AN EFFICIENT SUBPOPULATION-BASED MEMBERSHIP INFRINGEMENT ATTACK

Shahbaz Rezaei  
University of California  
Davis, CA, USA  
srezaei@ucdavis.edu  

Xin Liu  
University of California  
Davis, CA, USA  
xinliu@ucdavis.edu

ABSTRACT

Membership inference attacks allow a malicious entity to predict whether a sample is used during training of a victim model or not. State-of-the-art membership inference attacks have shown to achieve good accuracy which poses a great privacy threat. However, majority of SOTA attacks require training dozens to hundreds of shadow models to accurately infer membership. This huge computation cost raises questions about practicality of these attacks on deep models. In this paper, we introduce a fundamentally different MI attack approach which obviates the need to train hundreds of shadow models. Simply put, we compare the victim model output on the target sample versus the samples from the same subpopulation (i.e., semantically similar samples), instead of comparing it with the output of hundreds of shadow models. The intuition is that the model response should not be significantly different between the target sample and its subpopulation if it was not a training sample. In cases where subpopulation samples are not available to the attacker, we show that training only a single generative model can fulfill the requirement. Hence, we achieve the state-of-the-art membership inference accuracy while significantly reducing the training computation cost.

1 INTRODUCTION

Wide-spread deployment of machine learning models, often trained on private and sensitive data, has raised concerns over privacy of training data Rezaei et al. (2021). Recently, membership inference (MI) attacks have been introduced to identify if a sample has been used during the training of a victim model or not. Existing state-of-the-art MI attacks (Sablayrolles et al., 2019; Watson et al., 2021) often rely on training numerous shadow models to estimate the average output of a typical model. Then, any significant deviation from a typical output is an indication of the victim model being trained on the target sample. Despite their decent accuracy, these attacks require training numerous shadow models, which imposes huge computational cost.

In this paper, we propose a fundamentally different MI attack that achieves the same accuracy as SOTA while significantly reducing the shadow training computational overhead. Here, instead of comparing victim model’s confidence on the target sample versus the average confidence of typical models, which requires training numerous shadow models, we compare the victim model’s confidence on the target sample versus the victim model’s confidence on similar samples from the same subpopulation as the target sample. Hence, we obviate the need to train multiple shadow models. In other words, we calculate the expected value of subpopulation confidence rather than the expected value of shadow models’ confidence. However, in practice, the attacker may not have access to samples from the same subpopulation. To tackle this issue, we develop a BiGAN-like architecture to train a generator that craft samples from the subpopulation of a given image. In other words, our attack only needs training a single generator model once and then it can be used for even unseen samples.
2 Attack Overview

2.1 Background

Membership inference attacks aim to distinguish training samples from non-training samples. Training samples are often called member samples and non-training samples are called nonmember samples. Let \( x, y, \) and \( Y_v(.) \) be the target sample, target label, and victim model’s output, respectively. Now, let \( s(Y_v(x, y)) \) denote the membership score, where a higher score indicates the higher probability of a sample being member. MI attacks aim to introduce an accurate membership score function.

In the literature, there are two types of MI attacks: 1) with sample calibration, and 2) without sample calibration. The calibration process modifies the membership score of an MI attack such that it takes the target sample’s difficulty into account (Watson et al., 2021). The former attacks do not take the difficulty of the target sample into consideration. They essentially compare the victim model’s response on the target sample versus an average response to infer membership (e.g. via shadow models). They use (Shokri et al., 2017; Yeom et al., 2018) confidence output or loss values of target samples. The intuition is that models are often more confident on their training samples. For example, Yeom et al. (2018) uses loss function and Shokri et al. (2017) uses confidence value as a score function. However, it is shown that this leads to poor attack performance and high false positive mainly because well-generalized models output high confidence on majority of nonmember samples as well (Rezaei & Liu, 2021).

