SEDMA: Self-Distillation with Model Aggregation for Membership Privacy

Tsunato Nakai
Mitsubishi Electric Corporation
Kamakura, Kanagawa, Japan
Nakai.Tsunato@dy.MitsubishiElectric.co.jp

Ye Wang
Mitsubishi Electric Research Laboratories
Cambridge, MA, USA
yewang@merl.com

Kota Yoshida
Ritsumeikan University
Kusatsu, Shiga, Japan
y0sh1d4@fc.ritsumei.ac.jp

Takeshi Fujino
Ritsumeikan University
Kusatsu, Shiga, Japan
fujino@se.ritsumei.ac.jp

ABSTRACT

Membership inference attacks (MIAs) are important measures to evaluate potential risks of privacy leakage from machine learning (ML) models. State-of-the-art MIA defenses have achieved favorable privacy-utility trade-offs using knowledge distillation on split training datasets. However, such defenses increase computational costs as a large number of the ML models must be trained on the split datasets. In this study, we proposed a new MIA defense, called SEDMA, based on self-distillation using model aggregation to mitigate the MIAs, inspired by the model parameter averaging as used in federated learning. The key idea of SEDMA is to split the training dataset into several parts and aggregate multiple ML models trained on each split for self-distillation. The intuitive explanation of SEDMA is that model aggregation prevents model over-fitting by smoothing information related to the training data among the multiple ML models and preserving the model utility, such as in federated learning. Through our experiments on major benchmark datasets (Purchase100, Texas100, and CIFAR100), we show that SEDMA outperforms state-of-the-art MIA defenses in terms of membership privacy (MIA accuracy), model accuracy, and computational costs. Specifically, SEDMA incurs at most approximately 3 - 5% model accuracy drop, while achieving the lowest MIA accuracy in state-of-the-art empirical MIA defenses. For computational costs, SEDMA takes significantly less processing time than a defense with the state-of-the-art privacy-utility trade-offs in previous defenses. SEDMA achieves both favorable privacy-utility trade-offs and low computational costs.

KEYWORDS

membership inference attacks, privacy-preserving machine learning, self-distillation, model aggregation

1 INTRODUCTION

Machine learning (ML) has been widely used in a lot of areas, such as predictive analytics, image recognition, and natural language processing, where the input may include privacy-sensitive data. However, recent work has shown that ML models tend to memorize information from the training data (due to over-fitting), which poses serious privacy risks when training data includes privacy-sensitive information [4, 33, 44]. Membership inference attacks (MIAs), where an adversary aims to identify whether a target sample was used to train an ML model based on model behavior, are one of the most fundamental privacy attacks against ML models [12, 33]. The attacks can pose a serious privacy threat as they reveal privacy-sensitive information related to training data. For example, in the case of an ML system for hospital health analytics, an adversary can reveal that a victim was once a patient in the hospital (the victim's data was used for a training data of the ML system) by the MIAs. In addition, MIAs have also been applied to data extraction attacks [4] and the privacy-preserving assessment of ML models [15, 25].

MIA adversaries can obtain privacy-sensitive information related to the training data by simply accessing a prediction API in ML systems (called black-box MIAs), even if the providers of the ML systems are trusted. Black-box MIAs can be categorized into single-query [3, 23, 34, 36, 43–45] and label-only attacks [7, 19, 20]. Single-query attacks directly query a target model with only a target sample, typically to identify the target sample as members of the training data (used in training) or non-members (not used in training). Label-only attacks indirectly query a target model with multiple samples in the neighborhood of a target sample to identify the target sample as members or non-members.

MIA defenses need to design ML models to behave similarly between a sample of members and non-members because MIAs identify the sample as members or non-members based on the different model behavior caused by whether or not the sample is included in the training data [1, 8, 14, 16, 23, 32, 38]. MIA defenses can be categorized into provable privacy defenses and empirical membership privacy defenses, as shown in Table 1. Provable privacy defenses typically use differential privacy mechanisms to provide provable privacy guarantees for the all inputs of ML models. However, differential privacy-based defenses, such as DP-SGD [1], have been reported to significantly degrade model utility in a lot of ML models. Empirical membership privacy defenses aim to preserve the model utility and provide privacy empirically evaluated through practical MIAs without provable privacy guarantees. This study focuses on empirical membership privacy defenses and proposes a new defense strategy that has better trade-offs among membership

Proceedings on Privacy Enhancing Technologies 2024(1), 494–508

https://doi.org/10.56553/popets-2024-0029

Table 1: Two categories of MIA defenses: provable privacy-defenses and empirical membership privacy-defenses. The provable privacy defenses provide provable privacy guarantees however degrade model utility. The empirical membership privacy defenses maintain high model utility, but cannot provide provable privacy guarantees.

	Low model utility	High model utility
Provable privacy	DP-SGD [1]	No defences, so far (challenging)
Empirical membership privacy	-	AdvReg [23], MemGuard [16], KCD [8], SELENA [38], **SEDMA (Our defense)**

privacy, model utility, and computational costs, compared to state-of-the-art empirical defenses, such as AdvReg [23], MemGuard [16], KCD [8], and SELENA [38].

In this study, we introduce a new empirical membership privacy defense called SEDMA [1]. Our goal is to mitigate practical black-box MIAs, maintain high model utility, and have low computational costs. First, similar to KCD [8], SEDMA splits a training dataset into several subsets and trains multiple ML models (called sub-models) on each subset. Second, the trained sub-models are aggregated into several pairs (called model aggregation). Model aggregation means averaging the model parameters, such as performed in federated learning [21]; however, it is different from federated learning in that it is performed among several pairs of sub-models.

Next, SEDMA performs inference with several aggregated models on the subset not used in the training of each sub-model and obtains their corresponding prediction vectors. These prediction vectors were used as the soft labels for the original training dataset. In general, SEDMA trains a protected model on a soft-labelled training dataset. This process is called self-distillation, transferring prediction vectors (knowledge distillation) between ML models with the same model architecture. The state-of-the-art MIA defenses, KCD [8] and SELENA [38], have achieved favorable privacy-utility trade-offs by self-distillation on split training datasets. Note that the novelty of SEDMA lies in the approach of applying model aggregation in several pairs of sub-models for self-distillation. The intuitive explanation of SEDMA is that model aggregation prevents over-fitting by smoothing information of the original training data included in the trained models among the sub-models and preserving model utility such as in federated learning. In addition, SEDMA reduces the computational costs by generating sub-models to combine sub-models by using model aggregation, unlike SELENA which has state-of-the-art privacy-utility trade-offs in previous empirical defenses.

In our experiments, we evaluated SEDMA on three benchmark datasets (Purchase100, Texas100, and CIFAR100). We compared SEDMA with four existing empirical MIA defenses [8, 16, 23, 38], including KCD and SELENA. We conducted two types of black-box MIAs, single-query attacks (NN-based attacks and metric-based attacks) and label-only attacks (boundary distance attacks, data augmentation attacks, and likelihood ratio attacks) as existing attacks that have been used for evaluation in related work.

The experiments show that our defense achieves the lowest MIA accuracy in four existing MIA defenses (around 52%). Nevertheless, SEDMA incurs only a little drop model accuracy (at most approximately 3 - 5% drop compared to undefended models). Moreover, we discuss the best hyperparameters of SEDMA for favorable trade-offs, the computational costs of SEDMA, and the comparison with provable privacy defenses. For computational costs, compared to SELENA which has state-of-the-art favorable trade-offs, SEDMA takes significantly less processing time (only a seventh of the time on three benchmark datasets).

1.1 Contributions

In summary, the key contributions of this paper are as follows:

- We propose SEDMA, which is a new defense mechanism against MIAs by self-distillation with model aggregation. The novelty of SEDMA lies in the approach that reduces overfitting by smoothing information related to the training data among sub-models with model aggregation, similar to the model averaging performed in federated learning [21].
- We show that SEDMA mitigates practical black-box MIAs, maintains high model utility, and has low computational costs. We also demonstrate that SEDMA's performance depends on the content ratio of the original training data for a sub-model and discuss the best hyperparameters of SEDMA for favorable privacy-utility trade-offs. SEDMA has the advantage of reducing overfitting and adjusting the content ratio by model aggregation, while saving computation time.
- We evaluated SEDMA on three benchmark datasets (Purchase100, Texas100, and CIFAR100) with single-query and label-only attacks. We demonstrated that SEDMA outperforms state-of-the-art empirical defenses. SEDMA achieves the lowest MIA accuracy in the previous defenses (around 55%), with only a model accuracy drop of approximately 3 - 5% compared to undefended models.

2 PRELIMINARIES

In this section, we introduce our threat model and provide an overview of prior MIAs and MIA defenses. Based on the threat model and previous work, we present our goals.

2.1 Threat Model

We assume black-box MIAs, in which an adversary has black-box access to a target model. An adversary cannot access the target model parameters directly, however, it can query the target model through a prediction API and obtain the corresponding prediction vectors and / or labels. Thus, the adversary can perform single-query attacks with access to both prediction vectors and labels, or label-only attacks with access to only prediction labels. This threat model is standard black-box MIAs and is typically used for the evaluation of MIAs in prior studies.

We also assume that the adversary knows a small subset of the training dataset for the target model, similar to the assumption in prior works [23, 34, 38]. In other words, the adversary knows

[1] SElf-Distillation using Model Aggregation

Table 2: Two categories of black-box MIAs: single-query and label-only attacks. Single-query attacks directly query a target model with only a target sample typically. Label-only attacks indirectly query a target model with multiple samples in the neighborhood of a target sample.

Type	Typical Attacks
Single-query attacks	NN-based attacks [23, 33], Metric-based attacks (Correctness [45], Confidence [36, 44], Entropy [33], Modified entropy [34], Likelihood ratio [3, 43])
Label-only attacks	Boundary distance attacks [7, 19], Data augmentation attacks [7]

several members of the target model. In addition, we assume that the adversary knows the architecture of the target model, in case of NN (Neural Network) -based attacks [23, 33]. The adversary's goal is to identify the victim's data as members or non-members of the target model.

2.2 Related Work

We introduce an overview of prior MIAs and MIA defenses.

