EFFECTIVE UNBIASED MEMBERSHIP INFERENCE IN IMAGE AND LANGUAGE MODELS

Author
Eliott Zemour
Mathematics Section
EPFL

Supervisor
Bogdan Kulynych
SPRING Laboratory
EPFL

March 17, 2022

ABSTRACT

Membership inference attacks (MIAs) against machine learning aims to infer whether a data record was used to train a target model or not. When trained on sensitive and private data, vulnerability to MIA can be seen as a serious privacy threat. In this report, we present an effective and unbiased way to evaluate MIA vulnerability and compare it to the shadow model approach considered until then. Membership inference is performed on various classification and language models under a black-box setting. We investigate MIA performance as a function of the attack and the adversarial knowledge. We observe that the combination of adversary features is an efficient way of conducting membership inference, and threshold-based attacks often underperform compared to more complex methods. The attacks proposed are evaluated against models trained with differential privacy, and results show that this defense mechanism indeed mitigates the adversary success. Finally, we study membership inference on large language models (LMs) in the special case of fine-tuning. We consider a set of adversary features based on the perplexity measure, and demonstrate that fine-tuned LMs can be highly vulnerable to MIAs.

1 Introduction

In the recent past, Machine Learning (ML) models have demonstrated excellent ability for prediction and classification tasks in various fields such as image recognition, natural language processing (NLP), and even healthcare analysis. Such breakthroughs are often associated with large amounts of data available for training [12], which can consist of sensitive and private data such as user's private discussions or medical records. When trained with such data, assessing potential information leakage of (distributed) ML models is therefore essential, and constitutes a heated topic in technology law and policy. This concern is compounded by the tendency of predictive models to reproduce societal discrimination, bias, and to transform personal data into invasive insights [24].

One of the main privacy attacks against ML models is Membership Inference Attacks (MIAs), recently introduced by Shokri et al. [30], in which the adversary aims to infer whether a given data record was part of the model's training dataset or not. Although MIA does not recover the training data in itself, it can lead to severe privacy risks to individuals, and Veale et al. [32] argues that such vulnerability can be seen as a violation of EU's General Data Protection Regulation (GDPR). For example, Shokri et al. [30] demonstrated the feasibility of MIA against a hospital discharge model under a black-box setting, which could leak information about an individual illness given its presence in the training set. More generally, such vulnerability is concerning when it becomes possible to deduce sensitive information from inferred membership.

Since its introduction, there have been numerous studies that investigated MIA feasibility on ML models for regression [11], classification [30] and generation [13]. Therefore, the extension of MIA to Language Models (LMs) – fundamental to many NLP tasks – seems straightforward. Carlini et al. [3] demonstrated that generative sequence models are likely to unintentionally memorize rare or unique training-data sequences, which is once again problematic in the case of sensitive/private data records (e.g. users' private messages text: Google's Smart Compose [5]). The risks of LM's unintended memorization are mainly studied through training data extraction attacks [3, 4], but membership inference can be seen as the building block for such attacks. There exist examples of fine-tuned BERT models for Alzheimer's or dementia disease detection [1], in which membership insights already constitute a severe privacy violation. Indeed, a recent trend in NLP is the fine-tuning of large language models such as BERT and GPT-2 for multi-purpose tasks through APIs/pipelines made available to companies and individuals. For instance, Hugging Face library provides a very simple way of deploying LMs trained on personal and potentially sensitive data[1].

In this report, we propose an unbiased approach for assessing MIA feasibility on various language and classification models. Under a black-box setting, several attacks are introduced and their performance is investigated on different adversary features. The contributions are the following:

- The Holdout mode is an unbiased approach for evaluating models' vulnerability to MIAs, thus providing an alternative to shadow model (SMs) attacks proposed by Shokri et al. [30].
- Combining knowledge from multiple extracted features can lead to better performance in Membership Inference.
- Fine-tuned LMs such as GPT-2 are highly vulnerable to MIAs based on perplexity, especially when the base model (before fine-tuning) is available to the attacker.

2 Statement of the problem and theoretical background

2.1 MIAs against ML models: related work

First introduced by Shokri et al. [30], Membership Inference Attacks have been studied under the scope of various ML models, attack methods, adversarial knowledge and training algorithms [17]. The MIA problem can be formulated as follows:

Membership Inference Attack: *Let Ω be a population of examples (or data records), and consider a data generating distribution $\mathcal{D}$ over Ω. We indicate with $A(\cdot)$ the training algorithm that produces a model A_S given training data $S \in \Omega$. Once the training process is finished, the model A_S is able to make predictions $y(\mathbf{x})$ on (unseen) data $\mathbf{x} \in \Omega$. The goal of MIAs is therefore to infer whether a given example $\mathbf{x} \in \Omega$ is a member of the training set S.*

Prior work on membership inference exhibit two main frameworks for conducting such attacks, differing on the adversarial knowledge about the target model [28]: the *white-box* setting refers to an attacker having information about the training data distribution, the training procedure, the architecture and the learned parameters of the target model. In

[1] huggingface.co/docs/transformers/training

the case of *black-box* attacks, the attacker has access to the target model through black-box queries, and can obtain the model's decision $y(\mathbf{x})$, confidence scores $\hat{p}(y \mid \mathbf{x})$, or loss $\ell(\mathbf{x}, y(\mathbf{x}))$. The following will focus on the most common framework for attacking classification models: black-box MIA with knowledge of the confidence scores output [17]. Based on this, an attacker will further be able to compute the cross-entropy loss and other adversarial features (see 2.3).

Membership Inference Attacks are often associated with *overfitting* of ML models on training data, as many papers pointed out that overfitting is a key factor contributing to MIA vulnerability [30, 33]. An ML model is said to overfit when it performs much better on its training data than test data, the performance gap being usually measured through average loss on train and test sets. Indeed, ML models are often overparameterized, and thus have sufficient capacity to memorize information about the data they have been trained on [17]. This results in models exhibiting different behavior between member and non-member data records, even when the prediction is accurate on both sets: the confidence score is likely to be higher for an example from the training set, as the model has already been exposed to the very same data. However, Yeom et al. [33] have shown that overfitting is not a necessary condition for vulnerability to MIA, in the sense that the average-based definition may not capture a more complex discrepancy in the model's response on train and test data records (that an attacker could exploit). Kulynych et al. [22] introduces the notion of *distributional overfitting*, considering distributional information instead of difference in the average losses. This is supported by theoretical results linking worst-case vulnerability to MIAs with distributional overfitting under total variation distance. The attacks studied in this report intend to exploit such distribution discrepancies in adversarial extracted features and are introduced in 2.4.

As most of the previous works, this report will focus on the membership inference problem with balanced prior, i.e. the number of train data records to classify is the same as the number of test examples. Jayaraman et al. [20] mention that this assumption might be unrealistic as models are likely to face adversaries with imbalanced priors, and consider the case of skewed priors.

Finally, Differential Privacy (DP) – introduced by Dwork [8] – is a probabilistic privacy mechanism that provides an information-theoretical privacy guarantee, especially in the scope of MIAs. It can be shown that if the training algorithm $A(\cdot)$ satisfies ε-DP, the worst case vulnerability with any adversary features is upper bounded [33, 22]. In practice, DP-trained models are indeed less vulnerable to membership inference [17] and the attacks presented in this work shall be assessed on such models.

2.2 Metrics for membership inference

In this section, we will introduce the metrics that will be used for evaluating attack performance. As attacker's precision and accuracy will be expressed using terms from the classification framework (TP, FP, TN, FN), Table 1 introduces a taxonomy with the aim to avoid any misunderstanding: in the context of MIAs, a *positive* will denote an example classified as *member* of the training dataset.