The second category of attacks, achieving state-of-the-art MI performance, use some form of sample calibration to distinguish between hard-to-predict member samples from easy-to-predict nonmember samples (Watson et al., 2021). For instance, Sablayrolles et al. (2019) calibrates the score using the average score both when the target sample is in training data and when it is not, as follows:

\[
s_{\text{Sub}}(Y_v(x, y)) = s(Y_v(x, y)) - \frac{\mathbb{E}_{Y \sim A(x \in D)}[s(Y_v(x, y))] + \mathbb{E}_{Y \sim A(x \notin D)}[s(Y_v(x, y))]}{2},
\]

(1)

where \( A(c) \) is a randomized training algorithm that samples from the distribution of trained shadow models following a specified condition \( c \). \( D \) is the shadow model training dataset. The intuition is that if a sample is easy-to-predict, then the calibration term is also large. So, the total membership score is small. However, training overhead of this attack is large particularly if all target samples are not known during shadow training. In that case, each time a new sample is targeted, \( \mathbb{E}_{Y \sim A(x \in D)} \) should be calculated from scratch by training new shadow models. Watson et al. (2021) tackles this issue by estimating the calibration only on shadow models trained without the target sample. Simply put, their membership score is

\[
s_{\text{Watson}}(Y_v(x, y)) = s(Y_v(x, y)) - \mathbb{E}_{Y \sim A(x \notin D)}[s(Y_v(x, y))].
\]

(2)

For the base membership score, \( s(.) \), they explored confidence, loss, and gradient norm and showed that loss is slightly outperform others.

2.2 Our Subpopulation-based Attack

All MI attacks discussed above require training numerous shadow models to accurately estimate the expectation in the calibration term and thus are computationally heavy. In contrast, we propose a fundamentally different approach by running the expectation over similar samples rather than similar models. In other words, our attack estimates whether the victim model’s loss on the target sample is significantly smaller than the victim models’ loss on samples from the same subpopulation whose loss should be similar. We define subpopulation-based score as:

\[
s_{\text{ours}}(Y_v(x, y)) = s(Y_v(x, y)) - \mathbb{E}_{x' \sim G(x)}[s(Y_v(x', y))],
\]

(3)

where \( G(x) \) is a function sampling from the distribution of samples that belong to the same subpopulation as \( x \). The benefit of our subpopulation-based approach is that it obviates the need to train numerous shadow models. However, obtaining images from the same subpopulation is not trivial. How to define a subpopulation and to train \( G(x) \) is covered in the next section.
2.3 Crafting Subpopulations

Latent representation of deep learning models have been extensively used as a means of semantic closeness in various applications (Vu et al., 2020; Zhang & Zhu, 2018; Yang et al., 2019). We define a subpopulation of $x$, $X_{sub}$, such that their samples are close to $x$ in a latent space. Formally, a subpopulation is defined as follows

$$X_{sub} = \{x' : Dist(E_v(x'), E_v(x)) < \epsilon\}, \quad (4)$$

where $E_v$ is the latent representation of the victim model, and $Dist(.)$ is a distance metric. Here, we consider the output of the last fully connected layer before the softmax as the latent representation. When abundant samples are available to the attacker, she can easily use $X_{sub}$ to launch the subpopulation-based attack as explained in Section 2.2. The downside is that the attacker needs to have multiple extra samples for each target sample.

To solve this challenge, we propose a modified version of the BiGAN architecture (Donahue et al., 2016) to train a generator model, $G$. The generator learns the mapping from the latent space to the input space which can be used to obtain $X_{sub}$ for MI attacks. However, the original BiGAN architecture cannot be used directly for two reasons: 1) the encoder in BiGAN is trainable while the encoder in our case is the fixed victim model which we cannot change. 2) the original BiGAN forces the encoder to map the input to a uniformly distributed latent space. However, here, the victim model is fixed and there is no guarantee that the latent space follows a uniform or any known distribution.

Hence, to address these issues, we make the following changes:

- We replace the encoder with $E_v$ and block the back-propagation from training $E_v$.
- Instead of sampling from a uniform distribution, we obtain the latent representation of all samples in attacker’s dataset ($\neq$ victim training dataset) from which we sample as an input for the generator.

See Figure 1 for an overview illustration of our MI attack.
3 Experiments

We conduct experiments on multiple image classification benchmarks: MNIST (LeCun et al., 1998), FMNIST (Xiao et al., 2017), SVHN (Netzer et al., 2011), and CIFAR10/CIFAR100 (Krizhevsky et al., 2009). We divide the train set of these datasets into two parts: victim training dataset and attacker training dataset. The test set is only used for attack evaluation. For MNIST and FMNIST, we choose multi-layer perceptron (MLP) with 4 hidden layers as the victim model. For SHVN, we choose LeNet. For CIFAR10/100 we choose both LeNet and ResNet20. Details of the victim models, generators and discriminators are presented in Appendix A.1. We train all models using SGD with a learning rate of $0.1$. We reduce the learning rate by a factor of 10 at epoch 50 and 75.

