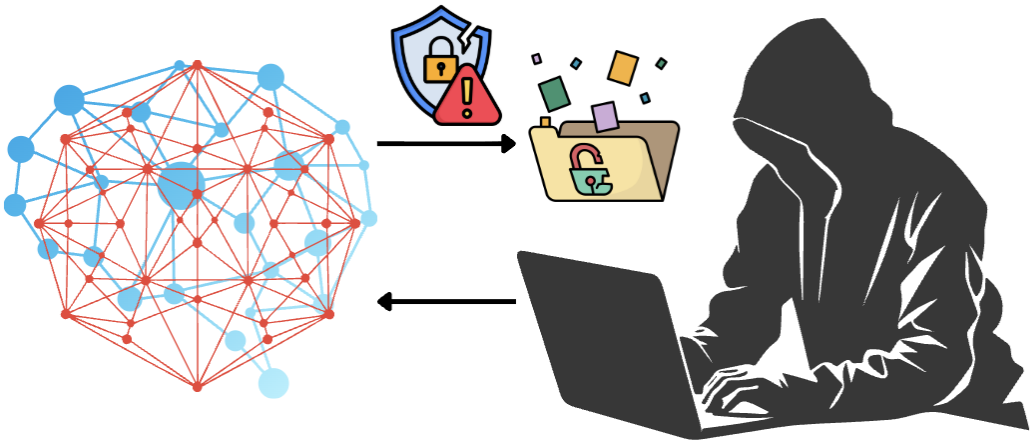
CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG



Is This Data Point In your Training Set? Similarity-based Inference Attacks

Performance Evaluation of Range Membership Attacks to Audit Privacy Risks in Machine Learning Models

Master's thesis in Computer science and engineering

Shubham Saha and Sifat Nawrin Nova

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2025

Master's thesis 2025

Is This Data Point In your Training Set? Similarity-based Inference Attacks

Performance Evaluation of Range Membership Attacks to Audit Privacy Risks in Machine Learning Models

Shubham Saha and Sifat Nawrin Nova

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2025

Is This Data Point In your Training Set? Similarity-based Inference Attacks
Performance Evaluation of Range Membership Attacks to Audit Privacy Risks in Machine Learning Models
Shubham Saha and Sifat Nawrin Nova

Supervisor: Romaric Duvignau, CSE
Advisor: Johan Östman & Fazeleh Hoseini, AI Sweden
Examiner: Rhouma Rhouma, CSE

Master's Thesis 2025
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Image illustrating how an adversary can exploit a machine learning model to infer sensitive data. The image is generated using Canva.

Typeset in LaTeX
Gothenburg, Sweden 2025

Is This Data Point In your Training Set? Similarity-based Inference Attacks
Performance Evaluation of Range Membership Attacks to Audit Privacy Risks in Machine Learning Models
Shubham Saha and Sifat Nawrin Nova
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Membership Inference Attacks (MIAs) pose a serious threat to the privacy of machine learning (ML) models by determining whether a specific data point was used during model training. A recent and powerful variant, the Range Membership Inference Attack (RaMIA), assesses privacy risks over a range of semantically similar data points. However, its practical application is limited by a high query overhead, as it requires querying the target model for every sample in the range. This thesis proposes and evaluates a novel approach designed to overcome this limitation by combining range queries with group testing principles to reduce the number of queries sent to the target model without losing the attack performance and making the attack more stealthy. Instead of testing every sample, this method first groups similar data points based on their extracted features and then queries only a small number of strategically chosen representatives.

All the experiments are conducted on the CIFAR-10 dataset, comparing its performance against the standard RaMIA baseline. The results demonstrate that RaMIA with group testing successfully reduces the number of queries by 84% in a setting of 50 augmentations. This work reveals that even minor enhancements in query design and decoding strategy can lead to substantial gains in auditing. Moreover, we provide practical recommendations for tuning key hyperparameters and integrate our attack into the LeakPro framework for reproducibility and broader adoption in privacy auditing of ML models.

Keywords: Machine Learning, Membership Inference Attacks, Range Membership Inference Attacks, Group Testing, Privacy Auditing, Decoding

Acknowledgements

We would like to express our sincere gratitude to our advisors on the LeakPro project at AI Sweden: Johan Östman and Fazeleh Hoseini. Without your expertise, encouragement, valuable insights, and feedback, this thesis would not have been possible.

We also extend our deepest thanks to our academic supervisor at Chalmers, Romaric Duvignau, for the helpful discussions and support throughout the project.
Finally, we would like to thank our examiner, Rhouma Rhouma, for taking the time to review and evaluate our thesis.

Shubham Saha and Sifat Nawrin Nova, Gothenburg, 2025-09-17

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

ML	Machine Learning
MLaaS	Machine Learning as a Service
MIA	Membership Inference Attack
RaMIA	Range Membership Inference Attacks
GT	Group Testing
ROC	Receiver Operating Characteristic
AUC	Area Under the Curve
TPR	True Positive Rate
FPR	False Positive Rate
LLR	Log-likelihood Ratio
OOD	Out-of-Distribution
RMIA	Robust Membership Inference Attack
RaMIA_BL	RaMIA Baseline
RaMIA_GT	RaMIA with Group Testing
RaMIA_BLe	RaMIA Baseline Exception

Contents

List of Figures

List of Tables

1

Introduction

Machine learning (ML) has become the backbone of many widely used technologies, from image and speech recognition to personalized recommendations and autonomous systems. These models are typically trained on large datasets generated by users such as purchase history, health records, browsing activity, or location data [1]. Many companies now offer "Machine Learning as a Service" (MLaaS), where users can upload a dataset, pay a fee, and receive access to a trained model. Access is usually provided through a black-box API, which returns predictions without revealing the model's internal structure or training data [1]. For example, an online store owner might use MLaaS to train a model that predicts customer behavior based on browsing history, enabling features like personalized recommendations or targeted ads via simple API calls. ML models often perform better on training data than on unseen data due to memorization and overfitting. They tend to learn specific details, especially rare patterns in long-tail distributions, leading to high confidence on familiar inputs but reduced generalization to new data [2], [3]. While strong training accuracy might suggest good performance, it also indicates a serious privacy risk. These models are often trained on sensitive data, raising the question: Can a model unintentionally reveal information about its training data? For example, if a model trained on hospital data reveals that a specific patient was part of its training set, it could indirectly disclose that the person has a certain medical condition, even without revealing their detailed health record.

To explore this risk, Shokri et al. introduced a privacy auditing method called membership inference attack (MIA) [1]. In this attack, the adversary interacts with the model through black-box access and decides, based on the model's output, whether a specific data point was part of the training set. MIAs simulate an adversary's perspective to test if a model unintentionally leaks training data, much like how penetration testing checks a system for security flaws. However, privacy leakage isn't limited to exact training samples. Models can also reveal sensitive information

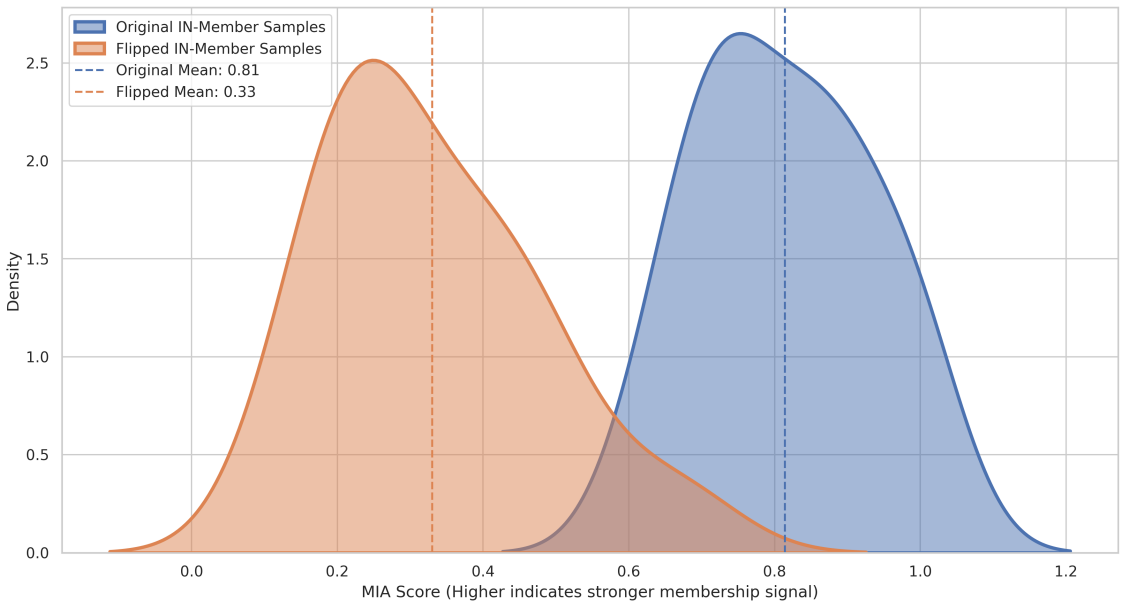


Figure 1.1: A density plot showing the distribution of membership scores computed using Robust Membership Inference Attack (RMIA) [4], for original in-member samples from CIFAR-10 (blue) and their horizontally flipped versions (orange). A simple flip causes the average membership score to drop from 0.81 to 0.33, resulting in the misclassification of members as non-members. This highlights the limitations of MIAs in capturing privacy leakage across semantically similar inputs.

when inputs are slightly modified. For example, a facial recognition model might memorize key features that identify a person, regardless of changes in background or lighting [2]. As a result, a different photo of the same person can still expose their presence in the training set. Traditional MIAs are not designed to handle such cases. They rely on a strict assumption: a data point must exactly match the training data to be considered a member. Any variation, no matter how small or semantically equivalent, is treated as a non-member, causing the attack to miss real privacy leaks. This issue is not just theoretical. As shown in Figure 1.1, original member images from CIFAR-10 receive high MIA scores (mean = 0.81). However, when these same images are flipped horizontally, a transformation that retains semantic meaning, the scores drop sharply (mean = 0.33), leading the attack to misclassify them as non-members despite the underlying leak. To address limitations of MIAs, Tao et al. introduced a new class of attacks known as Range Membership Inference Attacks (RaMIAs) [2]. In this thesis, we focus specifically on the attack method proposed by the authors, referred to as RaMIA, which implements this idea using range queries based on semantic similarity. This shifts the focus from exact matches to semantic similarity. Instead of asking, "Was this exact point in the training set?", RaMIA asks, "Was any point within a certain range of similarity a member?" The specific RaMIA

attack defines a range query using a central input, a semantic distance metric, and a radius that captures the region of interest. This allows the attack to detect not just memorization of exact samples, but also of semantically similar ones. For instance, if a model was trained on celebrity images, and an adversary submits a new photo of the same person, RaMIA can help reveal whether any image of that person was used during training, something MIAs would likely miss.