		Predicted Membership	
		Train	Test
True Membership	Train	True Positive (TP)	False Negative (FN)
	Test	False Positive (FP)	True Negative (TN)

Table 1: Taxonomy for membership inference attack metrics

Accuracy
Accuracy is one of the most commonly used metrics to measure the performance of a given attack. It gives information about the success probability of an adversary $\mathcal{A}$.

$$\text{Accuracy} = \frac{\text{\# Correctly classified examples}}{\text{\# Examples}} = \frac{\text{TP + TN}}{\text{TP + FP + TN + FN}} \tag{1}$$

Vulnerability
Mathematically, vulnerability of a model to an adversary $\mathcal{A}$ is expressed through its accuracy:

$$\mathcal{V}(\mathcal{A}) = 2\left(\text{Accuracy}_{\mathcal{A}} - \text{Accuracy}_{RG}\right) \tag{2}$$

where Accuracy_{RG} is the accuracy of the random guessing baseline. Vulnerability is also referred as 'Advantage' by Yeom et al. [33]. In the balanced setup we are considering, this quantity reduces to 0.5, and the vulnerability becomes $\mathcal{V}(\mathcal{A}) = 2\,\text{Accuracy}_{\mathcal{A}} - 1$, which is equivalent to the accuracy metric.

Precision
Precision gives the ratio of true members among all the positive membership predictions made by the adversary.

$$\text{Precision} = \frac{\text{\# Members correctly classified as members}}{\text{\# Membership predictions}} = \frac{\text{TP}}{\text{TP + FP}} \tag{3}$$

2.3 Adversarial knowledge: Adversary extracted features

In order to guess the membership of a data record $\mathbf{x}$, the attacker extract features w from the target model A_S:

$$w \leftarrow \phi(A_S, \mathbf{x}) \tag{4}$$

Among the potential features that can be extracted (depending on white/black box attacks), we propose the following set of features considered part of adversary knowledge on classification models:

- The model's *loss* $\ell(\mathbf{x}, y(\mathbf{x}))$, which will mostly denote the cross-entropy loss computed from the predictions $\hat{p}(y \mid \mathbf{x})$ and the true (one-hot) vector y over M classes:

$$\ell(\mathbf{x}, y(\mathbf{x})) = -\sum_{i=1}^{M} y_i \log(\hat{p}_i) \tag{5}$$

- The confidence outputs, that is the output layer of the classification model: $\hat{p}(y \mid \mathbf{x})$, and denoted '*confidence*'.
- The *entropy* of the prediction output $H(\hat{p}(y \mid \mathbf{x})) = -\sum_i \hat{p}_i \log(\hat{p}_i)$.

Note that extracting such features only requires access to the prediction output $\hat{p}(y \mid \mathbf{x})$ and can therefore be applied in the black-box framework.
The idea behind the attacker features is that their distribution should differ between train and test data. For example, the target model A_S has usually a larger loss and prediction entropy on its test data than its training data. In the same way, we expect the confidence output vector to have a larger maximum on the train set.

Finally, we will extend the set of features by considering a so-called *combined* feature, made of the concatenation (*loss, entropy, confidence*). For a given example with true label k ($y_k = 1$) and associated prediction vector $\hat{p}(y \mid \mathbf{x})$, the combined feature is given by the array $[-\log \hat{p}_k,\ H(\hat{p}(y \mid \mathbf{x})),\ \hat{p}_1, \ldots, \hat{p}_M]$.

2.4 The attacks

2.4.1 Threshold-based inference attacks

Most of inference attacks found in the literature are threshold-based [17]. Indeed, a common reason for the different behavior of ML model on train and test sets is (distributional) *overfitting*, and train data records tend to have a lower loss than others. When attacking, the adversary will make the following decision based on the extracted feature $w(\mathbf{x})$ (can be loss, entropy) and the threshold τ:

$$\text{Membership } \mathcal{M}(\mathbf{x}) = \begin{cases} \textit{Train} & \text{if } \ w(\mathbf{x}) \leq \tau, \\ \textit{Test} & \text{otherwise.} \end{cases} \tag{6}$$

Average Threshold
In average loss threshold attack, introduced by Yeom et al. [33], the adversary computes the expected value of the training loss and sets it as a threshold τ:

$$\tau = \mathop{\mathbb{E}}_{\mathbf{x} \in S_{\text{train}}} [\ell(\mathbf{x}, y(\mathbf{x}))]$$

The attacker therefore classifies a record as a member if its per-instance-loss is less than the expected training loss. We refer to this membership inference as 'av thr'. Note that this attack will also be applied on *entropy* features.

Optimal Threshold
In optimal threshold attacks, the adversary aims to find a threshold τ_{opt} that separates the losses from train and test sets in a way that maximizes accuracy (i.e maximizing TP and TN) [31]:

$$\tau_{\text{opt}} = \arg\max_{\tau} \mathop{\mathbb{E}}_{\mathbf{x} \in S_{\text{train}}} [\ell(\mathbf{x}, y(\mathbf{x})) \leq \tau] + \mathop{\mathbb{E}}_{\mathbf{x} \in S_{\text{test}}} [\ell(\mathbf{x}, y(\mathbf{x})) > \tau]$$

We refer to this membership inference as 'opt thr'. Note that unlike the average approach, this attack requires test data records and their features. Jayaraman et al. [20] propose alternatives for setting the decision threshold τ, by optimizing leakage constrained to a given expected maximum false positive rate. This attack will also be applied on *entropy* features.

2.4.2 ML-empowered attacks

Kernel Density Estimation
Kulynych et al. [22] showed that worst-case vulnerability to MIAs with adversary's features $w \sim W_{\text{train/test}}$ is equal to the distributional generalization gap under total-variation distance.

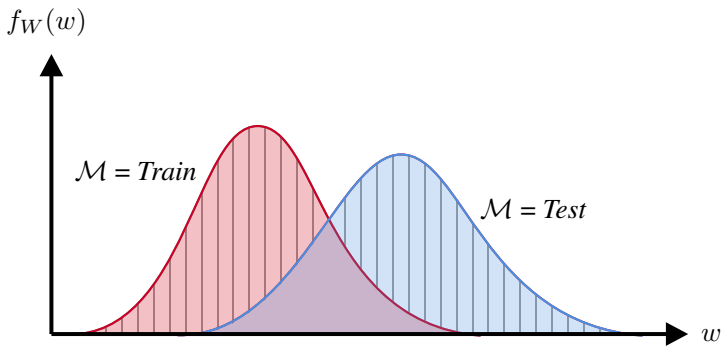


Figure 1: Taken from Kulynych et al. [22]. Example of train and test distribution for an adversary feature w. The striped area shows the distributional generalization gap: total variation between distributions of adversary's feature on training and outside. The size of the striped area exactly equals to the worst-case vulnerability to any adversary that uses feature w for distinguishing members from non-members.

The optimal adversary therefore follows this decision:

$$\text{Membership } \mathcal{M}(\mathbf{x}) = \begin{cases} \textit{Train} & \text{if } f_{W_{\text{train}}}(w(\mathbf{x})) \geq f_{W_{\text{test}}}(w(\mathbf{x})), \\ \textit{Test} & \text{otherwise.} \end{cases} \tag{7}$$

The attack presented in this section is motivated by this observation. Kernel density estimation (kde) is a non-parametric method that allows to estimate the probability density function of a random variable with a neighbor-based approach. After having computed the estimates $\tilde{f}_{W_{\text{train}}}$ and $\tilde{f}_{W_{\text{test}}}$, the design of an adversary following Eq.7 is possible. We refer to this membership inference as 'kde'.

F-BLEAU – K Nearest Neighbours
This attack directly uses F-BLEAU, a tool for estimating the information leakage of a (black-box) system[2]. Cherubin et al. [6] demonstrates with *universal consistency* that the Nearest Neighbours (NN) approach has convergence guarantees in a way that the vulnerability will approach the total-variation distance as $n \to \infty$. For the following, we will refer to this membership inference as 'fbleau'.

Random Forest Classifier
Finally, a decision-tree approach for Membership Inference is introduced with Random Forests (RF). For 1D features such as *loss* and *entropy*, the RF attack can implement multiple thresholds and better capture the distributional generalization gap than single-threshold attacks (average, optimal).In multiple dimensions (*confidence*, *combined* features), an RF classifier will be able to extract leakage from a combined set of features, or simply by identifying thresholds for each of the single features. We refer to this membership inference as 'forest'.

[2]See F-BLEAU's GitHub repository.

