduce the computational cost of the construction. For clarity, we use the terms 'activated pathway' and 'shared model' interchangeably throughout this chapter.

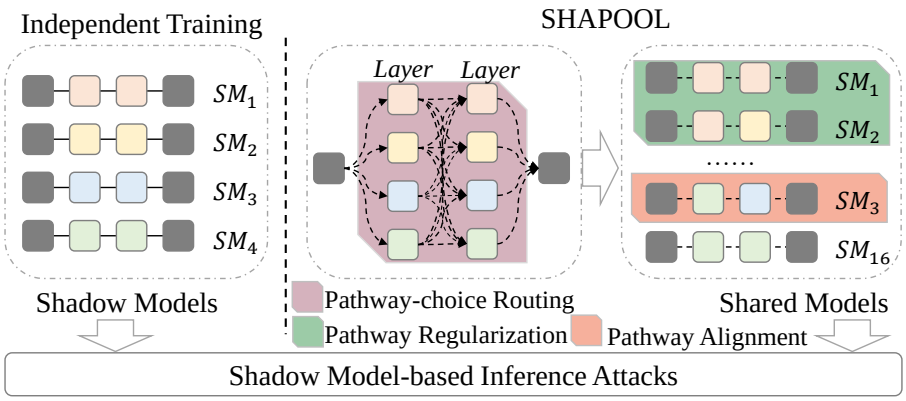


Figure 3.2: Illustration of SHAPOOL.

Module 1: Pathway-choice Routing. To deal with routing specialization and fluctuation, i.e., Challenge 1, we propose a pathway-choice routing strategy, which routes each input to fixed but randomly selected pathways and activates only one expert per layer during training and inference. This design enables us to maintain stable correlations between activated pathways and inputs, as well as to ensure a distinct data allocation for pathway learning.

We implement this pathway-choice routing strategy by establishing a hard mapping between pathways and disjoint training subsets. Given an MoE model with L layers and M experts per layer, we first enumerate all possible pathways as a set $\{\mathbf{P}_w\}_{w=1}^{M^L}$ where $\mathbf{P}_w \subset \mathbb{R}^L$ denotes the indices of activated experts in this pathway. Given a

training set D_{tr}, we define $\mathbf{B} \subset \mathbb{R}^{|D_{tr}| \times M^L}$ as a binary mapping matrix, where each element $\mathbf{B}_{i,w}$ indicates whether the w-th pathway is updated by the input x_i from the training set (1 for update, 0 otherwise). Formally, our pathway-choice router of the l-th layer for the w-th pathway is defined as

$$\mathcal{R}(x_i; w, l) = \mathbf{B}_{i,w} \cdot \mathbf{1}_{[\mathbf{P}_w[l]=k]}, \tag{3.2}$$

where $\mathbf{1}_{[\mathbf{P}_w[l]=k]}$ is an M-dimensional binary vector indicating which expert is activated at the layer for the w-th pathway. Then, the output of the l-th expert layer is formulated as:

$$h = \mathcal{R}(x_i; w, l) \cdot \mathbf{E}(x_i). \tag{3.3}$$

$\mathbf{E}(x_i)$ represents an output matrix of all experts, where each row refers to the output of a single expert for the input x_i. We establish a random yet stable assignment between experts and inputs by a mapping matrix $\mathbf{B}$. By controlling the overlap of data points across pathways, we ensure distinct training subsets for shared models and capture randomness in data distribution.

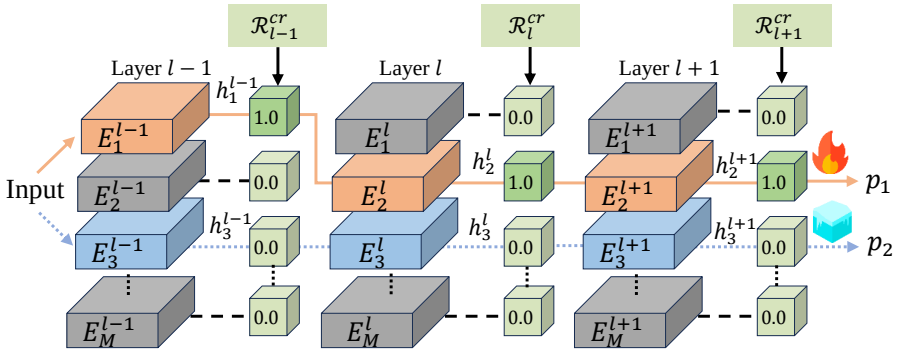


Figure 3.3: Illustration of the pathway regularization module.

Module 2: Pathway Regularization. SHAPOOL leverages pathway regularization to improve the diversity of shared shadow models to address Challenge 2. The core idea is to constrain the outputs of experts and produce diverse model predictions for the same input. We enforce the constraint on activated pathways by introducing

two loss terms: similarity regularizer and orthogonal regularizer. Inspired by contrastive learning [74, 75], SHAPOOL activates a pair of pathways in each iteration instead of a single pathway to enhance their differences. During training, we randomly select an additional pathway as a reference and update the model parameters of the activated pathway. Figure 3.3 illustrates one iteration of the training process for an input.

We first work on the issue of redundant model outputs from different pathways in SHAPOOL. Since model outputs are a key component in various inference attacks, particularly against black-box models [7, 13], we introduce a similarity regularizer to control the alignment of output probability among pathways. Let $p = f(x; P)$ denote the prediction probabilities of an MoE model f, where a set of experts along the pathway P are activated. As shown in Figure 3.3, given that two pathways are activated in one iteration, their prediction probabilities are represented as $p_1 = f(x; P_1)$ and $p_2 = f(x; P_2)$, where $P_1 = \{\dots, E_1^{l-1}, E_2^l, E_2^{l+1}, \dots\}$ and $P_2 = \{\dots, E_3^{l-1}, E_3^l, E_3^{l+1}, \dots\}$ respectively. We use the Kullback–Leibler (KL) divergence to measure the similarity of the output distributions between pairwise pathways [73]. The similarity regularizer $\mathcal{L}_{SR}$ is defined as the negative average of two KL divergence terms calculated from the prediction probabilities, as follows:

$$\mathcal{L}_{SR}(p_1; p_2) = -\frac{1}{2}(KL(p_1||p_2) + KL(p_2||p_1)). \tag{3.4}$$

We further introduce an orthogonal regularizer to eliminate feature correlation of experts within a single layer. This is important in extreme cases where a pair of activated pathways share all but one expert, leading to highly similar predictions for the same inputs. To address this issue, a strong constraint is imposed from the perspective of expert outputs to enhance their misalignment. In particular, we turn to orthogonal regularization [76, 77, 78] to improve the orthogonality effect in the representation spaces of experts. Rather than using common weight initialization methods, which may exhibit uncertain orthogonality in the convolutional layer [79]

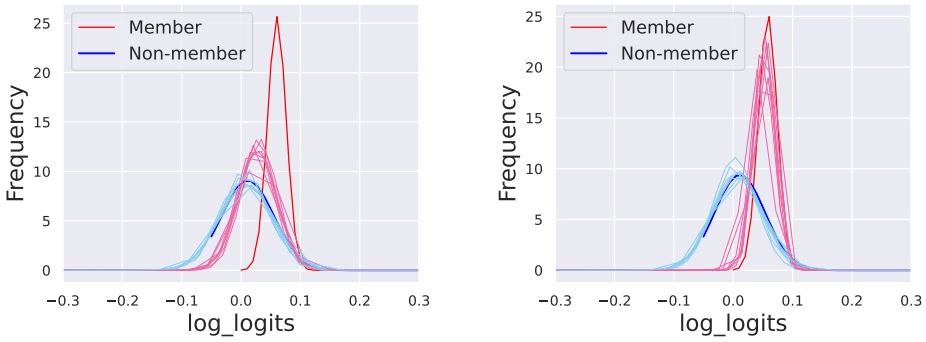
and impose overly stringent constraints [76], we apply the regularizer directly to the outputs of the experts and make their representation space as orthogonal as possible. Let H_1 and H_2 denote two sets of internal activations given a pair of pathways, respectively. We define the orthogonal regularizer $\mathcal{L}_{OR}$ as the sum of the pairwise inner products of the outputs from activated experts across all layers, formally expressed as:

$$\mathcal{L}_{OR}(H_1; H_2) = \sum_{l=1}^{L} \left\| h_i^l \cdot h_j^l(x) \right\|, \tag{3.5}$$

where h_i^l and h_j^l are the outputs of activated experts in the l-th layer for P_1 and P_2, respectively. Together with the commonly-used cross-entropy loss $\mathcal{L}_{CE}$ for model accuracy objective, the entire training objective of SHAPOOL with respect to a training sample (x, y) is to minimize:

$$\mathcal{L} = \mathcal{L}_{CE}(p_1; y) + \alpha \mathcal{L}_{SR}(p_1; p_2) + \beta \mathcal{L}_{OR}(H_1; H_2). \tag{3.6}$$

The hyperparameters α and β are introduced to balance model utility and the diversity strength of two regularization terms. The training objective in Eq.(3.6) forces all experts to minimize training accuracy errors while maximizing the diversity of their predictions.



(a) w/o PA (b) with PA (Epoch=5)

Figure 3.4: Distribution of shared models in a vanilla MoE and after alignment.

Module 3: Pathway Alignment. Challenge 3 points out that shared models in MoE tend to misalign with independent models in terms of generalization. Conse-

quently, this mismatch fails to mimic the behaviors of the target model and degrades the performance of inference attacks. To demonstrate this, we present the model output (i.e., scaled logits) distributions for both training (member) and test (non-member) data points using ResNet18 and CIFAR100 in Figure 3.4a. The red and blue curves represent the distributions of the target model, while the pink and sky-blue curves illustrate that of shared models from a shadow pool. Since the target model is trained independently, we align the outputs of shared models to better resemble it through pathway alignment. To address this issue, we aim to drive under-trained experts toward the performance of well-trained ones. The core idea is to enhance the memorization capacity of these activated pathways and align their behaviors with those of independent models across different data distributions through a key module in SHAPOOL termed pathway alignment.

We leverage the commonly used fine-tuning technique to achieve this alignment. A simple approach is to update specific layers of a pre-trained model while keeping the remaining weights frozen [80, 81]. Similarly, we update the model parameters along the activated pathway using a small dataset D_q randomly sampled from D_{tr} (around 10%). Specifically, after Modules 1 and 2, we randomly select n pathways from the pre-trained MoE model f and fine-tune them on D_q by minimizing $\mathcal{L}_{CE}$. We reuse a portion of samples from the overall training set D_{tr} to enrich the data available to each pathway, since each is initially trained on a considerably smaller subset compared to independent shadow models. These selected pathways are further trained on more examples, making their behavior more similar to that of a conventional shadow model and effectively reducing the mismatch with the target model, as shown in Figure 3.4b.

The Complete Algorithm: We present the pseudocode for SHAPOOL in Algorithm 2, a specialized MoE model designed for the shadow model construction problem, where activated pathways can replace conventional shadow models in various attacks. Prior to MoE training, we randomly assign training data to different pathways and record the mapping between each input and its corresponding activated

Algorithm 2 SHAPOOL: MoE-based Shadow Model Construction

Input: MoE model f, training dataset D_{tr}, fine-tuning dataset D_q, number of experts M, number of layers L, number of shared models n

Output: Trained MoE model f, data-to-pathway mapping matrix $\mathbf{B}$, selected pathway set S

// Before training: Pathway-choice routing
Randomly split D_{tr} into M^L subsets: D_{tr}^s
Enumerate all possible pathways $\{\mathbf{P}_w\}_{w=1}^{M^L}$
Construct mapping matrix $\mathbf{B}$ between D_{tr}^s and $\{\mathbf{P}_w\}_{w=1}^{M^L}$
// In training: Pathway regularization
for each $(x, y) \in D_{tr}$ **do**
    Randomly select two different pathways P_1, P_2
    Compute intermediate output h_1 using Eq. (3.3)
    Compute prediction $p_1 \leftarrow f(x; P_1)$
    Compute intermediate output h_2 using Eq. (3.3)
    Compute prediction $p_2 \leftarrow f(x; P_2)$
    Update model parameters of f using Eq. (3.6)
end for
// Fine tuning: Pathway alignment
Randomly select n pre-trained pathways S
for $i \leftarrow 1$ to $|S|$ **do**
    for each $(x, y) \in D_q$ **do**
        Compute prediction $p \leftarrow f(x; S_i)$
        Update model parameters of f using $\mathcal{L}_{CE}(p; y)$
    end for
end for
return f, $\mathbf{B}$, S

pathway. During training, we promote diversity and informativeness among shared models by regulating pairwise pathway similarity via Eq.(3.6). Finally, the pathway alignment module post-processes the trained pathways to align their behaviors with those of independently trained models across both training and test data distributions. Last but not least, similar to the shadow model technique, to learn more different aspects of both data distribution and the training process, we can develop multiple independent shadow pools, each providing shared models for inference attacks.

3.6 Empirical Evaluation

This section conducts a comprehensive evaluation of SHAPOOL, demonstrating its effectiveness and efficiency across existing inference attacks under two distinct scenarios:
(1) *Resource-Constrained Scenario*: There is a limit to training only a small number of shadow models, such as online inference attacks [53]. In this case, our goal is to enhance the effectiveness of existing attacks while consuming the same or similar amount of computational resources.
(2) *Resource-Unconstrained Scenario*: There are no explicit cost constraints, allowing for a large number of shadow models, such as offline data auditing or model vulnerability analysis [54, 55]. We aim to maintain comparable attack performance while significantly improving computational efficiency.

3.6.1 Experiment Setup

Datasets and Models. Our evaluation is conducted on three benchmark datasets: CIFAR100 [82], CIFAR10 [82] and CINIC10, and three typical network architectures: ResNet18 [83], VGG16 [84], and WideResNet28-10 [85]. For the configuration of SHAPOOL, we set the number of expert layers L to 4 (3 for WideResNet28-10), the number of experts M to 4, and the number of shared models n to 64.

Evaluation Protocol and Baseline. A shadow model-based inference attack pipeline generally consists of three stages: shadow model construction, attack model construction, and inference execution. We consider the conventional shadow model construction approach as the baseline (denoted by BASE), where each shadow model is independently trained on a randomly sampled subset of auxiliary data. Our proposed framework replaces the shadow model construction stage with shared models from the pool, while the other two stages remain unchanged.

Attack Setup. Following [7, 11, 48, 86, 87], we assume that an adversary knows the target model's network structure and possesses shadow datasets similar to the target data distribution. We evaluate the effectiveness of SHAPOOL by integrating it into two shadow model-based MIAs.

(1) *LiRA* [11] requires training multiple shadow models, where half are IN models trained on the query example, and the remaining half are OUT models that have not seen the example. It collects scaled logits from shadow models, fits them into two Gaussian distributions, and uses a likelihood-ratio test to determine the membership status of the query example. There are two attack modes: LiRA-online, which relies on two Gaussian distributions for members and non-members, and LiRA-offline, which relies solely on the Gaussian distribution of non-members.

(2) *RMIA* [48] introduces a fine-grained modeling of the null hypothesis (OUT models) within the likelihood ratio test framework. As with LiRA, it comes in two variants: RMIA-online and RMIA-offline. By introducing reference models and population data samples to reduce reliance on shadow models, both modes achieve comparable performance.

To extend our approach to LiRA and RMIA, we treat shared models from the fine-tuned pathways (i.e., after pathway alignment) trained on query examples as IN models, and those from the pre-trained pathways (i.e., before alignment) trained without query examples as OUT models. We report the attack performance on 1,000 query examples.

Evaluation Metrics. Following existing MIAs [11, 70, 88], we adopt two evaluation metrics as performance indicators, i.e., Area Under the ROC Curve (AUC) and True Positive Rate (TPR) at low False Positive Rate (FPR). We use TF1 and TF01 to denote TPR@FPR=1% and TPR@FPR = 0.1%, respectively. As for computational costs, we present the wall-clock time (in hours) required for shadow model construction, along with the corresponding percentage reduction in runtime. We report the average results over five independent runs with different random seeds.

3.6.2 Experimental Results

We evaluate the effectiveness of SHAPOOL in two different scenarios, followed by an investigation into how its several components contribute to the overall attack performance.

Inference Attacks under Low Computation Budget. We first analyze the performance of inference attacks under low computational resources, i.e., using a small number of shadow models. To achieve a similar cost, we use one shadow pool for SHAPOOL, while BASE uses four shadow models for LiRA-online and two for LiRA-offline. SHAPOOL removes pathway regularization and uses only a few fine-tuning epochs when conducting LiRA-offline.

Table 3.1 compares the attack performance of different construction methods. Overall, SHAPOOL demonstrates superior performance in most cases across various settings. For instance, SHAPOOL improves AUC by 5%–12% in LiRA-offline and by around 2% in LiRA-online, respectively. We also observe a relatively modest improvement on the WideResNet28-10 network compared to the other network types, attributed to its fewer expert layers and less diverse shared models. Besides, SHAPOOL shows a more significant improvement in LiRA-offline across evaluation metrics than in LiRA-online. This difference arises because the former attack is less sensitive to the output distribution of shadow models. These results indicate that SHAPOOL enhances inference attack performance by replacing independent shadow models with more shared models at a similar computational cost.

Ultimate Performance of Inference Attacks. We then evaluate the attack power when a large number of conventional shadow models can be trained. This scenario aims to validate SHAPOOL's training efficiency and ultimate attack performance. Following previous settings [11, 88], we prepare 128 shadow models for LiRA-online (half IN and half OUT) and 64 for LiRA-offline (all OUT). In this case, we construct four shadow pools, each randomly sampling 64 pathways as shared models. Table 3.2

Table 3.1: Attack performance and computational cost of LiRA under the constrained resource.

Dataset	Method	ResNet18			VGG16			WideResNet28-10		
		AUC	TF1	Cost	AUC	TF1	Cost	AUC	TF1	Cost
LiRA-offline attack										
CIFAR100	Base	0.73	0.22	0.5	0.69	0.16	0.4	0.76	0.30	2.4
	SHAPOOL	0.82	0.32	0.4	0.76	0.17	0.5	0.81	0.27	1.9
	Δ	+0.09	+0.10	-	+0.07	=	-	+0.05	-0.03	-
CIFAR10	Base	0.51	0.06	0.5	0.56	0.04	0.4	0.55	0.08	2.2
	SHAPOOL	0.63	0.09	0.5	0.64	0.09	0.4	0.60	0.08	1.6
	Δ	+0.12	+0.03	-	+0.08	+0.05	-	+0.05	=	-
LiRA-online attack										
CIFAR100	Base	0.90	0.34	1.0	0.86	0.25	0.9	0.90	0.36	3.6
	SHAPOOL	0.92	0.42	1.0	0.88	0.25	0.9	0.91	0.42	3.4
	Δ	+0.02	+0.08	-	+0.02	=	-	=	+0.06	-
CIFAR10	Base	0.69	0.11	1.0	0.67	0.07	0.9	0.67	0.09	4.4
	SHAPOOL	0.71	0.11	1.0	0.69	0.10	1.2	0.69	0.12	4.5
	Δ	+0.02	=	-	+0.02	+0.03	-	+0.02	+0.03	-

We use = to represent changes in the attack performance smaller than 0.02.

Table 3.2: Attack performance and computational cost of LiRA under the unconstrained resource.

Dataset	Method	ResNet18			VGG16			WideResNet28-10		
		AUC	TF1	Cost	AUC	TF1	Cost	AUC	TF1	Cost
LiRA-offline attack										
CIFAR100	Base	0.81	0.36	15.4	0.72	0.25	12.1	0.77	0.37	76.8
	SHAPOOL	0.84	0.37	1.6	0.78	0.21	2.0	0.82	0.36	7.7
	Δ	+0.03	=	↓ 90%	+0.06	-0.04	↓ 84%	+0.05	=	↓ 90%
CIFAR10	Base	0.57	0.11	16.8	0.54	0.06	13.7	0.56	0.09	70.8
	SHAPOOL	0.63	0.13	1.9	0.65	0.10	1.6	0.61	0.09	6.4
	Δ	+0.06	+0.02	↓ 89%	+0.11	+0.03	↓ 88%	+0.05	=	↓ 91%
LiRA-online attack										
CIFAR100	Base	0.95	0.55	30.7	0.90	0.36	24.3	0.93	0.50	153.6
	SHAPOOL	0.94	0.53	3.9	0.89	0.31	3.1	0.92	0.48	14.5
	Δ	=	-0.02	↓ 87%	=	-0.05	↓ 88%	=	-0.02	↓ 91%
CIFAR10	Base	0.71	0.16	33.7	0.71	0.14	27.5	0.71	0.13	141.6
	SHAPOOL	0.70	0.14	4.0	0.69	0.13	4.8	0.70	0.13	18.1
	Δ	=	-0.02	↓ 88%	-0.02	=	↓ 82%	=	=	↓ 87%

presents the attack performance and computational cost across different settings. First, for the LiRA-offline attack, our proposed method outperforms the baseline as

Table 3.3: Attack performance and computational cost of RMIA.

Method	RMIA-offline				RMIA-online			
	AUC	TF1	TF01	Cost	AUC	TF1	TF01	Cost
BASE	0.94	0.53	0.44	7.5	0.94	0.53	0.40	16.0
SHAPOOL	0.93	0.52	0.42	1.8	0.93	0.53	0.39	4.0
Δ	=	=	=	↓ 76%	=	=	-0.02	↓ 75%

the size of the shadow pools increases, mostly achieving higher attack performance while significantly reducing the computational cost of conventional shadow model training by an average of 88.6%. Regarding LiRA-online MIA, SHAPOOL achieves comparable performance in terms of AUC, with a slight reduction in TPR@FPR=1%. This is attributed to the high sensitivity and limited robustness of the online mode to the behavior of shadow models on the training data distribution [11]. Overall, our SHAPOOL framework improves the efficiency of shadow model training by approximately 7.7× compared to BASE on LiRA.

Extension to Other Membership Inference Attacks. To further validate the board applicability of SHAPOOL across different attacks, we report experimental results on RMIA using ResNet18 and CIFAR100, evaluating AUC, TPR@FPR=1%, and TPR@FPR=0.1% under the unconstrained setting. Specifically, we compare the performance of the original RMIA using 64 independent shadow models with its enhanced version that utilizes shared models from four shadow pools. Table 3.3 demonstrates that unlike previous methods tailored to specific attack strategies, our approach generalizes across different inference attacks with improved efficiency and adaptability.

3.6.3 Ablation Study

We investigate how various components in SHAPOOL collectively achieve comparable attack performance using CIFAR100 and ResNet18.