While RaMIA marks a significant step forward in privacy auditing, its practical implementation introduces notable challenges. The strategy requires sending each neighboring point within a defined range as an individual query to the target model. However, with a limited range size, the query process is manageable, but as the range expands, the number of neighboring points increases rapidly, making it computationally infeasible to query them all. Despite the cost, larger ranges are often desirable as they expand the search space and increase the chance that the model saw at least one semantically similar variation of the target sample during training. This is particularly relevant for high-dimensional data such as images, where augmentations (e.g., crops, rotations, or color shifts) can produce a rich set of plausible variants without altering semantic content. In real-world systems, repeatedly querying similar patterns of data can make the attack more detectable. As a result, the system may enforce stricter monitoring or limit black-box API access. In some cases, default rate limits may already restrict the number of queries allowed within a given time window. While RaMIA stands at the forefront of research, these practical constraints introduce challenges that highlight the need for further methodological improvements of the attack strategy.

1.1 Goals and Challenges

The primary goal of this study is to explore the privacy risks associated with ML models. We focus on RaMIA proposed by Tao et al.[2], where an adversary attempts to infer data points that are sufficiently similar to the training data, extending beyond traditional membership inference. This thesis aims to answer the following research question:

Can we reduce the number of queries sent to the target model without compromising the attack performance?

1.2 Limitations

This thesis adheres to the standard assumptions established in the field of MIAs to ensure a fair comparison with prior work [2]. The key assumptions and scope limitations are as follows:

- We assume that the adversary either has access to some real data from the same distribution as the target model's training data or can generate synthetic data that resembles it. This assumption is standard in MIA research, as the adversary needs candidate samples to test for membership in the training set.
- This work does not address models trained with explicit privacy-preserving defenses, such as Differential Privacy or Homomorphic Encryption. These techniques are addressed as defense mechanisms against ML model privacy leakage, which is a separate research direction and is considered out of the scope of this thesis.
- Given the time constraints, this thesis is limited to a single data modality: image data. The CIFAR-10 dataset [5] was chosen due to widespread use in research and manageable computational demands.

1.3 Structure of the Thesis

This thesis is organized into seven chapters, each building upon the previous one to develop a comprehensive understanding of the work. **Chapter 1** introduces the privacy risks in ML models, emphasizing the importance of assessing privacy leakage while defining the research problem and objectives. **Chapter 2** provides background on MIA, RaMIA, shadow models, group testing, and identifying key research gaps that motivate this study. **Chapter 3** details the method of the proposed framework, aimed at improving the attack strategy of existing range membership inference attacks. **Chapter 4** presents the experimental findings, comparing them against the benchmark. **Chapter 5** discusses the results, outlining key successes and challenges in achieving the research goals. It also suggests future directions in this field. Finally, **Chapter 6** concludes the study by summarizing the gained insights.

2

Background

This chapter provides the necessary background to understand the core ideas and motivations behind this thesis. We begin by discussing the concept of MIA, followed by RaMIA. The chapter also discusses key components such as shadow models, range queries, and group testing. Together, these elements establish the foundational knowledge necessary to understand the methodology discussed in subsequent chapters. We conclude by reviewing recent work in this area.

2.1 Membership Inference Attack(s)

One form of privacy attack is MIA, where the adversary aims to determine whether a specific data point was part of a model's training dataset [1]. This is defined as a binary classification problem: given a model and an input, can an adversary infer if that input was part of the training data?

MIAs serve as a foundational tool for auditing privacy risks in ML models by testing whether a specific data point was used during training. While the attack outputs a simple binary decision (member or non-member), even this limited information can lead to serious privacy violations [3], [4]. For example, if a model is trained on financial records of people who filed for bankruptcy, finding out that someone's data was used would likely reveal that they went bankrupt, exposing private financial details. MIAs often act as a backbone for more advanced attacks, such as attribute inference or data reconstruction [6], [7].

2.2 MIA(s) as a Security Game

Membership inference can be formally modeled as a game between a challenger and an adversary, as shown in Game 1, where the adversary employs a strategy $\mathcal{A}$ to extract information from a given machine learning model [1] based on their available

knowledge and resources. The inference game, where a challenger provides a data point to an adversary, is a formal framework designed to measure and quantify privacy leakage in a controlled manner. In practice, MIAs are primarily used as a privacy auditing tool, rather than as a direct attack method by a real-world adversary in every scenario.

To initiate the game, the challenger trains a target model θ on $\mathcal{D}$ (sampled from the entire data population π) by using a training algorithm $\mathcal{T}$. After that, the challenger flips a fair coin to select a data record from $\mathcal{D}$ (the training set) or from $\pi \setminus \mathcal{D}$ (the non-member set). Based on the outcome of the coin flip, a sample x_b is chosen, where $b = 1$ indicates that $x_b \in \mathcal{D}$ and $b = 0$ indicates $x_b \in \pi \setminus \mathcal{D}$. This record x_b is then sent to the adversary, who must predict whether it was part of the training set.

The adversary is assumed to have access to π, $\mathcal{T}$, and black-box access to θ. Based on the model's output, the adversary computes a membership score, $\text{Score}_{\text{MIA}}(x_b)$ that reflects how likely it is believed that θ was trained on x_b. This score is compared against a predefined threshold γ, which controls the adversary's tolerance for prediction error. If the score exceeds γ, the adversary outputs $\hat{b} = 1$ (predicting membership); otherwise, $\hat{b} = 0$ (predicting non-membership). The adversary is considered successful when $\hat{b} = b$, meaning their prediction matches the actual membership status. The effectiveness of the attack depends on how well the adversary can distinguish between members and non-members. This can be improved by designing stronger membership indicators such as confidence scores, loss values, or other statistical patterns that differentiate the model's behavior on training data versus unseen data [4] and by applying more refined statistical tests.

This inference game above formulates a generalized framework for MIA(s). The design of a MIA varies based on what the attacker is allowed to know and how they can interact with the model. A crucial factor that defines different attack settings is the adversary's level of access to the model. For instance, in white-box settings, the attacker is assumed to have complete visibility into the model's internal structure, including its architecture and learned parameters. In contrast, black-box settings limit the adversary to interacting with the model solely via an interface, such as an API. The nature of this API access further refines the attack scenario, where it may provide only hard-label predictions, class probabilities (logits), or even the loss values for given inputs. In the following sections, the necessary components related to MIA are discussed in detail.

Game 1 Basic Membership Inference Game [2]

Input: Data population π, training algorithm $\mathcal{T}$
Output: 1 if the adversary succeeds, 0 otherwise

Challenger:
1. Sample a training dataset $\mathcal{D} \sim \pi$.
2. Train target model $\theta \leftarrow \mathcal{T}(\mathcal{D})$.
3. Flip a fair coin: $b \leftarrow \text{rand}\{0, 1\}$.
4. **if** $b = 1$ **then**, sample $x_1 \sim \mathcal{D}$
   else, sample $x_0 \sim \pi$; where $x_0 \notin \mathcal{D}$.
5. Send θ and x_b to the adversary.

Adversary:
6. Compute score: $\text{Score}_{\text{MIA}}(x_b) \leftarrow \mathcal{A}(\theta(\cdot), x_b)$.
7. Predict membership: $\hat{b} \leftarrow \mathbb{1}[\text{Score}_{\text{MIA}}(x_b) \geq \gamma]$.

Outcome:
8. If $\hat{b} = b$, output 1 (success). Otherwise, output 0.

2.3 Shadow Models

In black-box settings, the adversary has limited access to the target model's internal structure or training data. To overcome this, Shokri et al. [1] introduced the shadow training technique. This involves training a set of models, called shadow models (or reference models), to approximate the behavior of the target model. These models are trained on datasets that are either sampled from the same distribution as the target model's training set or are synthetically generated to resemble it. The key idea is that machine learning models typically perform better on data they have seen during training (IN) compared to unseen data (OUT) [1]. Shadow models allow the adversary to exploit this behavior: since the adversary knows exactly which samples were used to train each shadow model, they can label data points as IN or OUT accordingly. By analyzing how the shadow models respond to these two classes, the adversary learns patterns that distinguish training members from non-members. These patterns, such as prediction confidence or loss values, are then used to train a membership inference classifier. This classifier compares the output of the target model on a given input to the known behaviors of the shadow models, inferring whether the target model likely saw the input during training [8].