2.5 Strategies for evaluating membership inference vulnerability

Given an adversary feature extracted from a trained model A_S, MIAs aim to infer whether this example was part of the training set S. Therefore, we will consider both *train* ($\mathbf{x} \in S$) and *test* ($\mathbf{x} \in \Omega \setminus S$) data, and we will separate these between *learn* and *eval* sets. For the purpose of evaluating ML models' vulnerability to various MIAs, we will train these attacks on a fixed *learn* set, and assess performance on the *eval* set. As the naming system can be confusing, Table 2 aims to properly define what these terms refer to. Moreover, we will consider a the following balance setup: the learn and eval sets will both contain the same number of train and test features. The accuracy and vulnerability metrics are therefore equivalent and only the former will be reported.

Name	Description
Train features	features extracted from the training set of the model
Test features	features extracted that are not from the training set
Learn set	set of features over which the attacks will be trained
Eval set	set of features to assess attacks' performance

Table 2: Naming system

Full dataset evaluation mode
After having extracted adversary features from the target model, the attacks are trained and evaluated on the same data: the *eval* set is the **same** as the *learn* set. We refer to this setting as 'Full mode'.

Holdout dataset evaluation mode
Holdout dataset evaluation mode differs from the Full one in the sense that the *learn* set and *eval* sets are now **disjoint**. We refer to this setting as 'Holdout mode' mode as it takes inspiration from Machine Learning standard practices: a model is evaluated on unseen data, i.e. examples it has not been trained on. Figure 2 aims to illustrate the procedure for assessing MIA performance in Holdout mode. Note that for a finite amount of extracted features from the target model, attacks in Holdout mode will be trained on a smaller set, usually 50% of the available data. Meanwhile, attacks in Full mode can be trained on a larger amount of records (since no learn-eval split is needed). Section 3 aim to compare the former two modes, under the scope of bias against control models.

Full and Holdout settings define a framework for the model's owner point of view. Indeed, the labeled data records are directly extracted from the target model, and are used to train models in a supervised way. MIA vulnerability in the Holdout setting suggests that an adversary having access to a fraction of the actual train and test sets would be able to infer membership on new, unseen examples. This hypothesis represents a realistic risk, as a common trend in ML models deployment is to release a small part of the training dataset. On the other side, the last approach presents a practical way to conduct MIAs without having access to any labeled data record from the target model.

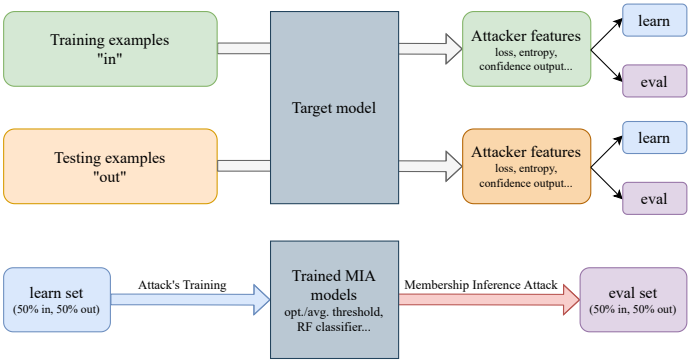


Figure 2: Holdout dataset evaluation mode with learn-eval split for membership inference attacks.

Shadow Model Attacks
Shadow Models (SMs) were proposed along with the introduction of MIAs by Shokri et al. [30]. In order to construct an attack training set (*learn* set), the adversary creates k shadow models using the same training algorithm $A(\cdot)$. The shadow training sets may overlap but we assume that these are disjoint from the private dataset used to train the target model. After having extracted a *learn* set of features from shadow models, the attacks are evaluated on an *eval* set directly obtained from the target model. Therefore, this approach differs with Holdout mode in the sense that the learn set is distributed similarly to the target model's eval dataset, meanwhile the Holdout mode implies that both sets are from the same underlying distribution.

3 Investigating MIA bias: Holdout vs Full-dataset mode

In this part, we will argue the following: *Holdout mode gives better vulnerability estimates that Full mode in the framework of membership inference attacks*. Indeed, although Full mode demonstrates better performance on *eval* set, the attacks are biased against control models and therefore cannot be considered as a fair measure of other models' vulnerability.

3.1 Experiment

The term *control model* refers to ML models whose output is completely independent on the input data. Typically, control models for classification tasks output constant or random prediction vectors $\hat{p}(y|\mathbf{x})$. For this experiment, the models considered are based on the German-Credit dataset from UCI Repository of Machine Learning Databases [7], which goal is to classify people described by a set of attributes as good (=1) or bad (=2) credit risks. The predict function of the control models considered is as follows:

1. For each $\mathbf{x}$, sample p uniformly at random in the range $[0, 1]$,
2. Set $\hat{p}(y = 1|\mathbf{x}) := p$, and $\hat{p}(y = 2|\mathbf{x}) := 1 - p$
3. Predicted class: $\hat{y} = \arg\max_{y\in\{1,2\}} \hat{p}(y|\mathbf{x})$

The bias experiment will be based N=100 random control models. The sizes of models' train and test sets are 800 and 200, respectively, and a random slicing will generate *learn* and *eval* sets. As this report focuses on MIAs with balanced priors, the learn and eval sets will both be of size 200 (each composed of 50% train and 50% test data). The adversary feature considered is the cross-entropy loss.
In this experiment, the expected vulnerability to any MIA is zero, that is the attack accuracy should not differ from random guessing baseline (50%). Indeed, the adversary's extracted features are sampled from the same uniform distribution, regardless of the true membership of the data record. The distributional generalization gap is therefore zero in expectation.

3.2 Results

Table 3 summarizes the averaged adversary accuracies obtained in both Holdout and Full modes when attacking the control models. As a result, one may observe the unbiasedness of all attacks presented in 2.4 under Holdout setting. Indeed, the reported accuracies are not statistically different from the random guessing baseline. Therefore, one might reasonably consider that any non-zero vulnerability to MIAs in Holdout mode is indeed due to distributional overfitting, as this quantity is zero when train and test features are sampled from the same underlying distribution. On the other side, Table 3 demonstrates that some attacks are biased against control models in Full mode: Optimal Threshold, Kernel Density Estimation, F-BLEAU and Random Forest. This is expected: as such attacks are evaluated on their training set, they manage to beat the random guessing baseline even when the distributional generalization gap is zero. One may conclude that the Full setting is not appropriate for evaluating MIA performance, as it results in high attack accuracy even when the actual vulnerability is none.

Attack	Holdout	Full
av thr	50.13 ± 0.61	49.62 ± 0.45
opt thr	50.14 ± 0.53	52.84 ± 0.33
kde	49.55 ± 0.59	55.51 ± 0.39
fbleau	50.11 ± 0.68	65.55 ± 0.39
forest	50.10 ± 0.65	73.98 ± 0.45

Table 3: Accuracies for bias experiment on German-Credit dataset. Grey filled cells represent (attack, mode) pairs that are unbiased, i.e. the accuracy is not statistically different from the random guessing baseline (50%). Results are averaged over N=100 random seeds and 95% CIs are provided: $\bar{x} \pm 1.96\,\sigma_x/\sqrt{N}$.

4 Holdout-mode attacks on various classification models

As motivated in the former section, the Holdout setting is an unbiased framework for evaluating MIAs. In this part, we will tackle the problem of membership inference on various classification models trained with the following datasets:

- **German-Credit** dataset [7] which we already described in Section 3, containing 1000 entries with 20 categorial attributes for predicting credit risks. The models considered will have 800 training entries (person) and 200 test ones.

- **CIFAR-10 and CIFAR-100** datasets, introduced by Krizhevsky et al. [21] and used in multiple benchmarks for privacy works. The CIFAR-10 dataset consists of 60000 32×32 colored images in 10 classes, with 6000 images per class. Similarly, CIFAR-100 has 100 classes containing 600 images each. In both cases, there are 50000 training images and 10000 test ones.
  Barz and Denzler [2] demonstrated that the test sets of both CIFAR datasets contain duplicate images, i.e. images that can also be found in very similar form in the training set or the test set itself. They propose variants so-called **ciFAIR-10 and ciFAIR-100** with modified test sets (and keeping the training sets unchanged). For the purpose of MIA experiments with SMs, these datasets will be used as one wants to ensure disjointedness between target and shadow sets.