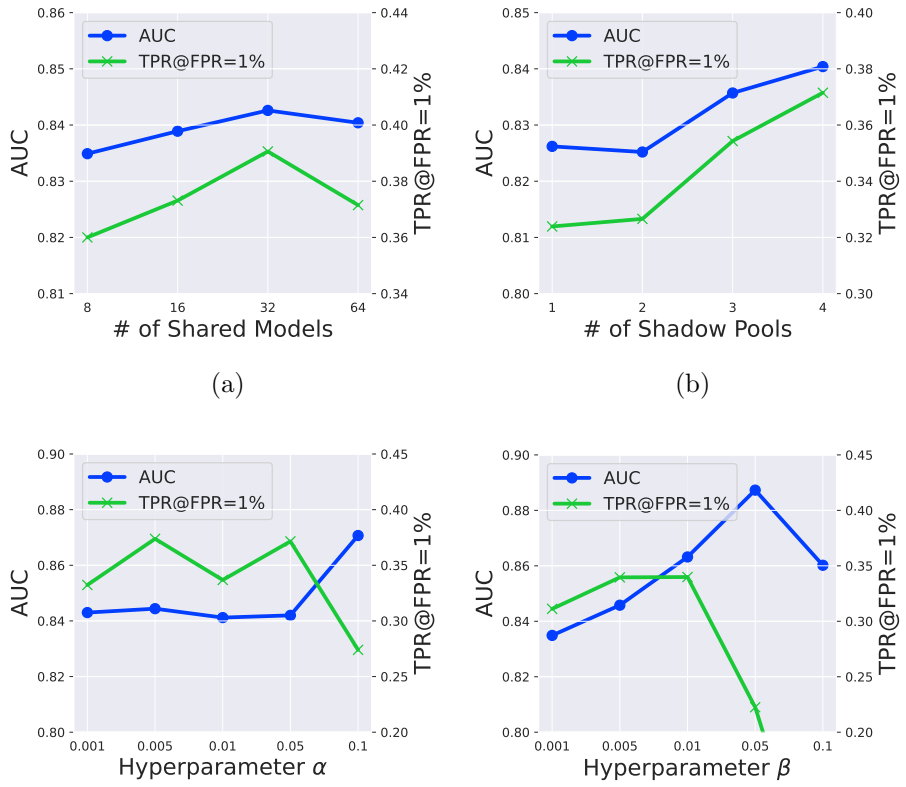


Figure 3.5: Impact of various hyperparameter settings on LiRA-offline.

Number of Shared Models. We evaluate the impact of the number of shared models sampled from the shadow pool on attack performance. We test LiRA-offline attacks using four different numbers of shadow models: 8, 16, 32, and 64, using CIFAR100 and ResNet18. Figure 3.5a illustrates a slight performance improvement in both AUC and TPR@FPR=1% as the number of shared models increases; however, this effect diminishes when the number exceeds 32, indicating that while a shadow pool can provide numerous potential shared models, their effectiveness is ultimately limited.

Number of Shadow Pools. To approximate the ultimate attack performance, we build multiple shadow pools for shared models in the above experiments. Here, we examine the impact of varying the number of shadow pools on attack performance, ranging from 1 to 4. Figure 3.5b shows that increasing the number of

Table 3.4: Impact of three loss terms.

$\mathcal{L}_{CE}$	$\mathcal{L}_{SR}$	$\mathcal{L}_{OR}$	AUC	TF1
✓			0.83	0.35
✓	✓		0.84	0.36
✓		✓	0.87	0.39
✓	✓	✓	0.87	0.42

Table 3.5: Effect of the fine-tuning set $|D_q|$.

$\lvert D_q \rvert$	LiRA -online	LiRA -offline
1000 (2.5%)	0.28	0.29
3000 (7.5%)	0.41	0.31
5000 (12.5%)	0.40	0.29

shadow pools improves both evaluation metrics, especially a more noticeable effect on TPR@FPR=1%. This upward trend suggests that SHAPOOL has the potential to further enhance attack performance beyond the results in Tables 3.2 and 3.3.

Training Objective. Path regularization employs a specialized training objective to enhance the diversity of shared models. Here, we examine the relative contributions of its loss terms: $\mathcal{L}_{CE}$, $\mathcal{L}_{SR}$, and $\mathcal{L}_{OR}$. The results in Table 3.4 indicate that $\mathcal{L}_{OR}$ plays a crucial role in attack performance, whereas adding $\mathcal{L}_{SR}$ results in only a slight improvement in AUC and TPR@FPR=1%, as $\mathcal{L}_{OR}$ imposes a more direct restriction on the experts.

Regularization Strength. We further investigate the impact of the regularization strengths α and β. To control the experiment, we set the other hyperparameter to zero when testing either one. Regarding the hyperparameter α, Figure 3.5c demonstrates that the attack performance remains stable as long as α is not very large (e.g., $\alpha \leq 0.05$). And β has a similar effect on attack performance in Figure 3.5d. As such, in our experiments, we set it to a value smaller than 0.01. The two regularizers positively influence attack performance when setting their contribution to an appropriate scale.

Table 3.6: Effect of the training set $|D_{tr}|$ on TF1.

$\lvert D_{tr} \rvert / \lvert D_{A_0} \rvert$	LiRA-online	LiRA-offline
1.0	0.33	0.22
1.2	0.37	0.28
1.4	0.38	0.30
1.6	0.36	0.32

Table 3.7: Effect of the number of experts.

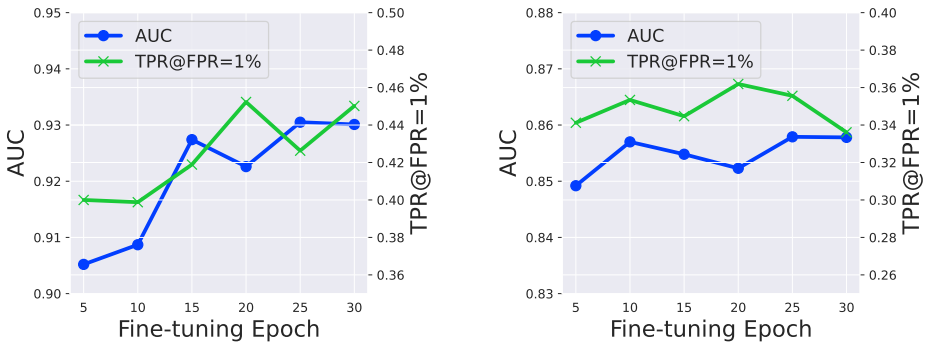
M	LiRA-online	LiRA-offline
3	0.39	0.35
4	0.42	0.33
5	0.44	0.32
6	0.47	0.28

Size of Training Set $|D_{tr}|$. SHAPOOL constructs a shadow pool with numerous shared models, resulting in more parameters compared to a single shadow model, which naturally demands more training data. We now examine the impact of training set size on attack performance, relative to the size of the training subset used for conventional shadow model training. Table 3.6 presents the results for different scales of training datasets. $|D_{A_0}|$ represents the size of the training subset for a shadow model trained independently. Both $|D_{tr}|$ and $|D_{A_0}|$ are sampled from the auxiliary dataset, but we use different notations to distinguish the training set of the shadow pool from that of an individual shadow model. We observe that the performance of SHAPOOL improves significantly after incorporating an additional 20% of data points, and then stabilizes with further increases.

Number of Experts. The above experiments use four experts in SHAPOOL for ResNet18 by default; now, we explore whether other settings influence the attack performance. As shown in Table 3.7, we find that introducing more experts improves attack performance, especially for the LiRA-online attack, as more experts enhance the diversity among shared models.

Size of Fine-tuning Set $|D_q|$. Pathway alignment fine-tunes the selected pathways to reduce generalization mismatch on the training set D_q. Here, we examine the effect of the fine-tuning training set size, as shown in Table 3.5. The percentage represents the ratio of D_q to a training set D_{tr}. Notably, there is an overlap between D_q and D_{tr}, indicating that the training set can be reused without introducing new data. We find that even a small fine-tuning dataset can achieve stable attack performance.

Fine-tuning Epochs. We explore the effect of the fine-tuning epochs in pathway alignment, as shown in Figure 3.6. The results show that the fine-tuning epoch has a limited effect on the LiRA-offline attack, while a larger epoch (e.g., greater than 15) leads to nearly stable attack performance for both attacks. This difference stems from specific algorithms: LiRA-offline relies on non-member output distributions, which are initially similar in Figure 3.4. This demonstrates that our pathway alignment



(a) LiRA-online (b) LiRA-offline

Figure 3.6: Effect of Fine-tuning Epoch.

requires minimal fine-tuning costs to mitigate generalization mismatch.

3.7 Chapter Summary

Targeting the efficiency bottleneck of shadow model construction, a core component of inference attacks, we propose an attack-agnostic solution to address this issue. Instead of independently training shadow models in a conventional manner, we construct a large number of shared models and train them jointly within a single process using the MoE mechanism. While this reuse of sub-networks improves the construction efficiency, dependent shadow models may negatively impact the attack performance. Our approach replaces conventional shadow models with shared models, significantly reducing construction costs while preserving competitive attack performance across various datasets. Moreover, it can be seamlessly integrated into a wide range of inference attacks. Future work can explore refined parameter-sharing strategies to further balance attack efficiency and effectiveness. Additionally, adaptive shadow model training mechanisms remain an interesting direction.

Chapter 4

Property Inference Attacks against Vertical Federated Learning

4.1 Introduction

With consideration for data privacy and security, FL [32] is proposed to train a high-quality model collaboratively across parties without centralizing their data. FL can be categorized into two types based on how data is partitioned within a feature and sample space: HFL and VFL. In contrast to HFL, where parties typically possess data with identical feature spaces, VFL operates on decentralized data sharing the same sample IDs while having different features [16, 89]. Application cases of VFL appear in finance, e-commerce, and healthcare [90, 91, 92]. As an example in Figure 4.1, for an e-commerce platform and bank institution, VFL offers a solution to merge distinct and distributed training samples, enabling them to develop a powerful loan prediction model without accessing original private records.

Typically, a VFL system has two types of parties: passive party that solely holds sample features, and active party that supplies both sample labels and features. Each party constructs a bottom model based on its own private dataset, uploads interme-

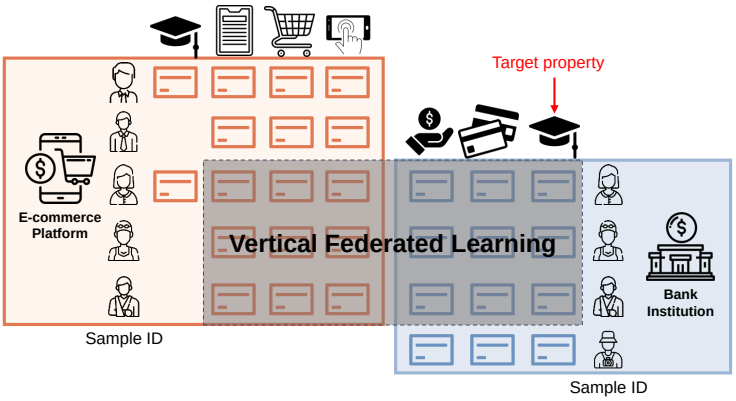


Figure 4.1: An example of PIA in a two-party VFL setting. Consider a VFL system where an e-commerce platform and a bank institution collaboratively develop a loan prediction model using their overlapping samples. The e-commerce platform infers the statistical distribution of educational degrees for common users and adjusts advertising policies for this population.

diate outputs to a top model controlled by an active party, and subsequently updates the bottom model based on intermediate gradients from the top model [93]. While VFL keeps each party's private data locally, it is susceptible to various privacy attacks. Recent studies have thoroughly investigated these threats, including feature inference attacks [17, 89, 94, 95, 96] and label inference attacks [16, 97, 98, 99]. The former entails adversaries reconstructing private data from model updates exchanged across parties, whereas the latter reveals sensitive label information of training samples [16]. Both attacks target the *record-level* privacy leakage of VFL as they operate on individual training samples.

Different from previous privacy studies, this chapter pioneers the investigation of PIAs against VFL, which target *distribution-level* privacy leakage. In PIAs, an attacker can infer the characteristics and statistical information about a specific property of interest (i.e., target property) in a training set. Such distribution-level information leakage may cause privacy breach [9, 13] or intellectual property infringement [13, 14], as an example provided in Figure 4.1. Understanding and exploring such privacy leakage helps shed light on privacy protection in VFL.

Unfortunately, existing works [100, 101] are not directly applicable to the VFL setting due to the following fundamental differences. At first, existing PIAs, designed for centralized learning or HFL, are impractical for VFL as they often require the training of shadow models on the full feature space [9, 13, 14, 100, 102]. However, an adversarial party in VFL has access to partial feature space, which makes it unable to replicate the complete training procedure. Besides, the heterogeneous feature spaces within VFL provide limited information for an adversarial party [101]. This restricts the effectiveness of PIAs based on attribute inference attacks (AIAs) [103, 104], which heavily relies on a strong correlation between the target property and the task label, as well as on the availability of adversarial knowledge [100]. These limitations motivate our contributions towards a specific approach and a thorough analysis for VFL settings.

To bridge this gap, we deeply investigate property information leakage within a partial feature space controlled by a malicious party in VFL. The essence of property inference is that different fractions of the training set on a target property may lead to distinction on model outputs or parameters [9, 13, 14]. Such a distinction can also be reflected in VFL intermediate results (i.e., outputs and gradients). This leads us to discover that the scale of divergences in the L_p-norm distribution of intermediate results is aligned with the fractional gap between two different populations. Building on this, we propose a novel PIA framework for VFL, named ProVFL, which infers the fraction of a target property based on the observed L_p-norm distributions of intermediate results. ProVFL comprises a distribution comparison module and a correlation augmentation module. In the first module, we randomly sample various populations with different fractions and learn the correspondence between L_p-norm distributions of intermediate results. To further enhance the attack effectiveness, we conduct a theoretical analysis of the factors influencing its performance and incorporate the correlation augmentation module. The second module incorporates two methods: (1) a label replacement technique that discreetly substitutes the original labels with the

target property, embedding more relevant information into the VFL system, and (2) a model refinement strategy that enhances the attacker's bottom model by utilizing supervised knowledge of the target properties and amplifying distribution disparities between populations. Experiment results demonstrate that both of them are effective in decreasing estimation errors of property inference.

The main contributions of this chapter are summarized as follows: (1) We propose a novel PIA framework for VFL, named ProVFL. To the best of our knowledge, this is the first work to investigate PIAs within VFL while achieving low estimation errors. (2) To achieve property inference, we devise a distribution comparison module by creating various intermediate-result populations with different proportions. This module can learn the relationship between L_p-norm distributions and their fractions. (3) Motivated by our theoretical analysis, we develop label replacement and model refinement approaches in the correlation augmentation module to enhance the effectiveness of PIAs. These approaches can amplify property information leakage while improving attack performance. (4) We validate the effectiveness of ProVFL on fourteen target properties in five real-world datasets. Also, several defenses are explored to safeguard against such leakage risk.

4.2 Related Work

4.2.1 Property Inference Attacks

Unlike privacy breaches targeting specific training samples [105, 106] or model parameters [107, 108], PIAs aim to infer statistical information about a training set, such as the proportion of samples related to a specific target property in the dataset. By constructing a batch of shadow models and training a meta-classifier, various PIAs have been introduced across different architectures, including neural networks [10, 102], generative adversarial networks [109], graph neural networks [100], and diffusion mod-

els [110, 111]. Their core idea is that similar training datasets result in comparable model performance, either in terms of model predictions [9, 102] or model parameters [10]. Another method based on AIAs recovers the attribute values of training samples and subsequently summarizes them [109, 111]. This approach relies on the correlation between inferred attributes and labels, as well as the adversarial knowledge available [7, 100].

PIAs have also been extended to collaborative learning contexts [57, 102, 103, 112]. Melis et al. [103] utilize snapshots of the global model to construct a batch property classifier in HFL. The classifier can infer properties related to a subset of the training inputs or detect when a specific property is present in the training data. Like shadow-training methods, [102] examines the disclosure of properties in a multi-party environment through model predictions. These studies focus on PIAs in HFL, while the risk of property leakage in VFL has received little attention. Our work bridges this gap by revealing that an adversarial party can make highly accurate inferences about target properties, highlighting the immediate threat of property information leakage in VFL.

4.2.2 Defenses against Property Inference Attacks

Intuitive defensive techniques, such as adding noise or removing sensitive properties, are not effective at protecting against property leakage. Previous research suggests that differential privacy [113], which is designed for individual privacy protection, offers limited mitigation [9, 13, 102]. Due to inherent correlations across features, removing sensitive attributes cannot solve property leakage inherently [102]. As for FL, another strategy is to reduce the amount of information exchanged between parties, such as by sharing fewer gradients [103, 114] and adding noise to model updates [10, 57, 115]. However, these techniques fail to provide adequate protection while maintaining acceptable model utility. Hence, there is a lack of rigorous and

powerful defenses against distribution-level information leakage [14, 47].

4.3 Threat Models

4.3.1 Record-level Intermediate Results in VFL

Following [93, 116], we consider a typical VFL setting in which K parties $\{P_1, P_2, ..., P_K\}$ use N overlapping data samples $D_{tr} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ to train a machine learning model. Each sample $\mathbf{x}_i$ is constructed by joining the distributed features $\mathbf{x}_i^k \in D_k$ across parties and the label $y_i \in \mathcal{Y}$ is provided by the active party. Upon preparing the training set, P_k utilizes its local dataset to train a bottom model f_{θ_k} parameterized with θ_k. Let $\mathbf{v}_i^k$ denote its intermediate output, i.e., $\mathbf{v}_i^k = f_{\theta_k}(\mathbf{x}_i^k)$. A top model h_ϕ aggregates all intermediate outputs in a certain way, e.g., by concatenation, where ϕ denotes the top model parameter. Let $\mathbf{v}_i = [\mathbf{v}_i^1, \mathbf{v}_i^2, \ldots, \mathbf{v}_i^K]$ denote the concatenated vector of $\mathbf{x}_i$. Subsequently, h_ϕ makes the final prediction as $h_\phi(\mathbf{v}_i)$ and computes corresponding gradients by minimizing the following objective function:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \ell \left(h_\phi \left(f_{\theta_1} \left(\mathbf{x}_i^1 \right), ..., f_{\theta_K} \left(\mathbf{x}_i^K \right) \right), y_i \right), \tag{4.1}$$

where ℓ is a loss function, e.g., the cross-entropy function. The active party calculates the loss using Eq.(4.1) and updates the top model using $\nabla_\phi \mathcal{L}$. Next, the active party computes the partial gradients for each bottom model and sends them back. Finally, each party updates its respective bottom model accordingly.

Unlike aggregated model updates in HFL, a VFL adversary could acquire fine-grained exchanged information, including record-level intermediate output and gradient. Intuitively, a party P_k can access the intermediate output of a training sample $\mathbf{x}_i^k$ with its bottom model, i.e., $\mathbf{v}_i^k$. Moreover, common sample IDs in VFL expose the intermediate gradients of individual data points to the active party who can obtain record-level intermediate gradients by replaying the optimization process, i.e.,

$\mathbf{g}_i^k = \nabla_{\mathbf{v}_i^k} \ell(h_\phi(\mathbf{v}_i), y_i)$. However, passive parties only observe aggregated gradients of multiple samples [89] due to lacking control over the top model or access to label information.

Before introducing possible adversarial knowledge, we define AO and AG as *record-level* intermediate outputs and gradients derived from the adversarial party's data points, respectively. Similarly, VO and VG refer to *record-level* intermediate outputs and gradients associated with the victim party's data points.

4.3.2 Adversary's Objective

We investigate PIAs in the context of VFL, where an adversarial party P_k infers statistical information about a sensitive attribute owned by a victim party. Formally, given a target property p, a training bottom model f_{θ_k} and external adversarial knowledge Ω, a PIA model can be defined as follows:

$$\mathcal{A} : \{p,\ f_{\theta_k},\ \Omega\} \rightarrow t, \tag{4.2}$$

where t is the predicted fraction of p in the victim party's training dataset.

4.3.3 Adversary's Capacity

We assume that an attacker can gather some auxiliary data, with each sample labeled as either having or not having the target property. This is an assumption commonly made by other PIAs [10, 13, 14], where a key difference is that we consider those data from the adversarial party's feature space rather than the victim party's, as accessing data from other parties is typically challenging in VFL. To achieve this, an attacker might inquire about inferred properties from a subset of their users or adopt other attacks, e.g., feature inference attacks [89, 117, 118] to reconstruct some noisy samples from the victim party's side in advance.

With auxiliary data, an adversarial party can receive different amounts of information, including knowledge of intermediate results and the top model, to cause property leakage. Considering the adversary's role in VFL, we examine its adversarial knowledge Ω under two cases:

(C1) Active party as the PIA adversary: Given that the active party provides label information and typically possesses more computational resources, it often undertakes the training of the top model [16]. Hence, in this scenario, the adversarial knowledge encompasses four types of intermediate results and a controllable top model, i.e., $\Omega = \{\text{AO}, \text{AG}, \text{VO}, \text{VG}, h_\phi\}$.

(C2) Passive party as the PIA adversary: Compared to the active party, a passive party has access to less adversarial knowledge because its bottom model is the only controllable component. Consequently, in this case, we explore property leakage of VFL when an attacker relies solely on its record-level intermediate outputs during the training process, i.e., $\Omega = \{\text{AO}\}$.

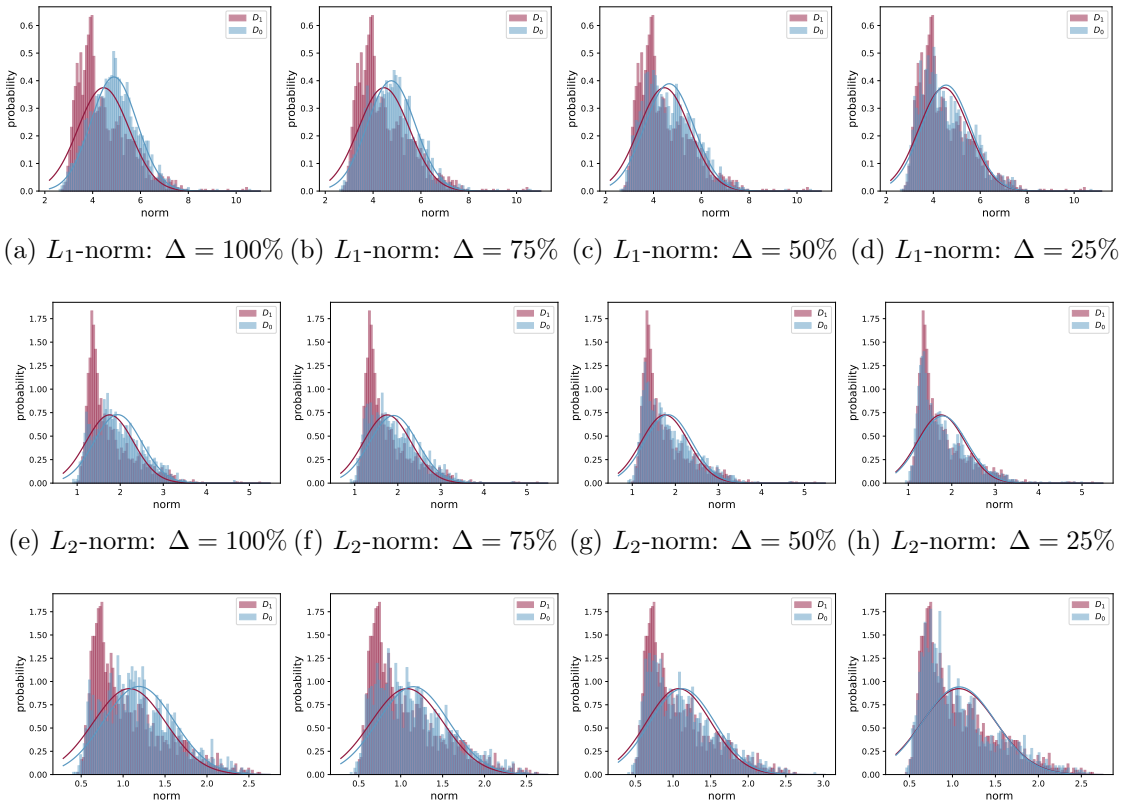
4.4 Key Observation

In this section, we first provide insights into what reveals property information to attackers in VFL, which motivates us to design a specific PIA for VFL.