Shadow models can be implemented in two different modes: online and offline. In the online mode, the adversary trains a new set of shadow models for each data point they wish to test [1], [8]. Half of these models are trained with the target

data point (IN models), and the other half without it (OUT models). Although this method is computationally expensive, as it requires training multiple models per query. To address this issue, an offline mode is introduced where a fixed set of shadow models is trained only once on auxiliary data [8]. This auxiliary data is a dataset sampled from the same distribution as the target model's training data but without any overlap (i.e., it does not contain any of the actual training samples used in the target model) [8]. These shadow models, particularly the OUT models, are then reused across multiple queries, eliminating the need to retrain models for every new target point. For instance, in the offline version of RMIA, Zarifzadeh et al. [4] leverage this strategy by using a pre-trained pool of OUT models to perform Likelihood Ratio (LR) tests. This avoids retraining and makes the attack scalable across many data points.

2.4 Robust Membership Inference Attack

Robust Membership Inference Attack (RMIA), proposed by Zarifzadeh et al. [4], introduces a statistically rigorous and computationally efficient method to perform MIA. Unlike prior attacks that require many shadow models or make simplifying assumptions [1], [3], RMIA reframes the problem as a binary hypothesis test and introduces an LR-based statistical framework. The binary hypothesis testing problem is defined as:
$H_{\mathbf{out}}$ (Null hypothesis): The data point x was not used in training the model θ.
$H_{\mathbf{in}}$: The data point x was part of the training data used to build model θ.
For a target data point x, the attack constructs a test against multiple plausible worlds where the target sample x is replaced by a random sample $z \sim \pi$, where π is the population distribution. The central test statistic is the pairwise LR:

$$\mathrm{LR}_\theta(x, z) = \frac{\Pr(\theta \mid x)}{\Pr(\theta \mid z)}$$

In the offline version of RMIA, IN models are unavailable, and it only has access to the OUT models that are trained without the target point x. Since it cannot compute $\Pr(x \mid \theta)$ for IN models directly, it uses Bayes' Rule to estimate the LR:

$$\mathrm{LR}_\theta(x, z) = \left(\frac{\Pr(x \mid \theta)}{\Pr(x)}\right) \cdot \left(\frac{\Pr(z \mid \theta)}{\Pr(z)}\right)^{-1}$$

Here, $\Pr(x \mid \theta)$ is typically approximated by the model's softmax confidence on x, and $\Pr(x)$ is estimated using a set of OUT models. The membership score in RMIA

is then computed by assessing how frequently the likelihood ratio $LR_\theta(x, z)$ exceeds a threshold t over the sampled non-member points z:

$$\text{Score}_{\text{MIA}}(x; \theta) = \Pr_{z \sim \pi}\left(LR_\theta(x, z) \geq t\right)$$

This score is compared against a threshold t to decide membership, where t controls the error (false positive rate) that the adversary is willing to tolerate. In the context of RaMIA and also our proposed work, RMIA is used as the membership scoring strategy.

2.5 Range Membership Inference Attack

Rather than deciding whether a specific data point was part of the training set, RaMIA aims to determine whether any training data exists within a defined neighborhood around a candidate input [2]. The structure of this attack is formalized as a game between a challenger and an adversary, as shown in Game 2.

While most of the steps are similar to the formulation described in Game 1, the key difference between the two games is that now the attacker receives a range of queries instead of point queries. After sampling the training dataset and training the model, the challenger flips a fair coin to sample a bit $b \in \{0, 1\}$. If $b = 1$, a data point is selected from the training set; if $b = 0$, the point is chosen from outside the training set. A range R_b is then constructed around this selected point, using a predefined similarity metric to generate semantically similar variants. Thus, R_1 may include training members, while R_0 is guaranteed not to contain any training data. The challenger sends the trained model θ and the corresponding range R_b to the adversary. The adversary then analyzes the model's behavior across all points within the range and predicts whether any of the inputs in R_b came from the training data. The adversary is considered successful when $\hat{b} = b$, meaning their prediction matches the actual membership status. As RaMIA differs from MIA in its use of range queries, it is crucial to understand the construction and sampling of ranges, as described in the following section.

2.5.1 Range queries

A range is defined as a set of data points located around a center data point based on a distance measure. A radius controls the size of the range, and a range function measures the proximity based on the semantic closeness of the points from the center.

Game 2 Range Membership Inference Game [2]

Input: Data population π, training algorithm $\mathcal{T}$
Output: 1 if the adversary succeeds, 0 otherwise

Challenger:
1. Sample a training dataset $\mathcal{D} \sim \pi$.
2. Train target model $\theta \leftarrow \mathcal{T}(\mathcal{D})$.
3. Sample a data record $x_0 \sim \pi \setminus \mathcal{D}$ and $x_1 \sim \mathcal{D}$.
4. Flip a fair coin: $b \leftarrow \text{rand}\{0, 1\}$.
5. **if** $b = 1$ **then**, construct range R_1 containing x_1
   else, construct range R_0 that containing x_0.
6. Send θ and R_b to the adversary.

Adversary:
6. Gets access to π and θ, and outputs a bit $\hat{b} \leftarrow \mathcal{A}(\theta(\cdot), R_b)$.

Outcome:
8. If $\hat{b} = b$, output 1 (success). Otherwise, output 0.

Formally, it can be written as: $R = \{x' : d(x', x) \leq \epsilon\}$ where x is the range center, d is the distance function, and ϵ is the range size. The choice of range function depends on the data modality and should retain meaningful aspects of the input. Common types of range functions include spatial measures (such as ℓ_p norms), transformation-based (such as geometric transformations), and semantic definitions (such as user identity or class labels) [2].

2.5.2 RaMIA as a framework

As mentioned earlier, a range is constructed around a central data point that consists of several augmented neighbors. Each range is labeled as IN if it contains at least one point from the model's training set, and OUT otherwise. Second, each sample within the range is evaluated using a membership scoring function. Any MIA mechanism, such as RMIA[4], LiRA[8], can be applied here. This function outputs a score for each sample, quantifying the likelihood that it was part of the training data based on the model's behavior. Finally, the individual scores from all samples in the range are aggregated into a single score for the entire range using the Trimming Average method. This trimming process is governed by two hyperparameters, q_s (quantile start) and q_e (quantile end), which define the portion of the sorted score distribution to be removed. As stated by Tao et al. [2], the tuning of these hyperparameters mostly depends on the nature of the data:

- **In-Distribution (ID) Data:** When the adversary samples from the same distribution as the model's training data, high scores are likely true members, while low scores are treated as noise. Thus, the strategy trims the lowest scores (by setting $q_s > 0$) and averages the rest. The upper trim point q_e is adjusted to control variance based on sample size.
- **Synthetic Data:** With synthetically generated data, high scores often result from out-of-distribution (OOD) and prone to false positives. To address this, the highest scores are trimmed ($q_e < 100$), and the lower bound q_s is increased when data quality decreases.

Formally, the trimming average is defined as:

$$\mathbb{P}(\theta|H_1) = \text{TrimmedAvg}(S, q_s, q_e; \mathbb{P}) = \text{Avg}_{x \notin [q_s, q_e]\text{-th quantiles}} \mathbb{P}(\theta|x \in D)$$

where S represents the set of sampled points in the range, q_s and q_e denote the quantile thresholds for trimming. The term $\mathbb{P}(\theta|x \in D)$ represents the membership score for a sample x.

2.6 Group Testing

Group testing is a powerful paradigm for efficiently identifying specific defective items within a large population by minimizing the number of tests required [9]. The core concept, first formalized by Robert Dorfman, was inspired by a real-world problem: efficiently screening U.S. soldiers for syphilis during World War II, where testing every individual was prohibitively expensive and time-consuming. Group testing operates on the principle that if the number of infected individuals is significantly smaller than the total population, then testing groups of items together can substantially reduce the number of required tests. In group testing, items are organized into overlapping groups or pools, and tests are performed on each group rather than on individual items. A test outcome is positive if at least one defective item exists in the group, and negative otherwise. Generally, a decoding operation is performed to infer which individuals are infected based on the test outcomes for the groups [10].

While the simplest group testing designs use separate, non-overlapping pools, more advanced frameworks often employ overlapping groups. In such designs, a single item can be a member of multiple test groups. Overlapping designs are particularly effective for improving the accuracy of the final inference with less probability of error, as they provide more information for each item as they are involved in several groups. For example, as shown in the Figure 2.1, if a soldier's blood appears in test

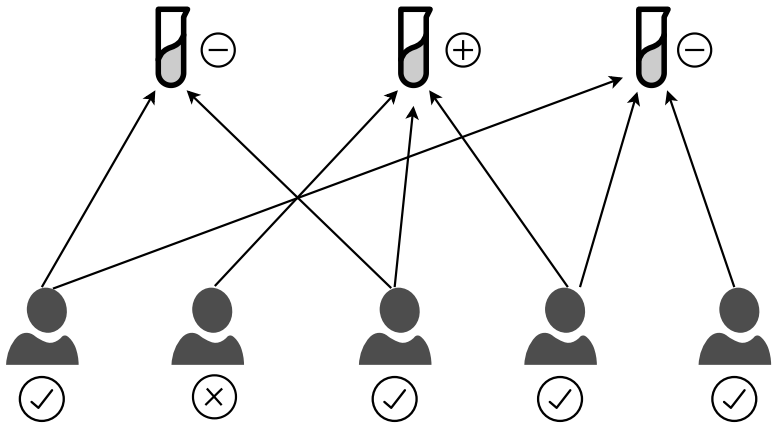


Figure 2.1: A representation of overlapping group testing. Arrows indicate which individuals are included in each test group. A positive test result suggests at least one infected individual in the group.

groups of both positive and negative results, it becomes highly probable that this soldier could not be the source of the infection. Because if so, then they would have also triggered a positive outcome in the negative test group. This overlap acts like cross-validation: even if one test result is noisy or misleading, other overlapping tests provide additional constraints that help resolve ambiguity.