The performance of the victim models are shown in Table 1.

We compare our attack with multiple SOTA MI attacks. Unless specified, we follow the same experimental settings to train attack models as suggested in their original paper. For Shokri et al. (2017), we train 100 shadow models for all datasets. Yeom et al. (2018) attack requires the knowledge about the average training loss to set the threshold which we assume it is known to the attacker for this particular attack. For attack with calibration, including ours, Watson et al. (2021), and Sablayrolles et al. (2019), we use the loss function as the base membership score before calibration. We train 30 shadow models for these two attacks as suggested in Sablayrolles et al. (2019). We also compare our attack with Jayaraman et al. (2021) because it is similar to our attack in an interesting way: if we define $X_{\text{sub}}$ as target samples with random noise, the two attacks would be essentially similar. However, we show that directly adding random noise to input space leads to poor performance. For Jayaraman et al. (2021), we first use $T = 100$ and $\sigma = 0.01$ as suggested in the original paper. Due to the poor performance, we perform random hyper-parameter tuning on $\sigma$, 10 times and we report the best result.

For our attack, we consider two scenarios: 1) when a large amount of samples is available to find subpopulation, 2) when training a generator is needed. For the first scenario, we only consider SVHN dataset because it is the only dataset with abundant extra data which is often ignored. We use cosine similarity in latent space to find subpopulations. We find no significant difference when using L2 distance. For the second scenario, we first train a generator in BiGAN-like architecture the details of which is presented in Appendix A.1. We get the latent representation of each sample using $E_V$. Then, we add small random Gaussian noise, $\epsilon$, with zero mean and standard deviation $\sigma$ proportional to each activation value. In other words, for the latent representation of $x$, denoted as $L_i = E_V(x)$, noisy latent representation is obtained by $L'_i = L_i + |L_i|\epsilon$. The purpose of the scale factor is to make sure that activations with small values remain small, otherwise it may change the subpopulation or even class of the image. We set $\sigma = 0.05$ for mnist and fmnist, and 0.5 for other datasets. Finally, we feed the noisy latent representation to the generator to craft a subpopulation for each sample. We craft 30 images per sample.

Figure 2 illustrates some examples of the target images and their corresponding subpopulation images crafted by our BiGAN generator. Generated images are often similar to the crafted versions with small difference in color, orientation, pattern, background, and texture. Clearly, crafted samples
Table 1: Accuracy of victim models

| Dataset   | MNIST | FMNIST | SVHN | C-10 | C-100 | C-10 | C-100 |
|-----------|-------|--------|------|------|-------|------|-------|
| Model     | MLP   | MLP    | LeNet| LeNet| LeNet | ResNet20| ResNet20|
| Victim train Acc | 100% | 100% | 99.89% | 95.13% | 95.27% | 98.51% | 96.71% |
| Victim test Acc   | 97.43% | 89.59% | 87.82% | 57.82% | 22.37% | 74.47% | 33.03% |

Table 2: AUC of various datasets, target models, and MI attack models

| Dataset   | MNIST | FMNIST | SVHN | C-10 | C-100 | C-10 | C-100 |
|-----------|-------|--------|------|------|-------|------|-------|
| Model     | MLP   | MLP    | LeNet| LeNet| LeNet | ResNet20| ResNet20|
| Yeom et al. (2018) | 51.58% | 54.84% | 57.54% | 77.56% | 91.98% | 70.62% | 93.03% |
| Shokri et al. (2017) | 51.98% | 57.69% | 58.07% | 75.52% | 84.72% | 67.72% | 87.75% |
| Jayaraman et al. (2021) | 52.20% | 55.78% | 56.45% | 75.01% | 80.64% | 68.58% | 85.97% |
| Watson et al. (2021) | 54.07% | 60.52% | 62.97% | 80.37% | 95.47% | 72.78% | 93.58% |
| Sablayrolles et al. (2019) | **55.54%** | 62.55% | **63.41%** | 81.56% | 96.10% | 74.84% | **95.21%** |
| Ours      | 54.66% | **62.88%** | 62.05% | **81.94%** | **96.23%** | 75.05% | 94.56% |
| Ours (Black-box) | 54.12% | 62.07% | 61.13% | 81.24% | 95.86% | 74.23% | 94.24% |
| Ours with natural subpopulation | - | - | 61.61% | - | - | - | - |

belong to the same subpopulation as the original images. Hence, a victim model output confidence should not be significantly greater than images from the same subpopulation, otherwise it is an indication that the target sample was used during training.

