Review of Membership Inference Attacks Against Machine Learning Models

Reza Bayat
reza.bayat76@gmail.com

February 2020

1 Introduction

In summary, the "Membership Inference Attacks Against Machine Learning Models" paper demonstrates and quantifies the problem of machine learning models leaking information about their training datasets. And it introduces a shadow learning technique that works with minimal knowledge about the target model and its training dataset. It shows how the leakage of membership information is related to model overfitting. Moreover, it discusses the root causes that make these attacks possible and quantitatively compares mitigation strategies such as limiting the model's predictions to top k classes, decreasing the precision of the prediction vector, increasing its entropy, or using regularization while training the model.[1]

2 Definition of privacy in machine learning

2.1 Inference about members of the population

A plausible notion of privacy, known in statistical disclosure control as the "Dalenius desideratum," states that the model should reveal no more about the input to which it is applied than would have been known about this input without applying the model.[1]

2.2 Inference about members of the training dataset

Determining whether a given data record was part of the model's training dataset or not.[1]

3 What is membership inference attacks?

Given a data record and black-box access to a model is used to determine if the record was in the model's training dataset.[1]

The targets of these attacks are machine learning models, and the purpose is to recognize the training data of the models. These can be dangerous because sometimes training data are sensitive and should not be identifiable to unauthorized persons.[1]

Membership inference against a target model uses adversarial machine learning to train custom inference models to recognise differences in the target model's predictions on the inputs that it trained on versus the inputs that it did not train on, so in a way membership inference problem is converted to a classification problem.[1]

4 Attack model

Membership inference attacks require the following steps:

- Availability of adequate training data collected from the model itself via sequential queries of possible inputs or gathered from available public or private datasets that the attacker has access to.[1]
- Create several shadow models that mimic the model (i.e., take similar inputs and outputs of the target model) and tuned with high precision and recall on samples of the training data that was collected.[1]
- Train a discriminator that learns the difference in output between the seen training data and the unseen test or validation data based on the input and output to the numerous shadow models. This discriminator is then used to evaluate the target API and determine if a data point is "in" or "out" of the target model, allowing the attacks to be executed in a "black-box" environment.[1]

5 Generate training data for shadow models

To train shadow models, the attacker needs training data that is distributed similarly to the target model's training data. Several methods offered for generating such data are stated below:

5.1 Model-based synthesis

If the attacker does not have real training data nor any statistics about its distribution, he can generate synthetic training data for the shadow models using the target model itself. The intuition is that records that are classified by the target model with high confidence should be statistically similar to the target's training dataset and thus provide excellent fodder for shadow models. Synthesis process runs in two phases:[1]

- **Search**, using a hill-climbing algorithm, the space of possible data records to find inputs that are classified by the target model with high confidence.

- **Sample** synthetic data from these records. After this process synthesizes a record, the attacker can repeat it until the training dataset for shadow models is full.

5.2 Statistics-based synthesis

The attacker may have some statistical information about the population from which the target model's training data was drawn, like having prior knowledge of the marginal distributions of different features.[1]

5.3 Noisy real data

The attacker may have access to some data that is similar to the target model's training data and can be considered as a "noisy" version thereof. This scenario forms the case where the training data for the target and shadow models are not sampled from exactly the same population, or else sampled in a non-uniform way.[1]

6 Effective factors of attack accuracy

6.1 Number of classes and training data per class

The number of output classes of the target model contributes to how much the model leaks. The more classes, the more signals about the internal state of the model are available to the attacker.[1]

6.2 Overfitting

The more overfitted a model, the more it leaks but only for models of the same type.[1]

7 Mitigate membership inference attacks

To defend against these attacks, we can reduce the available information to the attacker as much as possible. However, it not enough to overcome this issue. In the following, some ways illustrated to mitigate membership inference attacks.