2.2.1 Membership Inference Attacks (MIAs). MIAs can identify whether a target sample is a member based on prediction vectors and / or labels from the target model. MIAs are typically studied in a black-box scenario in which the adversary does not know the target model parameters [23, 33]. Black-box MIAs can be performed either by identifying a subset of the training dataset [23] or constructing shadow models from a dataset with the same distribution [33]. Black-box MIAs can be classified into two categories: single-query and label-only attacks, as shown in Table 2. There have been a lot of types of single-query attacks in prior studies [34, 36, 44, 45].

Single-query attacks, such as NN-based attacks or metric-based attacks, issue a query directly to a target model and use prediction vectors and labels to identify the query as members or non-members.

NN-based attacks [23, 33] build a neural network model I_{NN} for membership inference by using prediction vectors and labels of a target model. The adversary identifies a target sample as members or non-members with I_{NN}.

Metric-based attacks [23, 33, 34, 36, 44, 45] use the metrics, such as correctness (correctness-based attacks), confidences (confidence-based attacks), or entropy (entropy-based attacks or modified entropy-based attacks) of the prediction results from the target model to identify the target sample as members or non-members.

Correctness-based attacks [45] assume that a sample with correct prediction is likely to be a member. The attacks identify a target sample as members or non-members based on the difference between training accuracy and testing accuracy of a target model. The adversary builds the membership inference classifier (members as 1 and non-members as 0) I_{corr} to the target model F as follows:

$$I_{\mathrm{corr}}(F(x), y) = 1\{\arg\max_{\mathrm{i}} F(x)_i = y\}, \tag{1}$$

where y is the correct label of the input sample $\boldsymbol{x}$.

Confidence-based attacks [36, 44] exploit the fact that member's prediction confidence $F(\boldsymbol{x})_y$ is typically higher than non-member's one. The attacks identify a target sample as members when the prediction confidence is higher than either a class-dependent threshold τ_y or a class-independent threshold τ. The membership inference classifier I_{conf} for the attacks is as follows:

$$I_{\mathrm{conf}}(F(x), y) = 1\{F(x)_y \geq \tau_{(y)}\}. \tag{2}$$

Entropy-based attacks [33] exploit the fact that member's prediction entropy is typically lower than that of non-members. The attacks identify the target sample as a member when the prediction entropy is lower than either a class-dependent threshold τ_y or a class-independent threshold τ. The membership inference classifier I_{entr} for the attacks is as follows:

$$I_{\mathrm{entr}}(F(x), y) = 1\left\{-\sum_{\mathrm{i}} F(x)_i \log(F(x)_i) \leq \tau_{(y)}\right\}. \tag{3}$$

Modified entropy-based attacks [34] use a metric to combine prediction entropy and correct labels to improve entropy-based attacks. The attacks exploit the fact that member's values of the modified entropy metric $\mathrm{Mentr}(F(x), y)$ is typically lower than that of non-members. The adversary identifies a target sample as members when the modified entropy metric is lower than either a class-dependent threshold τ_y or a class-independent threshold τ. The membership inference classifier I_{mentr} for the attacks is as follows:

$$\begin{aligned}\mathrm{Mentr}(F(x), y) = &-(1 - F(x)_y)\log(F(x)_y) - \\ &\sum_{i \neq y} F(x)_i \log(1 - F(x)_i),\end{aligned} \tag{4}$$

$$I_{\mathrm{mentr}}(F(x), y) = 1\{\mathrm{Mentr}(F(x), y) \leq \tau_{(y)}\}. \tag{5}$$

Likelihood ratio attacks [3, 43] use a ratio of likelihood of a target sample on a target model and a reference model trained on samples from population distribution, instead of a loss of the target model. Carlini et al. [3] propose likelihood ratio attacks (LiRA), which achieves the state-of-the-art attack performance. LiRA trains N shadow models, of which half are IN models and the other half are OUT models, and fit Gaussians to the confidences of the IN and OUT models. The adversary measures the likelihood of the confidence of the target sample under each distribution of the IN and OUT models and identifies a target sample as members or not by whichever is more likely. Jiayuan et al. [43] propose Attack R, which is similar to LiRA and achieves higher performance than LiRA.

Label-only attacks, such as boundary distance attacks or data augmentation attacks, are based on the fact that a member's sample influences prediction vectors and labels on both itself and the other samples in its neighborhood [20]. The attacks exploit the fact that a target model is more likely to correctly classify samples around members' data than samples around non-members' data [7, 19]. The adversary issues multiple queries around a target sample indirectly to the target model and uses the prediction labels from the target model to identify the target sample as members or non-members. Therefore, obfuscating prediction confidence [16, 42], is not a defense against label-only attacks.

Boundary distance attacks [7, 19] assume that samples around members are more likely to correctly classify than samples around non-members. The attacks exploit the fact that the distance to the classification boundary for members is larger than that for non-members. The adversary computes the distance to the classification boundary by using samples added with small noise to a target sample or adversarial examples crafted to a target sample under the black-box scenario [2, 5, 6].

Data augmentation attacks [7] use data augmentation techniques in computer-vision for boundary distance attacks. The adversary computes the distance to the classification boundary by using augmented target samples with translations, rotations, or flips.

2.2.2 MIA Defenses. MIA defenses can be categorized into provable privacy defenses using differential privacy and empirical membership privacy defenses designed specifically to mitigate MIAs.

Provable privacy defenses which are based on differential privacy in ML, such as DP-SGD [1], add noise to ML models during the training process. However, applying differential privacy to ML models involves trade-offs between model accuracy loss and privacy guarantees [15, 29]. Improving the model accuracy loss are actively discussed in the provable privacy defenses.

PATE [26, 27] is a defense that leverages knowledge distillation with public data and differential privacy for a provable privacy guarantee. Knowledge distillation transfers knowledge of a model (teacher) to another model (student) by using the prediction output of the teacher model [11]. This defense splits training dataset for sub-models and adds noise to the trained sub-models to label public data for a student model. However, in a lot of realistic scenarios, it is difficult to prepare public data corresponding to original privacy-sensitive training data.

Currently, the state-of-the-art defense has shown that the model accuracy is significantly degraded on benchmark datasets (the model accuracy is 25% lower for $\epsilon \leqq 3$) [22, 28, 40]. Thus far, it has been difficult to achieve both acceptable utility loss and privacy guarantees with differential privacy [15, 29].

Empirical membership privacy defenses, such as AdvReg [23], MemGuard [16], DMP [32], KCD [8], SELENA [38], and MIAShield [14], aim to preserve model accuracy and provide privacy empirically evaluated through practical MIAs without provable privacy guarantees.

Adversarial Regularization (AdvReg) [23] is a defense that trains an ML model to mitigate different model behavior between members and non-members against MIAs. This defense is based on a game theoretic framework similar to MIAs extended to generative models [9]. The defense optimizes the training of ML models with an objective function of reducing the prediction loss while also minimizing the MIA accuracy.

MemGuard [16] is a defense that obfuscates prediction vectors with noise to confuse membership inference classifiers. This defense maintains the same model accuracy as the undefended model because it only obfuscates prediction vectors without changing the prediction labels. However, it is weak against metric-based attacks and does not defend against label-only attacks [34].

DMP [32] is a defense that leverages knowledge distillation with public data. This defense achieves favorable privacy-utility trade-offs because it can indirectly train an ML model on the privacy-sensitive training data by knowledge distillation with public data. It is difficult to prepare public data as well as PATE in a lot of realistic scenarios.

KCD [8] is a defense that trains multiple sub-models for each combination of split training data and then trains a protected model by distillation using the prediction outputs of the sub-models on one remaining split training data not used for training. This defense achieves favorable privacy-utility trade-offs without preparing public data which is the challenge of DMP.

SELENA [38] is a defense that trains multiple sub-models on each combination of split training data and then trains a protected model by distillation using the ensemble prediction output of the sub-models on the split training data not used for training. This defense differs from KCD in that it averages the multiple predictions outputs of sub-models for distillation. It also achieves favorable privacy-utility trade-offs without preparing public data, such as in KCD.

MIAShield [14] is a defense that trains each sub-model on multiple sub-models on each augmented split training dataset and then provides the ensemble prediction output of the sub-models, except for a sub-model trained on data related to an input in the inference phase. This defense is similar to SELENA; however, it has several limitations, such as applying only to augmentable computer-vision datasets and high-performance computing systems that can deploy multiple sub-models for ensemble prediction.

In addition, several regularization techniques, such as dropout [37], weight decay [41], and early-stopping [35], have been known to mitigate MIAs to a limited extent.

2.3 Our goals

In this study, we propose a defense that has favorable privacy-utility trade-offs and low computational costs against practical black-box MIAs.

2.3.1 Low MIA Accuracy. We aim to mitigate practical black-box MIAs, including both single-query and label-only attacks. Prior studies have often used accuracy of member inference, including both members and non-members, as an MIA evaluation. We focus on correct prediction for members because the adversary aims to identify whether a target sample was used to train an ML model. Thus, we evaluated not only the accuracy but also precision and recall on a dataset with the same number of members and non-members, based on a recent work [30]. Precision indicates the percentage of correct answers among the data inferred to be members, and recall indicates the percentage of correct answers among the members of the data. These metrics, accuracy, precision, and recall, have high values when the risk of membership privacy leakage is high. Let us denote TP as the number of true positives, FN as the number of false negatives, TN as the number of true negatives, and FP as the number of false positives. The accuracy, precision, and recall are given as follows:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}, \tag{6}$$

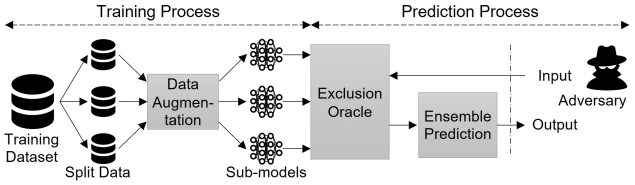

Figure 1: Overview of MIAShield. A privacy-sensitive training dataset is split and augmented. MIAShield trains multiple sub-models on each augmented split data. In the prediction process, the output is an ensemble of prediction results using multiple sub-models.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{7}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{8}$$

Thus, precision and recall about TP are important metrics in terms of the adversary's aim to identify whether a target sample is a member.