Table 4.1: The distance comparison of different norm options.

Δ	Norm Option	KL $\uparrow$	JS $\uparrow$	BC $\downarrow$
25%	L_1-norm	0.0017	0.0202	0.9684
	L_2-norm	0.0000	0.0012	0.9990
	L_∞-norm	0.0004	0.0095	1.0000
100%	L_1-norm	0.0327	0.0876	0.8692
	L_2-norm	0.0086	0.0458	0.9665
	L_∞-norm	0.0145	0.0592	0.9856

Δ means the fraction gap between the two distributions about the target properties.



(a) L_1-norm: $\Delta = 100\%$ (b) L_1-norm: $\Delta = 75\%$ (c) L_1-norm: $\Delta = 50\%$ (d) L_1-norm: $\Delta = 25\%$

(e) L_2-norm: $\Delta = 100\%$ (f) L_2-norm: $\Delta = 75\%$ (g) L_2-norm: $\Delta = 50\%$ (h) L_2-norm: $\Delta = 25\%$

(i) L_∞-norm: $\Delta = 100\%$(j) L_∞-norm: $\Delta = 75\%$(k) L_∞-norm: $\Delta = 50\%$(l) L_∞-norm: $\Delta = 25\%$

Figure 4.2: Distributions of VO between different training datasets.

The norms of latent outputs and gradients of a neural network hold a substantial amount of semantic information [119, 120, 121], which motivates many research areas, including person recognition [122, 123, 124] and model explanations [125, 126, 127]. We expect that such norms encompass side information about target properties and project intermediate results of training samples closer to those with similar properties.

To validate this conjecture, we present a case study about property information leakage of VFL in a real-world dataset. At first, we classify training samples with the target property as *property samples* and those without as *non-property samples*. We construct two kinds of datasets, D_0 and D_1, each comprising 2,000 sampled data points. Four different D_0 contain {0%, 25%, 50%, 75%} of property samples and the remaining are non-property samples, respectively, while D_1 only contains property

samples. Hence the fraction gaps between D_0 and D_1 on the target property are {100%, 75%, 50%, 25%}. Subsequently, we collect the victim party's intermediate outputs (i.e., VO) of each sample in these sets and present their L_1-norm, L_2-norm and L_∞-norm distributions in Figure 4.2. We find that the difference becomes less pronounced as the fractional gap decreases. For example, the overlap between distributions is significantly greater in the $\Delta = 25\%$ scenario than in the $\Delta = 75\%$ case, regardless of the norm option. We further present a quantitative analysis of distances between these distributions in Table 4.1, using KL divergence (KL), JS divergence (JS), and Bhattacharyya coefficient (BC). Regarding for different norm options, the results indicate that the L_1-norm type effectively captures differences between distributions, leading it as the default norm function in the following.

The above observation shows that the norms of intermediate results contain property-related information, indicating a correspondence between the L_p-norm distributions of intermediate-result populations and their inherent fractions. Drawing inspiration from this, we leverage distribution-level differences to identify the statistical characteristics of a target property. The following section details the process of our proposed PIAs by learning such correspondence in VFL.

4.5 Methodology

Exchanged intermediate results among VFL parties inadvertently expose property information to possible adversaries during training. Based on that, we propose a novel framework of PIAs in VFL, named ProVFL, in which an adversarial party can identify the exact fraction of a target property owned by other parties with low inference error. It involves two modules: *distribution comparison* (DC) exploits the correspondence between intermediate-result distributions and fractions of the target property to infer property information, and *correlation augmentation* (CA) increases the property leakage of VFL to boost the attack performance. Within the CA module,

we design two approaches, label replacement and model refinement, to achieve better attack performance. We introduce each module of ProVFL in detail below.

Algorithm 3 Distribution Comparison: Using Intermediate Results to Infer Property Information in VFL

Input: D_k^{ir} (intermediate results of D_k), $D_{\text{aux}}^{\text{ir}}$ (intermediate results of D_{aux}), R (attack training set size), Q (number of distribution queries), M (sample size per query), q (step size of fraction interval)
Output: Predicted property fraction in D_k^{ir}
Split $D_{\text{aux}}^{\text{ir}}$ into property and non-property sets: D_{rep}^{+}, D_{rep}^{-}
$D_{\text{att}} \leftarrow [\,]$ ▷ Initialize attack training set
for $i = 1$ to R **do**
    $r \leftarrow \texttt{random\_sampling}(\{0\%, q\%, \ldots, 100\%\}, 1)$
    $D_0 \leftarrow \texttt{random\_sampling}(D_{\text{rep}}^{+}, r \cdot M)$ ▷ Sample property instances
    $D_1 \leftarrow \texttt{random\_sampling}(D_{\text{rep}}^{-}, (1-r) \cdot M)$ ▷ Sample non-property instances
    $x \leftarrow L_1\text{\_norm}(D_0 \cup D_1)$ ▷ Extract distributional representation
    $D_{\text{att}} \leftarrow D_{\text{att}} \,||\, (x, r)$ ▷ Append to attack set
end for
$g_\varphi \leftarrow \texttt{TRAIN}(g_\varphi; D_{\text{att}})$ ▷ Train the attack model
$Pr \leftarrow [\,]$ ▷ Store predicted property fractions
for $i = 1$ to Q **do**
    $D_q \leftarrow \texttt{random\_sampling}(D_k^{\text{ir}}, M)$
    $x_q \leftarrow L_1\text{\_norm}(D_q)$
    $Pr \leftarrow Pr \,||\, g_\varphi(x_q)$
end for
return mean(Pr)

4.5.1 Distribution Comparison

The core idea behind DC is to create multiple intermediate-result distributions with different fractions and then compare these to the victim party's distribution. To achieve this, DC involves three phases: (1) Distribution generation constructs various intermediate-result populations from auxiliary data. (2) Regression ensemble learns the correspondence to target properties by developing a set of regression models. (3) Property inference identifies the fraction of the victim party's dataset. We assume that an adversarial party initiates a PIA at the e_{SEC}-th training epoch. As Algorithm 3 presents, DC is performed as follows:

1) Distribution generation (Lines 1-7). In this process, the adversary aims to construct pairwise training data points for an attack model by sampling intermediate result populations with varying fractions related to the target property.

Given a specific kind of intermediate result to use, such as one of {AO, AG, VO, VG}, the attacker, e.g., P_k, first extracts these from auxiliary dataset D_{aux} and the attacker's training dataset D_k, respectively. Each sample in the auxiliary dataset originates from the adversary's feature space and is additionally labeled to indicate the presence of the target property. Such data can be collected from a subset of users on the adversary's side or inferred using advanced feature inference attacks. We define them as D^{ir}_{aux} and D^{ir}_{k}, where each record corresponds to the intermediate result representation of an individual sample. Next, the adversary creates a fraction candidate set {0%, q%,..., 100%} with a q% interval. For its fraction element r, DC creates an intermediate-result subset where r of the samples corresponds to property samples, while the remaining $1 - r$ of the subset belongs to non-property samples. Subsequently, we construct the L1-norm distribution of this subset by concatenating the L_1-norm values of all elements. We use $\hat{\mathbf{z}} = [\|z_1\|_1, \|z_2\|_1, \ldots, \|z_M\|_1] \in \mathbb{R}^{1\times M}$, where $\|\cdot\|_1$ denotes the L_1-norm function and z_i is the intermediate result of a training sample. Finally, we obtain an attack data point $(\hat{\mathbf{z}}, r)$ and then establish an attack training set D_{att} for each kind of intermediate result in a similar way.

2) Regression ensemble (Line 8). After preparing the attack training set, DC employs regression models to learn the relationship between L_1-norm distributions and fractions. These regression models serve as the attack model for property inference.

We use g_φ to refer to a regression model with φ representing its parameters and train it to minimize the total sum of absolute errors between predicted and actual fractions in D_{att}, formulated as $\varphi \leftarrow \arg\min_{\varphi} \frac{1}{|D_{\text{att}}|} \sum_{(\hat{\mathbf{z}},r)\in D_{\text{att}}} \ell\left(g_\varphi(\hat{\mathbf{z}}), r\right)$, where ℓ denotes a loss function used in regression tasks, such as mean squared error.

When multiple types of intermediate results are observable in C1, we employ ensemble

methods by combining regression models trained on different types of attack training sets to improve overall prediction accuracy. Specifically, each regression model is initially trained independently using a specific type of intermediate result. The predictions from all models are then aggregated into an ensemble by averaging their outputs. This ensemble is ultimately used to predict property inference. For the C2 scenario, only a single regression model is necessary.

3) Property inference (Lines 9-13). To estimate the proportion of a target property within the victim party's dataset, an attacker queries the attack model using the L_1-norm distributions of the training samples they can access.

The attacker randomly selects M data points from D_k^{ir} to construct a distribution vector, uses it to query the attack model, and obtains the predicted fraction. To enhance the robustness and accuracy of property inference, we generate Q distribution vectors and use the average of their predicted fractions as the final result. When multiple types of intermediate results are accessed, the attacker performs the above process independently for each type. This involves obtaining distribution vectors for all kinds of intermediate results and ultimately querying the ensemble model to achieve property inference.

4.5.2 Theoretical Analysis

Intuitively, enhancing the representational capabilities is advantageous for inference attacks related to the target property, as it enables an adversary to more clearly differentiate between property and non-property samples. We now analyze and demonstrate this point from a theoretical perspective. Following previous works [13, 14], we reduce exact property estimation into a binary classification task to distinguish two worlds. Then, a simplified PIA is to classify the following worlds:
World 0: The VFL has an intermediate-result distribution D_0 whose proportion of the target property is t_0.

World 1: The VFL has an intermediate-result distribution D_1 whose proportion of the target property is t_1.

We introduce *distinguishing accuracy* (*DAcc*) to measure how well an adversary can distinguish between D_0 and D_1. This metric signifies the probability of an adversary accurately discerning between two distributions when it can access two datasets randomly sampled from D_0 or D_1, respectively. Now, we provide an upper bound of the distinguishing accuracy as follows.

Theorem 1 (The upper bound of distinguishing accuracy)**.** *Given that t^* represents the prior probability of a sample having the target property, q is the fraction difference between two datasets, the distinguishing accuracy is bounded as:*

$$DAcc \leq \frac{1+\sqrt{1-\left(1+\frac{q(c-t^*)}{t^*(1-c)}\right)^{-N}}}{2}, \tag{4.3}$$

where $c = \Pr[p=1|z]$ denotes the conditional probability of an observed intermediate result owning the target property, and N denotes the size of an intermediate-result dataset.

Proof. Let Z denote a particular type of intermediate results, i.e., $Z \in \{\text{AO}, \text{AG}, \text{VO}, \text{VG}\}$ and $z \in Z$ as one of its elements. According to the corresponding probability density function, two distributions D_0 and D_1 can be expressed as follows:

$$\begin{aligned} D_0 &: \rho_0 = t_0 \Pr[z|p=1] + (1-t_0)\Pr[z|p=0], \\ D_1 &: \rho_1 = t_1 \Pr[z|p=1] + (1-t_1)\Pr[z|p=0]. \end{aligned} \tag{4.4}$$

Without loss of generality, we consider the case when $q = t_0 - t_1$. Then, given $t^* = \Pr[p=1]$, and according to Eq.(4.4), we have:

$$\begin{aligned} \rho_0 &= (t_1+q)\frac{\Pr[z]\Pr[p=1|z]}{\Pr[p=1]} + (1-t_1-q)\frac{\Pr[z]\Pr[p=0|z]}{\Pr[p=0]} \\ &= (t_1+q)\frac{\Pr[z]c}{t^*} + (1-t_1-q)\frac{\Pr[z](1-c)}{1-t^*}. \end{aligned} \tag{4.5}$$

Similarly,

$$\rho_1 = t_1\frac{\Pr[z]c}{t^*} + (1-t_1)\frac{\Pr[z](1-c)}{1-t^*}. \tag{4.6}$$

Combining Eq.(4.5) and Eq.(4.6), we can derive:

$$\frac{\rho_0}{\rho_1} = 1 + \frac{q\,(c - t^*)}{t_1\,(c - t^*) + t^*\,(1 - c)} \quad \leq 1 + \frac{q\,(c - t^*)}{t^*\,(1 - c)}. \tag{4.7}$$

The KL-divergence distance d_{KL} between ρ_0 and ρ_1 can be written as

$$\begin{aligned} d_{KL}\left(\rho_0 || \rho_1\right) &= \int \rho_0\left(z\right) \log\left(\frac{\rho_0\left(z\right)}{\rho_1\left(z\right)}\right) \mathrm{d}z \leq \int \rho_0\left(z\right) \log\left(1 + \frac{q\,(c - t^*)}{t^*\,(1 - c)}\right) \mathrm{d}z \\ &= \log\left(1 + \frac{q\,(c - t^*)}{t^*\,(1 - c)}\right) \int \rho_0\left(z\right)\, \mathrm{d}z = \log\left(1 + \frac{q\,(c - t^*)}{t^*\,(1 - c)}\right). \end{aligned} \tag{4.8}$$

Based on the derivation above and previous work [47] , for two datasets D_0' and D_1' containing N data points from D_0 and D_1, respectively, we achieve that:

$$DAcc \leq \frac{1 + d_{TV}\left(D_0', D_1'\right)}{2} \leq \frac{1 + \sqrt{1 - \mathrm{e}^{-d_{KL}\left(D_0' || D_1'\right)}}}{2} \leq \frac{1 + \sqrt{1 - \left(1 + \frac{q(c - t^*)}{t^*(1 - c)}\right)^{-N}}}{2}, \tag{4.9}$$

where $d_{TV}\left(\cdot, \cdot\right)$ is the total variation distance between two probability measures. □

Based on it, we derive Corollary 1, which further indicates potential enhancement strategies by increasing the correlation between observed representations and the target property.

Corollary 1. *The upper bound of the distinguishing accuracy is a monotone increasing function about the posterior probability of an observed representation owning the target property.*

Proof. Let $f\left(x\right) = \sqrt{1 - \left(1 + x\right)^{-N}}$, where $x = \frac{q(c - t^*)}{t^*(1 - c)}$. Given that t^* is a constant and q is fixed, it is clear that as x increases, the function $f\left(x\right)$ also increases. We find the first derivative of x with respect to c as follows:

$$\frac{\partial x}{\partial c} = \frac{q t^*\,(1 - t^*)}{\left[t^*\,(1 - c)\right]^2} > 0. \tag{4.10}$$

Consequently, x monotonically increases with increasing c, resulting in the increase of $f\left(x\right)$. □

Theorem 1 presents an upper bound on the attack power of an adversary who seeks to distinguish between two distributions, while Corollary 1 points out that the bound arises with the conditional probability of an observed representation owning the target property. It implies avenues for improving the attack performance by controlling the correlation between observed representations and target property. Motivated by these insights, we present two enhancement approaches in the following module.

4.5.3 Correlation Augmentation

Based on the theoretical evidence, we introduce label replacement (LR) and model refinement (MR) in this module for two adversarial cases C1 and C2. The core idea is that an adversary transforms intermediate results into highly indicative representations of the target property and intentionally amplifies the disparity between the distributions of intermediate results.

Label Replacement. Inspired by existing poisoning attacks [13, 14], label replacement involves an attacker substituting the original task label and thus infusing supervised property information in the entire VFL system. Specifically, an adversary acted by the active party, e.g., P_k, possesses the capability to control label information. Before the VFL training, it secretly replaces the task label y_i with the property label p_i for each sample $\mathbf{x}_i^k$ in auxiliary data, where $p_i = 1$ if it belongs to a property sample, otherwise $p_i = 0$. Subsequently, the VFL system undergoes supervised learning about the target property, intensifying the association between intermediate results and the target property.

This method does not need access to the top model's training since the label manipulation can be carried out independently. Yet, only the active party with auxiliary data is empowered to use this enhancement. In addition, since most VFL systems are developed on extensive datasets, poisoning a small fraction of training data has a negligible impact on overall performance as demonstrated in our experiments later.

Model Refinement. Another approach to enhance the effectiveness of PIAs is to induce the adversary's bottom model to leak more property information through supervised learning and output constraints.

During the training process, the adversary controls its bottom model entirely. It at first adds randomly initialized layers on top of the bottom model, and refines the bottom model by learning supervised knowledge from auxiliary data. Formally, an adversarial party P_k introduces a prediction layer $f_{\theta_\mathcal{T}}$ with parameters $\theta_\mathcal{T}$. Using the intermediate output of its bottom model as input, $f_{\theta_\mathcal{T}}$ is used to make decisions for a binary property classification task. Then, the supervised learning objective on the adversary party's side is to minimize:

$$\mathcal{L}_s = \frac{1}{|D_{\text{aux}}|} \sum_{\left(\mathbf{x}_i^k, p_i\right) \in D_{\text{aux}}} \ell\left(f_{\theta_\mathcal{T}}\left(f_{\theta_k}\left(\mathbf{x}_i^k\right)\right), p_i\right), \tag{4.11}$$

where ℓ denotes as a loss function. By optimizing $\mathcal{L}_s$, more information related to the target properties is exposed to the adversarial party.

Besides supervised property information, the difference between intermediate results of property and non-property samples is another factor for a successful PIA. Intuitively, a wider distribution gap between property and non-property samples results in easier property inference. To exaggerate this distinction, we propose an output loss function for updating the adversary's bottom model as follows:

$$\mathcal{L}_o = -\frac{1}{|D_{\text{aux}}|} \sum_{\left(\mathbf{x}_i^k, p_i\right) \in D_{\text{aux}}} p_i' \left\| f_{\theta_k}\left(\mathbf{x}_i^k\right) \right\|_1, \tag{4.12}$$

where $\|\cdot\|_1$ is the L_1-norm function and $p_i' = -1$ for a non-property sample, and 0 otherwise. $\mathcal{L}_o$ increases the norm value of property samples and suppresses the norm value of non-property samples, making their intermediate output distributions more dispersed.

Finally, by combining the aforementioned two loss functions, the attacker aims to minimize the final objective function as follows:

$$\mathcal{L} = \mathcal{L}_s + \lambda \cdot \mathcal{L}_o, \tag{4.13}$$

where λ is a hyperparameter for balancing two terms. The attacker's bottom model is updated along with the optimization process. Conventional optimization techniques like stochastic gradient descent can be applied to solve the first term $\mathcal{L}_s$. Regarding the second term $\mathcal{L}_o$, we follow previous work [128] to optimize it and then update the adversary's bottom model.

Algorithm 4 Passive and Active Property Inference Attacks for Two-party VFL

Input: $\mathbf{x}^1$ (features from passive party), $\mathbf{x}^2$, y (features and labels from active party), p (property), η (learning rate), f_{θ_1} (passive party model), f_{θ_2} (active party model), h_ϕ (top model owned by active party), $f_{\theta_\mathcal{T}}$ (auxiliary model for inference attack)
Output: Trained models f_{θ_1}, f_{θ_2}, h_ϕ
for each epoch **do**
  for each mini-batch $(\mathbf{x}^1, \mathbf{x}^2, y)$ **do**
    $y \leftarrow p$ ▷ LR: Replace y with property p for attack
    $\mathbf{v}_1 \leftarrow f_{\theta_1}(\mathbf{x}^1)$ ▷ Passive party forward pass
    $\mathbf{v}_2 \leftarrow f_{\theta_2}(\mathbf{x}^2)$ ▷ Active party forward pass
    $\mathbf{v} \leftarrow [\mathbf{v}_1; \mathbf{v}_2]$ ▷ Concatenate representations
    $\mathcal{L} \leftarrow \ell(h_\phi(\mathbf{v}), y)$ ▷ Active party computes loss
    Compute gradients: $\nabla_{h_\phi}\mathcal{L}$, $\nabla_{\mathbf{v}_1}\mathcal{L}$, $\nabla_{\mathbf{v}_2}\mathcal{L}$
    Update f_{θ_1} using $\nabla_{\mathbf{v}_1}\mathcal{L}$ ▷ Passive party update
    Update f_{θ_2} and h_ϕ using $\nabla_{\mathbf{v}_2}\mathcal{L}$, $\nabla_{h_\phi}\mathcal{L}$ ▷ Active party update
    $D^{\text{ir}} \leftarrow \{\mathbf{v}_1, \mathbf{v}_2, \nabla_{\mathbf{v}_1}\mathcal{L}, \nabla_{\mathbf{v}_2}\mathcal{L}\}$ ▷ DC: Store for passive attack DC
    $h \leftarrow f_{\theta_\mathcal{T}}(\mathbf{v}_1)$ ▷ MR: Attack by passive party; use $\mathbf{v}_2$ for active
    $\mathcal{L}_o \leftarrow \ell(h, p)$ ▷ MR: Compute loss via Eq.(4.12)
    Update $f_{\theta_\mathcal{T}}$ and f_{θ_1} ▷ MR: Update attack model
  end for
end for
return f_{θ_1}, f_{θ_2}, h_ϕ

In summary, both LR and MR approaches benefit from supervised learning, enhancing the representative capacity of intermediate results regarding the target property. MR goes further by making distribution gaps more distinguishing by restricting the attacker's bottom model outputs. Additionally, MR operates across different adversary's capacities, allowing arbitrary parties in VFL, whether passive or active, to enhance attack performance with the help of auxiliary data. Therefore, for C1, we apply both MR and LR approaches simultaneously, while for the C2 scenario, we utilize MR solely. We present the complete attack pipeline involving the three modules

in Algorithm 4.

4.6 Empirical Evaluation

4.6.1 Experimental Setup

Datasets. As most of the datasets used in VFL research are tabular type [93], we evaluate our attack on five popular tabular datasets as follows:

1) *Adult* [129, 130]: The Adult dataset comprises 44,355 records of US census data. Following the exclusion of *final_weight* and duplicate entries, we frame a classification task aimed at predicting whether an individual's salary exceeds $50,000. Within this dataset, we use *sex*, *race*, and *workclass* as private attributes vulnerable to potential inference attacks.

2) *Census* [14, 129]: As a repository of US census information, the Census dataset offers greater depth than the Adult dataset, comprising 299,285 records with 41 features. Similar to the Adult dataset, we utilize a binary classifier trained on the Census data to predict whether an individual's income exceeds $50,000 annually.

3) *Bankmk* [14, 129]: The Bankmk dataset, also known as Bank Marketing, comprises 45,211 records detailing a banking institution's marketing campaigns. A binary classification model is trained on this dataset to forecast whether a client has subscribed to a term deposit. In this analysis, we designate *month*, *marital*, and *contact* as private attributes.