4.1 Experiments

The models considered for these experiments are adapted to the classification problem. For CIFAR datasets, we will use a Pytorch implementation [19] of the ResNet20 architecture from He et al. [14]. Meanwhile, a simple random forest classifier for the German-Credit dataset is used, where the following hyperparameters are fine tuned by grid search and cross validation: number of trees, maximum depth, criterion for quality of split. Table 4 summarizes the models considered in this part, and defines a naming system for conciseness.

For conducting these experiments, we will consider all adversary features defined in 2.3: *loss*, *entropy*, *combined*, *confidence*. All attacks from 2.4 will be evaluated on their set of supported features: 1D features for threshold and kde attacks, 1 and 2D for fbleau and forest. The size of the *learn* and *eval* sets are 10'000 for CIFAR models, and 200 for German-96/76. Results are averaged over N=10 seeds, using randomness in the learn-eval splits allowed by the Holdout setting.

Name	Dataset	Model	Train / Test Acc (%)	Generalization gap (%)
CIFAR10-96/91	CIFAR-10	ResNet20	96 / 91	5
CIFAR10-88/83	CIFAR-10	ResNet20	88 / 83	5
CIFAR100-81/65	CIFAR-100	ResNet20	81 / 65	16
German-96/76	German-Credit	Random Forest	96 / 76	20

Table 4: Description of classification models considered for evaluating MIAs.

4.2 Results

The MIA results are summarized in Tables 5,6,7,8. Green filled cells represent attacks with highest average accuracy/precision for a given adversary feature. From these results, we observe that the RF method (forest) is a strong attack compared to the others, both in single and multiple dimensions. We also notice that in models with large amount of data (M1, M2, M3), combining multiple features is a practical and efficient way to perform membership inference, as it consistently is among the highest precision and accuracy. Interestingly, the best MIA performances on model German-96/76 are reached on the *loss* feature, where the kde attack dominates the accuracy column and the average threshold obtains 65.24% precision. Similarly, on CIFAR models, the top performances with *loss* features are observed for CIFAR100-81/65, which has the highest overfitting gap (18%). One may conclude that membership inference based on *loss* is an appropriate approach for models demonstrating high overfitting (in the averaged sense).

Feature	Attack	Accuracy (%)	Precision (%)
loss	av thr	52.47 ± 0.15	51.49 ± 0.09
	opt thr	53.07 ± 0.07	51.71 ± 0.04
	kde	52.17 ± 0.59	51.58 ± 0.82
	fbleau	53.95 ± 0.34	53.59 ± 0.42
	forest	56.99 ± 0.25	55.15 ± 0.22
entropy	av thr	50.67 ± 0.18	50.44 ± 0.12
	opt thr	51.33 ± 0.09	50.76 ± 0.06
	kde	53.08 ± 0.37	54.36 ± 0.47
	fbleau	52.11 ± 0.29	51.99 ± 0.30
	forest	55.19 ± 0.24	53.97 ± 0.20
confidence	fbleau	52.79 ± 0.27	52.66 ± 0.25
	forest	58.64 ± 0.30	58.95 ± 0.30
combined	fbleau	54.14 ± 0.39	53.49 ± 0.37
	forest	59.39 ± 0.33	58.94 ± 0.39

Table 5: Holdout mode, CIFAR10-96/91 (ResNet20) Results are averaged over 10 random seeds and 95% CIs are provided.

Feature	Attack	Accuracy (%)	Precision (%)
loss	av thr	52.15 ± 0.13	51.36 ± 0.08
	opt thr	52.45 ± 0.16	51.42 ± 0.11
	kde	51.72 ± 0.63	51.53 ± 0.62
	fbleau	52.46 ± 0.26	52.43 ± 0.21
	forest	54.95 ± 0.30	53.34 ± 0.27
entropy	av thr	49.84 ± 0.14	49.88 ± 0.10
	opt thr	49.87 ± 0.17	44.35 ± 10.9
	kde	51.50 ± 0.41	51.82 ± 0.47
	fbleau	50.69 ± 0.33	50.93 ± 0.26
	forest	52.40 ± 0.30	51.70 ± 0.33
confidence	fbleau	52.27 ± 0.35	52.12 ± 0.34
	forest	59.21 ± 0.27	59.23 ± 0.31
combined	fbleau	53.62 ± 0.42	53.37 ± 0.37
	forest	59.32 ± 0.29	59.56 ± 0.35

Table 6: Holdout mode, CIFAR10-88/83 (ResNet20) Results are averaged over 10 random seeds and 95% CIs are provided.

Feature	Attack	Accuracy (%)	Precision (%)
loss	av thr	56.68 ± 0.32	55.20 ± 0.26
	opt thr	57.59 ± 0.27	54.84 ± 0.21
	kde	56.98 ± 0.62	55.15 ± 0.84
	fbleau	55.70 ± 0.32	54.65 ± 0.25
	forest	59.10 ± 0.30	57.28 ± 0.31
entropy	av thr	50.99 ± 0.27	50.87 ± 0.24
	opt thr	50.75 ± 0.28	50.68 ± 0.27
	kde	52.61 ± 0.29	52.66 ± 0.26
	fbleau	51.05 ± 0.30	51.06 ± 0.27
	forest	53.14 ± 0.18	53.90 ± 0.48
confidence	fbleau	51.36 ± 0.32	51.25 ± 0.26
	forest	62.36 ± 0.36	63.31 ± 0.54
combined	fbleau	54.47 ± 0.15	53.77 ± 0.18
	forest	62.30 ± 0.24	64.80 ± 0.31

Table 7: Holdout mode, CIFAR100-81/65 (ResNet20) Results are averaged over 10 random seeds and 95% CIs are provided.

Feature	Attack	Accuracy (%)	Precision (%)
loss	av thr	63.75 ± 1.63	65.24 ± 1.80
	opt thr	64.15 ± 1.92	62.02 ± 1.58
	kde	65.55 ± 1.82	62.28 ± 1.69
	fbleau	61.90 ± 1.66	61.63 ± 1.48
	forest	58.60 ± 1.71	58.12 ± 1.72
entropy	av thr	60.80 ± 1.59	64.95 ± 2.35
	opt thr	60.95 ± 1.77	62.35 ± 2.65
	kde	54.40 ± 4.06	55.17 ± 4.03
	fbleau	58.55 ± 2.11	59.52 ± 1.62
	forest	56.90 ± 2.05	57.17 ± 2.18
confidence	fbleau	58.55 ± 1.40	60.41 ± 1.95
	forest	56.50 ± 2.28	56.71 ± 2.39
combined	fbleau	61.70 ± 1.61	61.82 ± 1.71
	forest	62.05 ± 1.91	62.32 ± 2.01

Table 8: Holdout mode, German-96/76 (RF) Results are averaged over 10 random seeds and 95% CIs are provided.

4.3 Comparison with Shadow Model attacks

As mentioned in Section 2.5 there exists another unbiased setting for conducting Membership Inference Attacks: **Shadow Models**. The *eval* set still consists of adversary features extracted from the target model, but the *learn* set, that is the training set for the attacks, is now extracted from multiple SMs. We will attack a ResNet20 model on CIFAR-10 using $k = 3$ SMs described in Table 9. These models are trained with the exact same procedure as the target model. We recall that the training sets of SMs may overlap, but are strictly disjoint from the target's training set. We will omit the *entropy* feature for the sake of conciseness, but still include it in *combined* features as before.

Model	**Train / Test Acc (%)**	**Train / Test set size**
Target	78 / 70	10K / 10K
Shadow 1	79 / 72	10K / 10K
Shadow 2	70 / 64	10K / 10K
Shadow 3	79 / 72	10K / 10K

Table 9: Description of target and shadow models on CIFAR-10. All ResNet20 models are trained over 10'000 examples for 25 epochs.

The MIA results (Table 10) are computed as following: each SM will singly attack the target model using its set of features as *learn* set. Therefore, we will have $k = 3$ shadow models attacks. Note that Shokri et al. [30] suggests to consider features extracted from all k SMs as a whole, but the approach inspired by Salem et al. [29] allows for averaging. Consequently, the size of *learn/eval* sets for SM attacks is 10'000, while only 5'000 for the Holdout mode attack.