2.6.1 Decoding

Once tests are performed on the test groups, it is crucial to solve the inverse problem: given the set of positive and negative outcomes for the groups, the goal is to infer which specific individuals are infected. While simple logical decoders can work in noiseless settings, probabilistic decoders such as the BCJR algorithm, named after its creators Bahl, Cocke, Jelinek, and Raviv [10] are required when test outcomes are noisy or uncertain. Originally, it was developed for communications theory for decoding codes transmitted over noisy channels. Conceptually, BCJR represents the decoding problem on a graph structure called a trellis, where each node corresponds to a possible state (e.g., a specific configuration of which items could be defective) and each path through the trellis represents one possible sequence of such states consistent with the group testing design. Using a dynamic programming method known as the forward–backward algorithm, BCJR integrates evidence from all tests to estimate the likelihood of each configuration. The forward recursion computes the probability of reaching each state given all previous test results, while the backward recursion computes the probability of observing the remaining test results given that state. By combining the forward and backward probabilities, the algorithm calculates the

Log-likelihood Ratio (LLR) for each individual item under test, indicating how much the evidence supports that the item is defective versus non-defective. A conceptual trellis diagram illustrating this process is shown in Fig. 2.2, where paths represent different defect configurations, and the forward–backward passes aggregate evidence across all paths to produce LLR scores for each item.

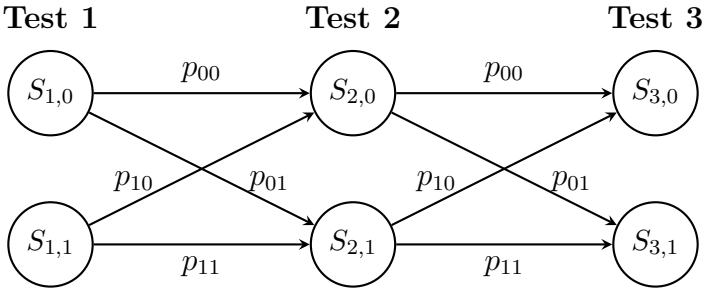


Figure 2.2: Example trellis diagram for group testing BCJR decoding. Each column represents a test step, each circle a possible state, and arrows denote possible transitions with their probabilities. Forward–backward passes traverse these edges to compute posterior probabilities for each state.

Formally, let $\alpha_\ell(\sigma)$ denote the forward metric of state σ at stage ℓ, and $\beta_\ell(\sigma)$ the backward metric. If $\gamma_\ell(\sigma', \sigma)$ is the branch metric associated with the transition from state σ' to σ, the recursions are defined as:

$$\alpha_\ell(\sigma) = \sum_{\sigma'} \alpha_{\ell-1}(\sigma')\, \gamma_\ell(\sigma', \sigma), \tag{2.1}$$

$$\beta_{\ell-1}(\sigma') = \sum_{\sigma} \beta_\ell(\sigma)\, \gamma_\ell(\sigma', \sigma). \tag{2.2}$$

The posteriori log-likelihood ratio (LLR) can be computed as:

$$LLR_\ell = \log\left(\sum_{(\sigma',\sigma)\in\mathcal{E}_\ell^0} \alpha_{\ell-1}(\sigma')\, \gamma_\ell(\sigma', \sigma)\, \beta_\ell(\sigma)\right) - \log\left(\sum_{(\sigma',\sigma)\in\mathcal{E}_\ell^1} \alpha_{\ell-1}(\sigma')\, \gamma_\ell(\sigma', \sigma)\, \beta_\ell(\sigma)\right), \tag{2.3}$$

where $\mathcal{E}_\ell^0$ and $\mathcal{E}_\ell^1$ represent the sets of edges connecting the trellis states at trellis depth $\ell - 1$ with the states at trellis depth ℓ. Then, the computed posterior LLRs are used to make a decision on the status of the item.

2.7 Evaluation Metrics in RaMIA

The evaluation of RaMIA follows a methodology designed to measure practical privacy risks. An attack's performance is quantified by its ability to distinguish

"IN-ranges" (those containing at least one training point) from "OUT-ranges" (those containing none). This is measured using two key rates:

- **True Positive Rate (TPR):** The proportion of correctly predicted IN-ranges out of all true IN-ranges. In other words, it measures how many of the ranges that actually contain training points were correctly identified as such.
- **False Positive Rate (FPR):** The proportion of OUT-ranges incorrectly classified as IN, out of all true OUT-ranges. This reflects how often the attack wrongly predicts membership when no training point is present in the range.

By varying an attack's decision threshold, these rates can be plotted to form a Receiver Operating Characteristic (ROC) curve, with the Area Under the Curve (AUC) serving as a summary of overall performance. However, as argued by Carlini et al., average-case metrics such as AUC are not appropriate for security evaluation [8]. This is because they can obscure critical, concentrated risks by averaging them with widespread, low-impact results, as it treats all errors equally. A real-world security assessment must be able to detect the most dangerous threats, not just measure average performance. To illustrate, compare two hypothetical attacks on a facial recognition model. One attack learns whether a large number of generic smiling face images were part of the training set. A second attack succeeds only once, but in doing so it uncovers that a celebrity's private photo was used during training. In these two cases, the first attack would achieve a much higher AUC because it succeeds moderately for many generic faces, whereas the second attack would appear to have a low AUC since it only succeeds for a single image. However, this comparison is misleading: the first attack is practically harmless despite its high AUC, while the second, far more damaging attack, exposes a critical privacy violation of an individual. When we are claiming there is a leak we need to be accurate about it. That means minimizing false accusation and keeping the false positive rate as low as possible, ideally close to zero. The ability to reveal the presence of even one sensitive image is therefore a more severe vulnerability than learning weak information about many. For a privacy audit to be considered trustworthy, it must operate with a very low tolerance for false accusations. Therefore, the standard methodology shifts focus away from averages and towards the low-FPR regime. By evaluating the TPR at a fixed, low FPR threshold (e.g., 1.0% and 0.1%), we answer the most critical question for a privacy auditor: "When the attack is required to make decisions with near-zero false positives, how many true members can it still correctly identify?" Thus, this thesis prioritizes TPR at low FPRs for a more accurate and practical assessment of privacy risks in Chapter 4.

3

Methods

This chapter presents the methodological foundation of our work. We begin with a precise formulation of the problem, reproducing the state-of-the-art RaMIA method proposed by Shokri et al. [2] to establish a consistent benchmark suite. This is followed by the proposed group testing-based solution, which reduces query overhead in the initially proposed method. It is important to note that RaMIA is demonstrated in the offline attack setting, which displays practical scenarios. Following this, we have implemented the benchmark and our proposed method in the offline setting.

3.1 Problem Formulation

RaMIA aims to determine whether a semantically similar data point to a given query, rather than the exact point itself, was present in the model's training set. To achieve this, it defines a range around the examined data point based on a chosen similarity metric. This range typically includes modified or augmented versions of the input sample, capturing its semantic neighborhood. However, in order to evaluate this range, RaMIA must query the target model with every point in it, which makes the attack computationally expensive for large ranges. This motivates the question: can we design a more efficient version of this attack that arrives at the same conclusion using significantly fewer queries to the model?

The problem can be mathematically formulated as, let π be the entire data population. A target model θ is trained using a learning algorithm $\mathcal{T}$ on a training subset $\mathcal{D} \subset \pi$, such that, $\theta = \mathcal{T}(\mathcal{D})$. A range, R_b, a set of n data points, is created that is either an IN-range ($b = 1$), meaning $R_1 \cap \mathcal{D} \neq \emptyset$, or an OUT-range ($b = 0$), meaning $R_0 \cap \mathcal{D} = \emptyset$. The adversary's task is to employ an attack strategy, $\mathcal{A}$, to predict the true label $\hat{b}$ given black-box access to the model θ and the set of points in the range R_b. The prediction is denoted as $\hat{b} = \mathcal{A}(\theta, R_b)$. This work addresses the query complexity of the attack $\mathcal{A}$. A baseline attack, $\mathcal{A}_{\text{base}}$, makes n queries to the model θ,

one for each of the n points in the range R_b. The problem is to design an alternative attack strategy, $\mathcal{A}'$, that satisfies:

- The number of queries made by $\mathcal{A}'$ is k, where ideally $k \ll n$.
- The success rate of $\mathcal{A}'$ (e.g., measured by TPR at low FPR) is comparable to that of the baseline attack, $\mathcal{A}_{\text{base}}$.

3.2 Dataset

For reproducing the baseline and our proposed method and experimental evaluation, we have used CIFAR-10. The dataset was curated by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton[5]. It is a widely used dataset in computer vision, especially for image classification and object recognition tasks. Derived from the larger 80 million tiny images collection, it contains 60,000 color images of 32×32 pixels across 10 distinct classes, sample images are shown in the figure 3.1.