4) *Health*: The Health dataset, called Health-Heritage, encompasses over 60,000 medical records. In line with prior research [102], we omit the *Year* and *MemberID* attributes before utilizing it to train a binary classifier aimed at predicting *max_CharIsonIndex*. In this context, we consider *Sex* and *AgeAtFirstClaim* as the target properties vulnerable to inference by potential adversaries.

5) *Lawschool* [1] [131, 132]: The Lawschool dataset, sourced from the Law School Admissions Council, comprises 96,584 records detailing various attributes of law students, including gender, LAST score, and GPA, among others. It constitutes a binary classification task to predict whether a student will secure admission. Within this dataset, we designate *race*, *resident*, and *gender* as attributes susceptible to inference.

6) *Texas* [133]: The Texas hospital dataset consists of inpatient stay records collected from multiple healthcare facilities, based on the Hospital Discharge Data from 2006 to 2009. It contains over 67,000 samples, each represented by 6,169 binary features, and covers 100 output classes. The dataset is characterized by its high dimensionality and extreme sparsity. In our experiments, we consider two different attributes from this dataset as target properties.

To simulate the VFL scenario, we evenly distribute the feature space of training data to different parties by default.

Table 4.2: Dataset description and target properties considered.

Dataset	Feature	Class	Size	Attribute	Num	Target	Fraction	Abbr.
Adult	14	2	44,355	sex	3	Male	66.21%	A-sex
				race	5	White	84.27%	A-race
				workclass	9	Private	67.52%	A-work
Census	41	2	299,285	sex	2	Female	52.05%	C-sex
				race	5	Black	10.20%	C-race
				education	17	Bachelor	9.94%	C-edu
Bankmk	16	2	45,211	month	12	May	30.45%	B-month
				marital	3	Married	60.19%	B-mart
				contact	3	Telephone	6.43%	B-cont
Health	17	2	61,700	sex	3	Female	16.13%	H-sex
				age	10	80+	2.68%	H-age
Lawschool	7	2	96,584	race	4	Black	8.63%	L-race
				resident	2	0.0	69.08%	L-res
				gender	2	0.0	44.27%	L-sex
Texas	6169	1000	67,330	Unknown-1	2	1.0	55.08%	T-u1
				Unknown-2	2	1.0	75.48%	T-u2

[1] `https://www.kaggle.com/datasets/danofer/law-school-admissions-bar-passage`

Selection of Target Properties. Our experiments include sixteen target properties across six datasets, all intrinsically private attributes such as gender, race, and age. Following previous studies [14], we comprehensively explore various fractions from small ratios (below 10%) to large ratios (above 50%), including binary and categorical attributes. Table 4.2 provides each sensitive attribute, the number of optional values in the selected attribute, target property, true fraction, and corresponding abbreviations.

Baseline. We choose *AIA* as our main baseline where an adversary estimates property information by inferring the specific value of the sensitive attribute and summarizing all results [100, 134]. We adopt a three-layer neural network as the binary classifier fed into intermediate results and make decisions about the target property. The predicted fraction is the ratio between the number of the predicted target property and the scale of a training dataset. Additionally, we explore other approaches, such as using the original training samples as input in AIA and simulating shadow model-based PIA within the VFL setting [13, 14, 57].

Model Architecture. Both the top and bottom models in VFL are constructed with their own neural networks. They utilize ReLU as the activation function and SGD as the optimizer. Specifically, the top model incorporates Sigmoid as its final layer and cross-entropy loss as its loss function. The aggregation of all intermediate outputs from the bottom models serves as the input for the top model. In our experiments, we mainly consider a two-party VFL while discussing multi-party VFL settings in the ablation study. Besides, we use XGBoost [135] as the default regression model.

Training and Attack Setting. For each dataset, we use 80% of data for training and the remaining 20% for testing in VFL. By default, we set D_{aux} with 2000 property samples and 2000 non-property samples (i.e., 1.3%-9.0% of original datasets). We configure the attack training set size $|R|$ to 200, the sampling size M to 2000, the fraction interval q to 1, and the number of distribution queries Q to 100. We perform PIAs in the penultimate epoch of the training process. All experiments are reported

on the average results of ten trials.

Evaluation Metric. Following previous work [109], we adopt mean absolute error (MAE) as the criteria to evaluate the attack performance. It is the average absolute difference between ground-truth fractions and predicted values. A lower MAE value indicates stronger attack performance.

4.6.2 Experimental Results

We validate the effectiveness of ProVFL in two adversarial cases: when the active party is the PIA adversary (C1) and when one of passive parties acts the PIA adversary (C2) during the VFL training process. For simplicity, we denote passive attacks as **ProVFL-B** when the adversary solely employs DC to infer target properties and active attacks as **ProVFL-M** when the adversary maliciously adopts LR or MR to enhance the performance of DC.

Performance of Passive Property Inference Attacks through Distribution Comparison. We first analyze the effectiveness of DC from the attack performance of ProVFL-B in both cases. Table 4.3 shows that ProVFL-B in C1 outperforms the baseline across all properties, significantly decreasing MAEs by nearly an order of magnitude (2.2× to 42.6×). Moreover, when employing AO solely in C2, ProVFL also achieves lower MAEs among most target properties. We also note that the baseline in C2 performs admirably, which results from the attack features used in the baseline coincidentally aligning well with the target property. Take L-sex as an example. The baseline can construct a high-quality classifier, achieving over 95% classification accuracy in practice. The performance difference can be attributed to inherent motivations: the baseline conducts inference at a record level, whereas our method performs property inference from a distribution-level perspective. Generally, it is easier to discern differences within an entire subset than within individual samples. Besides, when comparing ProVFL-B in different adversarial scenarios, the

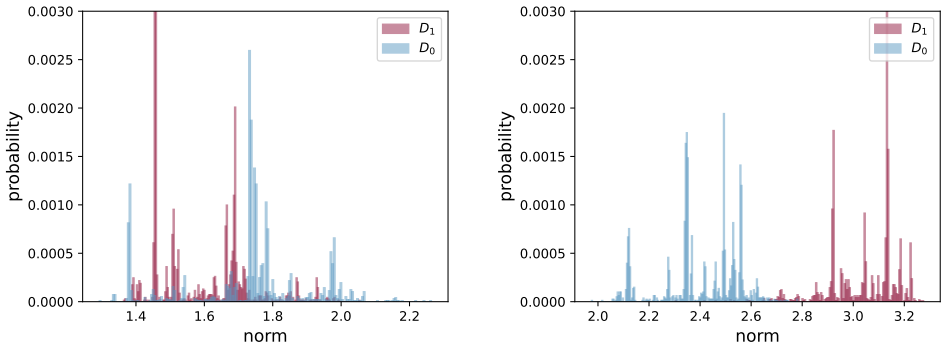
adversary in C1 often achieves better attack performance than in C2, especially in the Bankmk dataset. It demonstrates that DC effectively combines distribution information from different types of intermediate results and thus achieves low inference errors.

Table 4.3: Attack performance (MAE) of various attacks under different adversary's capacities.

Property	C1			C2		
	AIA	ProVFL-B	ProVFL-M	AIA	ProVFL-B	ProVFL-M
A-sex	0.1623	<u>0.0186</u>	**0.0097**	0.0181	<u>0.0164</u>	**0.0084**
A-race	0.2693	<u>0.0236</u>	**0.0143**	0.0388	<u>0.0209</u>	**0.0130**
A-work	0.2516	**0.0276**	<u>0.0280</u>	0.1777	<u>0.0483</u>	**0.0240**
C-sex	0.2127	**0.0175**	<u>0.0221</u>	0.0365	<u>0.0229</u>	**0.0184**
C-race	0.1913	<u>0.0310</u>	**0.0246**	0.2156	<u>0.0444</u>	**0.0255**
C-edu	0.1024	<u>0.0141</u>	**0.0106**	0.2149	**0.0125**	<u>0.0214</u>
B-month	0.2263	**0.0287**	<u>0.0362</u>	0.2189	<u>0.1135</u>	**0.0364**
B-mart	0.1370	<u>0.0372</u>	**0.0204**	0.0995	<u>0.0600</u>	**0.0325**
B-cont	0.2566	<u>0.1058</u>	**0.0236**	0.4039	<u>0.1967</u>	**0.0571**
H-sex	0.2853	**0.0067**	<u>0.0119</u>	<u>0.0221</u>	**0.0148**	0.0235
H-age	0.3564	<u>0.0290</u>	**0.0204**	0.0961	<u>0.0161</u>	**0.0130**
L-race	0.2066	<u>0.0449</u>	**0.0306**	0.2145	<u>0.0704</u>	**0.0281**
L-res	0.1930	<u>0.0858</u>	**0.0448**	0.1135	<u>0.0893</u>	**0.0500**
L-sex	0.1599	<u>0.0337</u>	**0.0185**	<u>0.0161</u>	0.0269	**0.0120**
T-u1	<u>0.0386</u>	0.0415	**0.0238**	0.0253	<u>0.0241</u>	**0.0195**
T-u2	0.1268	<u>0.0475</u>	**0.0251**	0.1490	<u>0.0802</u>	**0.0572**

Bold/<u>Underline</u>: the first/second smallest MAE in each case.

Performance of Active Property Inference Attacks through Correlation Augmentation. When an adversarial party adopts enhancement approaches, the inference estimation errors can be further reduced. Table 4.3 shows that an adversary party can achieve lower estimation errors for most target properties when it uses LR and MR approaches in C1. And for an adversary in C2, solely using the MR strategy further decreases the inference errors of target properties by 1.2× to 3.4×. Although ProVFL-B performs better in some properties, e.g., C-edu and H-sex, their differences are less than 1% MAE. These results demonstrate that CA enhances the performance of ProVFL-B and lowers inference errors regardless of the adversary's



(a) without MR (b) with MR

Figure 4.3: L_1-norm distribution of AO between property (D_1) and non-property samples (D_0) on the A-sex property.

role. We also observe that regarding target properties from Texas, although our proposed method still outperforms the baselines, the improvement of CA is less significant compared to other properties. This is partly because, for multi-class samples, we only apply MR to enhance the performance of ProVFL-B. Moreover, the number of task classes is substantially larger than the number of target attributes, resulting in limited supervised information being embedded in the intermediate representations. Besides, when comparing ProVFL-M in both cases, we observe that an adversary with less adversarial knowledge in C2 can sometimes achieve better attack performance, e.g., Adult dataset. We conjecture that this is because MR intentionally makes the attacker's AO more distinguishable and leads to significantly less overlap between their L_1-norm distributions, as shown in Figure 4.3. However, such a difference may be reduced when considering four types of intermediate results in C1.

Table 4.4: The performance of using the raw features under different levels of correlations.

Property	Attacks (MAE)		Correlation (PCC)	
	AIA*	ProVFL-B	p	$\mathcal{Y}$
B-month	0.1569	0.0287	0.4259	-0.1050
B-mart	0.0753	0.0372	0.2875	-0.0554
B-cont	0.2894	0.1058	0.1637	0.0131

Table 4.5: Attack performance (Accuracy) between SM-PIA and ProVFL-B.

Property	Values	SM-PIA	ProVFL-B
A-sex	30% vs. 50%	0.50	0.97
A-work	50% vs. 60%	0.55	0.88
A-race	5% vs. 10%	0.50	0.85

Other Baselines. Apart from the AIA baseline above, we additionally compare two possible methods with our ProVFL. 1) Using raw features rather intermediate results in AIA (named *AIA**): The raw features on the adversarial party's side can be used to deduce property information due to inherent correlation. Similar to the baseline, we use these features as the inputs of the binary classifier. Table 4.4 presents the attack performance, maximum correlation with the target property in adversarial party's features, and correlation with task labels using Pearson correlation coefficient (PCC). Although there is a high correlation, the AIA-based approach remains ineffective for property inference on the Bankmk dataset, e.g., B-month. In contrast, ProVFL demonstrates robustness by not relying heavily on correlations with the task label or other attributes, e.g., B-cont. 2) Shadow model-based PIA (named *SM-PIA*): We attempt shadow model training methods [13, 14, 57] by replicating the complete training process across the entire feature space. We adapt it and ProVFL-B to distinguish two predefined fraction values as shown in Table 4.5. As expected, ProVFL-B effectively handles both distributions regardless of the fraction gap, whereas SM-PIA fails to differentiate between them. In fact, the SM-PIA method assumes that an adversary has access to the full feature space, which is impractical for a party limited to its own feature subset in VFL.

Table 4.6: Attack Performance (MAE) and utility on misaligned auxiliary datasets.

Property	(C1) ProVFL-B	(C2) ProVFL-B	Utility (AUC)
A-sex	0.0091	0.0125	0.9016
A-race	0.0178	0.0247	0.9017
A-work	0.0661	0.0537	0.9016

Relaxed Assumptions on Auxiliary Data. Our above attacks are conducted under the assumption that auxiliary data are clean and aligned with other parties' samples. We now relax this assumption and discuss two general cases: **1)** *Noisy auxiliary data:* an adversary may use noisy samples to conduct ProVFL, e.g., by using feature inference attacks [17, 94, 136] to recover a handful of the victim's records. As shown in Figure 4.4a, we observe that for A-sex and H-age properties, the inference errors increase as the noise scale rises, but as for the other two properties, their attack performance remains relatively stable. Overall, when 10% auxiliary samples are mislabeled, the inference errors increase by 2%. It indicates that ProVFL is robust for noisy auxiliary data and maintains similar attack performance. **2)** *Misaligned auxiliary data:* An adversary cannot observe intermediate results during training if their sample IDs do not overlap with those of VFL parties. To simulate this scenario, we randomly replace the adversary's aligned training data with these samples prior to the VFL process. Table 4.6 demonstrates that ProVFL can successfully infer property information without significant performance deterioration in this scenario. The above results suggest that ProVFL remains robust and effective across different types of auxiliary data.

Table 4.7: Attack performance (MAE) under different CA strategies.

Property	w/o MR	w/o LR	with MR+LR
A-sex	0.0137/0.8917	0.0144/0.9009	0.0097/0.8931
A-race	0.0165/0.8950	0.0318/0.9017	0.0143/0.8960
A-work	0.0340/0.8907	0.0158/0.9009	0.0280/0.8907
B-month	0.0473/0.9129	0.0193/0.9057	0.0362/0.9047
B-mart	0.0226/0.9108	0.0313/0.8440	0.0204/0.7850
B-cont	0.0492/0.9045	0.1027/0.8840	0.0236/0.8360

/ means attack performance (MEA)/model utility (AUC).

Performance and Time Cost Analysis of LR and MR Approaches. Table 4.7 illustrates the attack performance and model utility of different enhancement approaches in C1. This suggests that neither LR nor MR plays a dominant role in CA, yet utilizing both consistently enhances attack performance without much loss

of model utility. Additionally, while CA introduces adjustments to VFL, it does not significantly increase time costs, as shown in Table 4.8. Vanilla DC has a lower computational overhead than the baseline, owing to its reliance on a regression model with small training costs. Regarding the CA module, LR incurs negligible additional computational overhead for VFL training. Although MR would require more time for the optimization process, the total cost is still lower than the baseline.

Table 4.8: Average time cost of various attacks on Adult dataset.

Attack	AIA	DC (ProVFL-B)	DC+MR	DC+MR+LR (ProVFL-M)
Time (s)	31.8	4.4	17.2	17.2

Table 4.9: Attack performance (MAE) of ProVFL-B and AIA on the NUS-WIDE dataset for different target properties.

Adversary Access	Method	Target Property		
		Grass	Animal	Water
Text Data	AIA	0.2997	0.1491	0.1853
	ProVFL-B	**0.0834**	**0.0324**	**0.0615**
Image Data	AIA	0.1814	0.1830	0.2120
	ProVFL-B	**0.0450**	**0.0273**	**0.0795**

Extension to Other Data Types. To validate the board generalizability of our proposed method, we conduct experiments on the multi-label image-text dataset NUS-WIDE [137], which contains 634 low-level image features and 1,000 textual tag features. Each type of feature is assigned to a separate client. We focus on the top-5 most frequent labels in our experiments. Unlike tabular data with structured feature columns, image and text data are inherently high-dimensional and unstructured. Therefore, we designate one type of label as the target property and use the remaining labels as the learning objectives for the VFL models. We perform our attacks on both feature types using ProVFL-B in C2, with the adversary holding image and text features, respectively. The results in Table 4.9 further demonstrate the effectiveness of our proposed method across different data modalities.

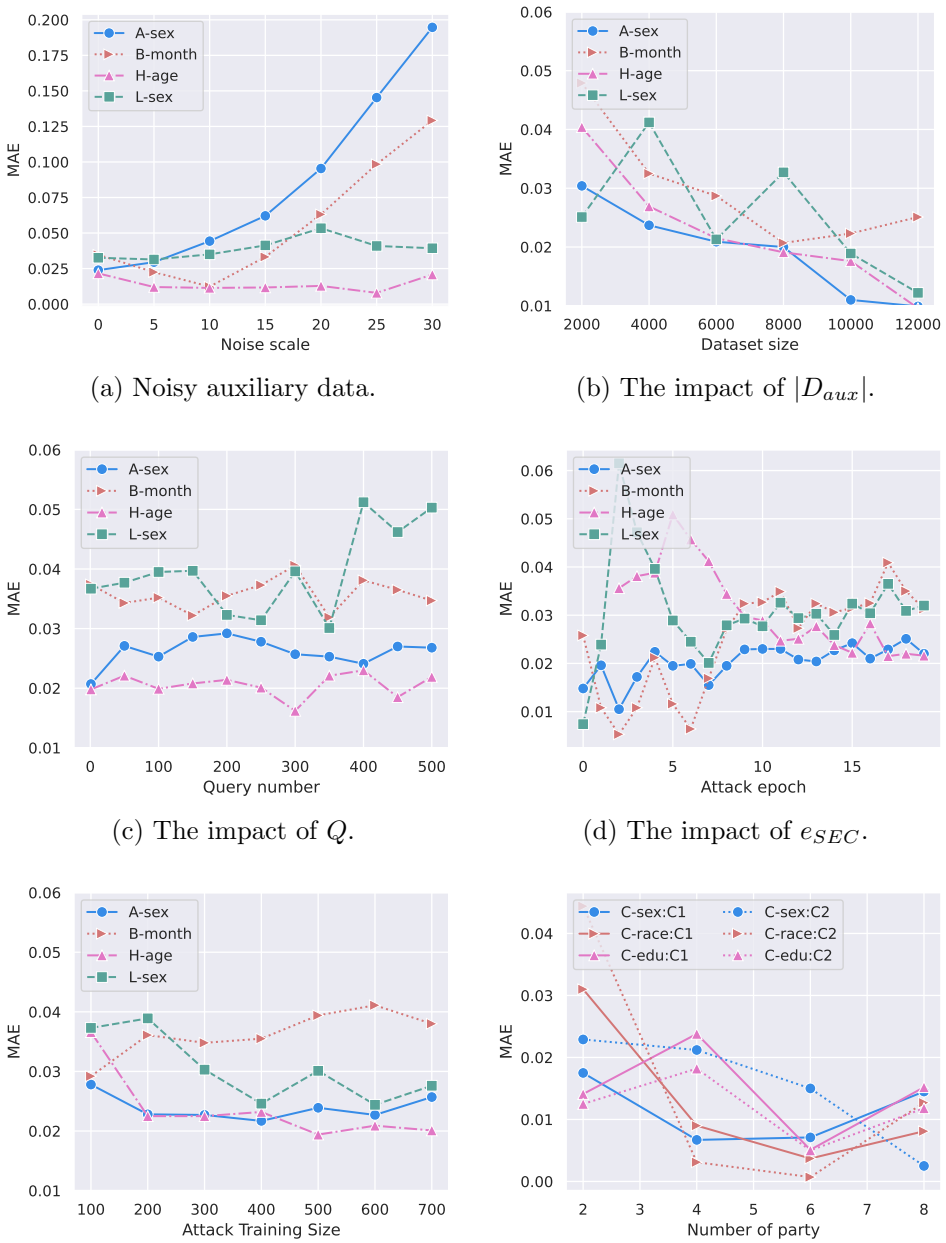
4.6.3 Ablation Study

In this section, we investigate the effects of different hyperparameter settings on attack performance, including the auxiliary dataset size $|D_{aux}|$, the number of distribution query Q, the attack epoch e_{SEC}, the attack training set size R, the fraction interval q, the architecture of regression models, and multi-party VFL settings. Considering the active party as the adversary, we delve into them in ProVFL-B across four target properties with distinct fractions, ranging from a larger ratio (e.g., A-sex: 66.21%) to a smaller one (e.g., H-age: 2.68%).

The Impact of Different $|D_{aux}|$. DC depends on an auxiliary dataset to prepare different populations of intermediate results. Consequently, we examine it to discern its impact on the attack's effectiveness by varying the size of the auxiliary dataset, as illustrated in Figure 4.4b. We observe that increasing the number of auxiliary samples improves the attack performance since the attack model can more effectively capture the differences across property distributions with more auxiliary samples.

The Impact of Different Q. We examine the effect on estimation errors by adjusting the number of distribution queries employed for inferring property information. Likewise, we carry out our ablation study on different levels of target properties in Figure 4.4c. In the beginning, as more queries are conducted, the attack performance improves by reducing the influence of outlier distributions randomly sampled from the training set. However, this improvement diminishes with further increases in the number of queries.

The Impact of Different e_{SEC}. We now explore the impact on attack performance by altering the epoch at which an adversary executes a PIA, as depicted in Figure 4.4d. We find a noticeable fluctuation when adversaries use intermediate outputs or gradients in the first several epochs. Nevertheless, the attack performance stabilizes as the attack epoch progresses. For instance, the MAE difference between two separate epochs is approximately 0.01 after the 10th epoch. As the training epoch



(a) Noisy auxiliary data.

(b) The impact of $|D_{aux}|$.

(c) The impact of Q.

(d) The impact of e_{SEC}.

(e) The impact of R.

(f) Multi-party settings.

Figure 4.4: Impact of various settings in ProVFL-B.

progresses, intermediate gradients and intermediate outputs demonstrate stability as the model converges, especially in the later stages of the training process. Hence, we suggest adversaries conduct an inference attack during the late epochs instead of at the beginning of training.

The Impact of Different R. We fix the size of the auxiliary dataset at 4000 and vary the size of the attack training set from 100 to 500. Figure 4.4e plots the corresponding attack performance. Upon surpassing a set size of 400, the improvement in attack performance becomes minimal. The estimation errors among target properties fluctuate within 0.01.

Table 4.10: Impact of different q on attack performance (MAE).

Property	0.1	1	5	10
A-sex	0.0257	0.0186	0.0232	0.0239
A-race	0.0250	0.0236	0.0264	0.0287
A-work	0.0374	0.0276	0.0315	0.0355

The Impact of Different q. We adopt the default setting of ProVFL-B in experiment settings but vary the interval of the fraction candidate set. Table 4.10 shows the attack performance under different q settings. When the fraction interval is 1, we can construct a distribution of appropriate granularity, achieving the lowest inference error.

Table 4.11: Attack Performance (MAE) among regression models.