It is apparent from Table 10 that Holdout and Shadow settings yield to similar vulnerability estimates for the CIFAR-10 target model. One may argue that the SM approach is more realistic in the attacker's point of view, and allows for larger *learn* set sizes. However, performing such attack requires access to enough data, information about the model's training procedure, and computational resources to train k models. The similarity of the results suggests that Holdout mode is an effective unbiased method for evaluating vulnerability to membership inference, and can be seen as a practical way to do so in the owner's point of view. Indeed, the Holdout setting only requires black-box queries to the target model, and some data with membership labels.

		Holdout mode		**Shadow Models**	
Feature	Attack	Accuracy (%)	Precision (%)	Accuracy (%)	Precision (%)
loss	av thr	54.16 ± 0.24	52.94 ± 0.17	54.30 ± 0.09	53.01 ± 0.20
	opt thr	54.07 ± 0.24	52.96 ± 0.21	54.23 ± 0.14	52.92 ± 0.39
	kde	54.02 ± 0.19	52.82 ± 0.13	54.14 ± 0.12	52.77 ± 0.26
	fbleau	51.37 ± 0.29	51.46 ± 0.21	51.47 ± 0.92	51.49 ± 0.98
	forest	53.84 ± 0.22	52.79 ± 0.18	54.23 ± 0.15	52.98 ± 0.17
confidence	fbleau	50.89 ± 0.21	50.97 ± 0.21	50.03 ± 0.59	50.38 ± 0.34
	forest	51.23 ± 0.25	51.27 ± 0.26	50.76 ± 0.35	50.73 ± 0.37
combined	fbleau	51.95 ± 0.31	51.79 ± 0.29	51.16 ± 0.62	51.14 ± 0.51
	forest	53.70 ± 0.24	52.97 ± 0.19	53.64 ± 0.39	52.75 ± 0.29

Table 10: Holdout mode results are averaged over 10 random seeds. We present aggregated results for SMs (one at the time)

4.4 Differencial Privacy

After having assessed various classification models' vulnerability to MIAs, this part focuses on a defense approach: **Differential Privacy**. To this end, a model on CIFAR-10 is trained with (ε, δ) differential privacy, following a Python implementation of a ConvNet model[3] provided by Pytorch Opacus [34]. The accuracies of the classifier on the training and test sets are 72% and 69%, respectively. Finally, the privacy parameters obtained are $\varepsilon = 7$ and $\delta = 10^{-5}$, which leads to an upper vulnerability bound of $(\exp(\varepsilon) - 1 + 2\delta)/(\exp(\varepsilon) + 1))$ = 99.82%, and therefore 99.91% on accuracy with balanced prior [18]. The theory provides pessimistic bounds on MIA vulnerability, and we already observe from the distribution of *loss* feature (Fig A.2) that this model should leak much less information about its training set (compared to the previous models trained on CIFAR-10).

Table 11 summarizes MIA performances on the DP-trained model. We found that threshold attacks on *loss* reached the best accuracy and precision among all the (feature, attack) pairs evaluated. Moreover, the metrics are very close to the random guessing baseline (50%) and the *combined* features setup does not seem to improve the adversary's success. Therefore, one might reasonably state that DP training effectively mitigated vulnerability to membership inference in Holdout mode.

[3] See on github: pytorch/opacus/blob/main/examples/cifar10.py

Feature	Attack	Accuracy (%)	Precision (%)
loss	av thr	51.53 ± 0.20	51.04 ± 0.14
	opt thr	51.74 ± 0.28	51.27 ± 0.21
	kde	51.25 ± 0.23	50.82 ± 0.14
	fbleau	50.95 ± 0.23	50.68 ± 0.22
	forest	50.95 ± 0.28	51.10 ± 0.21
entropy	av thr	50.64 ± 0.32	50.54 ± 0.26
	opt thr	50.27 ± 0.28	50.24 ± 0.27
	kde	50.09 ± 0.29	50.10 ± 0.26
	fbleau	50.46 ± 0.31	50.60 ± 0.25
	forest	50.32 ± 0.30	50.33 ± 0.32
confidence	fbleau	49.91 ± 0.18	50.35 ± 0.11
	forest	50.30 ± 0.17	50.30 ± 0.16
combined	fbleau	50.04 ± 0.32	50.38 ± 0.18
	forest	50.82 ± 0.22	50.71 ± 0.18

Table 11: Membership Inference on a DP-trained CIFAR-10 model with $\varepsilon = 7$, $\delta = 10^{-5}$ (3% generalization gap). Holdout mode – Results are averaged over 10 random seeds and 95% CIs are provided.

5 Membership inference attacks on language models

5.1 Unintended memorization in language models

Carlini et al. [3] have shown that neural networks, in particular LMs, have a tendency to memorize rare or unique sequences in the training data. This memorization problem is a particular form of vulnerability to MIAs, and can lead to serious privacy threats when such models are trained on sensitive sequences, for examples users' private messages [5]. Carlini et al. [4] demonstrated that large language models are indeed subject to unintended memorization by performing a training data extraction attack on GPT-2, a Transformer-based family of LMs released by OpenAI [26].

Generative Pretrained Transformer 2 (GPT-2) models are trained on (40GB of) text data scraped from the public internet, following outbound links from Reddit. This dataset is called WebText. Language models from the GPT-2 family differ on their number of parameters, starting from 117M (Small) and up to 1.5B (XL). In the scope of MIAs under Holdout setting, we require having access to both training and held out text data. However, the WebText dataset has not been released by openAI, although there have been attempts to perform the same scraping process in order to open-source it [10]. As a result, performing Holdout-mode attacks directly on GPT-2 is not possible, as one would require train and test data coming from the same underlying distribution (WebText), with associated membership labels.

Therefore, this report proposes to assess LMs vulnerability to MIAs in the special case of fine-tuning. By doing so, we ensure the similarity of the underlying distribution between train and test sequences. Indeed, the process of fine-tuning a pre-trained LM has become the de-facto standard for doing transfer learning in NLP [16]. While pre-training large LMs such as GPT-2 is computationally intensive, fine-tuning is a simple and efficient approach for reaching high level performances on almost any NLP task [25]. For example, the Hugging Face model repository allows users to fine-tune most state-of-the-art LMs with standard and custom datasets, and unaware individuals deploying such models that have been exposed to sensitive data represents a very practical privacy risk.

5.2 Experiment: fine-tuning GPT-2

As motivated in the previous section, this part will assess MIAs performance on GPT-2 fine tuned through the Hugging Face API. Mhapsekar et al. [23] states that *all Wikipedia links were removed from WebText, as it is a common source for other datasets which would have let to complications including overlapping training data.* Therefore, we propose to fine-tune the small version[4] (117M parameters) of GPT-2 on WikiCorpus [27], as it is a commonly used corpus made of large portions of the Wikipedia (based on a 2006 dump). By doing so, we control the conditions of the MIA experiment, which are the following:

- The GPT-2 LM is fine tuned following the sample code provided by Hugging Face[5].
- The number of epochs is varied (5, 10), in order to assess different degrees of memorization/overfitting.

[4] Small version of GPT-2 available here.
[5] See on GitHub.

– The `raw_en` (i.e. raw english) version of WikiCorpus is used, as this dataset contains multiple languages.
– Only a small number of WikiCorpus entries is considered: 10'000 for both train and test sets.

The models' vulnerability will be evaluated in Holdout mode, and a comparison with shadow models will eventually be provided. Concerning the SM attack, GPT-2 is fine-tuned on another subpart (10'000 entries) of WikiCorpus for 10 epochs and will constitute the only shadow model. The target model is the same as the model investigated in Holdout mode (10 epochs). For the purpose of obtaining robust vulnerability estimates, we will perform N=4 different SM attacks following the approach described in Fig. 3.