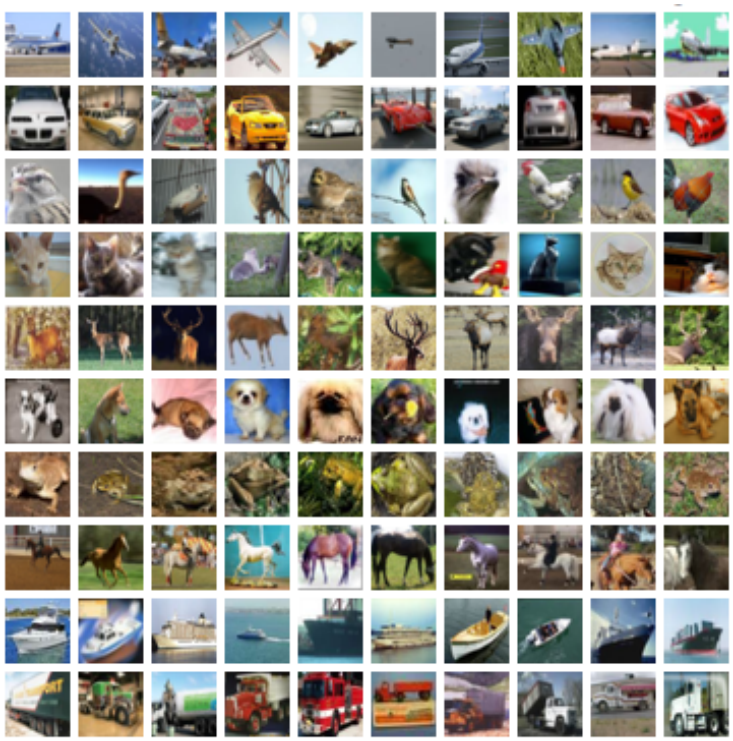


Figure 3.1: Sample images from the CIFAR-10 dataset, showcasing a diverse set of images across 10 object categories including airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks. Each class contains visually varied examples to support robust object recognition tasks. **Image source:** Kaggle [5]

Out of these, 50,000 images are designated for training and 10,000 for testing. The core of our approach is to create a neighborhood (a collection of nearby variants)

for each original image sample. To achieve this, we have applied a set of geometric augmentations, including horizontal flips, rotations, and color adjustments. This process yields a collection of modified versions of the original sample, which we refer to as the augmented dataset, used for our experiments.

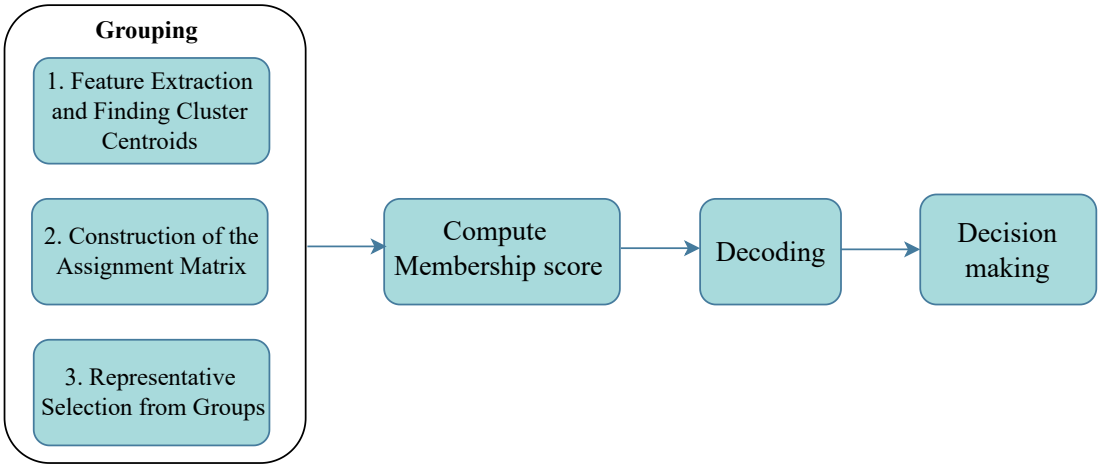


Figure 3.2: Overview of the integration of GT in RaMIA. The initial Grouping stage selects representative samples for efficient scoring. The resulting scores are then passed to a Decoder to infer membership and make a final decision.

3.3 Proposed Method

As discussed earlier in Section 2.5, RaMIA queries the target model for all neighboring points within the range. This becomes challenging when the range size is increased. In our proposed method, we aim to improve the methodology of RaMIA by incorporating group testing mechanism. Similar to the baseline, the first step of this proposed method is preparing the augmented dataset as mentioned in Section 3.2. The augmented dataset represents the ranges for each original sample that consists of its semantically similar neighboring samples. By borrowing ideas from group testing, we arrange samples within a range efficiently, thereby significantly reducing the number of range queries sent to the target model.

In traditional group testing, individual samples used to form groups are independent in terms of correlation [9]. But the efficiency of group testing can be significantly enhanced when groups are formed based on correlations between samples. Recent work has demonstrated that performance can be significantly improved by creating groups based on known or predicted correlations between samples [11]. This idea provides a strong motivation for the approach taken in this thesis. By intelligently forming our test groups based on feature similarity instead of random assignment,

we aim to leverage this inherent correlation to create more efficient and informative tests.

We refer to the proposed method as RaMIA with Group Testing (RaMIA_GT) to reflect this enhancement and for ease of reference throughout the paper. Figure 3.2 presents a high-level overview of the entire method, outlining the key stages from the initial grouping of samples to the final decision-making step. Each of these stages is explained in detail in the following sections.

3.3.1 Feature Extraction and Finding Cluster Centroids

Algorithm 1 Feature Extraction and Finding Cluster Centroids

Input:
  $I = \{I_1, I_2, \ldots, I_n\}$: Set of n augmented image samples.
  $f(\cdot)$: Pre-trained feature extractor (ResNet-18)
  $g(\cdot)$: Dimensionality reduction method (t-SNE)
  k: Number of clusters (test groups)
  m: Number of closest centroids to each sample ($m \leq k$)
Output:
  $\{c_i\}_{i=1}^{k}$: A set of k cluster centroids.

Step 1: Feature Extraction
for $j = 1$ **to** n **do**
  $x_j \leftarrow f(I_j)$ {Extract feature vector from image I_j}
end for

Step 2: Dimensionality Reduction
$\{z_j\}_{j=1}^{n} \leftarrow g(\{x_j\}_{j=1}^{n})$ {Map feature vectors to 2D space using t-SNE}

Step 3: Clustering
$\{c_i\}_{i=1}^{k} \leftarrow$ K-means$(\{z_j\}_{j=1}^{n}, k)$ {Find k cluster centroids}

return $\{c_i\}_{i=1}^{k}$ =0

We follow the initial steps similar to the baseline, which include preparing the augmented dataset and constructing the range. Afterwards, whereas RaMIA sends the augmented images to MIA module for membership scoring, our method integrates the steps of grouping. Before applying MIA, we consider an augmentation function where each image sample is passed through, from which we receive a set of augmented images, denoted by $I = \{I_1, I_2, \ldots, I_n\}$ to be distributed among k test groups, where n is the total number of samples. Each sample is assigned to exactly m distinct groups, where $m \leq k$ defines the degree of overlap. This means every sample belongs

to m of the k groups. The assignment of samples into test groups is described by an assignment matrix $A = (a_{i,j})$, where $i \in [k]$, $j \in [I]$. Here, $a_{i,j} = 1$ if sample j is assigned to group i, and $a_{i,j} = 0$ otherwise.

Now, to cluster augmented images, RaMIA_GT follows Algorithm 1, where it initially analyzes the content of each image sample through feature extraction. Each augmented image in the set I is passed through a pre-trained neural network based on ResNet-18 [12], which was initially trained on the ImageNet dataset [13]. This model processes each image and produces a feature vector that summarizes the important visual patterns in the image. We denote this vector as $x_j = f(I_j)$, where f is the feature extractor and x_j represents the feature vector for image I_j. Next, to reduce the dimensionality of these feature vectors, RaMIA_GT uses a dimensionality reduction technique called t-distributed Stochastic Neighbor Embedding (t-SNE) [14]. This method converts each feature vector x_j into a point z_j in a two-dimensional (2D) space, while preserving the similarity between samples. We opted for t-SNE over linear alternatives like Principal Component Analysis (PCA) because of its ability to preserve local similarity structures, which is crucial for grouping semantically similar images. Unlike PCA, which focuses on capturing global variance, t-SNE is specifically designed to maintain neighborhood relationships, ensuring that samples appearing close in the original feature space remain close in the 2D embedding. In this new 2D space, images with similar features are placed close together, revealing natural groupings among them. Once the 2D embeddings $\{z_j\}$ are obtained, RaMIA_GT applies the K-Means clustering algorithm [15] to identify k cluster centers (also called centroids), denoted as $\{c_i\}_{i=1}^{k}$. These centroids serve as reference points for forming the test groups.

3.3.2 Construction of the Assignment Matrix

To construct the final test groups, we use the k cluster centroids as anchors. The entire process is outlined in Algorithm 2. Each image sample j is assigned to the m groups whose centroids are nearest to it.

This assignment is based on the Euclidean distance in the 2D t-SNE space. The ultimate goal is to select one representative sample from each group to be a candidate for membership scoring. By assigning each sample to m out of k groups, we create an overlapping group that is a core requirement for group testing, as mentioned in Section 2.6. The entire grouping structure is formally captured in a binary assignment matrix $A \in \{0, 1\}^{k \times n}$. An entry $a_{i,j} = 1$ if sample j is assigned to group i (because centroid c_i is one of its m closest), and 0 otherwise.

Algorithm 2 Construction of the Assignment Matrix

Input:
$\{z_j\}_{j=1}^{n}$: Set of n samples in 2D space (from t-SNE)
$\{c_i\}_{i=1}^{k}$: Set of k cluster centroids (from Algorithm 1)
m: Number of closest centroids to each sample ($m \leq k$)
Output:
$A \in \{0,1\}^{k \times n}$: The final binary assignment matrix.

Step 1: Initialize the Assignment Matrix
Initialize A as a $k \times n$ matrix filled with zeros.

Step 2: Assign Each Sample to its m Nearest Groups
for $j = 1$ **to** n **do**
  Let `distances` be an empty list.
  for $i = 1$ **to** k **do**
    $d_{i,j} \leftarrow \|z_j - c_i\|_2$ {Calculate Euclidean distance from sample z_j to centroid c_i}
  end for
  `closest_group_indices` $\leftarrow$ Find indices of the **m** smallest values in the list of distances.
  for each `group_idx` in `closest_group_indices` **do**
    A[`group_idx`][j] $\leftarrow 1$ {Assign sample j to this group}
  end for
end for

return A =0

3.3.3 Representative Selection from Groups

After constructing the assignment matrix A, the final step of grouping is to select a single sample from each of the k groups. This sample acts as a representative of the entire group. To ensure it meaningfully captures the group's semantic center, we select the medoid of the sample whose 2D embedding has the lowest average distance to all other points in the same group. In other words, this representative is the most centrally located point in the feature space among its semantically similar neighbors. It is then sent as a query to the target model, and the resulting membership score is taken to represent the entire group.