Property	XGBoost	DTR	LNR	SVR
A-sex	0.0175	0.0228	0.0223	0.0197
A-race	0.0229	0.0297	0.0243	0.0508
A-work	0.0231	0.0250	0.0258	0.0459

The Impact of Regression Models. Our previous experiments employ an XGBoost as the attack model to infer property information. Now, we explore others to understand the effect of different regression models, including decision tree regression (DTR) [138], linear regression (LNR) [139], and support vector regression (SVR) [140]. Table 4.11 presents the performance of three regression models on the Adult dataset. It indicates that the non-linear XGBoost and DTR can more effectively capture the alignment of L_1-norm distribution and property fractions.

The Impact of Different λ. We explore the changes in the attack performance

Table 4.12: Impact of λ on attack performance (MAE) and utility (AUC).

λ	0.0	0.5	1.0	2.0
B-month	0.1135	0.0300	0.0364	0.0205
B-mart	0.0600	0.0389	0.0325	0.0297
B-cont	0.1967	0.0545	0.0571	0.0323
Utility	0.9134	0.8924	0.8779	0.8270

and VFL utility under different λ settings when using MR strategy, as presented in Table 4.12. Upon introducing the MR module, our attacks demonstrate much lower estimation errors on the Bankmk dataset. Although a larger λ setting tends to improve attack performance, an attacker has to balance the trade-off between property leakage and model utility.

The Impact of Multi-party VFL Setting. We present the attack performance in the multi-party VFL setting on the Census dataset due to its extensive feature space. Each party is allocated an equal portion of feature space. As Figure 4.4f shows, we present inference results on three target properties in C1 and C2. We observe that estimation errors become lower as the number of parties increases. This observation aligns with an intuition that, with a fixed feature space, more parties result in a smaller partial feature space, and the target property then becomes more dominant in the victim party's feature space.

The Impact of Non-IID VFL Setting. The experiments presented above were conducted under an even split of the feature space among parties. We now extend our investigation to non-IID feature distributions [141] and examine their impact on the proposed methods. Specifically, we consider scenarios where the feature space is unevenly partitioned, with different parties holding varying numbers of features. We define r as the ratio of the number of features held by the victim party to that of the other party. Table 4.13 shows that non-IID data in the feature space may reduce the property leakage, particularly when the victim party holds a large number of features.

4.6.4 Possible Defenses

In this section, we design several mechanisms to mitigate property leakage during the VFL training process, including common strategies as well as customized mitigation techniques specifically against our framework. We validate the defense performance against ProVFL-B in C1.

Table 4.13: Impact of Non-iid VFL setting on attack performance (MAE).

r	0.1	0.3	0.5	0.7	0.9
C-sex	0.0206	0.0141	0.0175	0.0273	0.0575
C-race	0.0552	0.0488	0.0310	0.0473	0.1169
C-edu	0.0275	0.0133	0.0141	0.0147	0.0330

D1. Add Noises to Gradient. A typical defensive strategy to mitigate information leakage in FL is adding noise into gradients [16, 57, 103]. In a vertical case, a trusted third party can add Laplacian noise to intermediate gradients before sending them to parties. We set the noise scale as the hyperparameter to adjust the effectiveness of this defense.

D2. Adopt Privacy-preserving Deep Learning. Following [16], we adopt a hybrid defense framework involving differential privacy [142, 143, 144] and gradient compression to mitigate property leakage. The r_G fraction of intermediate gradients are processed in the training process. A defender can vary r_G to trade-off between defense performance and model utility.

D3. Adopt Randomized Responses in Both Propagations. A recent state-of-the-art defense [115] for Split learning is proposed to disrupt knowledge transfer in both directions through a flexible, unified solution. It introduces a newly designed activation function, named R^3eLU, that transforms private smashed data in the forward pass and partial loss in the backward pass into randomized responses, respectively. We consider the privacy budget ϵ as the hyperparameter in our reproduction.

D4. Shuffle the Training Data. The correlation between intermediate results

and a target property is a key factor of information leakage. A possible defense is to disrupt such a correlation by shuffling the victim's training data. Specifically, the victim party randomly disorders some of his training data, which causes misalignment between its property information and the attacker's intermediate results. We treat the shuffling ratio of this mitigation as a hyperparameter.

D5. Withdraw Aligned Data Before Training. This defense indirectly changes the training data accessed by an adversarial party. After data index alignment, we can process intersecting features to hide the true statistic information of a sensitive attribute. To achieve this, the victim party deliberately withdraws part of intersecting samples containing the target property and thus changes its distribution in the training process. We consider different withdrawal ratios to validate its effectiveness.

Table 4.14: Defense Performance and utility of ProVFL-B on Adult Dataset.

	Setting	Value	Defense Performance ↑			Utility ↑
			A-sex	A-race	A-work	AUC
None	-	-	0.0186	0.0236	0.0276	0.9039
D1	Noise scale	10e-3	0.0219	0.0221	0.0370	0.9028
		10e-2	0.0239	0.0293	0.0407	0.9027
		10e-1	0.0185	0.0315	0.0499	0.8353
D2	r_G	0.50	0.0255	0.0349	0.0400	0.9009
		0.25	0.0219	0.0422	0.0410	0.8318
		0.10	0.017	0.0542	0.0428	0.7983
D3	ϵ	1.0	0.2154	0.3528	0.1605	0.5229
		4.0	0.1934	0.3355	0.1485	0.6303
		10.0	0.2058	0.3335	0.1248	0.6634
D4	Shuffling ratio	0.10	0.0211	0.0290	0.0387	0.8989
		0.20	0.0135	0.0383	0.0453	0.8948
		0.50	0.0083	0.0459	0.0351	0.8785
D5	Withdrawal ratio	0.10	0.0230	0.0281	0.0535	0.9014
		0.20	0.0617	0.0655	0.0828	0.9011
		0.50	0.1479	0.1345	0.1617	0.8979

We evaluate the above five mitigation mechanisms to defend our proposed method, where an attacker utilizes his bottom model and four kinds of intermediate results for property inference. Table 4.14 presents the defense performance across various con-

figurations. D1 and D2 offer slight mitigation against property leakage, accompanied by a clear degradation in model utility. Although D3 significantly reduces property leakage by simultaneously perturbing all intermediate results, it results in substantial utility loss across various target properties. Furthermore, such defense becomes less effective or even saturated as ϵ increases. Additionally, D4 proves ineffective in safeguarding against ProVFL, as shuffling merely disrupts alignment with the victim's data without altering the overall distribution of properties in the training set. As for D5, it diminishes the effectiveness of ProVFL with a small impact on model utility by removing property samples from aligned data, and thus alters the original property distribution. To further validate the defensive performance of D5, we evaluate it on a multi-classification task using the Texas dataset. As shown in Table 4.15, D5 effectively mitigates the performance of ProVFL-B while incurring minimal loss in classification accuracy.

This adjustment significantly weakens the adversary's ability to infer property information, offering a straightforward yet effective defense to safeguard sensitive private information.

Table 4.15: Defense Performance and utility of ProVFL-B on Texas Dataset.

	Setting	Value	Defense Performance ↑ T-u1	Defense Performance ↑ T-u2	Utility ↑ ACC
None	-	-	0.0415	0.0475	0.5607
D5	Withdrawal ratio	0.10	0.0777	0.0840	0.5480
		0.20	0.0717	0.1294	0.5458
		0.50	0.1276	0.1591	0.5511

4.7 Chapter Summary

VFL is a widely used framework in various applications, but its distribution-level privacy leakage has not been thoroughly explored. In this chapter, we reveal the risk of property leakage in VFL by introducing ProVFL, a novel framework designed to

perform property inference with low estimation errors. ProVFL is based on the empirical observation that the L_p-norm distribution of intermediate results unintentionally leaks private property information to potential adversarial parties. To address this issue, we propose a distribution-based comparison method for property inference and theoretically analyze its effectiveness. Furthermore, we introduce a correlation augmentation strategy to amplify property information leakage. Experimental results show that ProVFL is effective across diverse target properties, underscoring that shared intermediate representations in VFL can readily lead to privacy leakage. We hope this work provides valuable insights into the protection of private statistical information in VFL.

Chapter 5

Relative Membership Risk Measure for Machine Learning Models

5.1 Introduction

Thanks to the rapid advancements in machine learning and knowledge discovery, companies like Google [145], Amazon [146], and Microsoft [147] now offer Machine Learning as a Service (MLaaS). These platforms enable users to train and deploy ML models using private datasets through web interfaces or out-of-the-box APIs. However, such models are vulnerable to exposing private information from their training data. For instance, Carlini *et al.* [148] demonstrated how a downstream task of GPT-2, namely autocompletion, could be exploited to successfully extract sensitive information such as an individual's full name, physical address, email address, phone number, and fax number. As a result, users often aim to release a safer model when a set of candidates is provided, relying on privacy risk assessments to mitigate potential privacy breaches.

Among the attacks that pose significant risks to data owners' privacy, MIAs which determine whether a query example is part of a model's training dataset, is a fundamental cause of privacy breaches [149, 150]. The first study was proposed by Shokri *et al.* [149] using shadow training. Since then, a range of related works have extended the scope and capability of MIA in generative models [151, 152, 153], regression models [154] and pre-trained models [155, 156]. These attacks pose a significant threat to an individual's data used for training ML models. For instance, if an attacker discovers that Alice's medical record was used to train an AIDS prediction model, they could infer that Alice is likely diagnosed with AIDS, resulting in a severe violation of her privacy. Beyond direct privacy violations, MIAs also serve as a foundation for more advanced inference attacks, such as model stealing [157] and model inversion [158], enhancing their effectiveness and extending their impact. As the reliance on sensitive data to train machine learning models continues to grow, assessing membership privacy risks has become a crucial aspect of privacy protection.

The work focuses on evaluating membership risk across multiple model candidates and selecting a privacy-preserving model for model owners. Such scenarios commonly arise in real-world scenarios, as organizations often develop or acquire multiple machine learning models to solve the same task. These models may vary in their architecture, training setting, or development approach. For instance, an e-commerce platform might develop various recommendation algorithms, such as neural networks and collaborative filtering, to assess which model best balances user experience with privacy protection, minimizing the risk of exposing sensitive user preferences [57, 159, 160]. In federated learning, multiple local models are trained by different participants (e.g., hospitals or banks) and later aggregated into a global model [15, 106]. Evaluating privacy risks is crucial in this context to ensure the selected global model is resilient to inference attacks.

A straightforward approach to evaluating membership risk is to measure the success rate of existing MIAs [55, 161]. However, this approach is time-consuming when ap-

plied to numerous candidate models and may underestimate the privacy risks posed by stronger attacks. For example, [161] evaluates the privacy risk score of training examples using a modified confidence-based MIA. Similarly, ML-DOCTOR [55] estimates a model's membership risk using a shadow training-based attack [149]. The computational cost becomes even more prohibitive when multiple attacks are conducted, and relying on a single attack method may fail to capture the full membership risk exposed by more powerful attacks [11]. Thus, an ideal membership privacy risk metric should focus on the root cause of MIAs, remain independent of any particular attack method, and be adaptable to future MIA techniques [162].

Previous research proposes attack-agnostic solutions to address this problem, though with certain limitations. These approaches are based on the underlying causes of membership privacy leakage, such as overfitting [149, 163] and model stability [163, 164]. For instance, overfitting is not a prerequisite for MIA. It estimates the model risk according to the generalization gap between training and test data, ignoring the different effects of training examples. Even in a well-generalized model, an adversary can identify vulnerable training examples with high precision [165]. Model stability, which measures the influence of a particular training example on the model output, often leads to a high computational cost for the leave-one-out process [162, 164].

In this chapter, we propose an efficient and accurate MIA risk measure for machine learning models. The task of measuring the MIA risk of a target model is overcomplicated in practice as it is affected by many factors, e.g., model type [166], distribution of training data [149, 165], training strategy [163, 167, 168] and model memorization [7, 169]. Instead of measuring the membership privacy risk directly, we propose to measure Relative Membership Risk (RMR) of a model with respect to a *reference model*, which is acted by a trained model in the candidates without a retraining process. We treat the risk of a reference model as a *benchmark* and then derive the relative risk of target models on top of it. RMR is provable and practical to reveal the membership privacy of a training example or a machine learning model, which does

not depend on specific training strategies, for example, combined with differential privacy mechanisms [113]. Furthermore, we show that a high-risk reference model should be chosen to measure RMR accurately among a set of candidate models to assess. Therefore, we propose an iterative algorithm to find the most vulnerable model among those models efficiently. Besides, RMR can evaluate the privacy risk of an individual example to identify those vulnerable to membership attacks.

In summary, our contributions in this chapter are as follows: (1) We formally define a relative membership risk measure, namely RMR, to assess the membership inference risk of a training example or a model without depending on particular training strategies or data distribution beyond the example loss. (2) We analyze the impact of the risk of a reference model itself on the accuracy of RMR and devise an iterative algorithm to choose a suitable reference model in practice. (3) We conduct extensive experiments to demonstrate that RMR can reflect the practical MIA performance on various datasets and models. We also show that the MIA threats posed to a model can be mitigated by a risky-example-removal strategy based on the RMR measure.

5.2 Related Work

Owing to the widespread prevalence of MIAs and the privacy leakage risks they introduce, numerous efforts have focused on developing methods to quantify their impact on ML models. These approaches are broadly classified into attack-related methods, which evaluate risk based on specific attacks, and attack-agnostic methods, which focus on the underlying causes of membership privacy leakage.

Attack-related methods, which evaluate risk based on the success rate of existing MIAs, provide a straightforward approach to measuring membership privacy risk [55, 161]. However, these methods are time-intensive when applied to multiple candidate models and may underestimate the leakage risks. For instance, [161] employs a mod-

ified confidence-based MIA to calculate the privacy risk score of training examples. Similarly, [55] utilizes a shadow training-based attack [149] to estimate a model's membership risk. Such risk measurement approaches are heavily influenced by adversary knowledge, making them less reliable under varying conditions. Additionally, the computational cost escalates significantly when multiple attacks are performed. Moreover, relying on a single attack method may fail to capture the full extent of membership risk exposed by advanced attacks.

To keep independent of specific MIAs and adaptable to future attacks, attack-agnostic methods are designed to address membership leakage at its core. Existing works on this topic have explored the problem from two perspectives: the generalization gap and model stability. The generalization gap, also known as *overfitting*, is a sufficient but not necessary condition for a successful MIA [163]. Due to overparameterized networks and redundant training epochs, machine learning models often overfit and memorize training examples. While the connection between overfitting and MIA has been demonstrated both empirically [149, 170, 171, 172] and theoretically [163], recent studies have shown that certain MIAs can still succeed even when overfitting is not severe [148, 165]. Another limitation of overfitting as a measure of MIA risk is that it cannot measure the risk of a single training example. Attack-agnostic approaches based on model stability focus on how much a training example influences the model's output. Examples that significantly affect the model are believed to pose a higher risk of membership inference breaches. Long *et al.* [164] quantitatively assesses the sensitivity of each training example and used the most influential examples as an empirical MIA risk measure, similar to the leave-one-out technique. However, this approach requires training a shadow model for each example and is only feasible for small training sets. To address the computational burden of the naive leave-one-out strategy, SHAPr [162] is proposed as an approximation method based on Shapley values. Model stability measures have received particular attention in models trained with differential privacy mechanisms [113, 173], which limit the change in a model's

response to a specific example. [163] formally derives a bound for membership advantage in a differential private model, given by $e^{\epsilon} - 1$. Bernau *et al.* [167] obtain a tighter identifiability bound using Bayesian posterior belief. However, a range of works [163, 167, 174] show that these theoretical bounds are too loose to effectively measure membership risk in practice. Additionally, not all machine learning models, particularly those used in sensitive areas like healthcare or finance, can tolerate significant utility loss or be trained with differential privacy mechanisms.

Compared to existing approaches, our study aims to provide an effective, accurate, and attack-agnostic solution for estimating membership privacy risks for both training data records and models.

5.3 Methodology

This section presents the measure of privacy risk under membership inference attacks. The reason why MIAs work involves too many factors, including model architecture [166], training strategy [165, 175], data distribution [176] and overfitting of target model [163]. Hence it is intractable to consider all these factors and measure the membership risk directly. In this section, we take an alternative approach and define the risk in a relative manner. We first derive the absolute membership risk as the probability of an example being a member. Then we formally present the relative risk measure RMR to assess the MIA risk of a model.

5.3.1 Absolute Membership Risk

A supervised ML model learns from a labeled training set and makes predictions on unlabeled inputs [177]. Let $D = \{(x_i, y_i)\}_{i=1}^{N}$ denote the training set, where N is the total number of examples, and (x_i, y_i) is the i-th sample with a feature vector x_i labeled by y_i. We treat an ML model as a mapping function $f(x; \theta)$, where x is

an input example and θ is model parameters. Given a training set D, the model is trained to minimize a loss function $\ell\left(y, f\left(x;\theta\right)\right)$, or simply $\ell\left(x,\theta\right)$ over D.

Membership inference attacks aim to deduce from a model an example's existence in the training set. Consequently, the probability of an example being a member denotes the risk of private information leakage, referring to how much the amount of exposure to attackers. As such, we formulate the absolute membership risk (AMR) as follows.

Definition 1 (absolute membership risk)**.** *Given a target model θ_t trained on D and a training example x, the absolute privacy leakage risk of θ_t about x is:*

$$amr\left(x,\theta_t\right) = P\left(m=1|x,\theta_t,D\right), \tag{5.1}$$

where m denotes the membership status that is 1 for a member, otherwise 0.

The absolute membership risk is essentially the posterior probability distribution over a target model and an example. Inspired by [178], the absolute membership risk can be further expressed as:

$$P\left(m=1|x,\theta_t,D\right) = \sigma\left[-\ell\left(x,\theta_t\right) + \tau_p\left(x\right) + t_\lambda\right], \tag{5.2}$$

where

$$\tau_p\left(x\right) = -\log\left(\int_{\theta'} \exp\left(-\ell\left(x,\theta'\right)\right) P\left(\theta'|D\backslash\{x\}\right) \mathrm{d}\theta'\right), \tag{5.3}$$

$$t_\lambda = \log\left(\frac{P(m=1)}{P(m=0)}\right).$$

The term $P\left(\theta'|D\backslash\{x\}\right)$ in Eq.(5.3) denotes the posterior distribution in which the example x is removed from D, the term t_λ denotes the log ratio of prior distribution of members ($P(m=1)$) to that of non-members ($P(m=0)$), and the term σ is the sigmoid function, that is, $\sigma\left(x\right) = 1/\left(1+e^{-x}\right)$. As we take a further look at Eq.(5.2), the first two terms dominate the estimated membership probability, as the last term is a constant and approaches 0 if the query example has similar probabilities

of being a member and a non-member. Using Jensen's inequality and the optimum over parameter space [179], Eq.(5.2) can be approximated as follows:

$$P\left(m=1|x,\theta_t,D\right)\approx\sigma\left[-\ell\left(x,\theta_t\right)+\ell\left(x,\theta'\right)\right], \tag{5.4}$$

where θ' denotes an *auxiliary model* that is trained on the training set $D\backslash\{x\}$. To mitigate the impact of factors unrelated to the specific example x, θ' should have the same training settings as θ_t.

The challenge remains in deriving the auxiliary model for the whole training set, although Eq.(5.4) provides a practical solution to compute the absolute membership risk of a single instance. When measuring the membership risk of a model, we need to consider the privacy leakage risks of all examples. One possible approach to achieve this is to adopt the leave-one-out strategy for each example in the training set, similar to [164], which evidently incurs significant computational costs. Another solution [180] is to introduce a shadow dataset drawn from the same data distribution as the training set.

Our methodology proposed in the following subsection measures the relative risk against a reference model, thus canceling out the auxiliary model and circumventing the heavy computation. It requires an individual reference model for the whole training dataset and avoids introducing an additional dataset when measuring the model risk. When dealing with a set of candidate models, it avoids the need for retraining by selecting one model to serve as the reference.

5.3.2 Relative Membership Risk

In this subsection, we present the definition of Relative Membership Risk (RMR), a relative MIA risk measure to the aforementioned membership inference. A model's risk is relative to a preset *reference model* trained on the same dataset. In essence, RMR designates the additional privacy leakage of a target model with respect to the reference model. Formally,

Definition 2 (relative membership risk)**.** *Given a reference model θ_r, a target model θ_t and a training example x, the relative privacy leakage risk of θ_t about x compared with θ_r under membership inference attacks is the difference of their absolute membership risks:*

$$rmr_{\theta_r}(x, \theta_t) = amr\,(x, \theta_t) - amr\,(x, \theta_r)\,. \tag{5.5}$$

The following theorem provides an upper bound of RMR based on our discussion in Section 5.3.1.

Theorem 2. *Given a target model θ_t and a reference model θ_r, if $amr\,(x, \theta_t) \leq amr\,(x, \theta_r)$, the rmr of an example x with θ_t with respect to θ_r is bounded as:*

$$rmr_{\theta_r}(x, \theta_t) \leq k\sigma\left[-\ell\,(x, \theta_t) + \ell\,(x, \theta_r)\right] - 1, \tag{5.6}$$

where $k = 2 + e^C + e^{2C}$, and C is the upper bound of training example loss.

Proof. According to Eq.(5.4), we introduce an auxiliary model θ' that is trained on the same training set but without the sample x, then the relative risk is:

$$rmr_{\theta_r}(x, \theta_t) = \sigma\left[-\ell\,(x, \theta_t) + \ell\left(x, \theta'\right)\right] - \sigma\left[-\ell\,(x, \theta_r) + \ell\left(x, \theta'\right)\right].$$

According to Lemma 1, we have $\ell\,(x, \theta_t) \leq \ell\,(x, \theta')$. On the assumption that $amr\,(x, \theta_t) \leq amr\,(x, \theta_r)$, the loss of x of the auxiliary model θ' should be beyond that of θ_r and θ_t, i.e., $\ell\,(x, \theta_r) \leq \ell\,(x, \theta_t) \leq \ell\,(x, \theta')$. Besides, let $\alpha \overset{def}{=} -\ell\,(x, \theta_t) + \ell\,(x, \theta')$, $\beta \overset{def}{=} \ell\,(x, \theta_r) - \ell\,(x, \theta')$, and then $\alpha \geq 0$ and $\beta \leq 0$. Since an example's loss is at least zero and at most C, we have $-C \leq \alpha + \beta \leq 0$. Let $h\,(\alpha, \beta) = \frac{\sigma(\alpha)+\sigma(\beta)}{\sigma(\alpha+\beta)}$, and we have:

$$h\,(\alpha, \beta) = \frac{2 + e^{-\alpha} + e^{-\beta}}{1 + \left(e^{-\alpha} + e^{-\beta}\right) / \left(1 + e^{-(\alpha+\beta)}\right)} \leq 2 + e^{-\alpha} + e^{-\beta} \leq 2 + e^C + e^{2C}.$$

Therefore, we can derive that:

$$\begin{aligned} &rmr_{\theta_r}(x, \theta_t) = \sigma\,(\alpha) + \sigma\,(\beta) - 1 = h\,(\alpha, \beta)\,\sigma\,(\alpha + \beta) - 1 \\ &\leq \left(2 + e^C + e^{2C}\right)\sigma\left[-\ell\,(x, \theta_t) + \ell\,(x, \theta_r)\right] - 1. \end{aligned}$$

□

Lemma 1. *Let θ_1 and θ_2 be different converged models under identical training settings, except that θ_1's training dataset contains a data point x, while θ_2's does not. Then, the following inequality holds:*

$$\ell(x, \theta_1) \leq \ell(x, \theta_2),$$

where $\ell(x, \theta)$ denotes the loss of model θ on the data point x.