2.3.2 High Model Accuracy. Our defense aims to achieve favorable trade-offs between privacy and utility, that is, to protect membership privacy without significantly degrading model accuracy. Prior work has shown that defenses using knowledge distillation on split training datasets, such as KCD [8] and SELENA [38], achieves favorable privacy-utility trade-offs. In addition, these defenses do not require public datasets (PATE [26, 27] and DMP [32] require public data). In this study, we focus on KCD and SELENA as benchmarks to evaluate privacy-utility trade-offs.

2.3.3 Low Computational costs. Our defense aims to achieve low computational costs. Even if a defense has favorable trade-offs between privacy and model accuracy, it is not practical if the additional computational costs for the defense is significant. MIAShield [14] adds computational costs corresponding to the number of sub-models for each inference because it makes inferences by using multiple sub-models trained on each split training data, as shown in Figure 1. However, KCD and SELENA provide one protected model by knowledge distillation, therefore they have no additional computational costs during the inference phase. In addition, it is difficult to deploy MIAShield to embedded devices with limited resources or federated learning which share one trained model [24].

In this study, MIAShield is out of scope and we focus on the computational costs of KCD and SELENA during the training phase. KCD requires additional training of sub-models for each combination of split training datasets, as shown in Figure 2. According to KCD's experiments [8], KCD provides favorable trade-offs with more than 10 sub-models. SELENA also requires additional training depending on the combination of sub-models, as shown in Figure 3. According to SELENA's experiments [38], the 25 sub-models are required for favorable trade-offs. Thus, the defenses using knowledge distillation on split training datasets provide favorable privacy-utility trade-offs, however, the computational costs of training sub-models becomes a main concern.

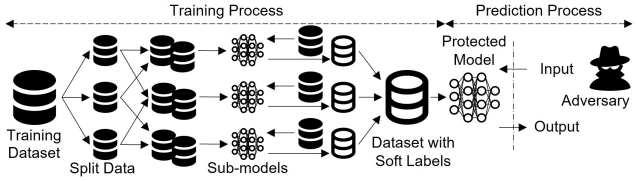

Figure 2: Overview of KCD. KCD makes combinations of split training data and trains multiple sub-models on each combination. The sub-models label prediction results for each one remaining untrained split data and make a dataset with the labeled data (soft labels). In the prediction process, it outputs a prediction result for an input by using a protected model trained on the dataset with soft labels.

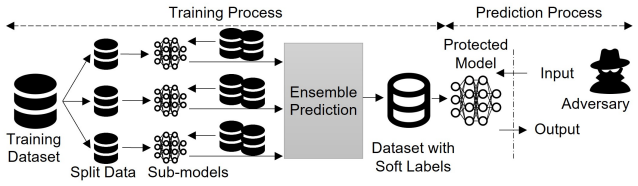

Figure 3: Overview of SELENA. SELENA trains multiple sub-models on each split training data and obtains prediction results with the sub-models for each untrained split data. A protected model is trained on a dataset labeled ensembles of the prediction results. In the prediction process, it outputs a prediction result for an input using a protected model trained on a dataset labelled by ensembled prediction results.

3 OUR PROPOSED DEFENSE

In this section, we present the concept and details of our defense.

3.1 Concept

MIAs result from the fact that ML model behavior differs between members and non-members. For example, ML models exhibit the behaviors, such as different prediction accuracy between members and non-members [45], or higher prediction confidence for members [31, 34, 36, 44]. We propose a new defense strategy to mitigate these differences, based on model aggregation and self-distillation.

An overview of our proposed defense system is shown in Figure 4 and Algorithm 1. Our defense system first splits a privacy-sensitive training dataset D into n subsets $D_1, D_2, ..., D_n$. Sub-models $f_1, f_2, ..., f_n$ are trained on each subsets $D_1, D_2, ..., D_n$. Our defense aggregates the sub-models $f_1, f_2, ..., f_n$ with $\binom{n}{k}$ combinations of k sub-models. Model aggregation refers to averaging the parameters of the models, as used in federated learning. However, it differs from federated learning in that it is performed among each combination of the sub-models. In the case of $k = n - 1$, the sub-models are aggregated into $\binom{n}{n-1} = n$ aggregated models. Our defense obtains prediction results with the aggregate models to each subset not used for training of original sub-models. The prediction results are used for each soft-labelled subset $D'_1, D'_2, ..., D'_n$. Finally, an ML model F is trained on a training dataset D' consisting of the soft-labelled subsets. The

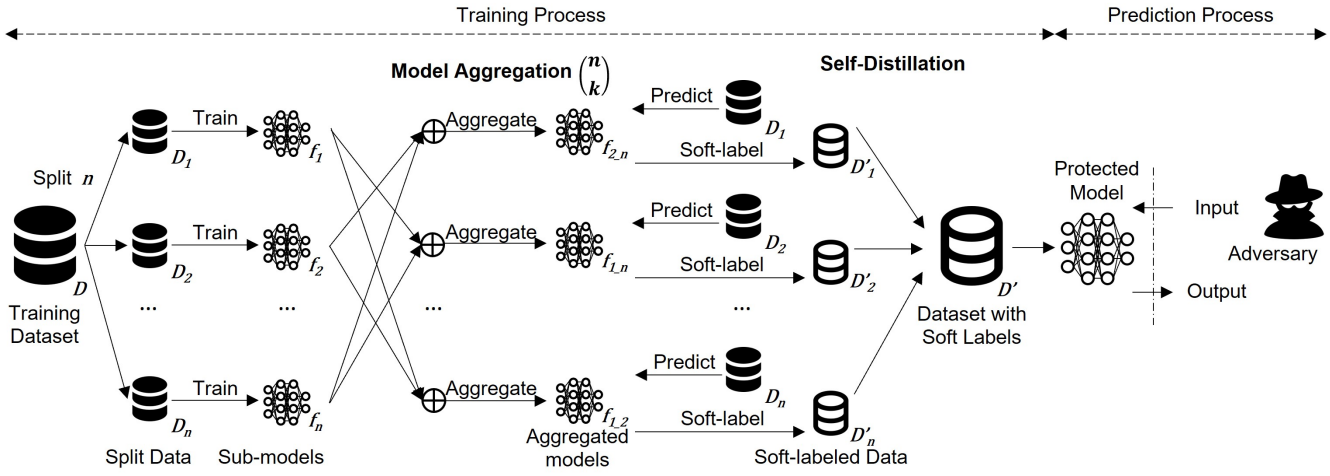

Figure 4: Overview of SEDMA. SEDMA trains multiple sub-models on each split training dataset and aggregates the sub-models into each combination. The aggregated models label the split training data not used for training of the sub-models in each combination. The labeled data is gathered as a dataset with soft labels. In the prediction process, SEDMA outputs a prediction result for an input by using a protected model trained on the dataset with soft labels.

Algorithm 1 Training algorithm of SEDMA

Input: a training dataset $D \subset \{(x, y)|x \in \boldsymbol{R}, y \in \{0, 1\}\}$, a training model f, a loss function L, the number of subsets n, and the number of sub-models aggregated to create each aggregated model k.

Output: a protected model F.

Split D into n disjoint subsets $\{D_i\}_{i=1}^{n}$, such as $D = \bigsqcup_{i=1}^{n} D_i$.

for $i = 1$ to n **do**

Train sub-model f_i by using D_i, $\sum_{(x,y)\in D_i} L(f_i(x), y)$.

end for

Aggregate the trained sub-models $f_1, f_2, ..., f_n$ into $\binom{n}{k}$ combination set M of aggregated models $f'_1, f'_2, ..., f'_{\binom{n}{k}}$,

$f'_{1,2,...,\binom{n}{k}} = \frac{1}{k} \sum_{i\in\{M_{1,2,...,\binom{n}{k}}\}} (f_i)$.

for $i = 1$ to n **do**

Let D'_i be a subset D_i with soft labels, $D'_i = \{(x, f'(x))|\exists y : (x, y) \in D_i\}$.

end for

Train F on a dataset D' consisting of subsets $D'_1, D'_2, ..., D'_n$, $D' = \sum_{i=1}^{n} D'_i$, $\sum_{(x,y)\in D'} L(F(x), y)$.

return F.

model F is expected to be a protected model to mitigate MIAs as it is trained on the new training dataset D' instead of the original privacy-sensitive training data D.

3.2 Description

The key attribute of our defense is to mitigate the over-fitting of a trained model on a privacy-sensitive training dataset by using model aggregation of multiple sub-models trained on split training data. Our defense necessitates that the trained model contains less additional information about the training data. The sub-models can have similar behaviors to that of non-members for the other split dataset, not used for training. Therefore, KCD soft-labels the split data using the sub-model trained on the other split dataset and trains a protected model on the soft-labelled dataset. SELENA soft-labels the split data by using the ensemble prediction results of the sub-models trained in the other split data. Our defense, however, soft-labels the split data by using the aggregated model consisting of the sub-models trained on the other split data. The model aggregation aims to reduce the over-fitting of trained sub-models by smoothing the information of the training data included in the trained models among multiple sub-models while preserving the model accuracy, such as in federated learning. The degree of over-fitting on sub-models differs between our defense and KCD or SELENA, even when we use the same number of sub-models. Our defense generates multiple aggregated models from sub-models by performing model averaging for every few sub-models. Each aggregated model soft-labels to subsets not used for training of the sub-models, which consist of the aggregated model. In KCD, each sub-model soft-labels to one remaining subset excluded during training of the sub-model.

Another key attribute of our defense is that additional computational costs for the defense is lower than that of SELENA. SELENA has the state-of-the-art privacy-utility trade-offs, however, it has additional costs to train a large number of sub-models (approximately 25) to obtain prediction results for an ensemble. Our defense, however, combines sub-models by using model aggregation (simple averaging of model parameters) and therefore only trains on each split training dataset. The training for each combination of the split training datasets, such as in SELENA, is unnecessary. If the model accuracy of the aggregated models is not too high, only a

Table 3: Three datasets for the evaluation of SELENA: Purchase100, Texas100, and CIFAR100. "Train" is the amount of the training data. "Test" is the amount of test data. "Known" is the amount of the training data that the adversary can know and exploit for MIAs. "Target" is the amount of data to infer membership (half of the data is from members, and the other half is from non-members).