3.3.4 Compute Membership Score

With the set of k representative samples selected, we now need to determine their likelihood of being training members. This stage is where the interaction with the target model occurs, but in an efficient manner. Instead of scoring all n augmented images as the baseline RaMIA does, RaMIA_GT only sends the k representative samples to the membership scoring function. The scoring function itself is identical to the one used in the baseline. As described in Section 2.5.2, this can be any suitable MIA mechanism, like RMIA. The membership score reflects how closely the model's response resembles its behavior on known training data.

3.3.5 Decoding and Decision Making

The final stage of the attack is to use the scores of the k representatives to make a single decision for the entire group, whether it is an IN-range or an OUT-range. This is achieved through a two-step decoding process.

First, the continuous membership scores of the k representatives are converted into binary test outcomes. This is done by applying a simple threshold: if a representative's score is above a pre-defined global threshold, its outcome is 1 (positive); otherwise, it is 0 (negative). This gives us a binary vector of k outcomes, where each outcome corresponds to one group.
In the second step, the decoder takes two inputs:

- The assignment matrix A, which describes which of the original n samples belong to which of the k groups.
- The binary test outcomes for each of the k groups.

Then the decoding algorithm processes this information to produce LLR scores for

each sample within a group. This process will be continued for all ranges. Finally, it will return LLR scores for every augmented sample within all the ranges. This score represents the logarithm of the ratio of two probabilities: the overall probability of all evidence if the sample were a non-member versus the overall probability if it were a member. A lower LLR provides strong evidence that the sample is a member, while a higher LLR indicates it is likely a non-member.

Afterwards, we perform the trimming average on the LLR scores to remove outliers that might skew the results. In this case, we trim the higher LLR values based on the hyperparameter q_e. After trimming, the rest are averaged to get the best range score that can be compared against a decision threshold to classify the range as IN or OUT. If a range is classified as IN, it means the range contains at least one training member.

4

Results

This chapter presents the empirical evaluation of our proposed method, RaMIA_GT, against the baseline, RaMIA_BL. Additionally, we have compared the proposed method with the baseline in an exceptional case (RaMIA_BLe), which sends the same number of queries as RaMIA_GT to validate its necessity in practice. We keep the distinct notations to avoid any ambiguity. This chapter outlines the experimental environment, including the frameworks, model configurations, and hyperparameters used. Afterwards, it presents the primary attack results, focusing on the trade-off between attack performance and query efficiency. Finally, a series of ablation studies is conducted to justify our choice of the number of groups and overlaps.

4.1 Experimental Setup

All experiments were conducted using Python (version 3.12) and PyTorch deep learning framework. The core attack logic and evaluation pipeline were built upon the LeakPro privacy auditing framework, which provided the necessary infrastructure for model training, data handling, and membership inference evaluation. The target and shadow models share the same architecture to ensure a fair and realistic simulation of the adversary's capabilities in an offline setting. The specific settings are detailed in Table 4.1.

Table 4.1: Target and Shadow Model Settings.

Parameter	**Value/Setting**
Model Architecture	ResNet-18
Dataset	CIFAR-10
Optimizer	Adam
Learning Rate	0.001
Epochs	30
Shadow Models	8

For image classification, the convolutional neural network architecture RestNet-18

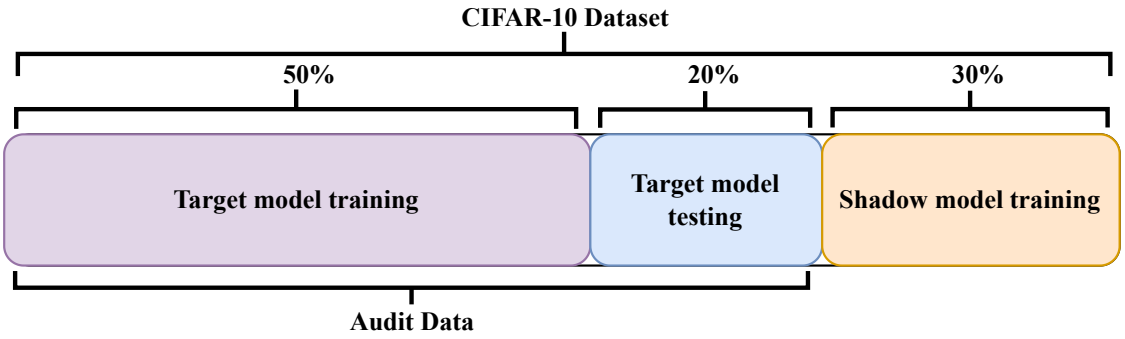


Figure 4.1: Data partitioning strategy for the CIFAR-10 dataset. The dataset is divided into three main sections: 50% for training the target model, 30% for training the shadow models, and 20% for testing the target model. The target model's training data (members) and testing data (non-members) are combined to form the audit dataset, which is used to evaluate the attack.

[12] is used. The data distribution for training the target and shadow models follows a carefully designed strategy. To set up our experiments, we partition the CIFAR-10 dataset as illustrated in Figure 4.1. A total of 50% of the data is used to train the target model. A separate 30% (excluding target model training and testing data) is used for shadow model training. This setup simulates a realistic setting of an adversary with limited data access.
The remaining 20% of the dataset, which serves as the target model's test set, is combined with the target's training data (the initial 50%) to form the audit dataset. This combined set is essential for the privacy evaluation phase. The data points from the target model's training set act as the members for our attack, while the data points from its test set act as the non-members. By drawing samples from this audit data, we can measure how effectively an attacker can distinguish between data used for training and data not used for training.

Table 4.2: Hyperparameter Settings for the Attack.

Hyperparameter	**Value/Setting**
Number of audit samples	1200
n (Number of augmentations per sample)	50
q_s (Trim Start Quantile for RaMIA_BL)	0
q_e (Trim End Quantile for RaMIA_BL)	40
k (Number of Groups)	8
m (Number of groups each sample belongs to)	3
GTq_s (Trim Start Quantile for RaMIA_GT)	40
GTq_e (Trim End Quantile for RaMIA_GT)	100
Membership Scoring Algorithm	RMIA

To configure our attack experiments for both RaMIA_BL and RaMIA_GT, the used hyperparameters are listed in Table 4.2. For all experiments, we define a

range as a set of $n = 50$ augmented samples where 1200 data points are sampled from the audit dataset. This choice is primarily driven by computational feasibility, given the extensive augmentation and evaluation required for each sample within a reasonable timeframe. The k controls the number of test groups, and m controls the number of groups each sample can be assigned to. The underlying membership score for individual samples is calculated using the RMIA. The base implementation of RMIA has already been implemented in LeakPro. A critical distinction between the baseline and our proposed method lies in the score aggregation strategy, governed by different trimming quantiles. For the RaMIA_BL, we follow the standard approach for in-distribution data mentioned in their study [2]. The RMIA scoring function produces higher scores for likely members. Therefore, we set $q_s = 0$ and $q_e = 40$, meaning we trim the bottom 40% of scores (considered noise from non-members) and average the remaining top 60% to get the final range score.
For our RaMIA_GT method, the process is different. The BCJR decoder yields LLR scores where a lower value indicates a stronger signal for membership. To analyze the most significant evidence, we must focus on these low scores. We set $GTq_s = 40$ and $GTq_e = 100$, which instructs our method to select the lowest 40% of the LLRs generated by the decoder and discard the rest. These selected scores are then averaged to calculate the final range score.

4.2 Attack Results

The primary goal of our work is to reduce the number of queries sent to the target model without significantly compromising the performance of the attack. We evaluate RaMIA_GT against RaMIA_BL in table 4.3 by measuring the TPR at low FPR of 1.0% and 0.1%, which is the standard for security evaluations, as mentioned in Section 2.7. Here, we report the mean ± standard deviation of the TPR across five runs. The key difference is that RaMIA_BL queries all samples within a range, while RaMIA_GT strategically selects and queries only the group representatives.

The above comparison shows that RaMIA_GT sends only 8 queries, whereas the baseline sends 50 queries to the target model. However, we must distinguish whether the effectiveness of RaMIA_GT comes from its grouping strategy or if a similar result could be achieved by simply reducing the range size of the baseline to the number of queries sent by RaMIA_GT.
To experiment, RaMIA_BL is configured to execute with 8 augmentations, and, consequently, sends exactly 8 queries to the target model, which we refer to as RaMIA_BL exception or RaMIA_BLe to avoid confusion. This is equivalent to the

Table 4.3: Attack performance comparison among the baseline with RaMIA_GT for 50 augmentations. Our proposed RaMIA_GT reduces the query cost per sample from 50 to 8 in total queries for an audit of 1,200 samples, while maintaining comparable attack performance to the baseline. Lastly, we report the performance of the baseline with 8 queries, i.e., RaMIA_BLe, to validate the necessity of RaMIA_GT.

Method	**Queries per sample**	**Total queries sent to the target model**	**TPR@1.0% FPR**	**TPR@0.1% FPR**
RaMIA_BL	50	(50x1200)=60,000	3.92 ± 1.49	1.31 ± 0.81
RaMIA_GT	8	(8x1200)=9,600	3.34 ± 1.02	1.03 ± 0.80
RaMIA_BLe	8	(8x1200)=9,600	1.19 ± 0.46	0.37 ± 0.36

same number of queries sent by RaMIA_GT, which justifies the implication and necessity of the proposed method. In Figure 4.2, we show all attack results in the ROC curve, generated by thresholding the MIA scores at varying levels. This reveals how the attack performance of the baseline is affected by the reduced number of queries.