Proof. According to the principle of Empirical Risk Minimization (ERM), a machine learning model is trained to minimize the loss over all data points in the training dataset. Formally, this can be expressed as:

$$\theta = \arg\min_{\theta} R(D, \theta) = \arg\min_{\theta} \frac{1}{N} \sum_{x' \in D} \ell(x', \theta),$$

where N represents the number of samples in D. Given the training sets D and D' for training θ_1 and θ_2, $D = D' \cup \{x\}$, the gradients of the empirical risk functions are expressed as follows:

$$\nabla_\theta R(D, \theta_1) = \frac{1}{N} \sum_{x' \in D} \nabla_\theta \ell(x', \theta_1),$$

and

$$\nabla_\theta R(D', \theta_2) = \frac{1}{N-1} \sum_{x' \in D'} \nabla_\theta \ell(x', \theta_2).$$

Given that θ_1 and θ_2 are converged, they satisfy: $\nabla_\theta R(D, \theta_1) = 0 \quad \text{and} \quad \nabla_\theta R(D', \theta_2) = 0$. under the same training conditions and assuming a smooth loss function, θ_1 can be viewed as an offset of θ_2. Specifically, we have:

$$\theta_1 = \theta_2 + \Delta\theta,$$

where $\Delta\theta$ represents the offset between the two parameter sets. Meanwhile, we have $\nabla_\theta R(D', \theta_1) = \nabla_\theta R(D', \theta_2) + H\Delta\theta$, where H denotes the Hessian matrix of $R(D', \theta)$ with respect to θ and is positive definite. Then, we obtain:

$$\Delta\theta = -\frac{1}{N-1} H^{-1} \nabla_\theta \ell(x, \theta_1).$$

By substituting $\Delta\theta$ and applying Taylor's formula, we obtain:

$$\ell(x,\theta_1) = \ell(x,\theta_2) - \frac{1}{N-1}\nabla_\theta \ell(x,\theta_2)^\top H^{-1} \nabla_\theta \ell(x,\theta_1).$$

The second-order Taylor expansion can be safely neglected, as θ_1 and θ_2 have converged and are close, making both $\Delta\theta$ and $\nabla^2_\theta \ell(x,\theta_2)$ sufficiently small. Finally, as the directions of $\nabla_\theta \ell(x,\theta_1)$ and $\nabla_\theta \ell(x,\theta_2)$ are aligned, we conclude that $\ell(x,\theta_1) \leq \ell(x,\theta_2)$. □

Thanks to Theorem 2, we can approximate RMR using Inequality (5.6), and further simplify it by removing the constant coefficients and offsets. This simplification is reasonable for well-trained models which have already fit the training data with small losses. As such, we redefine RMR as:

$$rmr_{\theta_r}(x,\theta_t) \stackrel{def}{=} \sigma\left[-\ell(x,\theta_t) + \ell(x,\theta_r)\right]. \tag{5.7}$$

The following corollary further shows that RMR preserves the order of the absolute membership risk so that it can provide accurate comparison results of the risks of two models under membership inference attacks.

Corollary 2 (order-preserving property)**.** *Given models θ_1 and θ_2 trained on D and a reference model θ_r, for a training example x, if $amr(x,\theta_1) \geq amr(x,\theta_2)$, then $rmr_{\theta_r}(x,\theta_1) \geq rmr_{\theta_r}(x,\theta_2)$.*

Proof. Since $amr(x,\theta_1) \geq amr(x,\theta_2)$, according to Eq.(5.4), we have:

$$\sigma\left[-\ell(x,\theta_1) + \ell(x,\theta')\right] \geq \sigma\left[-\ell(x,\theta_2) + \ell(x,\theta')\right],$$

where θ' is an auxiliary model. By the monotonicity property of sigmoid function, we have $\ell(x,\theta_1) \leq \ell(x,\theta_2)$. So given a reference model θ_r, we have

$$\sigma\left[-\ell(x,\theta_1) + \ell(x,\theta_r)\right] \geq \sigma\left[-\ell(x,\theta_2) + \ell(x,\theta_r)\right].$$

□

Based on Eq.(5.7) and Corollary 2, the RMR of a target model trained on D is defined as the average risk of the whole training set given a reference model:

$$rmr_{\theta_r}(D, \theta_t) \overset{def}{=} \mathbb{E}_{x \sim D}\left[\sigma\left(-\ell\left(x, \theta_t\right) + \ell\left(x, \theta_r\right)\right)\right]. \tag{5.8}$$

The advantage of RMR for a machine learning model lies in its efficiency, as it requires only a single model to evaluate the entire training set, rather than needing an auxiliary model for each individual training example [179, 180]. Moreover, RMR strictly maintains the ranking of the absolute membership risks of target models, as established in Corollary 2, making it a lightweight relative risk indicator for membership inference attacks. As a final note, while our method derives RMR from the posterior distribution [178], this relative measure is adaptable to other attack scenarios with similar closed-form posterior probabilities as Eq.(5.2), which serve as the basis for Theorem 2 and Corollary 2.

5.3.3 Reference Model Selection

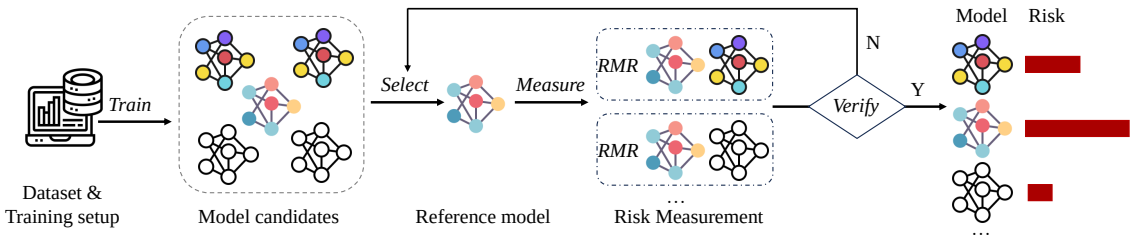


Figure 5.1: Overview of our proposed algorithm for measuring membership risks across multiple target models.

In this section, we explore methods for measuring the membership risks associated with multiple target models. This scenario often arises when a user attempts training configurations, such as model architectures and hyperparameters, resulting in a set of candidate models for potential release. Beyond predictive accuracy, assessing the privacy leakage risk of these models is crucial, particularly in sensitive applications like medical or financial domains. In such scenarios, where the goal is to select the

safest model against MIAs from a set of candidate models, we propose an iterative selection algorithm that identifies a reference model without requiring the retraining of an inference model.

As illustrated in Fig. 5.1, the candidate models are pre-trained, off-the-shelf target models provided by the user. According to Theorem 2, a valid reference model must exhibit a higher AMR, i.e., greater risk, compared to any target model as measured by RMR. A straightforward approach is to select the model with the highest RMR within the candidate set. The following corollary provides a simple method to verify whether this prerequisite is satisfied.

Corollary 3 (prerequisite of Theorem 2)**.** *Given a reference model θ_r, a target model θ_t, and a training example x, if $rmr_{\theta_r}(x, \theta_t) \leq 0.5$, then $amr(x, \theta_r) \geq amr(x, \theta_t)$. Consequently, Theorem 2 is satisfied.*

Proof. Since $rmr_{\theta_r}(x, \theta_t) \leq 0.5$, it follows that $\ell(x, \theta_r) \leq \ell(x, \theta_t)$. By the monotonicity property of the sigmoid function, for any auxiliary model θ', it holds that $amr(x, \theta_r) \geq amr(x, \theta_t)$. □

According to Corollary 3, the average risk of θ_t trained on the whole training set D is also no larger than 0.5, i.e, $rmr_{\theta_r}(D, \theta_t) \leq 0.5$. Based on this condition, we propose an iterative reference model selection algorithm that finds the riskiest model in a candidate set. In each iteration, it (1) chooses the current riskiest model as the reference model, and (2) checks whether any measurement violates the prerequisite of RMR. Since this algorithm returns the entire list of RMRs after the reference model is finalized, it also reveals the safest target model under MIAs for a user to choose for release.

The pseudo-code is presented in Algorithm 5. First, it randomly selects an initial reference model from the target model set Θ (Line 4). Then the algorithm iteratively computes the RMR of each model by (5.8) (Line 7). When any RMR is larger than

Algorithm 5 Iterative Reference Model Selection and RMR Sorting

1: **Input:** $D = \{x_1, \ldots, x_N\}$ (training dataset), $\Theta = \{\theta_1, \ldots, \theta_M\}$ (target model set)
2: **Output:** RMR measurement list R
3: $R \leftarrow \mathbf{0}_{[M]}$; $St \leftarrow \emptyset$; $unserved \leftarrow \Theta$
4: $\theta_r \leftarrow \texttt{randomSelect}(\Theta)$ ▷ Initialize reference model
5: **while** True **do**
6: **for** $m = 1$ to M **do**
7: $R[m] \leftarrow rmr_{\theta_r}(D, \theta_m)$ ▷ Compute RMR
8: **end for**
9: $unserved \leftarrow \texttt{delete}(unserved, \theta_r)$ ▷ Remove served model
10: $St \leftarrow \texttt{sort}(\Theta, R)$ ▷ Sort Θ by RMR
11: **for** $m = 1$ to M **do**
12: **if** $R[m] > 0.5$ **then**
13: $\theta_r \leftarrow \texttt{unserSelect}(unserved, St)$ ▷ Select riskiest model
14: **break**
15: **end if**
16: **end for**
17: **if** $m == M$ **then**
18: **break** ▷ Early stop
19: **end if**
20: **end while**
21: **return** R

0.5, that is, unsuccessfully validated by Corollary 3, it selects a current unserved model with the highest RMR value as the reference model (Line 12) and updates the RMR list in the next iteration. Otherwise, the riskiest reference model is identified, and the algorithm will terminate (Line 15). It is easy to prove that Algorithm 5 can terminate within M (the size of Θ) iterations due to M candidate models. Therefore, considering a set of M candidate models, the time complexity of Algorithm 5 is $\mathcal{O}(NM^2)$, where $N \gg M$ is the size of the training set. Hence, when the final RMR list can be successfully validated, the algorithm can choose the riskiest model as the reference model and measure the model risk accurately. When the average risk of the entire training set as the validation criterion, some training examples might exhibit greater risk in a safe model than in a risk one. Nevertheless, as shown in the Section 5.4, such exceptional cases only account for a small fraction.

5.4 Empirical Evaluation

This section evaluates the effectiveness of RMR to measure the membership risk of models, and then present a case study on its application in risky example removal. We attempt to answer the following questions:

Q1: Is RMR effective in measuring membership privacy risk for a ML model?

Q2: How does the reference model impact when evaluating a set of trained models?

Q3: What are the factors that may impact the performance of RMR?

Q4: How is RMR employed to pinpoint high-risk examples for an individual model?

5.4.1 Experimental Setup

Datasets. We adopt several datasets for classification tasks, namely, Purchase[1], Location[2], FMNIST, and STL10. Purchase and Location encompass tabular information, while the FMNIST and STL10 datasets consist of image samples. The Location dataset comprises check-in records, which include 4,000 data samples with 446 binary features in our experiments. The Purchase dataset contains shopping records, which consist of 39,464 data samples with 600 binary features. FMNIST is a collection of 70,000 grayscale images categorized into ten classes, with an equal number of images per class. STL10 consists of ten classes, each comprising 1,300 color images. The details are shown in Table 5.1.

Following previous work [55], we evenly split each dataset into four equal disjoint parts: target training set, target test set, shadow training set, and shadow test set. Similar to [55, 149, 163], the number of member examples is the same as that of non-members, so a random guess results in 50% membership inference attack accuracy.

Off-the-shelf Attacks. To evaluate the effectiveness of our proposed risk measurement, we first need to determine the ground-truth membership risk for each target

[1] `https://www.kaggle.com/c/acquire-valued-shoppers-challenge`

[2] `https://sites.google.com/site/yangdingqi/home/foursquare-dataset`

Table 5.1: Dataset Statistics.

dataset	type	network	#class	#feature	#size	M
Location	Tabular	MLP	30	446	4,000	80
Purchase	Tabular	MLP	100	600	39,464	70
STL10	Image	CNN	10	27648	13,000	44
FMNIST	Image	CNN	10	1024	70,000	33

model. However, since obtaining the ground-truth risk is not feasible, we adopt an alternative approach: we consider the highest attack performance as the *real membership risk* of the model, based on state-of-the-art metric-based and NN-based membership inference attacks. We consider the following six existing attacks:

- ***M-CR***: [161] relies on prediction correctness for membership inference. An input sample is inferred as a member if it is correctly predicted; otherwise, it is a non-member.
- ***M-CF***: [161, 170] infer membership by comparing the prediction confidence of a sample with a threshold. If the confidence exceeds the threshold, the sample is classified as a member; otherwise, as a non-member.
- ***M-ET***: [161] uses the entropy of confidence scores to infer membership by comparing it against a threshold.
- ***M-ME***: [161] relies on modified prediction entropy, incorporating the ground truth label and setting distinct thresholds per class. A sample is classified as a member if its modified entropy is below the threshold; otherwise, it is not.
- ***N-SM***: [149] uses a binary classifier to distinguish a target model's behavior on its training members from non-members. Following previous works [55, 149], the attack model is a 3-layer perceptron, taking the full confidence scores from the target model and the ground-truth labels as inputs.
- ***N-LiRA***: [11] adopts a statistical method to determine if a target sample was in the training set. It trains shadow models with query examples (IN) and without query examples (OUT), fits the scaled logits to two Gaussian distributions, and uses a likelihood-ratio test to infer membership. We train 16 shadow models (8

IN and 8 OUT) and evaluate the membership risk of a given model using 2,000 query examples (1,000 for the Location dataset).

In our experiments, all thresholds are optimized on the shadow datasets to achieve the best attack performance.

Training Setup. We implement different network architectures on various datasets, with the specific architecture type used for each dataset presented in Table 5.1. The MLP is trained on tabular datasets with three hidden layers containing 1024, 512, and 256 neurons. The CNN, designed for image data, consists of three convolutional layers and two fully connected layers, with ReLU as the activation function. In the ablation study, we also evaluate our method using AlexNet and DenseNet. Each model is trained using the SGD optimizer with a learning rate of 10^{-2}, momentum of 0.9, training epoch of 100, and a batch size of 128.

Before evaluating the effectiveness of various measures, we first train M target models with different configurations, including ℓ_2-regularization, dropout, and varying network architectures. We then perform five off-the-shelf attacks to obtain the real membership risk.

Evaluation Metrics. Following existing works [149, 161, 164, 178, 180], we use Membership Accuracy (MAcc) to evaluate attack performance. MAcc is defined as the ratio of successful attacks to the total number of attacks. Given an even dataset split, we sample an example from either the training or test set with a probability of 0.5.

To evaluate the effectiveness of RMR, we use two correlation metrics: Pearson Correlation Coefficient (PCC, denoted by ρ) and Kendall Rank Correlation Coefficient (KRC, denoted by τ) [181]. PCC measures the linear relationship between two distributions, indicating how well RMR (denoted as Y) aligns with the actual membership risks (denoted as X) on the target models, expressed as $\rho = \frac{\mathrm{cov(X,Y)}}{\sigma_X \sigma_Y}$ where cov is the covariance, and σ_X (or σ_Y) is the standard deviation of X (or Y). And KRC

measures the ordinal association between two risk ranks,

$$\tau = \frac{2}{n\,(n+1)} \sum_{i<j} \operatorname{sgn}\,(x_i - x_j) \operatorname{sgn}\,(y_i - y_j),$$

where $x_i \in X$, $y_i \in Y$, n is the size of X or Y, and $\operatorname{sgn}\,(\cdot)$ is the sign function.

Comparison Baselines. We use two risk measurement approaches as baselines. The first, referred to as *overfit* [149, 163], is a commonly employed method for evaluating the membership risk of a machine learning model. It is determined by the difference in classification accuracy between the training and test sets. The second approach, SHAPr [162], is an attack-agnostic and detailed method that approximates leave-one-out computations. For a given instance x, SHAPr extracts the feature embedding from the target model, then applies a k-NN classifier to compute x's contribution to the training set. The highest SHAPr value across all training examples is used to quantify the model's risk.

Experiment Environment. Our experiments are implemented in Pytorch and performed on an NVIDIA RTX-3090 server with Ubuntu operating system. All experiments are repeated five times and report the average results.

5.4.2 Experimental Results

To address Q1, we validate whether the relative risk measure RMR is effective in measuring the membership privacy risk of machine learning models.

Table 5.2 presents the performance of RMR and two baseline methods against real membership risks for target models. Overall, RMR consistently outperforms the baselines in both PCC and KRC metrics, with particularly strong advantanges in the image domain. For instance, on the STL10 dataset, RMR achieves a PCC of 0.88, while *overfit* scores only 0.2086, and SHAPr results in -0.0451. Fig. 5.2 further visualizes the distributions of attack performance and risk measures across four datasets. We observe strong correlations between the RMR measurement and the

real membership accuracy, especially in the tabular datasets, where the PCCs are 0.9752 and 0.9030 for the Location and Purchase datasets, respectively. In contrast, SHAPr shows very low alignment with the real membership risk, and overfit fails to capture it in the STL10 dataset. These results highlight that RMR serves as an effective indicator of the performance of membership inference attacks.

Table 5.2: Risk measure performance comparison.

Metric	Method	Location	Purchase	STL10	FMNIST
PCC	overfit	0.9455	0.8589	0.2086	0.7156
	SHAPr	-0.0144	0.1137	-0.0451	0.1840
	RMR	**0.9752**	**0.9030**	**0.8800**	**0.8609**
KRC	overfit	0.9038	0.8094	0.5378	0.7014
	SHAPr	0.4845	0.4572	0.4461	0.4779
	RMR	**0.9286**	**0.9302**	**0.8502**	**0.7102**

(a) (b) (c) (d)

Figure 5.2: Comparison of various risk measures with membership accuracy.

Since we use the average RMR of all training examples as the model's risk in Eq.(5.8), individual example risk variations may occur. In such a case, a model with higher risk might leak less membership privacy than a model deemed safer for certain examples. Table 5.3 shows the percentage of such violations across all datasets when the riskiest model is selected as the reference model from a set of target models. Although not all training examples may strictly adhere to Corollary 3, these violations are infrequent, remaining below 10%. Furthermore, Table 5.4 presents the improvement rate of RMR when we remove these violating examples. The performance change is minimal, indicating that these violations do not significantly impact the model risk results, thereby justifying the use of the average RMR as the model risk measure.

Table 5.3: Percentage of individual example risk violations.

	Location	Purchase	STL10	FMNIST
Viol.per	3.4%	2.6%	9.8%	3.6%

Table 5.4: RMR improvement rate by removing violating examples.

Metric	Location	Purchase	STL10	FMNIST
PCC	+0.8%	+0.1%	-0.1%	-0.05%
KRC	+0.2%	+0.3%	-0.2%	+0.5%

The above results demonstrate the effectiveness of RMR in measuring the membership risk across multiple models. Additionally, it proves useful for monitoring the risk changes of an individual model throughout its training process. We prepare a reference model trained on the same training dataset without any defense mechanism and then observe the attack performance on the target model during its training process. Fig. 5.3 shows the change in both the RMR and the real membership risk of a single model across different epochs. We observe nearly perfect alignment between the two metrics during the training epochs, particularly in the Location and Purchase datasets. This result encourages us to use RMR as an indicator for early stopping when considering privacy leakage.

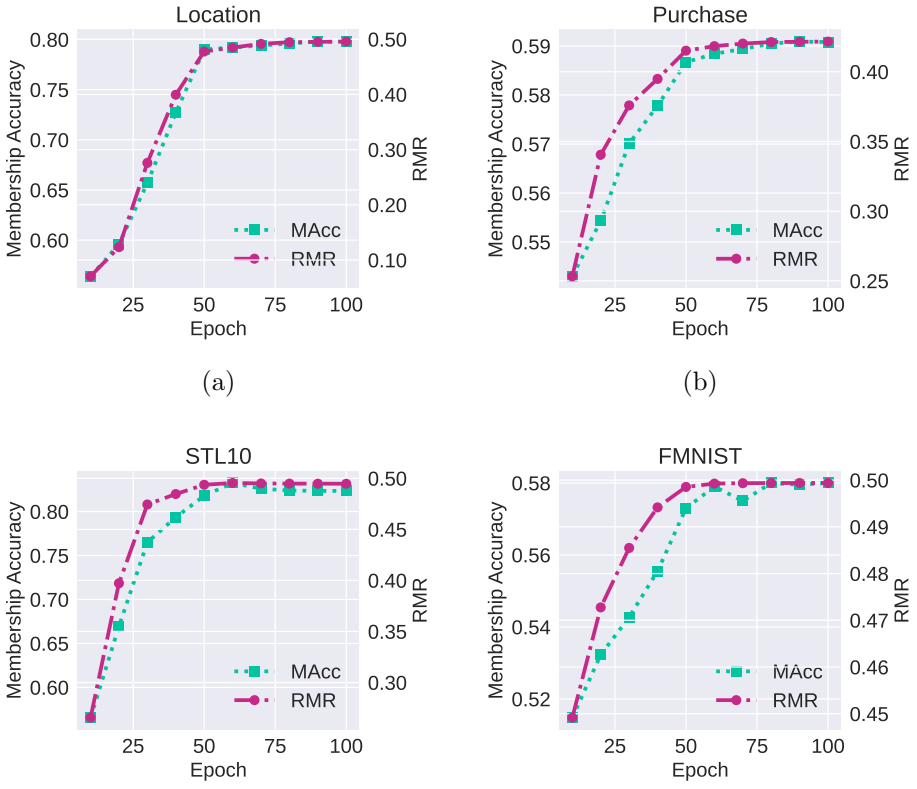


Figure 5.3: RMR performance across different training epochs.

An effective membership privacy metric must also be efficient, which makes the computational cost of various measures an important consideration. To this end, we present the average time cost required to assess the membership risk of an individual model within the candidate pool in Table 5.5. The results indicate that RMR incurs comparable or lower costs than baselines, while offering superior performance in measuring membership risk.

Table 5.5: Time cost comparison.

Method	Location	Purchase	STL10	FMNIST
overfit	1x(0.74s)	1x(1.46s)	1x(3.0s)	1x(5.87s)
SHAPr	9.5x	18.5x	8.0x	4.0x
RMR	0.5x	1.4x	0.6x	0.7x

Time results are normalized based on the overfit approach.

5.4.3 Ablation Study

In this subsection, to solve Q2 and Q3, we examine possible factors contributing to the success of RMR in measuring membership risk, including the selection of the reference model, the type of network used, the training example ratio, and the number of target models.

Table 5.6: Impact of reference model selection on PCC.