Table 1 illustrates the attack performance of all MI attacks. Sablayrolles et al. (2019) outperforms all existing attacks in literature while our attack achieves similar or better performance. Although Watson et al. (2021) is more efficient, its AUC is lower than Sablayrolles et al. (2019) attack. All previous MI attacks that do not consider sample difficulty (calibration) are substantially worst. Our subpopulation-based MI attack is on a par with Sablayrolles et al. (2019) while obviating the need to train a large number of shadow models. A comparison of computational cost of training multiple shadow model versus a generator model is reported in Appendix A.2. Moreover, our attack achieves a good performance when abundant data is available in which case no model training is needed. It shows that for the membership inference purpose our proposed BiGAN-like architecture achieves the same performance as natural images.

3.1 Black-box Setting

To get the subpopulation of a target sample, our attack needs to know the latent representation to generate or find semantically similar samples. This is done by using the last layer before the softmax of the victim model. However, in practice, the victim model’s intermediate layers might not be available to the attacker. In this case, the attacker also trains an encoder during the BiGAN training instead of using the victim model. As shown in Table 2, the black-box scenario barely changes the attack performance.

4 Conclusion

In this paper, we propose a fundamentally different approach towards membership inference. Instead of comparing the victims model output versus shadow models’ output, we essentially compare the victim model’s output on the target sample versus victim model’s output on samples from the same subpopulation. This new way of approaching membership inference obviates the need to train dozens to hundreds of shadow models and makes MI attacks more computationally efficient. Moreover, we show that when samples from the same subpopulation is not available, we can train a single generator using BiGAN-like architecture to craft samples of subpopulations. Hence, in the worst case, we only need to train a single generator. Our evaluation results demonstrate that our attack can achieve the state-of-the-art MI attack accuracy with no shadow model training.
REFERENCES

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

Bargav Jayaraman, Lingxiao Wang, Katherine Knipmeyer, Quanquan Gu, and David Evans. Revisiting membership inference under realistic assumptions. *In Proceedings on Privacy Enhancing Technologies (PoPETs)*, 2021.

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

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop*, 2011.

Shahbaz Rezaei and Xin Liu. On the difficulty of membership inference attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Shahbaz Rezaei, Zubair Shafiq, and Xin Liu. Accuracy-privacy trade-off in deep ensemble: A membership inference perspective. *arXiv preprint arXiv:2105.05381*, 2021.

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

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18. IEEE, 2017.

LyVu, Quang Uy Nguyen, Diep N Nguyen, Dinh Thai Hoang, Eryk Dutkiewicz, et al. Learning latent representation for iot anomaly detection. *IEEE Transactions on Cybernetics*, 2020.

Lauren Watson, Chuan Guo, Graham Cormode, and Alex Sablayrolles. On the importance of difficulty calibration in membership inference attacks. *arXiv preprint arXiv:2111.08440*, 2021.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Ruihan Yang, Dingsu Wang, Ziyu Wang, Tianyao Chen, Junyan Jiang, and Gus Xia. Deep music analogy via latent representation disentanglement. *arXiv preprint arXiv:1906.03626*, 2019.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pp. 268–282. IEEE, 2018.

Quanshi Zhang and Song-Chun Zhu. Visual interpretability for deep learning: a survey. *arXiv preprint arXiv:1802.00614*, 2018.