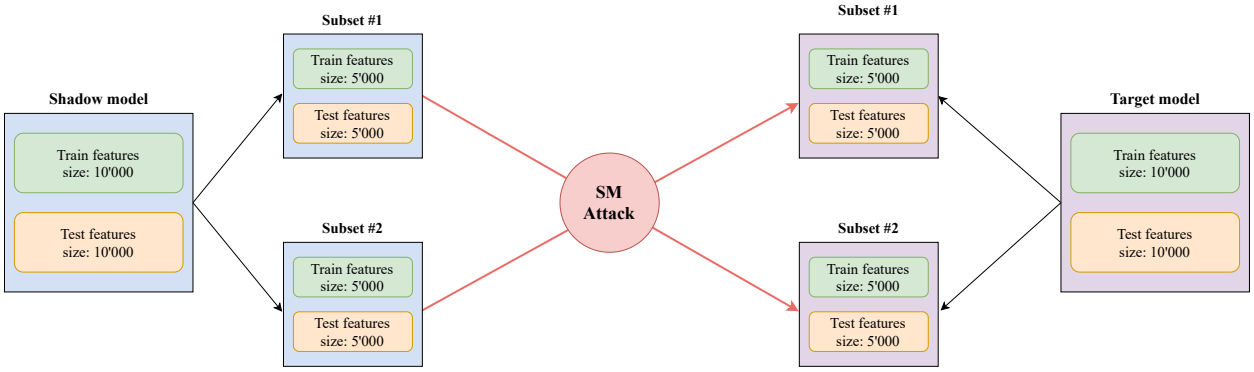


Figure 3: The train and test data of both shadow and target models are split in half. Instead of having two datasets of size 10'000 + 10'000 (train + test), one will have four samples of size 5'000 + 5'000. By doing so, we obtain $N = 4$ different shadow models attacks. The averaged results for this experiment are summarized in Table 14.

5.3 Adversarial knowledge on language models

Previous work on membership inference relied on the observation that models' confidence scores were higher on examples they have already been exposed to (i.e. from the training set). Since LMs like GPT-2 are probabilistic, Carlini et al. [3] proposes *perplexity* as a feature to assess sequence memorization. Indeed, perplexity is a natural likelihood measure of how well a given LM "predicts" the sequence of tokens. Let us denote θ the parameters of the probabilistic model $f_\theta(\cdot)$. The log-perplexity of a sequence $x_1, \ldots, x_n$ is given by:

$$\log \mathcal{P} = -\frac{1}{n} \sum_{i=1}^{n} \log f_\theta(x_i \mid x_1, \ldots, x_{i-1}) \tag{8}$$

The intuition is the following: tokens that are well 'predicted' by the model (in their context) are such that $f_\theta(x_i \mid \cdot)$ will be close to 1, and the log-perplexity will therefore be low. The higher the perplexity, the more the LM will be 'surprised' about the sequence. This is similar to the *loss* feature for classification models: training examples are likely to have lower perplexity (loss) than held-out records.

Furthermore, Carlini et al. [4] introduce multiple metrics based on log-perplexity to distinguish between memorized and non-memorized sequences. These features are inspired by their work and adapted for the MIA framework as following:

– *log-perplexity*, as defined in Eq.8.
– *zlib entropy ratio*: the ratio of the sequence's log-perplexity and the zlib [9] entropy of the text (the number of bits of entropy when the sequence is compressed with zlib compression). This adversary feature aims to ignore sequences with very low perplexity due to repeating substrings [15] (as any text-compressor measure will be small as well).
– *lowercase ratio*: the ratio of the log-perplexities of the original sequence and its lowercased version. The underlying idea is that memorized sequences might be case sensitive. Therefore, overfitted examples that expect a particular casing will have a very small log-perplexity compared to the lowercased one.
– *pre-trained ratio*: this feature is inspired by Carlini et al. [4] idea of computing the ratio of log-perplexities between the original model and a smaller one. In this case, we divide the log-perplexity of the fine-tuned model by the one from the pre-trained model (original GPT-2). This feature will also allow us to verify if the model has learned at all during fine-tuning, in which case the ratio should be below one.

Finally, the features defined above are such that a vulnerable text sequence from the training set will find itself at the left of all distributions. For the attacks, we will consider *log-perplexity* as an adversary feature in itself (to perform threshold and kde attacks), while the *combined* setting will refer to a concatenation of all four metrics. The fbleau attack is not considered in this experiment.

Table 12 summarizes the average log-perplexities for the pre-trained and fine-tuned models considered in this experiment. As a result, one may argue that the 5-epochs fine-tuned model (GPT2-WIKI-5) has reached an acceptable performance, as its average perplexity on the test set is lower than the original GPT-2. However, the LM fine-tuned on 10 epochs (GPT2-WIKI-10) demonstrates an obvious overfitting pattern, as the performance on the train set is significantly improved, but not on the test set. As shown by Figs. A.3 and A.4, both models exhibit significant distributional overfitting on the adversary features, suggesting potentially high vulnerability to MIAs.

Model	Train set	Test set
Pre-Trained GPT-2	3.51 [0.58]	3.50 [0.59]
Fine-Tuned 5 epochs	2.95 [0.78]	3.33 [0.87]
Fine-Tuned 10 epochs	2.74 [0.86]	3.53 [0.99]

Table 12: Description of fine tuned GPT-2 models considered for evaluating MIAs. The average log-perplexities are reported along with the standard deviation for both train and test sets (of size 10'000).

5.4 Results and discussion

The results of MIAs against fine-tuned GPT-2 are shown in Tables 13 (Holdout mode) and 14 (SMs). We find that these models are highly vulnerable to membership inference, with the combined + forest setup reaching 81% accuracy (62% vulnerability) on GPT2-WIKI-5 and 90% accuracy (80% vulnerability) on GPT2-WIKI-10 in Holdout mode. Interestingly, we observe that the average threshold attack on log-perplexity reaches high precision, although not being among the most accurate overall. This indicates that very few log-perplexities from the test set are below the average threshold – hence few False Positives – and that there is also an important part of the train records having a perplexity above the threshold, resulting in False Negatives.

The results in Table 13 also suggest the pertinence of the extended set features built from the log-perplexities. For example, we measure a 10% rise in both accuracy and precision between forest attacks on loss and on combined features. Indeed, the fine-tuning experiment allows the adversary to have access to a pre-trained model and its perplexities before it has been exposed to training data. This is a powerful but realistic setting for the attacker, considering the growing trend in fine-tuning large LMs released to the public.

Moreover, we also notice that the models considered are already considerably vulnerable to threshold attacks on log-perplexity only. This confirms the overfitting patterns (in the average sense) demonstrated by the models' average perplexities (Table 12). Therefore, such results must be interpreted with caution, as fine-tuned models might not exhibit the same overfit on their training set. However, this experiment is relevant in the scope of privacy risk analysis. Indeed, with such models and fine-tuning APIs being available to anyone, it is likely that an individual deploys a model that overfits sensitive data at large scale.

		GPT2-WIKI-5 (5 epochs)		**GPT2-WIKI-10** (10 epochs)	
Feature	Attack	Accuracy (%)	Precision (%)	Accuracy (%)	Precision (%)
log-perplexity	av thr	65.96 ± 0.38	70.40 ± 0.31	74.42 ± 0.43	85.33 ± 0.25
	opt thr	67.00 ± 0.21	67.74 ± 0.28	78.47 ± 0.15	79.47 ± 0.93
	kde	65.65 ± 0.96	64.38 ± 1.98	78.27 ± 0.29	78.13 ± 1.30
	forest	67.05 ± 0.21	68.19 ± 0.39	78.58 ± 0.18	80.02 ± 0.68
combined	forest	81.37 ± 0.22	79.15 ± 0.26	90.20 ± 0.08	89.48 ± 0.21

Table 13: Results for the LM experiments (Holdout mode). Both models are GPT2 fine-tuned on wikitext for 5 and 10 epochs. Results are averaged over 10 random seeds and 95% CIs are provided.

Similarly to Section 4.3, we provide a comparison of the results obtained in Holdout mode with Shadow Model attacks. The results shown in Table 14 are averaged over 4 attacks, using the methodology described in Fig. 3. As *learn* and *eval* sets are of the same size between Holdout mode and SMs, these results can be directly compared with ones obtained above on GPT2-WIKI-10. We find very similar metrics on on all feature/attack pairs, suggesting once again that Holdout mode is an effective unbiased way of assessing MIA vulnerability, in the same way as SMs.