	Location	Purchase	STL10	FMNIST
Iterative	**0.9752**	**0.9030**	**0.8800**	**0.8609**
Random	0.9619	0.8698	0.8636	0.7637

First, we discuss the impact of the reference model on RMR. As highlighted in Section 5.3.3, the choice of reference model is crucial for the accuracy of RMR. To address this, we propose an iterative approach in Algorithm 5 for selecting the highest-risk model as the reference model. Table 5.6 compares this iterative strategy with a random reference model selection approach in terms of PCC. Our iterative method shows a clear advantage across nearly all datasets.

Table 5.7: RMR performance on image datasets under various model architectures.

Dataset	Metric	DenseNet	AlexNet	MixNet*
STL10	PCC	0.9631	0.9201	0.9617
	KRC	0.9510	0.9091	0.8896
FMNIST	PCC	0.8817	0.8924	0.8116
	KRC	0.9412	0.8030	0.7920

* MixNet includes CNN, AlexNet and DenseNet.

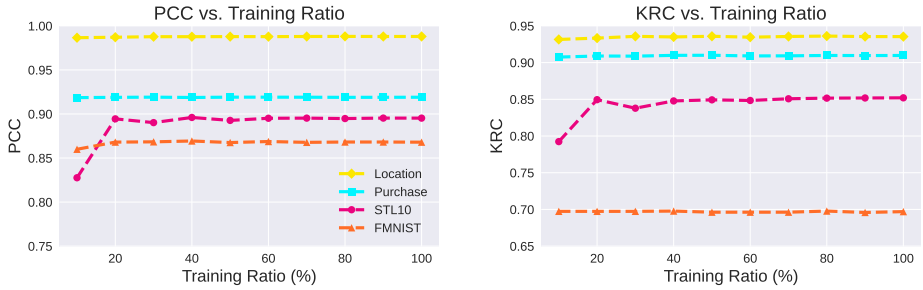
The previous experiments were conducted using simple architectures, such as MLP and CNN. Now, we explore the effectiveness of RMR by applying it to more complex network types and examining its performance across heterogeneous target models. We incorporate two widely adopted architectures, AlexNet and DenseNet. For each architecture type, we prepare 18 target models with varying dropout and regularization settings, and then evaluate their membership risks. Table 5.7 presents the

performance of RMR across various network types on image datasets. On the one hand, RMR demonstrates strong effectiveness on complex models, achieving high correlation in both metrics. Furthermore, we observe that the estimation task on models with these two architectures achieves higher performance compared to a simple CNN. This discrepancy is not due to differences in network types but rather the difficulty of the evaluation task itself. Specifically, tasks are relatively easier when the differences between models are more pronounced. For instance, on the STL10 dataset, the membership accuracy variance for models with DenseNet is 0.0106, whereas for CNN, it is only 0.0004.

Furthermore, we also explore the effectiveness of RMR in scenarios where different types of network architecture are used. As shown in Table 5.7, for a mixture of heterogeneous models, MixNet, RMR experiences a slight decrease in both metrics compared to the best-performing model among the three. However, it still maintains a high correlation with the real membership accuracy, demonstrating that RMR is a robust risk indicator across different architectures. Next, we discuss the impact of training examples used for RMR computation in Eq. (5.8). We vary the ratio of examples by randomly selecting them from the training set and evaluate the predicted membership risk using RMR. Fig. 5.4 shows the performance of RMR across different training example ratios. We observe that the ratio of training examples does not significantly affect the effectiveness of RMR once it exceeds 20%. However, for the small-scale STL10 dataset, fewer training examples are insufficient for accurately modeling the complex image data distribution.

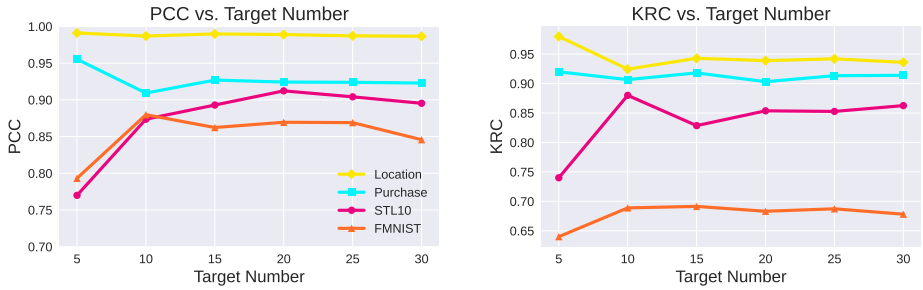
Finally, we vary the number of target models and plot the performance of RMR under different target model sizes in Fig. 5.5. While RMR remains unaffected in simple MLP networks trained on tabular datasets, it shows improvement with more target models in CNN models trained on image datasets. This difference can be attributed to the selection of the reference model. When there is a small number of model candidates trained under similar settings, the reference model may be too close

to the others in terms of membership risk, leading to more violation examples. This negatively impacts the accuracy of the estimation results. For small-scale MLPs used in Location and Purachse data, similar variations in training settings, such as dropout and ℓ_2-regularization, result in more distinct model candidates and less influenced by the number of target models. Even across different scenarios, the experimental results show that the evaluation stabilizes when the number of target models exceeds 10.



(a) (b)

Figure 5.4: Impact of training example ratio.

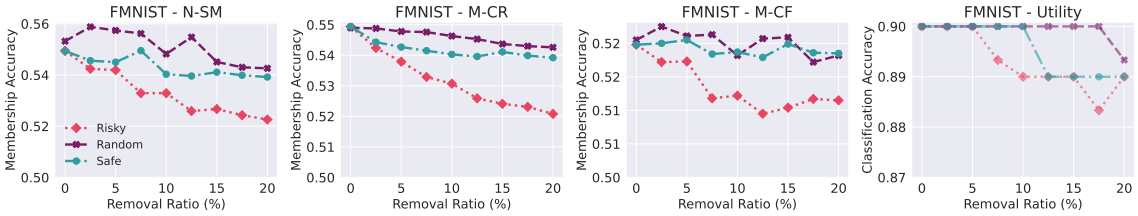


(a) (b)

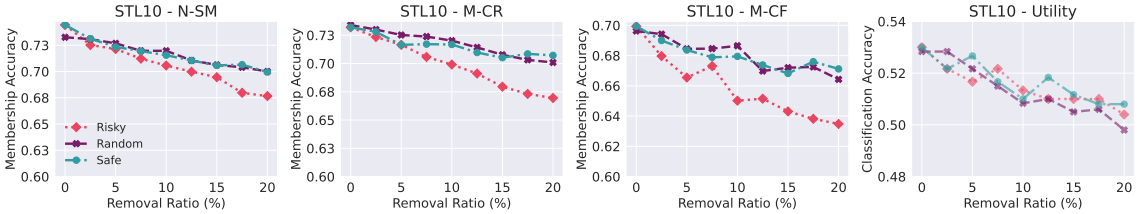
Figure 5.5: Impact of target model number.

5.4.4 Case Study: Risky Example Removal

The experimental results above demonstrate that RMR is an effective model-level risk measure. To further evaluate its effectiveness as a sample-level risk measure, we



(a) FMNIST



(b) STL10

Figure 5.6: RMR performance across different removal ratios.

present a case study that utilizes RMR to reduce model risk by identifying and removing easy-to-attack examples in image datasets [165]. We first prepare a target model protected by defense mechanisms, such as dropout and ℓ_2-regularization, and then construct a reference model trained without any countermeasures. For this target model, we rank all training examples in descending order of their RMR values computed by Eq.(5.7), as determined by the predefined reference model, and subsequently remove the top-ranked examples. In addition to the *risky-example-removal* strategy, we implement two additional strategies for comparison: *random-* and *safe-example-removal* (the inverse of risky-example removal). Finally, we train a new model from scratch using the modified dataset, excluding the removed examples, and evaluate its membership inference accuracy.

Fig. 5.6 illustrates the performance of various attacks under different removal strategies with respect to the ratio of removed examples. We observe that the risky-example-removal strategy significantly reduces the accuracy of all membership inference attacks by up to 6%, whereas the other two strategies have a smaller and comparable impact. However, the drop is still not proportional to the ratio of re-

moved examples. For example, removing 20% of risky examples in STL10 can ideally achieve a 10% membership accuracy drop (i.e., a 100% accuracy down to a 50% random guess accuracy), but the actual drop is only 6%. There are two possible explanations for this. First, achieving 100% accuracy is not possible with current non-optimal membership inference attacks, even on the riskiest examples. Second, as recently noted by [182], the onion effect may be at play—after the riskiest examples are removed, the remaining examples (particularly those with the second-highest risk) may experience an increase in their membership risk.

Furthermore, we investigate the effect of different removal strategies on model utility and present their classification performance on the STL10 and FMNIST datasets in Fig. 5.6. We find that STL10 is more sensitive to the removal strategy due to its smaller training dataset compared to the FMNIST dataset. However, for both datasets, model utility is more negatively impacted when more than 10% of the risky training examples are removed. This is because examples near the decision boundary are more easily identified and are typically the most vulnerable ones [183, 184, 185]. In conclusion, there is a trade-off between membership risk and model utility when using RMR to remove risky training examples.

5.5 Chapter Summary

Since MIAs pose a fundamental threat to machine learning models trained on private data and serve as a basis for more advanced attacks, measuring the membership risk of foundation models is a critical issue. This chapter presents a relative membership risk measure, RMR, that captures both record-level and model-level membership risk. Using a reference model, RMR efficiently quantifies membership risk for both training examples and machine learning models. We also propose an iterative strategy for selecting the reference model when assessing multiple target models. The experimental results demonstrate that RMR makes accurate predictions that closely align with

real membership risks across various datasets and model architectures. For future work, we aim to extend RMR to address other privacy risks associated with machine learning models, such as property inference and model inversion attacks.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This thesis explores three distinct methodologies targeting MIAs and PIAs, with a focus on training data privacy leakage risks in ML models. Our proposed approaches range from improving existing inference attacks, to exploring new attack surfaces, and to quantifying the privacy risk of trained models and their data.

In the first methodology, we address the efficiency bottleneck of shadow model construction by proposing an attack-agnostic solution to address this issue. We present a novel shadow pool training framework SHAPOOL, which constructs multiple shared models and trains them jointly within a single process. In particular, we leverage the Mixture-of-Experts mechanism as the shadow pool to interconnect individual models, enabling them to share some sub-networks and thereby improving efficiency. To ensure the shared models closely resemble independent models and serve as effective substitutes, we introduce three novel modules: path-choice routing, pathway regularization, and pathway alignment. These modules guarantee random data allocation for pathway learning, promote diversity among shared models, and maintain consistency with target models. We evaluate SHAPOOL in the context of various membership

inference attacks and show that it significantly reduces the computational cost of shadow model construction while maintaining comparable attack performance.

In the second methodology, we reveal the risk of property leakage in VFL by introducing ProVFL, a novel framework designed to perform property inference with low estimation errors. ProVFL is based on the empirical observation that the L_p-norm distribution of intermediate results unintentionally leaks private property information to potential adversarial parties. To address this issue, we first design a distribution comparison method for property inference and analyze its effectiveness from a theoretical standpoint. We further propose correlation augmentation to enhance the leakage of property information. Our experimental results demonstrate that ProVFL performs effectively across various target properties, highlighting how shared intermediate results in VFL can easily expose private information. Additionally, we explore existing defenses against such leakage and propose a simple yet effective mitigation approach for our attacks and provide valuable insights into the protection of private statistical information in VFL.

Finally, in the third methodology, we quantify the membership risk of foundation models which captures both record-level and model-level membership risk from a comparative standpoint. RMR calculates the difference in prediction loss for training examples relative to a predefined reference model, enabling risk comparison across models without needing to delve into details like training strategy, architecture, or data distribution. We also explore the selection of the reference model and show that using a high-risk reference model enhances the accuracy of the RMR measure. To identify the most vulnerable reference model, we propose an efficient iterative algorithm that selects the optimal model from a set of candidates. Through extensive empirical evaluations on various datasets and network architectures, we demonstrate that RMR is an accurate and efficient tool for measuring the membership privacy risk of both individual training examples and the overall ML model.

In summary, we characterize ML privacy risks from both practical attacks and inher-

ent risk perspectives. The first two works study inference attacks in practice, focusing on attack efficiency and new attack surfaces, while the third work quantifies privacy risk at a more fundamental level. Together, they form a holistic view of ML privacy that bridges empirical and theoretical attack analysis across both centralized and distributed settings.

6.2 Future Work

We conclude the thesis by outlining future directions, focusing on two promising avenues:

(1) Exploration of Systemic Privacy Risks. Future efforts can investigate systemic privacy risks by jointly considering multiple inference attacks rather than examining them in isolation. Prior research has demonstrated that one type of attack can amplify the effectiveness of another. For example, poisoning attacks can strengthens MIAs, highlighting the interdependence between different threats [186]. In real-world scenarios, adversaries are unlikely to rely on a single, isolated technique but may strategically combine attacks to maximize their success. This suggests that developing defenses tailored to individual attacks may provide only limited protection, leaving models vulnerable when exposed to compound or sequential attack strategies. A more holistic approach to studying and mitigating inference attacks is therefore critical for building resilient and practical privacy-preserving ML systems.

(2) Extensions to Large Language Models. Although our current work focuses on relatively small-scale ML models, its core principles can be extended to LLMs. For SHAPOOL, the idea of shared shadow models remains relevant, especially given the widespread use of the MoE architecture in modern LLMs; however, the extreme parameter scale makes training multiple shadow models a major challenge and is therefore often impractical in existing attacks. In contrast, ProVFL and RMR are largely

architecture-agnostic and can be more readily adapted to LLM settings. Nevertheless, because LLMs are trained on massive corpora, privacy leakage is less tied to individual samples or simple distribution estimates, and more related to memorization and exposure through generation behaviors, which calls for rethinking attack objectives and privacy risk definitions under LLM-specific training and inference paradigms.

(3) Exploration of Large Language Models. Building on the extension to large language models, future work should further investigate privacy challenges in complex and emerging model architectures, where inference attack techniques designed for conventional ML models may no longer be effective. Due to their unique architectures and training paradigms, LLMs are characterized by massive parameter scales, intricate architectures, and highly complex feature spaces, making typical approaches—such as shadow model training—computationally infeasible or unrealistic [187]. Moreover, the pretraining–fine-tuning paradigm and the strong memorization capacity of LLMs introduce new and underexplored privacy leakage mechanisms. Understanding these unique vulnerabilities, and developing tailored inference attacks and corresponding defenses, therefore represents an important and timely direction for future research in ML privacy.

References

[1] Hafsa Habehh and Suril Gohel. Machine learning in healthcare. *Current genomics*, 22(4):291–300, 2021.

[2] Matthew F Dixon, Igor Halperin, Paul Bilokon, et al. *Machine learning in finance*, volume 1170. Springer, 2020.

[3] Mrinal R Bachute and Javed M Subhedar. Autonomous driving architectures: insights of machine learning and deep learning algorithms. *Machine Learning with Applications*, 6:100164, 2021.

[4] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3):50–60, 2020.

[5] Inc. Hugging Face. Hugging face hub, 2016.

[6] Amazon Web Services, Inc. Machine learning (ML) - Amazon Web Services (AWS). `https://aws.amazon.com/ai/machine-learning/`, 2024.

[7] Congzheng Song, Thomas Ristenpart, and Vitaly Shmatikov. Machine learning models that remember too much. In *Proceedings of the 2017 ACM SIGSAC Conference on computer and communications security*, pages 587–601, 2017.

[8] Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. Ml-leaks: Model and data independent membership in-

ference attacks and defenses on machine learning models. In *Proceedings of the 26th Annual Network and Distributed System Security Symposium (NDSS)*, 2019.

[9] Giuseppe Ateniese, Luigi V. Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks*, 10:137–150, 2015.

[10] Karan Ganju, Qi Wang, Wei Yang, Carl A. Gunter, and Nikita Borisov. Property inference attacks on fully connected neural networks using permutation invariant representations. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 619–633, 2018.

[11] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1897–1914. IEEE, 2022.

[12] Lauren Watson, Chuan Guo, Graham Cormode, and Alexandre Sablayrolles. On the importance of difficulty calibration in membership inference attacks. In *Proceedings of the Tenth International Conference on Learning Representations (ICLR)*, 2022.

[13] Saeed Mahloujifar, Esha Ghosh, and Melissa Chase. Property inference from poisoning. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, pages 1120–1137. IEEE, 2022.

[14] Harsh Chaudhari, John Abascal, Alina Oprea, Matthew Jagielski, Florian Tramer, and Jonathan Ullman. Snap: Efficient extraction of private properties with poisoning. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, pages 400–417. IEEE, 2023.

[15] Yang Liu, Yan Kang, Tianyuan Zou, Yanhong Pu, Yuanqin He, Xiaozhou Ye, Ye Ouyang, Ya-Qin Zhang, and Qiang Yang. Vertical federated learning: Concepts, advances, and challenges. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[16] Chong Fu, Xuhong Zhang, Shouling Ji, Jinyin Chen, Jingzheng Wu, Shanqing Guo, Jun Zhou, Alex X Liu, and Ting Wang. Label inference attacks against vertical federated learning. In *Proc. USENIX Security Symp. (USENIX Security)*, pages 1397–1414, 2022.

[17] Xinjian Luo, Yuncheng Wu, Xiaokui Xiao, and Beng Chin Ooi. Feature inference attack on model predictions in vertical federated learning. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*, pages 181–192. IEEE, 2021.

[18] Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. A survey on mixture of experts. *arXiv preprint arXiv:2407.06204*, 2024.

[19] Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[20] Georgios Drainakis, Konstantinos V Katsaros, Panagiotis Pantazopoulos, Vasilis Sourlas, and Angelos Amditis. Federated vs. centralized machine learning under privacy-elastic users: A comparative analysis. In *2020 IEEE 19th International Symposium on Network Computing and Applications (NCA)*, pages 1–8. IEEE, 2020.

[21] Vladimir Vapnik. Principles of risk minimization for learning theory. *Advances in neural information processing systems*, 4, 1991.

[22] David Saad. Online algorithms and stochastic approximations. *Online Learning*, 5(3):6, 1998.

[23] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations (ICLR)*. ICLR, 2015.

[25] Jie Xu, Benjamin S Glicksberg, Chang Su, Peter Walker, Jiang Bian, and Fei Wang. Federated learning for healthcare informatics. *Journal of Healthcare Informatics Research*, 5:1–19, 2021.

[26] Dinh C Nguyen, Quoc-Viet Pham, Pubudu N Pathirana, Ming Ding, Aruna Seneviratne, Zihuai Lin, Octavia Dobre, and Won-Joo Hwang. Federated learning for smart healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(3):1–37, 2022.

[27] Latif U Khan, Walid Saad, Zhu Han, Ekram Hossain, and Choong Seon Hong. Federated learning for internet of things: Recent advances, taxonomy, and open challenges. *IEEE Communications Surveys & Tutorials*, 23(3):1759–1799, 2021.

[28] Dinh C Nguyen, Ming Ding, Pubudu N Pathirana, Aruna Seneviratne, Jun Li, and H Vincent Poor. Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 23(3):1622–1658, 2021.

[29] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

[30] Yutao Huang, Lingyang Chu, Zirui Zhou, Lanjun Wang, Jiangchuan Liu, Jian Pei, and Yong Zhang. Personalized cross-silo federated learning on non-iid data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7865–7873, 2021.

[31] Othmane Marfoq, Chuan Xu, Giovanni Neglia, and Richard Vidal. Throughput-optimal topology design for cross-silo federated learning. *Advances in Neural Information Processing Systems*, 33:19478–19487, 2020.

[32] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 54, pages 1273–1282, 2017.

[33] Qinbin Li, Zeyi Wen, and Bingsheng He. Practical federated gradient boosting decision trees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4642–4649, 2020.

[34] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

[35] Junpeng Zhang, Mengqian Li, Shuiguang Zeng, Bin Xie, and Dongmei Zhao. A survey on security and privacy threats to federated learning. In *2021 International Conference on Networking and Network Applications (NaNA)*, pages 319–326. IEEE, 2021.

[36] Yang Liu, Yan Kang, Chaoping Xing, Tianjian Chen, and Qiang Yang. A secure federated transfer learning framework. *IEEE Intelligent Systems*, 35(4):70–82, 2020.

[37] Dashan Gao, Yang Liu, Anbu Huang, Ce Ju, Han Yu, and Qiang Yang. Privacy-

preserving heterogeneous federated transfer learning. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 2552–2559. IEEE, 2019.

[38] Shreya Sharma, Chaoping Xing, Yang Liu, and Yan Kang. Secure and efficient federated transfer learning. In *2019 IEEE international conference on big data (Big Data)*, pages 2569–2576. IEEE, 2019.

[39] Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S Yu, and Xuyun Zhang. Membership inference attacks on machine learning: A survey. *ACM Computing Surveys (CSUR)*, 2021.

[40] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction {APIs}. In *25th USENIX security symposium (USENIX Security 16)*, pages 601–618, 2016.

[41] Binghui Wang and Neil Zhenqiang Gong. Stealing hyperparameters in machine learning. In *2018 IEEE symposium on security and privacy (SP)*, pages 36–52. IEEE, 2018.

[42] Yuheng Zhang, Ruoxi Jia, Hengzhi Pei, Wenxiao Wang, Bo Li, and Dawn Song. The secret revealer: Generative model-inversion attacks against deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 253–261. IEEE, 2020.

[43] 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)*, pages 268–282. IEEE, 2018.

[44] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1322–1333, 2015.

[45] Christopher A Choquette-Choo, Florian Tramer, Nicholas Carlini, and Nicolas Papernot. Label-only membership inference attacks. In *International conference on machine learning*, pages 1964–1974. PMLR, 2021.

[46] Wanrong Zhang, Shruti Tople, and Olga Ohrimenko. Leakage of dataset properties in {Multi-Party} machine learning. In *30th USENIX security symposium (USENIX Security 21)*, pages 2687–2704, 2021.

[47] Anshuman Suri and David Evans. Formalizing and estimating distribution inference risks. *Proceedings on Privacy Enhancing Technologies (PET)*, 2022(4):528–551, 2022.

[48] Sajjad Zarifzadeh, Philippe Liu, and Reza Shokri. Low-cost high-power membership inference attacks. In *Forty-first International Conference on Machine Learning*, 2024.

[49] Martin Bertran, Shuai Tang, Aaron Roth, Michael Kearns, Jamie H Morgenstern, and Steven Z Wu. Scalable membership inference attacks via quantile regression. *Advances in Neural Information Processing Systems*, 36, 2024.

[50] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1930–1939, 2018.

[51] Min Lin, Jie Fu, and Yoshua Bengio. Conditional computation for continual learning. *arXiv preprint arXiv:1906.06635*, 2019.

[52] Joan Puigcerver, Carlos Riquelme, Basil Mustafa, Cedric Renggli, André Susano Pinto, Sylvain Gelly, Daniel Keysers, and Neil Houlsby. Scalable transfer learning with expert models. *arXiv preprint arXiv:2009.13239*, 2020.

[53] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang. When machine unlearning jeopardizes privacy. In *Proceedings of the 2021 ACM SIGSAC conference on computer and communications security*, pages 896–911, 2021.

[54] Zonghao Huang, Neil Zhenqiang Gong, and Michael K Reiter. A general framework for data-use auditing of ml models. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 1300–1314, 2024.