Dataset	Train	Test	Known	Target
Purchase100	19,372	19,372	9,866	9,866
Texas100	10,000	5,000	5,000	5,000
CIFAR100	50,000	5,000	5,000	5,000

small amount of additional training is required, because the initial model parameters are based on aggregated parameters from the sub-models. In addition, because our defense does not use an ensemble strategy such as SELENA, a large number of sub-models are not required.

4 EXPERIMENTS

In this section, we introduce the experimental setup and evaluation results compared to undefended models and state-of-the-art defenses.

4.1 Setup

4.1.1 Datasets. We use three benchmark datasets widely used in prior studies on MIAs. The benchmark datasets are Purchase100, Texas100, and CIFAR100, as shown in Table 3. Purchase100 is a dataset provided by Kaggle's Acquire Valued Shoppers Challenge [17]. We use the dataset that Shokri et al. processed [33]. The dataset has 197,324 records with 600 binary features regarding items customers purchased. The data is classified into 100 classes regarding purchase styles. Texas100 is a dataset provided by the Texas Department of State Health Services [39]. We also use the dataset that Shokri et al. processed [33]. The dataset has 67,330 records with 6,170 binary features about patients. The data is classified into 100 classes about procedures that patients underwent. CIFAR100 is a dataset provided as a typical benchmark for image classification algorithms [18]. The dataset has 60,000 images of various objects. The data is classified into 100 classes of object names.

We set the number of training and test data, as shown in Table 3. The adversary's prior knowledge corresponds to half of the training data and half of the test data, according to prior studies [23, 38]. Note that this does not make any adjustments to the datasets, such as removing high-entropy data [8].

4.1.2 Model Architectures. We use a 4-layer fully connected neural network with layer sizes of [1024, 512, 256, 100] for Purchase100 and Texas100, following prior work [23, 38]. For CIFAR100, we used ResNet-18 [10] which is widely used in computer-vision tasks. Each model is trained and tested on the dataset, as listed in Table 3.

4.1.3 MIAs for Evaluations. We conducted single-query and label-only attacks, as shown in Table 2. The results show the median and standard error from the ten runs. Single-query attacks consist of five attacks, and we evaluated defenses based on the most successful attack. For the label-only attacks, boundary distance attacks were conducted. Data augmentation attacks were conducted on CIFAR100 which is the computer-vision task. The code of these attacks is based on the code [2] of SELENA. We used 15 shadow models for LiRA and Attack R and applied both logit and linear scaling to the model's confidence on Attack R. We used accuracy, precision, and recall to evaluate the attacks according to Section 2.3.1. The larger the metrics, the higher the risk of membership privacy leakage.

4.1.4 Defenses for Comparison. We compared SEDMA with MemGuard [16], AdvReg [23], KCD [8], and SELENA [38] as the state-of-the-art defenses from Section 2.2.2. For KCD, we set the number of split training data $n = 10$ and the hyperparameter $\alpha = 1$ which is a setting that protects privacy sufficiently. According to Chourasia et al. [8], the larger n generally implies better privacy and utility, and the performance converges around $n = 10$. For SELENA, we set the number of split training data (sub-models) $K = 25$ and the number of sub-models for the ensemble $L = 10$ with reference to the study's setup [38]. The hyperparameters of SEDMA are the number of split training data n and the number of the sub-model combination for model aggregation $\binom{n}{k}$. We set $n = 7$ and $\binom{n}{k} = \binom{7}{3}$ for SEDMA in the experiments. We discuss the best hyperparameters of our defense in Section 5.2.

4.2 Results

Table 4 summarizes the model accuracy and best attack accuracy, precision, and recall for each attack type, including comparison with undefended models and previous defenses.

4.2.1 Model Accuracy. We first compare our defense with undefended models on MIA accuracy and model accuracy. According to Table 4, the highest MIA accuracy in two types of attacks against undefended models is 68.23% on Purchase100, 67.51% on Texas100, and 74.51% on CIFAR100. However, the MIA accuracy against SEDMA is no higher than 52.67% on Purchase100, 52.46% on Texas100, and 52.76% on CIFAR100. Our defense significantly reduces the risk of membership privacy leakage by approximately 15-20%, compared to the undefended models. The model accuracy of the undefended models on the test dataset is 84.20% on Purchase100, 52.32% on Texas100, and 77.69% on CIFAR100. The model accuracy of SEDMA is 79.55% on Purchase100, 55.18% on Texas100, and 74.35% on CIFAR100. Compared with undefended models, our defense incurs, at most, an accuracy drop of approximately 3 - 5%. Our defense only has a small drop in model utility and achieves a low risk of membership privacy leakage. Therefore, our defense has favorable privacy-utility trade-offs. We next show that our defense achieves better utility-privacy trade-offs compared to previous defenses, such as AdvReg, MemGuard, KCD, and SELENA.

4.2.2 Comparison with Previous Defenses. We compare our defense with previous defenses on MIA accuracy and model accuracy. Figure 5 shows the relationship between the model accuracy and the single-query attack accuracy against undefended models, AdvReg, MemGuard, KCD, SELENA, and SEDMA on (a) Purchase100, (b)

[2] https://github.com/inspire-group/MIAdefenseSELENA

Table 4: Comparison of membership privacy and model accuracy. SEDMA is compared with undefended models and previous defenses on three datasets. We evaluate membership privacy based on the most successful attack in each single-query and label-only attack. The attack results show the median and standard error from the ten runs. The lowest attack accuracy, precision, and recall on each dataset is bold.

Dataset	Defense	Mode accuracy on training data	Model accuracy on test data	Best single-query attack			Best label-only attack		
				Accuracy	Precision	Recall	Accuracy	Precision	Recall
Purchase100	None	99.98%	84.20%	69.00%(0.45)	70.95%(0.92)	63.63%(1.46)	65.84%(0.23)	60.39%(0.17)	92.04%(0.89)
	AdvReg	90.10%	76.83%	62.33%(0.09)	67.47%(0.64)	47.64%(1.22)	58.19%(0.60)	55.79%(0.39)	78.88%(1.31)
	MemGuard	99.98%	**84.20%**	65.26%(0.09)	68.99%(0.62)	55.45%(1.42)	61.98%(0.69)	61.98%(0.69)	93.89%(1.21)
	KCD	82.07%	74.83%	64.34%(0.09)	69.02%(0.62)	52.04%(1.33)	60.08%(0.58)	56.98%(0.41)	82.31%(1.08)
	SELENA	82.85%	79.07%	59.25%(0.16)	64.18%(0.66)	41.86%(1.07)	55.61%(0.55)	55.76%(0.59)	54.34%(0.70)
	SEDMA	83.26%	79.55%	**58.40%(0.20)**	**62.56%(0.68)**	**41.85%(1.07)**	**52.25%(0.61)**	**52.31%(0.64)**	**50.82%(1.00)**
Texas100	None	81.98%	52.32%	72.64%(0.23)	76.37%(0.11)	65.57%(0.50)	62.29%(0.22)	60.38%(0.30)	71.50%(0.55)
	AdvReg	56.94%	46.52%	58.97%(0.01)	80.28%(0.02)	23.78%(0.04)	58.24%(0.45)	57.94%(0.46)	60.14%(0.65)
	MemGuard	81.98%	52.32%	59.96%(0.01)	82.12%(0.02)	25.46%(0.04)	65.68%(0.68)	62.87%(0.68)	76.62%(0.50)
	KCD	52.23%	47.35%	59.43%(0.01)	81.59%(0.02)	24.48%(0.04)	53.61%(0.52)	54.80%(0.74)	**41.24%(0.66)**
	SELENA	58.37%	53.43%	56.89%(0.01)	76.22%(0.02)	20.04%(0.03)	53.90%(0.44)	54.49%(0.51)	47.34%(0.65)
	SEDMA	61.70%	**53.89%**	**56.45%(0.01)**	**75.22%(0.02)**	**19.23%(0.03)**	**52.17%(0.34)**	**52.47%(0.39)**	46.02%(0.63)
CIFAR100	None	99.98%	77.69%	84.96%(3.01)	87.58%(0.86)	81.48%(6.27)	77.77%(0.13)	69.25%(0.13)	99.90%(0.07)
	AdvReg	89.13%	71.39%	63.02%(1.21)	81.78%(1.17)	33.49%(2.58)	59.24%(0.12)	56.37%(0.10)	81.78%(0.09)
	MemGuard	99.98%	**77.69%**	65.06%(1.35)	84.06%(1.05)	37.16%(2.86)	68.34%(0.14)	61.30%(0.10)	99.46%(0.09)
	KCD	76.14%	69.77%	61.33%(1.05)	81.96%(1.16)	29.05%(2.24)	55.28%(0.13)	54.90%(0.13)	**59.18%(0.11)**
	SELENA	78.24%	74.43%	59.34%(0.88)	80.51%(1.22)	24.64%(1.90)	53.90%(0.14)	52.97%(0.11)	69.50%(0.11)
	SEDMA	79.83%	74.25%	**57.91%(1.39)**	**76.12%(2.84)**	**23.04%(2.92)**	**53.03%(0.15)**	**52.50%(0.12)**	63.60%(0.15)

Texas100, and (c) CIFAR100. Figure 6 shows the relationship between the model accuracy and the label-only attack accuracy. The plots towards the upper left in Figure 5 and Figure 6 show better defenses for the utility-privacy trade-offs.

Compared with AvdReg, our defense archives higher model accuracy and lower attack accuracy across all three datasets. The highest MIA accuracy against AvdReg is 62.33% on Purchase100, 58.97% on Texas100, and 63.02% on CIFAR100. Our defense achieves approximately 5-6% lower attack accuracy than AvdReg. The model accuracy of AvdReg is 76.83% on Purchase100, 46.54% on Texas100, and 71.39% on CIFAR100. The model accuracy of our defense is approximately 3 - 9% higher than AvdReg. Our defense has better privacy-utility trade-offs than AvdReg.