Feature	Attack	**GPT2-SHADOW** (10 epochs) Accuracy (%)	Precision (%)
log-perplexity	av thr	74.92 ± 0.51	84.97 ± 0.31
	opt thr	78.50 ± 0.42	80.05 ± 0.69
	kde	77.34 ± 0.55	75.57 ± 1.46
	forest	78.40 ± 0.41	79.48 ± 0.84
combined	forest	90.14 ± 0.09	89.10 ± 0.17

Table 14: Results for the LM experiments (Shadow model attacks). Both models are GPT2 fine-tuned on wikitext for 5 and 10 epochs. Results are averaged over 4 slices and 95% CIs are provided.

6 Conclusion

Throughout this report, we investigated Membership Inference Attacks (MIAs) under the scope of various ML models, attack methods, and adversarial knowledge. We introduced an effective membership inference approach, namely Holdout mode, which we proved to be unbiased against control models. This setting allows one to evaluate models' vulnerability to MIAs without having to define and train multiple shadow models (SMs). The comparisons provided with SMs suggested that the Holdout mode is indeed a practical unbiased approach that can be readily applied by the scientist before deploying a model trained on sensitive data. The situation where an adversary has access to a fraction of the data with membership labels can also be seen as realistic, as it is common to release a small part of the train set to the public.

Furthermore, we presented various attacks and evaluated them on extended sets of adversary features that can be extracted through black-box queries: loss, entropy, confidence outputs. We observed that the combination of adversarial knowledge is often an efficient way of conducting membership inference. Moreover, threshold-based attacks are found to be less powerful than more complex methods (KNN, RF) when inferring membership on models with a small overfitting gap (in the average sense). These results suggest that threshold inference might not be the optimal approach for exploiting distributional overfitting.

Finally, we proposed to study membership inference on large language models (LMs) with the particular case of fine-tuning. First suggested by the problem of unintended memorization [3], large LMs can indeed be considered as potential targets for MIAs, and it has become very common to fine-tune GPT-2 or BERT models on custom and potentially sensitive data records. The experiments carried out with GPT-2 trained on WikiCorpus demonstrated an important vulnerability to MIAs based on the perplexity measure as adversarial knowledge. This leads to substantial privacy concerns regarding large language models, especially since their performance led researchers to design fine-tuned models on sensitive medical data [1].

References

[1] A. Balagopalan, B. Eyre, J. Robin, F. Rudzicz, and J. Novikova. Comparing pre-trained and feature-based models for prediction of alzheimer's disease based on speech. *Frontiers in aging neuroscience*, 13:189, 2021.

[2] B. Barz and J. Denzler. Do we train on test data? purging cifar of near-duplicates. *Journal of Imaging*, 6(6):41, 2020.

[3] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, and D. Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 267–284, 2019.

[4] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021.

[5] M. X. Chen, B. N. Lee, G. Bansal, Y. Cao, S. Zhang, J. Lu, J. Tsay, Y. Wang, A. M. Dai, Z. Chen, et al. Gmail smart compose: Real-time assisted writing. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2287–2295, 2019.

[6] G. Cherubin, K. Chatzikokolakis, and C. Palamidessi. F-BLEAU: fast black-box leakage estimation. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 835–852. IEEE, 2019.

[7] D. Dua and C. Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

[8] C. Dwork. Differential privacy. In *International Colloquium on Automata, Languages, and Programming*, pages 1–12. Springer, 2006.

[9] J.-l. Gailly and M. Adler. Zlib compression library. 2004.

[10] A. Gokaslan and V. Cohen. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

[11] U. Gupta, D. Stripelis, P. K. Lam, P. M. Thompson, J. L. Ambite, and G. V. Steeg. Membership inference attacks on deep regression models for neuroimaging. *arXiv preprint arXiv:2105.02866*, 2021.

[12] A. Halevy, P. Norvig, and F. Pereira. The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24(2): 8–12, 2009.

[13] J. Hayes, L. Melis, G. Danezis, and E. De Cristofaro. Logan: Membership inference attacks against generative models. In *Proceedings on Privacy Enhancing Technologies (PoPETs)*, volume 2019, pages 133–152. De Gruyter, 2019.

[14] 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*, pages 770–778, 2016.

[15] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

[16] J. Howard and S. Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

[17] H. Hu, Z. Salcic, G. Dobbie, and X. Zhang. Membership inference attacks on machine learning: A survey. *arXiv preprint arXiv:2103.07853*, 2021.

[18] T. Humphries, S. Oya, L. Tulloch, M. Rafuse, I. Goldberg, U. Hengartner, and F. Kerschbaum. Investigating membership inference attacks under data dependencies, 2021.

[19] Y. Idelbayev. Proper ResNet implementation for CIFAR10/CIFAR100 in PyTorch. https://github.com/akamaster/pytorch_resnet_cifar10.

[20] B. Jayaraman, L. Wang, K. Knipmeyer, Q. Gu, and D. Evans. Revisiting membership inference under realistic assumptions. *arXiv preprint arXiv:2005.10881*, 2020.

[21] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[22] B. Kulynych, M. Yaghini, G. Cherubin, M. Veale, and C. Troncoso. Disparate vulnerability to membership inference attacks. *Proceedings on Privacy Enhancing Technologies*, 2022(1):460–480, 2022.

[23] M. Mhapsekar, P. Mhapsekar, A. Mhatre, and V. Sawant. Advanced computing technologies and applications, 2020.

[24] E. Ntoutsi, P. Fafalios, U. Gadiraju, V. Iosifidis, W. Nejdl, M.-E. Vidal, S. Ruggieri, F. Turini, S. Papadopoulos, E. Krasanakis, et al. Bias in data-driven artificial intelligence systems—an introductory survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(3):e1356, 2020.

[25] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, pages 1–26, 2020.

[26] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[27] S. Reese, G. Boleda, M. Cuadros, L. Padró, and G. Rigau. Wikicorpus: A word-sense disambiguated multilingual wikipedia corpus. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, 2010.

[28] A. Sablayrolles, M. Douze, C. Schmid, Y. Ollivier, and H. Jégou. White-box vs black-box: Bayes optimal strategies for membership inference. In *International Conference on Machine Learning*, pages 5558–5567. PMLR, 2019.

[29] A. Salem, Y. Zhang, M. Humbert, P. Berrang, M. Fritz, and M. Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *arXiv preprint arXiv:1806.01246*, 2018.

[30] 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)*, pages 3–18. IEEE, 2017.

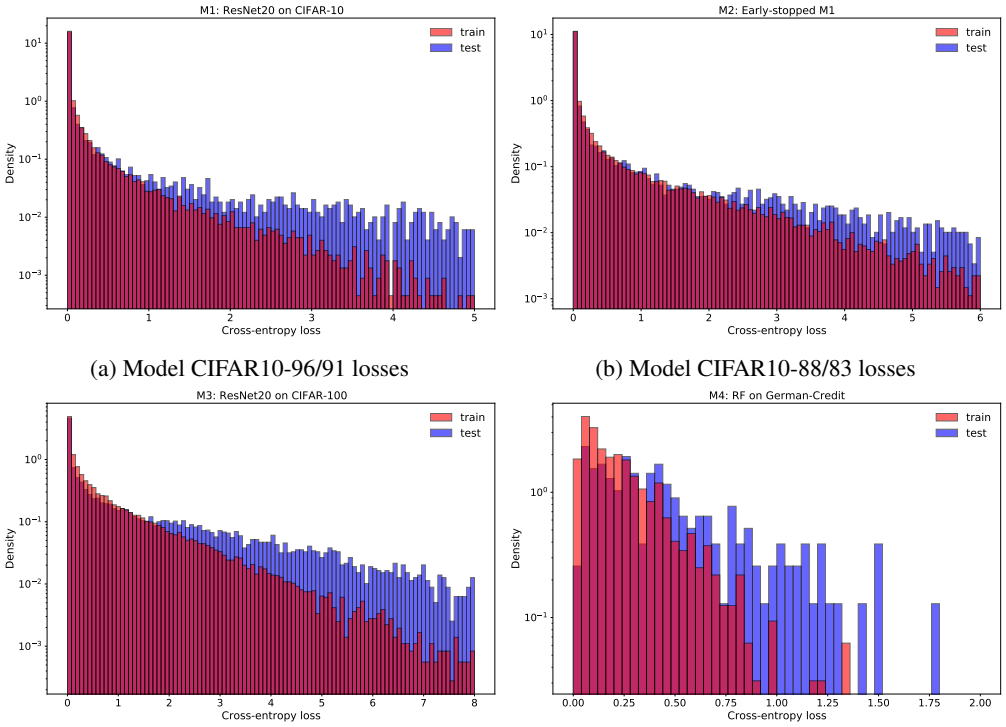
[31] R. Shokri, M. Strobel, and Y. Zick. On the privacy risks of model explanations. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pages 231–241, 2021.