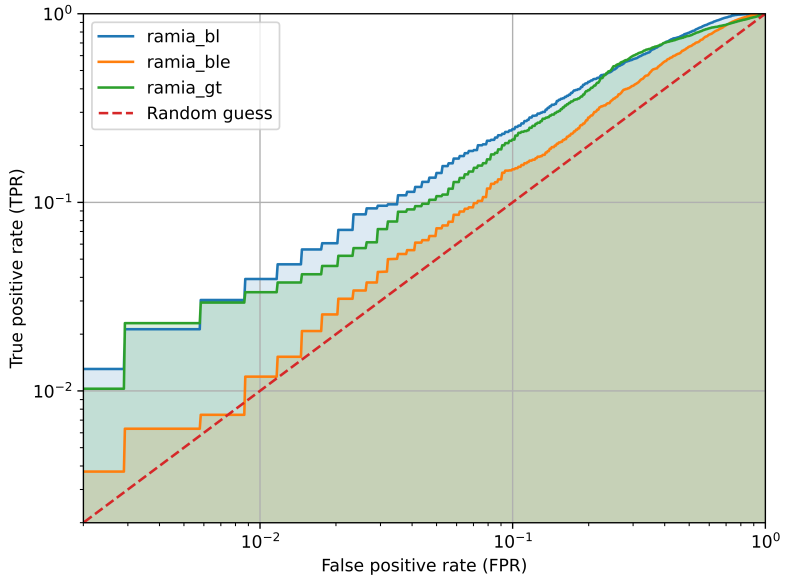
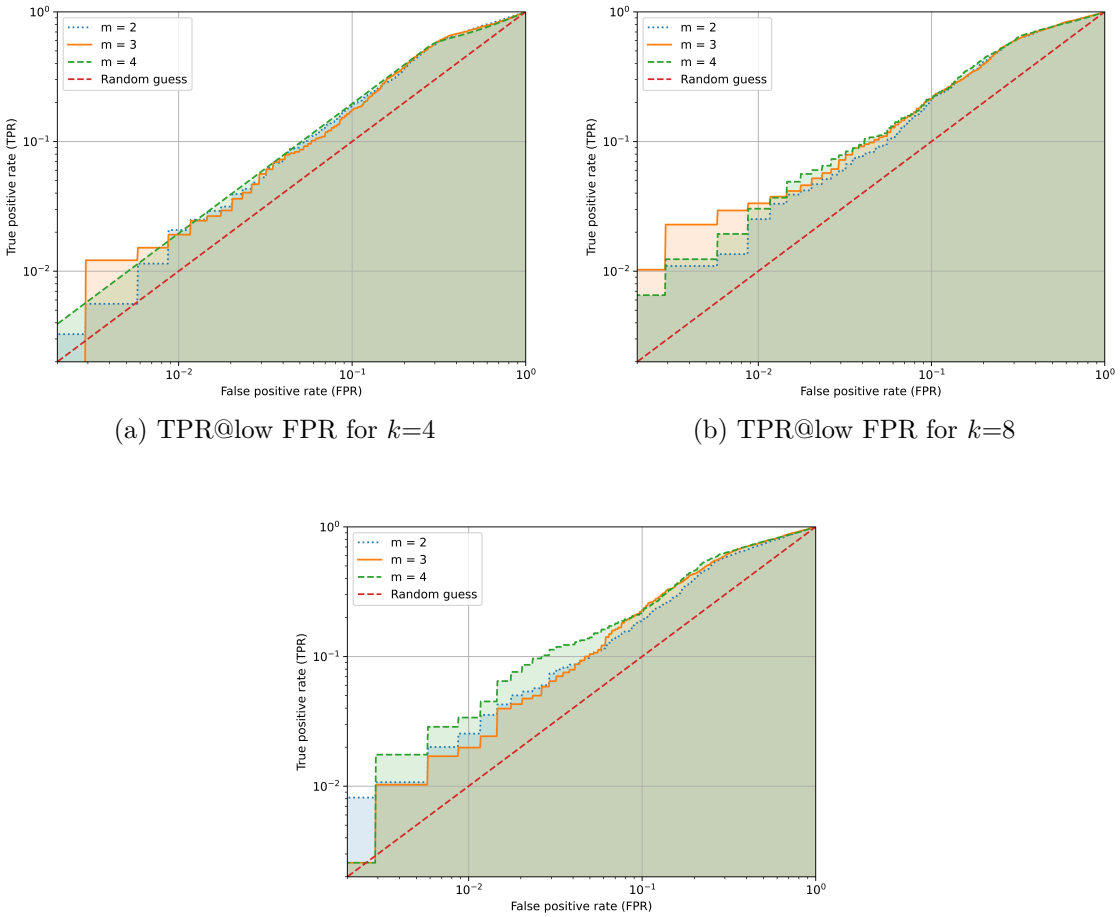


Figure 4.2: Comparison of RaMIA_BL with RaMIA_GT and RaMIA_BLe. Results are shown in ROC curve, where it shows the mean TPR across five runs.

4.3 Ablation Studies

The performance of our proposed RaMIA_GT method depends on the careful selection of its group testing parameters: k (the number of test groups) and m (the degree of overlap, i.e., how many groups each sample belongs to). To justify our choice of $k = 8$ and $m = 3$ in the attack results presented in Section 4.2, and to understand the sensitivity of our method to these parameters, we conducted a series

of ablation studies. In these experiments, we systematically varied k and m while keeping the underlying range size fixed at $n = 50$ augmentations. The results are summarized in Table 4.4, where in each experiment for a fixed value of k, $(k = 4, 8, 12)$ we have varied the size of m with 2, 3, and 4. In the table, we report the mean $\pm$ standard deviation of the TPR across five runs. To visualize these trade-offs, Figure 4.3 shows the ROC curves. The observation reveals that the choice of the (k, m) pair significantly impacts the attack performance.



(a) TPR@low FPR for k=4

(b) TPR@low FPR for k=8

(c) TPR@low FPR for k=12

Figure 4.3: ROC curves for RaMIA_GT showing the effect of varying group overlap m for fixed group sizes: (a) k=4, (b) k=8, and (c) k=12. These plots visualize the trade-off between redundancy (m) and performance, also visually complement the numerical results presented in Table 4.4.

To select the best (k, m) pair, their balance is critical. The parameter k determines the number of queries sent to the target model, and m controls the redundancy in the group testing design. A poor balance between these two can hurt the overall performance of RaMIA_GT.

Analyzing the data in Table 4.4 and Figure 4.3 together, we can observe its impact.

Table 4.4: TPR@ low FPR (%) for RaMIA_GT across different k and m settings.

	m=2		**m=3**		**m=4**	
TPR@FPR(%)	**1.0%**	**0.1%**	**1.0%**	**0.1%**	**1.0%**	**0.1%**
k= 4	2.08 ± 0.64	0.33 ± 0.29	1.91 ± 0.74	0.19 ± 0.16	1.96 ± 0.16	0.20 ± 0.02
k= 8	2.52 ± 1.12	0.65 ± 0.39	3.34 ± 1.02	1.03 ± 0.80	3.03 ± 1.23	0.65 ± 0.62
k= 12	2.54 ± 1.18	0.82 ± 0.48	1.98 ± 1.10	0.26 ± 0.17	3.38 ± 0.61	0.26 ± 0.33

For $m = 3$, increasing k from 4 to 8 results in a significant performance improvement, with the TPR at 1% FPR jumping from 1.91% to 3.34%. This is because using more groups provides the decoder with more granular information, making it easier to infer the membership status of the underlying samples correctly. However, this trend does not continue indefinitely. When k is increased further to 12, the performance unexpectedly drops to 1.98%. A possible explanation is that with a fixed range size of 50 augmentations, creating too many groups makes each group smaller and less distinct. This can lead to noisy or less representative group selections, which diminishes the quality of the information passed to the decoder and ultimately hurts performance. On the other hand, for $k = 8$, increasing the overlap from $m = 2$ to $m = 3$ improves the TPR from 2.52% to 3.34%. This suggests that the additional redundancy helps the decoder make more confident and accurate decisions. However, when we increase the overlap further to $m = 4$, the performance decreases slightly to 3.03%. This indicates that there are diminishing returns to adding more redundancy. Too much overlap may not provide new, independent information and can make the groups too similar to one another, which does not further aid the decoder.

The interplay between k and m: Based on our results from Table 4.4, where $k = m$ (*e.g.*, $k = 4, m = 4$), we can observe an extreme case, where every augmented sample is placed into every single group. As a result, all four groups would be identical, and all four queries would test the exact same set of samples. This defeats the purpose of group testing, as there is no independent information to cross-reference; if one test fails, they all fail. Nevertheless, for $k = 4$, increasing the overlap from $m = 2$ to $m = 3$ actually lowers the performance (from 2.08% to 1.91% TPR). With only four groups available, $m = 3$ makes the groups too similar to one another.
Thus, the best configuration comes down to two pairs: $(k = 8, m = 3)$ and $(k = 12, m = 4)$. While the $(12, 4)$ setting achieves a slightly higher TPR at 1% FPR (3.38%*vs.*3.34%), we selected $(8, 3)$ as the optimal configuration. This decision is based on the crucial principle of maintaining consistent and reliable performance across different security standards. For a privacy auditing tool to be trustworthy, it must perform well not just under average conditions, but especially under strict ones. The 0.1% FPR threshold represents a significantly higher standard of evidence

than the 1% threshold, meaning we are allowing very few false accusations. At this stricter level, the $(12, 4)$ configuration's performance collapses to a mere 0.26% TPR. In contrast, the $(8, 3)$ configuration remains strong and dependable, delivering a much higher 1.03% TPR. A tool that is effective under one setting but fails under a stricter one is not reliable. Furthermore, the $k = 12$ setting requires 50% more queries than the $k = 8$ setting (12 queries vs. 8). The minor benefit that $(12, 4)$ offers at the 1% FPR level does not justify this significant increase in cost, especially given its unreliability at the more critical 0.1% FPR level.