Compared with Memguard, our defense has an advantage of a defense against label-only attacks because MemGuard only obfuscates prediction confidence. Note that the results of label-only attacks without prediction confidence against MemGuard are the same as the results of the undefended model. The highest MIA accuracy against MemGuard is 65.26% on Purchase100, 65.58% on Texas100, and 68.37% on CIFAR100. Our defense achieves greater than 10% lower attack accuracy than MemGuard. However, because MemGuard maintains the same model accuracy as the undefended models, our defense has a disadvantage in the model accuracy compared to MemGuard. Our defense only has a small drop in the model accuracy and defends against label-only attacks.

Compared with KCD, our defense achieves higher model accuracy and lower attack accuracy across all three datasets. The highest MIA accuracy against KCD is 64.34% on Purchase100, 59.43% on Texas100, and 61.33% on CIFAR100. The attack accuracy of our defense is approximately 3-6% lower than that of KCD. The model accuracy of KCD is 74.83% on Purchase100, 47.35% on Texas100, and 69.77% on CIFAR100. Our defense achieves approximately 5 - 8% higher model accuracy than KCD. Therefore, our defense has better privacy-utility trade-offs than KCD.

Compared with SELENA, our defense has an advantage in the attack accuracy across all three datasets, although the model accuracy is only a small difference from SELENA. The highest MIA accuracy against SELENA is 59.25% on Purchase100, 56.89% on Texas100, and 59.34% on CIFAR100. Our defense achieves approximately 0.4 - 2% lower attack accuracy than SELENA. The model accuracy of SELENA is 79.07% on Purchase100, 53.43% on Texas100, and 74.43% on CIFAR100. The model accuracy of our defense is 79.55% on Purchase100, 53.89% on Texas100, and 74.25% on CIFAR100. Our defense has the same model accuracy as SELENA on Purchase100 and Texas100 or slightly less on CIFAR100. Our defense is comparable in model accuracy and slightly better than SELENA in attack accuracy. However, there are significant differences between SELENA and our defense in computational costs, which will be discussed in the next section.

We also evaluate the precision and recall of MIAs on three datasets against previous defenses and our defense. Figure 7 shows the relationship between the precision and recall of single-query attacks against undefended models, AdvReg, MemGuard, KCD, SELENA, and SEDMA on (a) Purchase100, (b) Texas100, and (c) CIFAR100. Figure 8 shows the relationship between the precision and recall of label-only attacks. The plots towards the bottom left in Figure 7 and Figure 8 are better defenses for membership privacy.

The precision and recall related to the percentage of correct answers are important metrics in MIAs because the adversary tries to know if a sample is a member. In Figure 7 and Figure 8, because our defense is plotted in the bottom left across all three datasets, it achieves the lowest MIA risk in comparison with previous defenses. Our defense has a slight difference in the attack accuracy compared to SELENA, however, it has a clear advantage in comparison to the

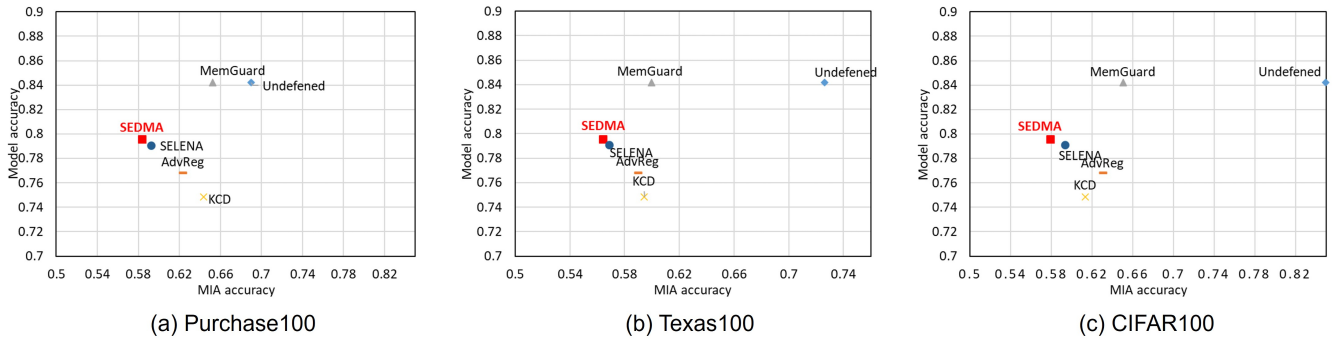

Figure 5: Comparison of the model accuracy and single-query attack accuracy against undefended models, AdvReg, MemGuard, KCD, SELENA, and SEDMA on (a) Purchase100, (b) Texas100, and (c) CIFAR100. The vertical axis is model accuracy on the test dataset from Table 4. The horizontal axis is the MIA accuracy of the best single-query attack from Table 4. The plots towards the upper left show better defenses for the privacy-utility trade-offs.

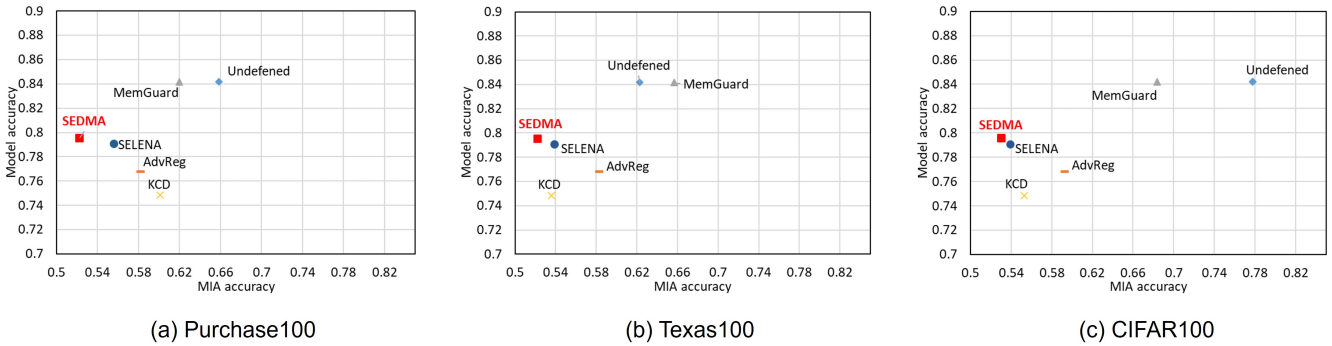

Figure 6: Comparison of the model accuracy and label-only attack accuracy against undefended models, AdvReg, MemGuard, KCD, SELENA, and SEDMA on (a) Purchase100, (b) Texas100, and (c) CIFAR100. The vertical axis is model accuracy on the test dataset from Table 4. The horizontal axis is the MIA accuracy of the best label-only attack from Table 4. The plots towards the upper left show better defenses for the privacy-utility trade-offs.

precision and recall. Therefore, our defense outperforms state-of-the-art empirical defenses in terms of privacy-utility trade-offs.

5 DISCUSSION

In this section, we discuss the best hyperparameters of our defense for favorable privacy-utility trade-offs, the computational costs, the comparison with provable privacy defenses, and limitations of our defense.

5.1 Best hyperparameters

Our defense has the number of split training data n and the combination for model aggregation $\binom{n}{k}$ as hyperparameters. Because our defense splits the training data for the sub-models and aggregates the trained sub-models, the performance against MIAs is expected to depend on the over-fitting of the sub-models. Therefore, we focus on the content ratio of the original training data for a sub-model in terms of the over-fitting of the sub-models. We compare nine different SEDMAs shown in Table 5. The content ratio is determined by the combination $\binom{n}{k}$. For example, in the case of SEDMA_6-3, the content ratio is 3 / 6 = 0.5, which means that 3 of the 6 dataset splits are used to train each sub-model.

Figure 9 shows the relationship between the model accuracy on Purchase100 and the content ratio of the original training data against each different SEDMA. Figure 10 shows the relationship between the attack accuracy of the best single-query attack and the content ratio against each different SEDMA. The model accuracy tends to be increased by the content ratio of the original training data because the sub-models are assumed to fit the training data more. However, the attack accuracy also tends to increase with the content ratio, as shown in Figure 10.

The over-fitting of the sub-models is assumed to be related to the content ratio of the original training data. If the content ratio is high, the sub-model contains more information about the training data. Therefore, we need to set the hyperparameters for SEDMA, considering the trade-offs between attack accuracy and model accuracy due to the content ratio. The best parameter in this study is the combination $\binom{n}{k} = \binom{7}{3}$ with a content ratio of 0.43. We consider

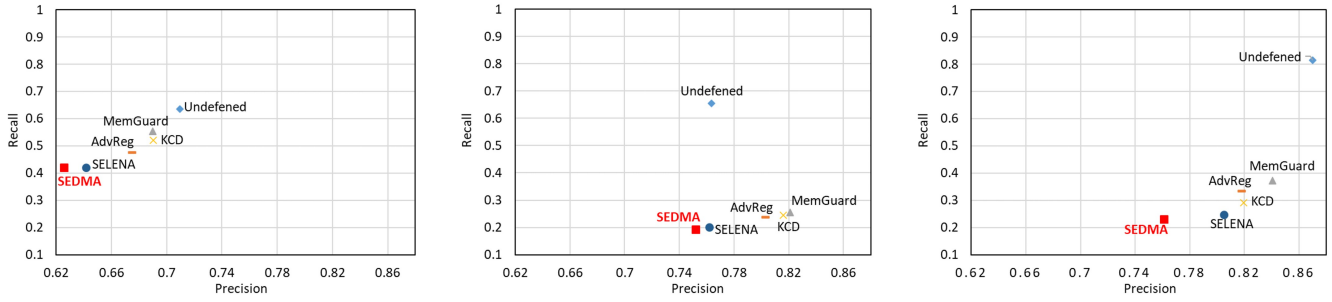

Figure 7: Comparison of precision and recall of single-query attacks against an undefended model, AdvReg, MemGuard, KCD, SELENA, and SEDMA on (a) Purchase100, (b) Texas100, and (c) CIFAR100. The vertical axis is the recall of the best single-query attack from Table 4. The horizontal axis is the precision of the best single-query attack from Table 4. The points towards the bottom left are better defenses for membership privacy.