[32] M. Veale, R. Binns, and L. Edwards. Algorithms that remember: model inversion attacks and data protection law. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 376(2133): 20180083, 2018.

[33] 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)*, pages 268–282. IEEE, 2018.

[34] A. Yousefpour, I. Shilov, A. Sablayrolles, D. Testuggine, K. Prasad, M. Malek, J. Nguyen, S. Ghosh, A. Bharadwaj, J. Zhao, et al. Opacus: User-friendly differential privacy library in pytorch. *arXiv preprint arXiv:2109.12298*, 2021.

A Distribution of adversarial features



(a) Model CIFAR10-96/91 losses

(b) Model CIFAR10-88/83 losses

(c) Model CIFAR100-81/65 losses

(d) Model German-96/76 losses

Figure A.1: Distribution of adversary features for models studied in Section 4. Densities (y-axis) are in log-scale. The outliers are excluded from the figures, i.e. the distribution is shown up to the 99th percentile for each model.

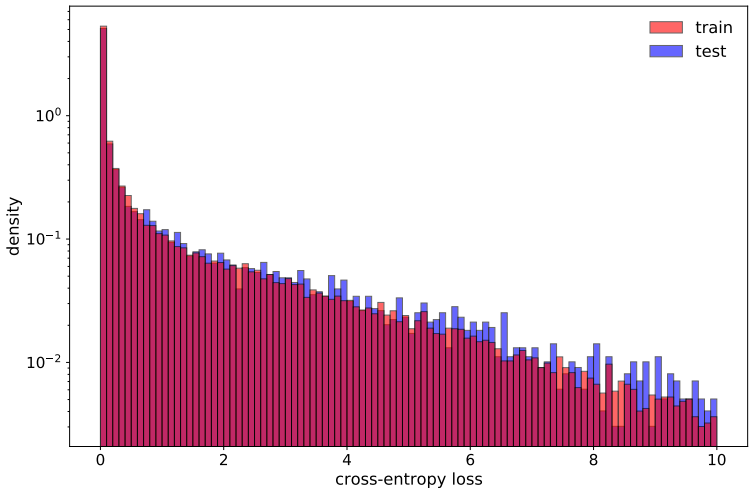


Figure A.2: Distribution of *loss* feature of the DP-trained model in 4.4.

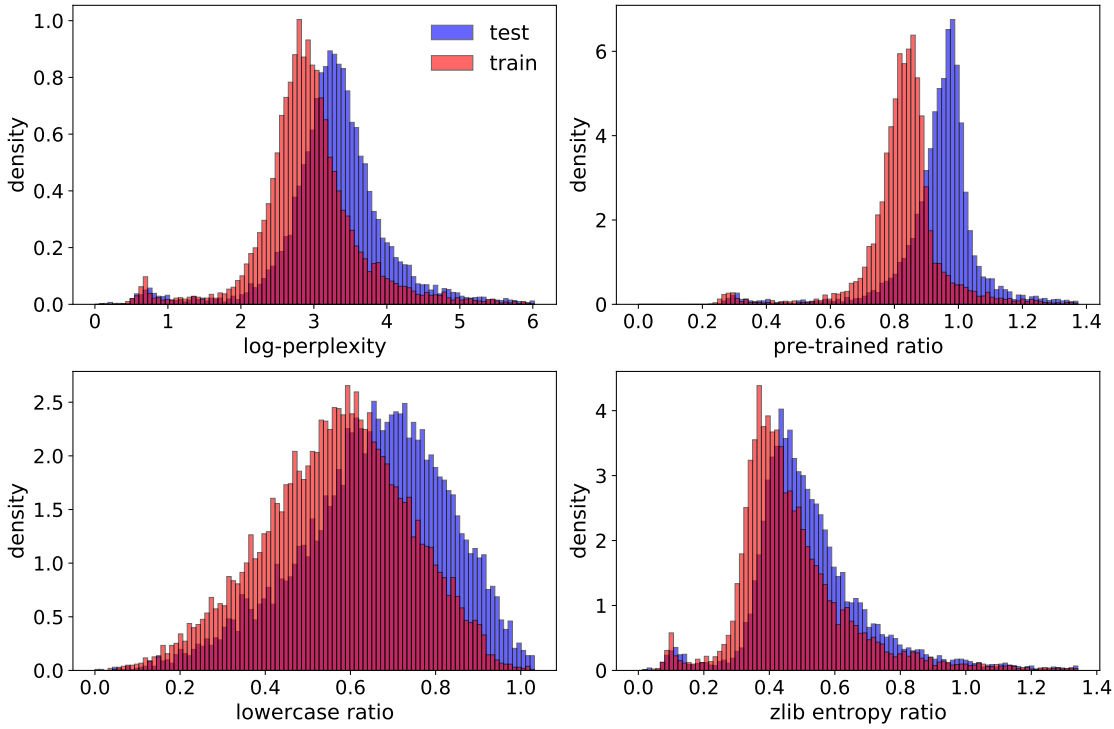


Figure A.3: Distribution of adversary features for MIAs on GPT2-WIKI-5.

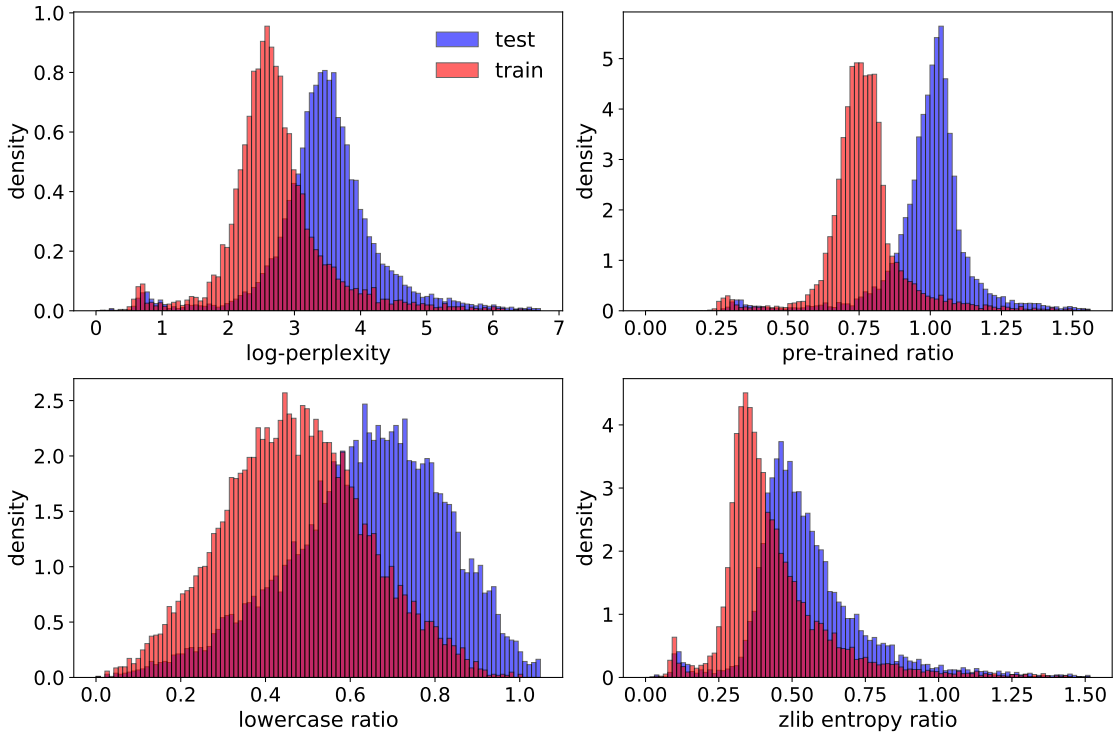


Figure A.4: Distribution of adversary features for MIAs on GPT2-WIKI-10.