5

Discussion

This chapter presents a comprehensive analysis of the performance evaluation of the proposed method, RaMIA_GT, which aims to reduce the number of queries sent to the target model without degrading the overall attack performance. We demonstrate how an increased number of ranges can impact attack performance and the role of RaMIA_GT when the range size is expanded. This chapter examines the significance of the query-performance trade-off, discusses the insights gained from the hyperparameter tuning process, and outlines the limitations of this study, as well as the corresponding directions for future research.

5.1 Result Analysis

To evaluate the effectiveness of our proposed method, we have experimented with 50 augmentations assembled into 8 groups, where the degree of overlap is $m = 3$, and compared it with the baseline RaMIA_BL. For this setting, the results demonstrate that integrating group testing achieves a remarkable reduction in query overhead and maintains a decent attack performance. RaMIA_BL reported a TPR of 3.92% @ 1.0% FPR and 1.31% at a 0.1%, where RaMIA_GT maintained a comparable TPR of 3.34% and 1.03% under the same stringent conditions. This outcome represents the central success of this work, where we have shown it is possible to make the attack computationally feasible and stealthy. From Figure 4.2, we also see that the performance of the baseline RaMIA_BLe with fewer number of queries drops to 1.19% TPR@ 1.0% FPR and 0.37% TPR@ 0.1% FPR, whereas under the same conditions, RaMIA_GT gives a more stable performance close to the baseline with 50 augmentations. Hence, we can visually validate how the performance of the baseline improves when the range size is increased, With a more intelligent strategy of grouping samples based on similar features, RaMIA_GT demonstrates that even with an increased range size, the attack can be more potent and stealthy by reducing

the number of queries sent to the target model.

5.2 Choosing the Optimal k and m

Based on our experiments, it seemed that RaMIA_GT requires careful considerations while designing the test groups, followed by a balanced degree of overlap. It is crucial to note that these two parameters highly depend on the actual range size n. For instance, in our experiments with 50 augmentations, if we select $(k, m) = (25, 1)$, there will be many small groups that may not carry enough similarity among the samples. Overpartitioning of groups with sporadic selection of overlap will result in noisy groups with no systematic information retrieved, consequently hurting the overall performance.
Our results suggest that a reasonable starting point for k is a value between roughly 15% and 25% of n. For our $n = 50$ samples, this suggests a k between 7 and 12, which aligns perfectly with our finding that $k = 8$ is optimal. Second, once a range for k is established, the overlap m should be chosen to provide robust redundancy without making the groups overly similar. An overlap m that is too close to k offers little new information. A good rule of thumb is to set m to be around one-third to one-half of k. In our optimal case of $k = 8$, this suggests an m between 2 and 4. Our empirical result for $m = 3$ fits squarely within this recommendation. In conjecture, for a given n, one should start by selecting a k that is a small fraction of n, and then select an m that is a fraction of k. This provides a principled starting point for experimentation.

5.3 Future Work

In the proposed method, the decoder's runtime is exponential to the number of test groups (k) and the number of groups a single sample belongs to (m). This makes the decoding step computationally infeasible if we increase the range size, as it will also require a higher number of test groups and a greater degree of overlap accordingly. This creates an opportunity to use a suboptimal decoder that can handle such computational complexity. We have tried to integrate a novel soft-decision iterative decoder for quantitative group testing proposed by [16]. However, we were unable to establish this integration successfully, as this decoder does not support noisy group tests. This leaves room for future work that can address a more compatible, suboptimal decoder with fewer computational demands.

5.4 Ethical Considerations

This thesis is grounded in ethical intent with the primary objective of auditing and improving the privacy of ML models. While the development of a more efficient attack strategy may initially seem ethically sensitive, our goal is not to provide tools for malicious actors but to better understand and reveal hidden vulnerabilities from the perspective of an attacker. The core ethical argument for this research is based on the principles of transparency and awareness, where vulnerabilities are proactively discovered and disclosed to foster stronger defenses. Adversaries, who are often more motivated and better resourced than defenders, will inevitably discover and exploit such efficient techniques independently. In such a scenario, the attack would be deployed silently, and defenders would not even know what to look for. By publicly documenting and analyzing a more stealthy and efficient attack methodology like RaMIA_GT, we aim to help defenders more than adversaries.

By integrating our method into the LeakPro privacy auditing tool, we provide defenders with a practical way to assess how well their models withstand stronger attacks, such as the ones we studied. This means our research does not just remain theoretical; it is also applicable in practical scenarios that can help improve ML model security. Given the rapid pace of advancement in attack strategies, it is difficult to anticipate the form or strength of future threats. However, by continuously benchmarking models against increasingly potent attacks, such as RaMIA_GT, we can estimate an evolving upper bound on potential data leakage. This proactive evaluation, constrained by current knowledge, helps practitioners remain one step ahead in securing ML models. Our contribution is not the attack concept itself, which is a known vulnerability, but rather a novel, query-efficient methodology for executing it, which serves to make privacy auditing more practical and effective.

6

Conclusion

This thesis addressed a fundamental challenge of RaMIA. While increasing the size and diversity of the sample range can potentially boost the effectiveness of membership inference attacks by capturing stronger privacy signals, it also introduces a major drawback of requiring a separate query for every single sample. Hence, the attack becomes expensive and impractical for an expanded range size. To resolve this issue, this thesis introduces and evaluates a new approach that addresses this limitation by integrating range queries with group testing techniques, effectively reducing the number of queries to the target model while preserving attack performance and making the attack more stealthy. Instead of querying every data point, this new method uses a more intelligent approach to group similar samples, query only a small number of representatives, and then infer membership for the entire range.

The main contribution of this work is therefore a more feasible and efficient method of RaMIA for auditing the privacy of ML models. By making the attack significantly more stealthy, this new method allows auditors to investigate privacy leakage in ML models more thoroughly. It provides an effective way to audit ML models in realistic settings, where the number of queries may be limited, and repeated probing could trigger security defenses. This research also highlights that the effectiveness of RaMIA_GT depends on the careful selection of its parameters, specifically the number of test groups and their degree of overlap. We have established a practical recommendation for choosing these values, providing a reasonable starting point for future experiments. Moreover, this study highlights the scalability challenge of the decoding step, which becomes a bottleneck as the range size increases. We outline this challenge and suggest directions for addressing it in future research.

In conclusion, this thesis successfully develops and validates a novel approach that significantly advances the state of privacy auditing. By integrating this method into the LeakPro framework, we believe this will benefit the auditors in assessing privacy in ML models more thoroughly and support the development of stronger defensive

mechanisms in the future.

Bibliography

[1] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE Symposium on Security and Privacy (SP)*, 2017, pp. 3–18. DOI: 10.1109/SP.2017.41.

[2] J. Tao and R. Shokri, *Range membership inference attacks*, 2024. arXiv: 2408.05131 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2408.05131.

[3] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy risk in machine learning: Analyzing the connection to overfitting," in *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, 2018, pp. 268–282. DOI: 10.1109/CSF.2018.00027.

[4] S. Zarifzadeh, P. Liu, and R. Shokri, *Low-cost high-power membership inference attacks*, 2024. arXiv: 2312.03262 [stat.ML]. [Online]. Available: https://arxiv.org/abs/2312.03262.

[5] W. Cukierski, *Cifar-10 - object recognition in images*, https://kaggle.com/competitions/cifar-10, Kaggle, 2013.

[6] N. Carlini, F. Tramer, E. Wallace, *et al.*, *Extracting training data from large language models*, 2021. arXiv: 2012.07805 [cs.CR]. [Online]. Available: https://arxiv.org/abs/2012.07805.

[7] A. Salem, G. Cherubin, D. Evans, *et al.*, *Sok: Let the privacy games begin! a unified treatment of data inference privacy in machine learning*, 2023. arXiv: 2212.10986 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2212.10986.

[8] N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, and F. Tramer, *Membership inference attacks from first principles*, 2022. arXiv: 2112.03570 [cs.CR]. [Online]. Available: https://arxiv.org/abs/2112.03570.

[9] R. Dorfman, "The detection of defective members of large populations," *The Annals of Mathematical Statistics*, vol. 14, no. 4, pp. 436–440, 1943, ISSN: 00034851. [Online]. Available: http://www.jstor.org/stable/2235930 (visited on 05/03/2025).

[10] M. Xhemrishi, J. Östman, A. Wachter-Zeh, and A. G. i. Amat, "Fedgt: Identification of malicious clients in federated learning with secure aggregation," *IEEE Transactions on Information Forensics and Security*, vol. 20, pp. 2577–2592, 2025. DOI: 10.1109/TIFS.2025.3539964.

[11] Y.-J. Lin, C.-H. Yu, T.-H. Liu, C.-S. Chang, and W.-T. Chen, "Positively correlated samples save pooled testing costs," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 3, pp. 2170–2182, 2021. DOI: 10.1109/TNSE.2021.3081759.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, Ieee, 2009, pp. 248–255.

[14] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. 11, pp. 2579–2605, 2008.

[15] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the royal statistical society: series c (applied statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[16] M. Xhemrishi, J. Östman, and A. Graell i Amat, "Soft-decision decoding for ldpc code-based quantitative group testing," in *2025 14th International ITG Conference on Systems, Communications and Coding (SCC)*, 2025, pp. 1–6. DOI: 10.1109/IEEECONF62907.2025.10949110.