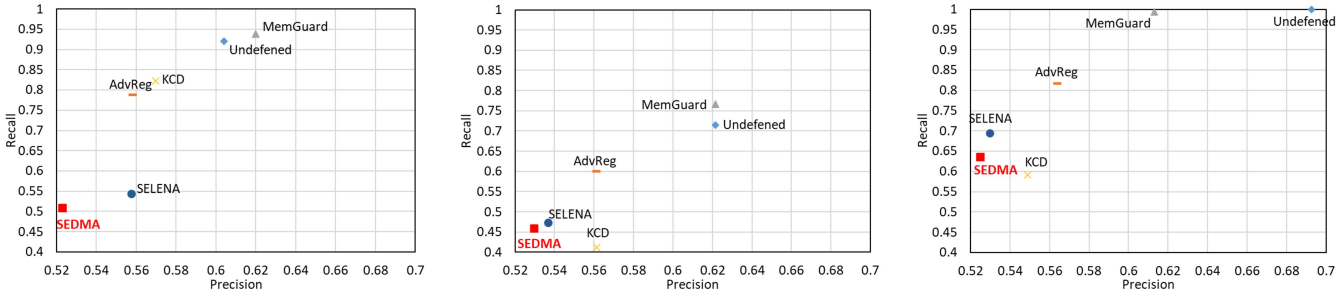

Figure 8: Comparison of precision and recall of label-only attacks against undefended model, AdvReg, MemGuard, KCD, SELENA, and SEDMA on (a) Purchase100, (b) Texas100, and (c) CIFAR100. The vertical axis is the recall of the best label-only attack from Table 4. The horizontal axis is the precision of the best label-only attack from Table 4. The points towards the bottom left are better defenses for membership privacy.

that the hyperparameters of SEDMA depend to some extent on the kinds of training datasets or model architectures.

5.2 Computational costs

Table 6 shows the comparison of the processing time in the training and inference phase of previous defenses and SEDMA on three datasets. We measure the processing time on a GeForce RTX 2080 SUPER in our experimental setup. The processing time is averaged from the three runs in each phase. We set the batch size to 512 for Purchase100, 128 for Texas100, 256 for CIFAR100 in the training phase. The number of epochs for each dataset is 30, 20, and 200, respectively. In the inference phase, we set the batch size to 1 and run 1,000 samples.

In the training phase, our defense takes approximately six times longer than the undefended model. Compared to the previous defenses, our defense requires only a seventh of the processing time than SELENA which has favorable privacy-utility trade-offs. For example, SELENA had the main cost of training 25 sub-models on the 10 split training data in the experiments. However, our defense had the main cost to train only 7 sub-models and aggregate them.

Table 5: Nine different SEDMAs, with varying content ratio, of how much training data is used for training each sub-model. The content ratio depends on the combination for model aggregation $\binom{n}{k}$.

Our defense	Combination $\binom{n}{k}$	Content ratio
SEDMA_5-4	$\binom{5}{4} = 5$	4 / 5 = 0.80
SEDMA_6-2	$\binom{6}{2} = 15$	2 / 6 = 0.33
SEDMA_6-3	$\binom{6}{3} = 20$	3 / 6 = 0.50
SEDMA_6-4	$\binom{6}{4} = 15$	4 / 6 = 0.67
SEDMA_7-2	$\binom{7}{2} = 21$	2 / 7 = 0.29
SEDMA_7-3	$\binom{7}{3} = 35$	3 / 7 = 0.43
SEDMA_7-4	$\binom{7}{4} = 35$	4 / 7 = 0.57
SEDMA_15-14	$\binom{15}{14} = 15$	14 /15 = 0.93

Table 6: Comparison of processing time on an undefended model, AdvReg, MemGuard, KCD, SELENA, and SEDMA on Purchase100. The processing time is the average of three runs on a GeForce RTX 2080 SUPER.

Dataset	Processing Time	Undefended model	AdvReg	MemGuard	KCD	SELENA	**SEDMA**
Purchase100	Training	11.49s	78.80s	12.57s	40.95s	433.45s	69.21s
	Inference	0.46s	0.46s	461.78s	0.46s	0.46s	0.46s
Texas100	Training	13.75s	160.12s	14.20s	42.64s	440.43s	72.43s
	Inference	0.46s	0.47s	460.73s	0.47s	0.46s	0.46s
CIFAR100	Training	1.79h	23.89h	1.78h	9.84h	30.37h	12.20h
	Inference	13.45s	13.88s	505.02s	14.20s	14.03s	13.85s

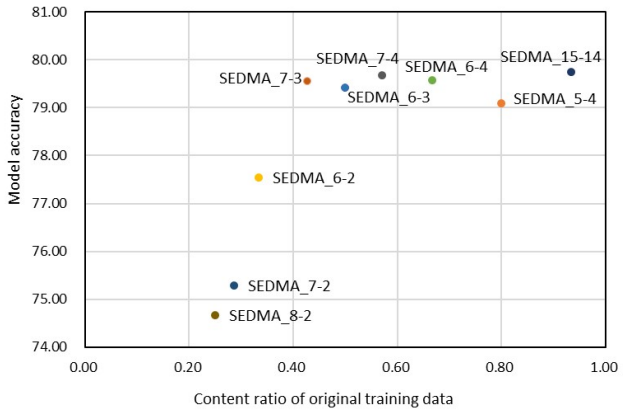

Figure 9: Relationship between the model accuracy and the content ratio against nine different SEDMAs. The vertical axis is the model accuracy on the test dataset of Purchase100. The horizontal axis is the content ratio of the original training data for one sub-model from Table 5.

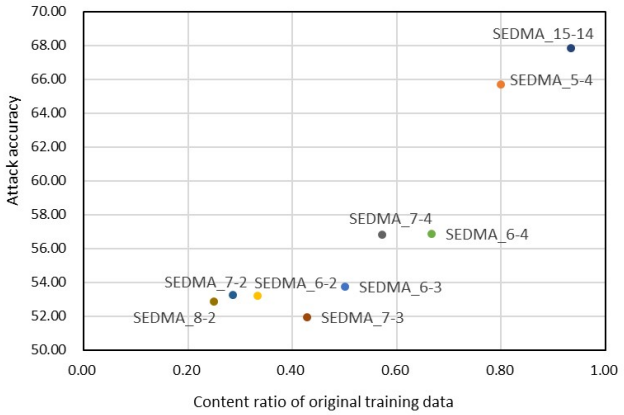

Figure 10: Relationship between the attack accuracy and the content ratio against nine different SEDMAs. The vertical axis is the MIA accuracy of the best single-query attack on Purchase100. The horizontal axis is the content ratio of the original training data for one sub-model from Table 5.

The difference in processing time between SELENA and our defense is due to the number of trained sub-models. Note that SELENA can be accelerated by parallel execution of sub-models [38]; however, this can also be applied in our defense.

KCD is approximately 60 - 80% faster than our defense in the training phase. In the experiments, KCD had the main cost of training 10 sub-models and labels the training data by using the 10 sub-models. However, our defense labelled the training data by using the 35 sub-models which is the combination $\binom{7}{3}$ of the 7 sub-models. The difference in processing time between KCD and our defense is due to the number of sub-models used for labeling. Note that our defense has much better privacy-utility trade-offs than KCD. In addition, MemGuard has costs in the inference phase because it processes prediction vectors of the trained undefended model when outputting inference results. We next discuss the costs in the inference phase.

In the inference phase, the processing time of our defenses is the same as that of the undefended model. The previous defenses, with the exception of MemGaurd, have the same processing time in the inference phase. This is because the protected models trained in the training phase are used for inference, the same as the undefended model. MemGuard is approximately 1,000 times slower than the others because it solves an optimization problem to obfuscate prediction vectors for every input in the inference phase.

In conclusion, our defense achieves both favorable privacy-utility trade-offs and low computational costs. The total computation costs of our defense are lower than those of previous defenses, except for KCD. However, KCD has worse privacy-utility trade-offs than our defense.

5.3 Comparison with Provable Privacy Defenses

We discuss our defense compared with DP-SGD [1] as a provable privacy defense. According to the study [38], DP-SGD ($\epsilon = 4$) results in a model accuracy on Purchase100 of 56.0% and an attack accuracy, for the best single-query, of 52.8%. However, our defense resulted in a model accuracy of 79.6% and an attack accuracy of 52.0% from Table 4. DP-SGD provides a differential privacy guarantee, however, it incurs a significant model accuracy loss of approximately 14% compared to our defense. Compared to the undefended model, DP-SGD incurs an 18% drop in model accuracy The attack accuracy is

slightly different between DP-SGD and our defense. Our defense has no privacy guarantee, however, in terms of empirical privacy guarantees, it only incurs at most approximately 3 - 5% model accuracy drop with a low membership privacy risk, compared to the undefended model.

In conclusion, our defense is not generally comparable to provable privacy defenses, such as DP-SGD, in terms of provable privacy guarantees, however we empirically achieve the best privacy-utility trade-offs.

5.4 Limitations

SEDMA may not be suitable for settings where provable privacy guarantees are necessary because it is an empirical membership privacy defense system. The appropriate use cases are for ML systems that demand high levels of both MIA mitigation and model accuracy. We assume that there may not be enough resources for sub-models and split datasets of SEDMA when training on small embedded devices. SEDMA, on the other hand, has fewer sub-models than SELENA and can be suitable for several embedded devices. The appropriate use cases are for embedded device with resources for repeated training, such as clients in federated learning systems. SEDMA can also be suitable on resource-rich cloud servers.

6 CONCLUSION

In this study, we propose a new defense system using self-distillation with model aggregation against MIAs. Our defense system mitigates the over-fitting of the privacy-sensitive training data by aggregating multiple sub-models trained on split training data. In addition, we achieve low computational costs because our defense trains sub-models and aggregates the combinations of them rather than training a lot of sub-models for self-distillation, such as a state-of-the-art defense, SELENA. We performed the experimental evaluation with major benchmark datasets (Purchase100, Texas100, and CIFAR100) using the black-box MIAs, including single-query and label-only attacks. We demonstrate that our defense outperforms previous defenses in achieving the lowest risk of membership privacy leakage in comparison with previous empirical defenses, such as AdvReg, MemGuard, KCD, and SELENA.