A APPENDIX

A.1 FULL TRAINING REPORT

The MLP model for MNIST and FMNIST consists of 5 hidden layers of size 1024, 512, 256, 128, and 100 with LeakyReLU activation followed by a softmax layer. The last layer before the softmax is used as a latent representation. As the result, the generator has a reverse architecture, starting from an input of dimension 100 followed by 5 layers of size 128, 256, 512, 1024, and 784 (input image size). We use LeakyReLU in the generator as well. For the discriminator, we use an architecture similar to the encoder with a few changes: 1) the input is the concatenation of the input image of size 784 and latent representation of size 100, and 2) the last softmax layer is replaced with a dense layer of 1 neuron as the task is binary classification.
For SVHN, CIFAR10, and CIFAR100 datasets, we train a well-known LeNet model as the victim/encoder model. Here, the internal representation is a vector of size 84. Hence, our generator has input dimension of 84 followed by a dense layer of size 512. Then, the model reshapes it to (1, 1, 512) followed by four convolutional blocks of size 512, 256, 128, and 64. Each convolutional block consists of a 2D Convolutional Transpose layer (filter size of (2, 2) and strides of (2, 2)), LeakyReLU, and 2D Convolutional (filter size of (3, 3)). The number of filters is specified by the block size. However, the number of filters of 2D Convolutional of the last convolutional block is set to 3 to make sure that the output size is (32, 32, 3).

In our experimental evaluations, a shallow discriminator based on LeNet architecture suffers from mode collapse. So, we use a deeper convolutional model consists of four 2D convolutional layers of size 128, 256, 512, and 1024 (filter size of (3, 3) and strides of (2, 2)) followed by LeakyReLU activation after each 2D convolutional layer. Then, we use a flatten layer and concatenate the latent representation here. We find that concatenating latent representation at the middle of the model achieves better convergence than at the beginning of the model. Finally, there is a dense layer of size 64, followed by a LeakyReLU activation and the last dense layer of size 1.

For CIFAR10 and CIFAR100 datasets, we also use ResNet20 as victim/encoder model. However, we use the same generator and discriminator as we use in LeNet case. In our BiGAN training, we find that when a Cosine similarity loss \(^1\) is around 20% of the first epoch, the generated images are good enough. For CIFAR10 and CIFAR100, we find that training a GAN model for a few epochs and using the pre-trained generator in the BiGAN architecture prevents mode collapse in BiGAN training. The overall training time is similar since the BiGAN converges faster with a pre-trained generator.

A.2 Training Cost Comparison

We use Python 3.6.12 and Tensorflow 2.3, and a server with Intel(R) Xeon(R) Platinum 8168 CPU @ 2.70GHz and NVIDIA GeForce RTX 2080 Ti GPU using Ubuntu 18.04. Table 3 shows the training time required for each attack on minutes using a single GPU instance. Note that we group experiments that has similar training size and model architecture because they essentially have similar training time. As shown in Table 3, our attack significantly reduces the overall training time overhead. Although training a BiGAN is computationally more expensive than a single discriminator (shadow) model, previous MI attacks require training more than one shadow model. Here, both Watson et al. (2021) and Sablayrolles et al. (2019) is trained using 30 shadow models, as suggested in Sablayrolles et al. (2019). Additionally, Sablayrolles et al. (2019) attack requires the average shadow model output of cases where the shadow model is trained with the target sample. Hence, for each new sample to investigate, the MI attack needs to compute \(E_{Y \leftarrow A(x \in D)}[s(Y, (x, y))]\) from scratch, meaning training 15 new shadow models. Our attack and Watson et al. (2021) do no require training new models for each new target sample. Moreover, our attack is also more effective when the victim model is deeper, such as in ResNet20. In this case, our generator and discriminator architecture is still the same as the LeNet case. Although it might take longer for the generator model to converge and find the mapping from the latent representation to the input space, it is significantly more efficient than training 30 ResNet20 models.

\(^1\)The loss is defined in TensorFlow as tf.keras.losses.CosineSimilarity().
Table 3: Training time comparison of MI attacks in minutes.

| Dataset            | MNIST/FMNIST | SVHN | C-10/ C-100 | C-10/C-100 |
|--------------------|--------------|------|--------------|-------------|
| Model              | MLP          | LeNet| LeNet        | ResNet20    |
|                    |              |      |              |             |
| When all target samples are known before the attack |             |      |              |             |
| Watson et al. (2021) | 46.37       | 86.72| 56.42        | 281.60      |
| Sablayrolles et al. (2019) | 46.37       | 86.72| 56.42        | 281.60      |
| Ours               | 4.97         | 31.26| 24.73        | 46.36       |
| Ours without BiGAN | 0            | 0    | 0            | 0           |
| Training time per new sample |             |      |              |             |
| Watson et al. (2021) | 0            | 0    | 0            | 0           |
| Sablayrolles et al. (2019) | 23.185     | 43.36| 28.21        | 140.80      |
| Ours               | 0            | 0    | 0            | 0           |
| Ours without BiGAN | 0            | 0    | 0            | 0           |