[55] Yugeng Liu, Rui Wen, Xinlei He, Ahmed Salem, Zhikun Zhang, Michael Backes, Emiliano De Cristofaro, Mario Fritz, and Yang Zhang. Ml-doctor: Holistic risk assessment of inference attacks against machine learning models. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 4525–4542, 2022.

[56] Borja Balle, Giovanni Cherubin, and Jamie Hayes. Reconstructing training data with informed adversaries. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1138–1156. IEEE, 2022.

[57] Zhibo Wang, Yuting Huang, Mengkai Song, Libing Wu, Feng Xue, and Kui Ren. Poisoning-assisted property inference attack against federated learning. *IEEE Transactions on Dependable and Secure Computing*, 20(4):3328–3340, 2022.

[58] Ahmed Salem, Apratim Bhattacharya, Michael Backes, Mario Fritz, and Yang Zhang. {Updates-Leak}: Data set inference and reconstruction attacks in online learning. In *29th USENIX security symposium (USENIX Security 20)*, pages 1291–1308, 2020.

[59] Hao Li, Zheng Li, Siyuan Wu, Chengrui Hu, Yutong Ye, Min Zhang, Dengguo Feng, and Yang Zhang. Seqmia: Sequential-metric based membership inference attack. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 3496–3510, 2024.

[60] Zhiwen Fan, Rishov Sarkar, Ziyu Jiang, Tianlong Chen, Kai Zou, Yu Cheng, Cong Hao, Zhangyang Wang, et al. M^3vit: Mixture-of-experts vision transformer for efficient multi-task learning with model-accelerator co-design. *Advances in Neural Information Processing Systems*, 35:28441–28457, 2022.

[61] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[62] Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*, 2020.

[63] Svetlana Pavlitska, Christian Hubschneider, Lukas Struppek, and J Marius Zöllner. Sparsely-gated mixture-of-expert layers for cnn interpretability. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–10. IEEE, 2023.

[64] Lam Pham, Huy Phan, Ramaswamy Palaniappan, Alfred Mertins, and Ian McLoughlin. Cnn-moe based framework for classification of respiratory anomalies and lung disease detection. *IEEE journal of biomedical and health informatics*, 25(8):2938–2947, 2021.

[65] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34:8583–8595, 2021.

[66] Ibomoiye Domor Mienye and Yanxia Sun. A survey of ensemble learning: Concepts, algorithms, applications, and prospects. *IEEE Access*, 10:99129–99149, 2022.

[67] Xumeng Gong, Cheng Yang, and Chuan Shi. Ma-gcl: Model augmentation tricks for graph contrastive learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 4284–4292, 2023.

[68] Yingwei Li, Song Bai, Yuyin Zhou, Cihang Xie, Zhishuai Zhang, and Alan Yuille. Learning transferable adversarial examples via ghost networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 11458–11465, 2020.

[69] Han Wu, Guanyan Ou, Weibin Wu, and Zibin Zheng. Improving transferable targeted adversarial attacks with model self-enhancement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24615–24624, 2024.

[70] Yuyang Long, Qilong Zhang, Boheng Zeng, Lianli Gao, Xianglong Liu, Jian Zhang, and Jingkuan Song. Frequency domain model augmentation for adversarial attack. In *European conference on computer vision*, pages 549–566. Springer, 2022.

[71] Liwei Song and Prateek Mittal. Systematic evaluation of privacy risks of machine learning models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2615–2632, 2021.

[72] Damai Dai, Li Dong, Shuming Ma, Bo Zheng, Zhifang Sui, Baobao Chang, and Furu Wei. Stablemoe: Stable routing strategy for mixture of experts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 7085–7095, 2022.

[73] Simiao Zuo, Xiaodong Liu, Jian Jiao, Young Jin Kim, Hany Hassan, Ruofei Zhang, Tuo Zhao, and Jianfeng Gao. Taming sparsely activated transformer with stochastic experts. *arXiv preprint arXiv:2110.04260*, 2021.

[74] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6707–6717, 2020.

[75] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[76] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[77] Andrew Brock. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[78] Yishi Li, Kunran Xu, Rui Lai, and Lin Gu. Towards an effective orthogonal dictionary convolution strategy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1473–1481, 2022.

[79] Jiayun Wang, Yubei Chen, Rudrasis Chakraborty, and Stella X Yu. Orthogonal convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11505–11515, 2020.

[80] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.

[81] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *ACL 2018-56th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 328–339. Association for Computational Linguistics, 2018.

[82] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[83] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[84] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[85] Sergey Zagoruyko. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[86] Florian Tramèr, Reza Shokri, Ayrton San Joaquin, Hoang Le, Matthew Jagielski, Sanghyun Hong, and Nicholas Carlini. Truth serum: Poisoning machine learning models to reveal their secrets. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 2779–2792, 2022.

[87] Zitao Chen and Karthik Pattabiraman. A method to facilitate membership inference attacks in deep learning models. *arXiv preprint arXiv:2407.01919*, 2024.

[88] Yuxin Wen, Arpit Bansal, Hamid Kazemi, Eitan Borgnia, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Canary in a coalmine: Better membership inference with ensembled adversarial queries. *arXiv preprint arXiv:2210.10750*, 2022.

[89] Xiao Jin, Pin-Yu Chen, Chia-Yi Hsu, Chia-Mu Yu, and Tianyi Chen. Cafe: Catastrophic data leakage in vertical federated learning. In *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, volume 34, pages 994–1006, 2021.

[90] Peng Chen, Xin Du, Zhihui Lu, Jie Wu, and Patrick C. K. Hung. Evfl: An explainable vertical federated learning for data-oriented artificial intelligence systems. *Journal of Systems Architecture*, 126:102474, 2022.

[91] JianFei Zhang and YuChen Jiang. A vertical federation recommendation method based on clustering and latent factor model. In *Proceedings of the International Conference on Electronic Information Engineering and Computer Science (EIECS)*, pages 362–366. IEEE, 2021.

[92] Tianyi Chen, Xiao Jin, Yuejiao Sun, and Wotao Yin. Vafl: a method of vertical asynchronous federated learning. *arXiv preprint arXiv:2007.06081*, 2020.

[93] Yang Liu, Yan Kang, Tianyuan Zou, Yanhong Pu, Yuanqin He, Xiaozhou Ye, Ye Ouyang, Ya-Qin Zhang, and Qiang Yang. Vertical federated learning: Concepts, advances, and challenges. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–20, 2024.

[94] Peng Ye, Zhifeng Jiang, Wei Wang, Bo Li, and Baochun Li. Feature reconstruction attacks and countermeasures of dnn training in vertical federated learning. *arXiv preprint arXiv:2210.06771*, 2022.

[95] Yuzheng Hu, Tianle Cai, Jinyong Shan, Shange Tang, Chaochao Cai, Ethan Song, Bo Li, and Dawn Song. Is vertical logistic regression privacy-preserving? a comprehensive privacy analysis and beyond. *arXiv preprint arXiv:2207.09087*, 2022.

[96] Chaohao Fu, Hongbin Chen, and Na Ruan. Privacy for free: Spy attack in vertical federated learning by both active and passive parties. *IEEE Transactions on Information Forensics and Security*, 20:2550–2563, 2025.

[97] Oscar Li, Jiankai Sun, Xin Yang, Weihao Gao, Hongyi Zhang, Junyuan Xie, Virginia Smith, and Chong Wang. Label leakage and protection in two-party split learning. *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2022.

[98] Jiankai Sun, Xin Yang, Yuanshun Yao, and Chong Wang. Label leakage and protection from forward embedding in vertical federated learning. *arXiv preprint arXiv:2203.01451*, 2022.

[99] Sanjay Kariyappa and Moinuddin K. Qureshi. Exploit: Extracting private labels in split learning. In *Proceedings of the IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pages 165–175. IEEE, 2023.

[100] Xiuling Wang and Wendy Hui Wang. Group property inference attacks against graph neural networks. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 2871–2884, 2022.

[101] Timothy Castiglia, Shiqiang Wang, and Stacy Patterson. Flexible vertical federated learning with heterogeneous parties. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[102] Wanrong Zhang, Shruti Tople, and Olga Ohrimenko. Leakage of dataset properties in multi-party machine learning. In *Proceedings of the USENIX Security Symposium (USENIX Security)*, pages 2687–2704, 2021.

[103] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, pages 691–706. IEEE, 2019.

[104] Congzheng Song and Vitaly Shmatikov. Overlearning reveals sensitive attributes. In *Proceedings of the International Conference on Learning Representations (ICLR)*. ICLR, 2020.

[105] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *Proceedings of the USENIX Security Symposium (USENIX Security)*, pages 267–284. USENIX, 2019.

[106] Li Bai, Haibo Hu, Qingqing Ye, Haoyang Li, Leixia Wang, and Jianliang Xu. Membership inference attacks and defenses in federated learning: A survey. *ACM Computing Surveys*, 57(4):1–35, 2024.

[107] Haoyang Li, Li Bai, Qingqing Ye, Haibo Hu, Yaxin Xiao, Huadi Zheng, and Jianliang Xu. A sample-level evaluation and generative framework for model inversion attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 39, pages 18287–18295. AAAI, 2025.

[108] Xinwei Zhang, Haibo Hu, Qingqing Ye, Li Bai, and Huadi Zheng. Mer-inspector: Assessing model extraction risks from an attack-agnostic perspective. In *Proceedings of the ACM Web Conference (WWW)*, pages 4300–4315. ACM, 2025.

[109] Junhao Zhou, Yufei Chen, Chao Shen, and Yang Zhang. Property inference attacks against gans. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*. NDSS, 2022.

[110] Hailong Hu and Jun Pang. Prisampler: Mitigating property inference of diffusion models. *arXiv preprint arXiv:2306.05208*, 2023.

[111] Xinjian Luo, Yangfan Jiang, Fei Wei, Yuncheng Wu, Xiaokui Xiao, and Beng Chin Ooi. Exploring privacy and fairness risks in sharing diffusion models: An adversarial perspective. *IEEE Transactions on Information Forensics and Security*, 19:8109–8124, 2024.

[112] Dario Pasquini, Giuseppe Ateniese, and Massimo Bernaschi. Unleashing the tiger: Inference attacks on split learning. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 2113–2129. ACM, 2021.

[113] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[114] Haoxin Yang, Yi Wang, and Bin Li. Individual property inference over collabo-

rative learning in deep feature space. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2022.

[115] Yunlong Mao, Zexi Xin, Zhenyu Li, Jue Hong, Qingyou Yang, and Sheng Zhong. Secure split learning against property inference, data reconstruction, and feature space hijacking attacks. In *Proceedings of the European Symposium on Research in Computer Security (ESORICS)*, volume 14347, pages 23–43, 2023.

[116] Pengyu Qiu, Xuhong Zhang, Shouling Ji, Yuwen Pu, and Ting Wang. All you need is hashing: defending against data reconstruction attack in vertical federated learning. *arXiv preprint arXiv:2212.00325*, 2022.

[117] Xue Jiang, Xuebing Zhou, and Jens Grossklags. Comprehensive analysis of privacy leakage in vertical federated learning during prediction. *Proceedings on Privacy Enhancing Technologies (PET)*, 2022(2):263–281, 2022.

[118] Zecheng He, Tianwei Zhang, and Ruby B. Lee. Model inversion attacks against collaborative inference. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, pages 148–162. ACM, 2019.

[119] Ruijia Xu, Guanbin Li, Jihan Yang, and Liang Lin. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1426–1435. IEEE, 2019.

[120] Wenjie Wang, Yufeng Shi, Shiming Chen, Qinmu Peng, Feng Zheng, and Xinge You. Norm-guided adaptive visual embedding for zero-shot sketch-based image retrieval. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1106–1112, 2021.

[121] Christian Sohler and David P. Woodruff. Subspace embeddings for the l1-norm with applications. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 755–764. ACM, 2011.

[122] Di Chen, Shanshan Zhang, Jian Yang, and Bernt Schiele. Norm-aware embedding for efficient person search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12615–12624, 2020.

[123] Yandong Guo and Lei Zhang. One-shot face recognition by promoting underrepresented classes. *arXiv preprint arXiv:1707.05574*, 2017.

[124] Yitong Wang, Dihong Gong, Zheng Zhou, Xing Ji, Hao Wang, Zhifeng Li, Wei Liu, and Tong Zhang. Orthogonal deep features decomposition for age-invariant face recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 738–753, 2018.

[125] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Proceedings of the International Conference on Learning Representations (ICLR) Workshop*, 2014.

[126] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017.

[127] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 3319–3328, 2017.

[128] Hanting Chen, Yunhe Wang, Chang Xu, Zhaohui Yang, Chuanjian Liu, Boxin Shi, Chunjing Xu, Chao Xu, and Qi Tian. Data-free learning of student networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3514–3522. IEEE, 2019.

[129] D. Dua and C. Graff. Uci machine learning repository. `http://archive.ics.uci.edu/ml`, 2017.

[130] Ron Kohavi et al. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 202–207. ACM, 1996.

[131] Linda F. Wightman. Lsac national longitudinal bar passage study. lsac research report series. 1998.

[132] Mark Vero, Mislav Balunović, Dimitar Iliev Dimitrov, and Martin Vechev. Tableak: Tabular data leakage in federated learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 35051–35083. ICML, 2023.

[133] Texas Department of State Health Services. Texas hospital inpatient discharge public use data file. `https://www.dshs.texas.gov/thcic/hospitals/Inpatientpudf.shtm`, 2006.

[134] Congzheng Song and Ananth Raghunathan. Information leakage in embedding models. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 377–390. ACM, 2020.

[135] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 785–794. ACM, 2016.

[136] Haiqin Weng, Juntao Zhang, Feng Xue, Tao Wei, Shouling Ji, and Zhiyuan Zong. Privacy leakage of real-world vertical federated learning. *arXiv preprint arXiv:2011.09290*, 2020.

[137] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yantao Zheng. Nus-wide: A real-world web image database from national university of singapore. In *Proceedings of the ACM International Conference on Image and Video Retrieval (CIVR)*, pages 1–9. ACM, 2009.

[138] Wei-Yin Loh. Classification and regression trees. *WIREs Data Mining and Knowledge Discovery*, 1(1):14–23, 2011.

[139] Hsiang-Fu Yu, Fang-Lan Huang, and Chih-Jen Lin. Dual coordinate descent methods for logistic regression and maximum entropy models. *Machine Learning*, 85:41–75, 2011.

[140] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):1–27, 2011.

[141] Khalil Jahani, Behzad Moshiri, and Babak Hossein Khalaj. A survey on data distribution challenges and solutions in vertical and horizontal federated learning. *Journal of Artificial Intelligence and Applications (JAIA)*, 1(2):55–71, 2024.

[142] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1310–1321. ACM, 2015.

[143] Leixia Wang, Qingqing Ye, Haibo Hu, Xiaofeng Meng, and Kai Huang. Ldp-purifier: Defending against poisoning attacks in local differential privacy. In *Proceedings of the International Conference on Database Systems for Advanced Applications (DASFAA)*, volume 14853 of *Lecture Notes in Computer Science*, pages 221–231. Springer, 2024.

[144] Jianping Cai, Qingqing Ye, Haibo Hu, Ximeng Liu, and Yanggeng Fu. Boosting accuracy of differentially private continuous data release for federated learning. *IEEE Transactions on Information Forensics and Security*, 19:10287–10301, 2024.

[145] Prediction API - pattern matching and machine learning. `https://cloud.google.com/prediction/`, 2016.

[146] Amazon machine learning - predictive analytics with AWS. `https://aws.amazon.com/machine-learning/`, 2016.

[147] Machine learning - Microsoft Azure. `https://azure.microsoft.com/en-us/services/machine-learning/`, 2016.

[148] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021.

[149] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

[150] Hongyang Yan, Shuhao Li, Yajie Wang, Yaoyuan Zhang, Kashif Sharif, Haibo Hu, and Yuanzhang Li. Membership inference attacks against deep learning models via logits distribution. *IEEE Transactions on Dependable and Secure Computing*, 2022.

[151] Aoting Hu, Renjie Xie, Zhigang Lu, Aiqun Hu, and Minhui Xue. Tablegan-mca: Evaluating membership collisions of gan-synthesized tabular data releasing. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 2096–2112, 2021.

[152] Jamie Hayes, Luca Melis, George Danezis, and ED Cristofaro. Logan: Evaluating information leakage of generative models using generative adversarial networks. *arXiv preprint arXiv:1705.07663*, 2017.

[153] Benjamin Hilprecht, Martin Härterich, and Daniel Bernau. Monte carlo and reconstruction membership inference attacks against generative models. *Proc. Priv. Enhancing Technol.*, pages 232–249, 2019.

[154] Umang Gupta, Dimitris Stripelis, Pradeep K Lam, Paul Thompson, José Luis Ambite, and Greg Ver Steeg. Membership inference attacks on deep regression models for neuroimaging. In *Medical Imaging with Deep Learning*, pages 228–251. PMLR, 2021.

[155] Hongbin Liu, Jinyuan Jia, Wenjie Qu, and Neil Zhenqiang Gong. Encodermi: Membership inference against pre-trained encoders in contrastive learning. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 2081–2095, 2021.

[156] Zhikun Zhang, Min Chen, Michael Backes, Yun Shen, and Yang Zhang. Inference attacks against graph neural networks. In *Proceedings of the 31th USENIX Security Symposium*, pages 1–18, 2022.

[157] Xuefei Yin, Yanming Zhu, and Jiankun Hu. A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions. *ACM Computing Surveys (CSUR)*, 54(6):131:1–131:36, 2022.

[158] Yucheng Long, Zuobin Ying, Hongyang Yan, Ranqing Fang, Xingyu Li, Yajie Wang, and Zijie Pan. Membership reconstruction attack in deep neural networks. *Information Sciences*, 634:27–41, 2023.

[159] Hyeyoung Ko, Suyeon Lee, Yoonseo Park, and Anna Choi. A survey of recommendation systems: recommendation models, techniques, and application fields. *Electronics*, 11(1):141, 2022.

[160] Minxing Zhang, Zhaochun Ren, Zihan Wang, Pengjie Ren, Zhunmin Chen, Pengfei Hu, and Yang Zhang. Membership inference attacks against recommender systems. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 864–879, 2021.

[161] Liwei Song and Prateek Mittal. Systematic evaluation of privacy risks of ma-

chine learning models. In *30th USENIX Security Symposium*, pages 2615–2632, 2021.

[162] Vasisht Duddu, Sebastian Szyller, and N Asokan. Shapr: An efficient and versatile membership privacy risk metric for machine learning. *arXiv preprint arXiv:2112.02230*, 2021.

[163] 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)*, pages 268–282. IEEE, 2018.

[164] Yunhui Long, Vincent Bindschaedler, and Carl A Gunter. Towards measuring membership privacy. *arXiv preprint arXiv:1712.09136*, 2017.

[165] Yunhui Long, Lei Wang, Diyue Bu, Vincent Bindschaedler, Xiaofeng Wang, Haixu Tang, Carl A Gunter, and Kai Chen. A pragmatic approach to membership inferences on machine learning models. In *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 521–534. IEEE, 2020.

[166] Stacey Truex, Ling Liu, Mehmet Emre Gursoy, Lei Yu, and Wenqi Wei. Demystifying membership inference attacks in machine learning as a service. *IEEE Transactions on Services Computing*, 2019.

[167] Daniel Bernau, Günther Eibl, Philip W Grassal, Hannah Keller, and Florian Kerschbaum. Quantifying identifiability to choose and audit ϵ in differentially private deep learning. *Proceedings of the VLDB Endowment*, 14(13):3335–3347, 2021.

[168] Bargav Jayaraman, Lingxiao Wang, Katherine Knipmeyer, Quanquan Gu, and David Evans. Revisiting membership inference under realistic assumptions. *Proceedings on Privacy Enhancing Technologies*, 2021.

[169] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural

networks. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 267–284, 2019.

[170] Ahmed Salem, Yang Zhang, Mathias Humbert, Mario Fritz, and Michael Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. In *Network and Distributed Systems Security Symposium*. Internet Society, 2019.

[171] Dingfan Chen, Ning Yu, Yang Zhang, and Mario Fritz. Gan-leaks: A taxonomy of membership inference attacks against generative models. In *Proceedings of the 2020 ACM SIGSAC conference on computer and communications security*, pages 343–362, 2020.

[172] Klas Leino and Matt Fredrikson. Stolen memories: Leveraging model memorization for calibrated {White-Box} membership inference. In *29th USENIX security symposium (USENIX Security 20)*, pages 1605–1622, 2020.

[173] Cynthia Dwork. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*, pages 1–19. Springer, 2008.

[174] Thomas Humphries, Simon Oya, Lindsey Tulloch, Matthew Rafuse, Ian Goldberg, Urs Hengartner, and Florian Kerschbaum. Investigating membership inference attacks under data dependencies. *arXiv preprint arXiv:2010.12112*, 2020.

[175] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[176] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against

centralized and federated learning. In *2019 IEEE symposium on security and privacy (SP)*, pages 739–753. IEEE, 2019.

[177] Stuart J Russell. *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010.

[178] Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, Yann Ollivier, and Hervé Jégou. White-box vs black-box: Bayes optimal strategies for membership inference. In *International Conference on Machine Learning*, pages 5558–5567. PMLR, 2019.

[179] Dingfan Chen, Ning Yu, Yang Zhang, and Mario Fritz. Gan-leaks: A taxonomy of membership inference attacks against generative models. In *Proceedings of the 2020 ACM SIGSAC conference on computer and communications security*, pages 343–362, 2020.

[180] Lauren Watson, Chuan Guo, Graham Cormode, and Alexandre Sablayrolles. On the importance of difficulty calibration in membership inference attacks. In *International Conference on Learning Representations*, 2021.

[181] Hervé Abdi. The kendall rank correlation coefficient. *Encyclopedia of measurement and statistics*, 2:508–510, 2007.

[182] Nicholas Carlini, Matthew Jagielski, Nicolas Papernot, Andreas Terzis, Florian Tramer, and Chiyuan Zhang. The privacy onion effect: Memorization is relative. *Advances in Neural Information Processing Systems*, 2023.

[183] Zheng Li and Yang Zhang. Membership leakage in label-only exposures. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 880–895, 2021.

[184] Christopher A Choquette-Choo, Florian Tramer, Nicholas Carlini, and Nicolas Papernot. Label-only membership inference attacks. In *International conference on machine learning*, pages 1964–1974. PMLR, 2021.

[185] Yutong Wu, Han Qiu, Shangwei Guo, Jiwei Li, and Tianwei Zhang. You only query once: An efficient label-only membership inference attack. In *The Twelfth International Conference on Learning Representations*, 2024.

[186] Yufei Chen, Chao Shen, Yun Shen, Cong Wang, and Yang Zhang. Amplifying membership exposure via data poisoning. *Advances in Neural Information Processing Systems*, 35:29830–29844, 2022.

[187] Zhan Li, Yongtao Wu, Yihang Chen, Francesco Tonin, Elias Abad Rocamora, and Volkan Cevher. Membership inference attacks against large vision-language models. *Advances in Neural Information Processing Systems*, 37:98645–98674, 2024.