In addition, our defense also achieves favorable privacy-utility trade-offs and low computational costs. Specifically, our defense incurs, at most, a model accuracy drop of approximately 3 - 5% compared to undefended models. That is approximately 5 - 8% higher model than that of KCD and the same as SELENA. Our defense requires only a seventh of the processing time than SELENA on the datasets. Future work will include analyzing our defense performance against practical white-box MIAs [13] and theoretically guaranteeing the privacy of the defense.

REFERENCES

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

[2] Wieland Brendel, Jonas Rauber, and Matthias Bethge. 2017. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248* (2017).

[3] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. 2022. Membership Inference Attacks From First Principles. In *2022 IEEE Symposium on Security and Privacy (SP)*. 1897–1914. https://doi.org/10.1109/SP46214.2022.9833649

[4] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting Training Data from Large Language Models.. In *USENIX Security Symposium*, Vol. 6.

[5] Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*. Ieee, 39–57.

[6] Jianbo Chen, Michael I Jordan, and Martin J Wainwright. 2020. Hopskipjumpattack: A query-efficient decision-based attack. In *2020 ieee symposium on security and privacy (sp)*. IEEE, 1277–1294.

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

[8] Rishav Chourasia, Batnyam Enkhtaivan, Kunihiro Ito, Junki Mori, Isamu Teranishi, and Hikaru Tsuchida. 2022. Knowledge Cross-Distillation for Membership Privacy. *Proc. Priv. Enhancing Technol.* 2022, 2 (2022), 362–377. https://doi.org/10.2478/popets-2022-0050

[9] Jamie Hayes, Luca Melis, George Danezis, and Emiliano De Cristofaro. 2017. Logan: Membership inference attacks against generative models. *arXiv preprint arXiv:1705.07663* (2017).

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

[11] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

[12] Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S. Yu, and Xuyun Zhang. 2022. Membership Inference Attacks on Machine Learning: A Survey. 54, 11s (2022). https://doi.org/10.1145/3523273

[13] Bo Hui, Yuchen Yang, Haolin Yuan, Philippe Burlina, Neil Zhenqiang Gong, and Yinzhi Cao. 2021. Practical blind membership inference attack via differential comparisons. *arXiv preprint arXiv:2101.01341* (2021).

[14] Ismat Jarin and Birhanu Eshete. 2022. MIAShield: Defending Membership Inference Attacks via Preemptive Exclusion of Members. *arXiv preprint arXiv:2203.00915* (2022).

[15] Bargav Jayaraman and David Evans. 2019. Evaluating differentially private machine learning in practice. In *USENIX Security Symposium*.

[16] Jinyuan Jia, Ahmed Salem, Michael Backes, Yang Zhang, and Neil Zhenqiang Gong. 2019. Memguard: Defending against black-box membership inference attacks via adversarial examples. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*. 259–274.

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

[18] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. 2009. *CIFAR-100 (Canadian Institute for Advanced Research)*. http://www.cs.toronto.edu/~kriz/cifar.html

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

[20] Yunhui Long, Vincent Bindschaedler, Lei Wang, Diyue Bu, Xiaofeng Wang, Haixu Tang, Carl A Gunter, and Kai Chen. 2018. Understanding membership inferences on well-generalized learning models. *arXiv preprint arXiv:1802.04889* (2018).

[21] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.

[22] Milad Nasr, Reza Shokri, et al. 2020. Improving deep learning with differential privacy using gradient encoding and denoising. *arXiv preprint arXiv:2007.11524* (2020).

[23] Milad Nasr, Reza Shokri, and Amir Houmansadr. 2018. Machine learning with membership privacy using adversarial regularization. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*. 634–646.

[24] Milad Nasr, Reza Shokri, and Amir Houmansadr. 2019. 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)*. IEEE, 739–753.

[25] Milad Nasr, Shuang Songi, Abhradeep Thakurta, Nicolas Papernot, and Nicholas Carlin. 2021. Adversary instantiation: Lower bounds for differentially private machine learning. In *2021 IEEE Symposium on security and privacy (SP)*. IEEE, 866–882.

[26] Nicolas Papernot, Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. 2016. Semi-supervised knowledge transfer for deep learning from private training data. *arXiv preprint arXiv:1610.05755* (2016).

[27] Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. 2018. Scalable private learning with PATE. *arXiv preprint arXiv:1802.08908* (2018).

[28] Nicolas Papernot, Abhradeep Thakurta, Shuang Song, Steve Chien, and Úlfar Erlingsson. 2021. Tempered sigmoid activations for deep learning with differential

privacy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 9312–9321.
[29] Md Atiqur Rahman, Tanzila Rahman, Robert Laganière, Noman Mohammed, and Yang Wang. 2018. Membership Inference Attack against Differentially Private Deep Learning Model. *Trans. Data Priv.* 11, 1 (2018), 61–79.
[30] Shahbaz Rezaei and Xin Liu. 2021. On the difficulty of membership inference attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7892–7900.
[31] Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. 2018. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *arXiv preprint arXiv:1806.01246* (2018).
[32] Virat Shejwalkar and Amir Houmansadr. 2021. Membership Privacy for Machine Learning Models Through Knowledge Transfer. In *AAAI Conference on Artificial Intelligence*.
[33] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*. IEEE, 3–18.
[34] Liwei Song and Prateek Mittal. 2021. Systematic Evaluation of Privacy Risks of Machine Learning Models.. In *USENIX Security Symposium*, Vol. 1. 4.
[35] Liwei Song and Prateek Mittal. 2021. Systematic Evaluation of Privacy Risks of Machine Learning Models. In *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, 2615–2632. https://www.usenix.org/conference/usenixsecurity21/presentation/song
[36] Liwei Song, Reza Shokri, and Prateek Mittal. 2019. Privacy risks of securing machine learning models against adversarial examples. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 241–257.
[37] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.
[38] Xinyu Tang, Saeed Mahloujifar, Liwei Song, Virat Shejwalkar, Milad Nasr, Amir Houmansadr, and Prateek Mittal. 2022. Mitigating membership inference attacks by Self-Distillation through a novel ensemble architecture. In *31st USENIX Security Symposium (USENIX Security 22)*. 1433–1450.
[39] Texas100 2014. Hospital Discharge Data Public Use Data Fil. https://www.dshs.texas.gov/texas-health-care-information-collection/general-public-information/hospital-discharge-data-public.
[40] Florian Tramer and Dan Boneh. 2020. Differentially private learning needs better features (or much more data). *arXiv preprint arXiv:2011.11660* (2020).
[41] Stacey Truex, Ling Liu, Mehmet Emre Gursoy, Lei Yu, and Wenqi Wei. 2018. Towards demystifying membership inference attacks. *arXiv preprint arXiv:1807.09173* (2018).
[42] Ziqi Yang, Bin Shao, Bohan Xuan, Ee-Chien Chang, and Fan Zhang. 2020. Defending model inversion and membership inference attacks via prediction purification. *arXiv preprint arXiv:2005.03915* (2020).
[43] Jiayuan Ye, Aadyaa Maddi, Sasi Kumar Murakonda, and Reza Shokri. 2021. Enhanced Membership Inference Attacks against Machine Learning Models. *CoRR* abs/2111.09679 (2021). arXiv:2111.09679 https://arxiv.org/abs/2111.09679
[44] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. 2018. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*. IEEE, 268–282.
[45] Samuel Yeom, Irene Giacomelli, Alan Menaged, Matt Fredrikson, and Somesh Jha. 2020. Overfitting, robustness, and malicious algorithms: A study of potential causes of privacy risk in machine learning. *Journal of Computer Security* 28, 1 (2020), 35–70.

A DETAILED RESULTS OF EACH ATTACK

In this appendix, we show detailed results of single-query and label-only attacks from Table 4. Figure 7 shows the detail results of single-query and label-only attacks. We evaluate membership privacy by using the highest attack accuracy in single-query and label-only attacks.

B THE NUMBER OF TP, FN, FP, AND TN OF EACH BEST ATTACK

In this appendix, we show the number of TP, FN, FP, and TN of the best single-query and label-only attack from Table 4. Figure 8 shows the number of TP, FN, FP, and TN in the best single-query attack. Figure 9 shows the number of TP, FN, FP, and TN in the best label-only attack. We compute the attack accuracy, precision, and recall with TP, FN, FP, and TN from Equation (6), (7), and (8).

Table 7: Detail results of single-query and label-only attacks from Table 4. The best attack accuracy on each dataset is bold.