7.1 Restrict the prediction vector to top k classes

When the number of classes is large, many classes may have very small probabilities in the model's prediction vector. The smaller k is the less information the model leaks. In the extreme case, the model returns only the label of the most likely class without reporting its probability.[1]

7.2 Coarsen precision of the prediction vector

Round the classification probabilities in the prediction vector down to d floating point digits. The smaller d is the less information the model leaks.[1]

7.3 Increase the entropy of the prediction vector

One of the signals that membership inference exploits is the difference between the prediction entropy of the target model on its training inputs versus other inputs.[1]

7.4 Use regularization

Regularization techniques are used to reduce overfitting in the learning process, so prevent leaking information.[1]

8 Improve the attacks model

I recommend two improvements:

- Generating a dataset using simulated-annealing and GANs.
- Using other techniques to prevent overfitting.

8.1 Generating a dataset using simulated-annealing and GANs

Generating data for the shadow models have a dramatic effect on the performance attack model. The paper proposed that if we do not have adequate data for each class, the membership inference attack fails against the target model. The relationship between the amount of training data per class and the accuracy of membership inference is complicated, but in general, the more data in the training dataset is associated with a given class, the lower the attack precision for that class.

Paper introduces the hill-climbing algorithm in the model-based synthesis algorithms to generate candidates for training and test data. I reviewed some techniques such as hill-climbing, simulate-annealing, and genetic algorithm also compared their advantages and disadvantages, and find out that simulate-annealing will have more considerable accuracy for generating datasets.

A hill-climbing algorithm never makes a move towards a lower value guaranteed to be incomplete because it can get stuck on a local maximum. Furthermore, if the algorithm applies a random walk, by moving a successor, then it may complete but not efficient. Simulated-annealing is an algorithm that yields both efficiency and completeness.

In the simulated-annealing in which the algorithm picks a random move, instead of picking the best move. If the random move improves the state, then

it follows the same path. Otherwise, the algorithm follows the path which has a probability of less than one, or it moves downhill and chooses another path.

The simulate-annealing algorithm includes the following steps:

1. Generate a random solution
2. Calculate its cost using some cost function
3. Generate a random neighbor solution and calculate its cost
4. Compare the cost of old and new random solution
5. If $\text{Cost}_{\text{old}} > \text{Cost}_{\text{new}}$ then go for old solution otherwise go for a new solution
6. Repeat steps 3 to 5 until reach an acceptable optimized solution of a problem

Future works: In [2] proposed the use of Generative Adversarial Networks (GAN) to generate artificial training data for machine learning tasks. The authors generated totally synthetic data for a binary classification problem (cancer detection). It seems interesting to use this approach to generate data for shadow models.

8.2 Using other techniques to prevent overfitting

Although overfitting is not the only factor that causes a model to be vulnerable to membership inference, but it is an important reason why machine learning models leak information about their training datasets. The more overfitted a model, the more it leaks. After some study, I discovered some methods to prevent overfitting.

The following techniques prevent overfitting, and they could be applied to the target models to prevent from leaking information.

- **Early stopping** is a form of regularization while training a model with an iterative method, such as gradient descent. Since all the neural networks learn exclusively by using gradient descent, early stopping is a technique applicable to all the problems. This method updates the model to make it better fit the training data with each iteration. Up to a point, this improves the model's performance on data on the test set. Past that point, however, improving the model's fit to the training data leads to increased generalization error. Early stopping rules guide as to how many iterations can be run before the model begins to overfit.
- **Cross-validation** is a considerable preventative measure against overfitting. The idea is to use initial training data to generate multiple mini train-test splits. Use these splits to tune models. In standard k-fold cross-validation, partitioned the data into k subsets, called folds. Then, iteratively models trained the algorithm on k-1 folds while using the remaining fold as the test set (called the "holdout fold"). Cross-validation

allows tuning hyperparameters with the only original training set. This allows us to keep the test set as a genuinely unseen dataset for selecting the final model.

- **Train with more data** will not work every time, but training with more data can help algorithms detect the signal better.

References

[1] R. Shokri, M. Stronati, C. Song and V. Shmatikov, "Membership inference attacks against machine learning models," in IEEE Symposium on Security and Privacy (SP), Oakland, 2017.

[2] F. H. K. dos S. Tanaka and C. Aranha, "Data Augmentation Using GANs," 2019.