Dataset	Defense	Single-query attacks							Label-only attacks	
		NN-based	Correctness	Confidence	Entropy	Modified-ent.	LiRA	Attack R	Boundary	Data aug.
Purchase100	None	67.21%(0.03)	54.78%(0.09)	65.88%(0.08)	66.32%(0.07)	66.45%(0.09)	61.45%(0.15)	**69.00%(0.45)**	**65.84%(0.23)**	-%(-)
	AdvReg	53.34%(0.01)	55.35%(0.02)	56.62%(0.07)	55.12%(0.11)	56.21%(0.13)	56.35%(0.12)	**62.33%(0.09)**	**58.19%(0.60)**	-%(-)
	MemGuard	59.21%(0.04)	62.44%(0.13)	61.07%(0.09)	58.19%(0.10)	61.43%(0.10)	58.35%(0.11)	**65.26%(0.09)**	**61.98%(0.69)**	-%(-)
	KCD	60.88%(0.05)	52.30%(0.14)	59.74%(0.23)	55.54%(0.08)	55.12%(0.21)	57.86%(0.10)	**64.34%(0.08)**	**60.08%(0.58)**	-%(-)
	SELENA	51.21%(0.10)	51.62%(0.21)	51.18%(0.16)	52.61%(0.09)	52.65%(0.18)	53.91%(0.18)	**59.25%(0.16)**	**55.61%(0.55)**	-%(-)
	SEDMA	52.13%(0.07)	51.34%(0.04)	51.45%(0.11)	51.22%(0.07)	54.81%(0.11)	53.02%(0.23)	**58.40%(0.20)**	**52.25%(0.61)**	-%(-)
Texas100	None	62.91%(0.13)	63.34%(0.21)	67.38%(0.31)	57.75%(0.22)	68.23%(0.32)	**72.64%(0.23)**	71.97%(0.03)	**62.29%(0.22)**	-%(-)
	AdvReg	51.45%(0.07)	54.98%(0.45)	58.01%(0.21)	55.34%(0.21)	58.34%(0.12)	58.63%(0.10)	**58.97%(0.01)**	**58.24%(0.45)**	-%(-)
	MemGuard	63.05%(0.05)	63.12%(0.13)	**63.43%(0.33)**	57.15%(0.19)	63.39%(0.11)	59.93%(0.11)	59.96%(0.01)	**65.68%(0.68)**	-%(-)
	KCD	59.10%(0.11)	54.23%(0.22)	52.89%(0.12)	55.17%(0.17)	56.65%(0.21)	59.36%(0.11)	**59.48%(0.01)**	**53.61%(0.52)**	-%(-)
	SELENA	52.71%(0.04)	52.31%(0.31)	53.23%0.35)	52.61%(0.20)	53.81%(0.09	55.96%(0.08)	**56.89%(0.01)**	**53.90%(0.44)**	-%(-)
	SEDMA	51.10%(0.05)	53.20%(0.14)	51.80%(0.23)	53.41%(0.15)	50.99%(0.11)	55.39%(0.08)	**56.45%(0.01)**	**52.17%(0.34)**	-%(-)
CIFAR100	None	74.20%(0.07)	60.98%(0.11)	74.46%(0.39)	74.22%(0.15)	75.76%(0.41)	84.95%(3.32)	**84.96%(3.01)**	70.86%(0.42)	**77.77%(0.13)**
	AdvReg	57.23%(0.05)	59.11%(0.21)	58.33%(0.32)	56.87%(0.17)	57.98%(0.31)	62.98%(1.50)	**63.02%(1.21)**	58.43%(0.21)	**59.24%(0.12)**
	MemGuard	52.00%(0.06)	62.12%(0.12)	64.78%(0.29)	62.34%(0.13)	63.32%(0.28)	65.03%(1.60)	**65.06%(1.35)**	69.86%(0.22)	**68.34%(0.14)**
	KCD	56.85%(0.13)	52.34%(0.13)	57.24%(0.21)	59.02%(0.15)	56.93%(0.22)	61.30%(1.30)	**61.33%(1.05)**	52.96%(0.33)	**55.28%(0.13)**
	SELENA	54.62%(0.11)	55.21%(0.12)	54.32%(0.21)	53.89%(0.15)	56.77%(0.27)	59.30%(1.13)	**59.34%(0.88)**	53.01%(0.12)	**53.90%(0.14)**
	SEDMA	52.01%(0.12)	53.38%(0.18)	52.32%(0.27)	51.03%(0.15)	51.23%(0.33)	57.85%(1.74)	**57.91%(1.39)**	50.43%(0.34)	**53.03%(0.15)**

Table 8: In the best single-query attack from Table 4, TP as the number of true positives that are correct with members, FN as the number of false negatives that are incorrect with non-members, TN as the number of true negatives that are correct with non-members, and FP as the number of false positives that are incorrect with members.

Dataset	Defense	TP	FN	FP	TN	Total	Accuracy	Precision	Recall
Purchase100	None	6234(136.38)	3564(136.38)	2553(122.51)	7381(122.51)	19732	69.00%(0.45)	70.95%(0.92)	63.63%(1.46)
	AdvReg	4663(119.18)	5126(119.18)	2248(129.27)	7541(129.27)	19732	62.33%(0.09)	67.47%(0.64)	47.64%(1.22)
	MemGuard	5428(138.75)	4361(138.75)	2440(140.27)	7349(140.27)	19732	65.26%(0.09)	68.99%(0.62)	55.45%(1.42)
	KCD	5094(130.20)	4695(130.20)	2286(131.44)	7503(131.44)	19732	64.34%(0.09)	69.02%(0.62)	52.04%(1.33)
	SELENA	4098(104.73)	5691(104.73)	2287(131.46)	7502(131.46)	19732	59.25%(0.16)	**64.18%(0.66)**	41.86%(1.07)
	SEDMA	4096(104.70)	5693(104.70)	2451(140.93)	7338(140.93)	19732	**58.40%(0.20)**	62.56%(0.68)	**41.85%(1.07)**
Texas100	None	3279(24.97)	1721(24.97)	1015(1.73)	3986(1.73)	10000	72.64%(0.23)	76.37%(0.11)	65.57%(0.50)
	AdvReg	2328(3.98)	7461(3.98)	572(1.69)	9217(1.69)	10000	58.97%(0.01)	80.28%(0.02)	23.78%(0.04)
	MemGuard	2492(4.26)	7297(4.26)	543(1.60)	9246(1.60)	10000	59.96%(0.01)	82.12%(0.02)	25.46%(0.04)
	KCD	2396(4.10)	7393(4.10)	541(1.60)	9248(1.60)	10000	59.43%(0.01)	81.59%(0.02)	24.48%(0.04)
	SELENA	1961(3.35)	7828(3.35)	612(1.81)	9177(1.81)	10000	56.89%(0.01)	76.22%(0.02)	20.04%(0.03)
	SEDMA	1882(3.22)	7902(3.22)	620(1.83)	9169(1.83)	10000	**56.45%(0.01)**	**75.22%(0.02)**	**19.23%(0.03)**
CIFAR100	None	4074(313.44)	926(313.44)	578(24.42)	4422(24.42)	10000	84.96%(3.01)	87.58%(0.86)	81.48%(6.27)
	AdvReg	3279(252.27)	6510(252.27)	731(30.87)	9058(30.87)	10000	63.02%(1.21)	81.78%(1.17)	33.49%(2.58)
	MemGuard	3637(279.86)	6152(279.86)	690(29.13)	9099(29.13)	10000	65.06%(1.35)	84.06%(1.05)	37.16%(2.86)
	KCD	2844(218.83)	6945(219.83)	626(26.44)	9163(26.44)	10000	61.33%(1.05)	81.96%(1.16)	29.05%(2.24)
	SELENA	2412(185.58)	7377(185.58)	584(24.66)	9205(24.66)	10000	59.34%(0.88)	80.51%(1.22)	24.64%(1.90)
	SEDMA	2255(286.26)	7534(286.26)	707(29.89)	9082(29.89)	10000	**57.91%(1.39)**	**76.12%(2.84)**	**23.04%(2.92)**

Table 9: In the best label-only attack from Table 4, TP as the number of true positives that are correct with members, FN as the number of false negatives that are incorrect with non-members, TN as the number of true negatives that are correct with non-members, and FP as the number of false positives that are incorrect with members.

Dataset	Defense	TP	FN	FP	TN	Total	Accuracy	Precision	Recall
Purchase100	None	9081(87.89)	785(87.89)	5956(83.78)	3910(83.78)	19732	65.84%(0.23)	60.39%(0.17)	92.04%(0.89)
	AdvReg	7782(129.56)	2084(129.56)	6166(71.55)	3700(71.55)	19732	58.19%(0.60)	55.79%(0.39)	78.88%(1.31)
	MemGuard	9263(119.43)	603(119.43)	6900(44.47)	2966(44.47)	19732	61.98%(0.69)	61.98%(0.69)	93.89%(1.21)
	KCD	8121(106.16)	1745(106.16)	6132(75.46)	3734(75.46)	19732	60.08%(0.58)	56.98%(0.41)	82.31%(1.08)
	SELENA	5361(68.81)	4505(68.81)	4254(88.70)	5612(88.70)	19732	55.61%(0.55)	55.76%(0.59)	54.34%(0.70)
	SEDMA	5014(98.45)	4852(98.45)	4571(61.41)	5295(61.41)	19732	**52.25%(0.61)**	**52.31%(0.64)**	**50.82%(1.00)**
Texas100	None	3575(27.54)	1425(27.54)	2346(45.13)	2654(45.13)	10000	62.29%(0.22)	60.38%(0.30)	71.50%(0.55)
	AdvReg	3007(32.67)	1993(32.67)	2183(40.63)	2817(40.63)	10000	58.24%(0.45)	57.94%(0.46)	60.14%(0.65)
	MemGuard	3831(25.19)	1169(25.19)	2263(63.80)	2737(63.80)	10000	65.68%(0.68)	62.87%(0.68)	76.62%(0.50)
	KCD	2062(33.03)	2938(33.03)	1701(37.24)	3299(37.24)	10000	53.61%(0.52)	54.80%(0.74)	**41.24%(0.66)**
	SELENA	2367(32.64)	2633(32.64)	1977(37.74)	3023(37.74)	10000	53.90%(0.44)	54.49%(0.51)	47.34%(0.65)
	SEDMA	2301(31.57)	2699(31.57)	2084(33.77)	2916(33.77)	10000	**52.17%(0.34)**	**52.47%(0.39)**	46.02%(0.63)
CIFAR100	None	4995(3.47)	5(3.47)	2218(13.55)	2782(13.55)	10000	77.77%(0.13)	69.25%(0.13)	99.90%(0.07)
	AdvReg	4089(4.64)	911(4.64)	3165(14.23)	1835(14.23)	10000	59.24%(0.12)	56.37%(0.10)	81.78%(0.09)
	MemGuard	4973(4.73)	27(4.73)	3139(14.14)	1861(14.14)	10000	68.34%(0.14)	61.30%(0.10)	99.46%(0.09)
	KCD	2959(5.54)	2041(5.54)	2431(13.51)	2569(13.51)	10000	55.28%(0.13)	54.90%(0.13)	**59.18%(0.11)**
	SELENA	3475(5.35)	1525(5.35)	3085(11.17)	1915(11.17)	10000	53.90%(0.14)	52.97%(0.11)	69.50%(0.11)
	SEDMA	3180(7.36)	1820(7.36)	2877(11.72)	2123(11.72)	10000	**53.03%(0.15)**	**52.50%(0.12)**	63.60%(0.15)