Mitigating Membership Inference Attacks by Self-Distillation Through a Novel Ensemble Architecture

Xinyu Tang\(^1\)  Saeed Mahloujifar\(^1\)  Liwei Song\(^1\)  Virat Shejwalkar\(^2\)  Milad Nasr\(^2\)

Amir Houmansadr\(^2\)  Prateek Mittal\(^1\)

\(^1\)Princeton University, \(^2\)University of Massachusetts, Amherst
Membership inference attack against machine learning (ML) models

**Membership Inference Attack (MIA)**

- Guess whether the target sample is used to train the target machine learning model or not.

\[(x, y)\]

**(x, y)**

(\text{Target sample})

\[\text{ML model} \rightarrow \text{Prediction of } x\]

\[y\]

\[\text{Member or Non-member?}\]
Why we design defense against MIA?

MIA can lead to **substantial privacy leakage**.
- MIA reveals private information in real-world scenarios.
- MIA can be foundation of stronger attacks.
MIA’s intuition: Machine learning models show different behaviors on members and non-members

**Members**: in the training set.  **Non-members**: not in the training set.

Defense intuition: We should enforce the model to **behave similarly** on all samples (members and non-members).
Our contribution

SELENA: a new defense to enforce the model to behave similarly on members and non-members.

Our defense SELENA can achieve a good utility-privacy trade-off.

- Utility: high classification accuracy.
- Privacy: effectively mitigate practical MIAs.
Design intuition: How to enable the model to behave similarly on members and non-members?

Sample A is in the training set. Sample B is not in the training set.

How to make these two queries to follow the same distribution?
Design intuition: How to enable the model to behave similarly on members and non-members?

Sample A is in the training set. Sample B is not in the training set.

How to make these two queries to follow the same distribution?
Design intuition: How to enable the model to behave similarly on members and non-members?

Sample A and B both are not in the training set.

Remove A from training set before model training?

Problem: we cannot remove all points from training set as we still need training data to train models.
Design intuition: How to enable the model to behave similarly on members and non-members?

This motivates our Split-AI.

How about multiple models?

ML model
ML model
ML model
Split-AI: training

$K$ sub-models in total. -> $K$ subsets of training set.

- Each training sample: member for $(K-L)$ sub-models, non-member for $L$ sub-models.
Intuition for adaptive inference follows our design intuition.
Split-AI: adaptive inference

\(U(x)\): sub-models not trained with member sample \(x\). \(|U(x)|=L\).

For queried member sample \(x\),

\[
\frac{1}{L} \sum_{i \in U(x)} F_i(x)
\]
Split-AI: adaptive inference

$U(x)$: sub-models not trained with member sample $x$. $|U(x)|=L$.

For queried non-member sample $x$, randomly select member sample $x'$,

$$\frac{1}{L} \sum_{i \in U(x')} F_i(x)$$
Privacy implication of Split-AI

Direct single-query attack: Attacker only queries the target sample once. The attacker \textbf{cannot} infer the membership of this sample.
Attacker is not limited to only directly querying the target sample.

Sample A is in the training set. The **perturbed member attack** motivates the design of our Self-Distillation.
Self-Distillation

We want a new model to have similar behavior as Split-AI.
We want a new model to have similar behavior as Split-AI.
Self-Distillation

We want a new model to have **similar behavior** as Split-AI.
Why Self-Distillation helps?

The new model has the **similar good property** as Split-AI.

The new model **solves the limitation** of Split-AI in perturbed member attack.

---

**Diagram:**

Member A

Query each **exact** training sample *once*

Non-member B

Soft labels of training set

(Self-)Distillation

A new model

Queries
Experiment setup

Datasets and models.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Purchase100</th>
<th>Texas100</th>
<th>CIFAR100</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Training samples</td>
<td>19,732</td>
<td>10,000</td>
<td>50,000</td>
</tr>
<tr>
<td>Models</td>
<td>4-layer FC networks</td>
<td>4-layer FC networks</td>
<td>ResNet-18</td>
</tr>
</tbody>
</table>

- Also investigate different model architectures for ablation study.

State-of-the-art MIA defenses to be compared.
- Adversarial Regularization (AdvReg, CCS 2018) and MemGuard (CCS 2019).
- Also included undefended model as a baseline.

- Utility: Classification accuracy.
- Privacy: MIA average attack accuracy (baseline: random guess 50%).
Experiment: evaluation of utility privacy trade-off

Compared to undefended models/MemGuard.
- Our SELENA only incurs **3.9% classification accuracy drop**.
- Best **MIA attack accuracy** is much **lower**.
  - 13% lower than undefended model, 11.5% lower than MemGuard.

<table>
<thead>
<tr>
<th>Defense method</th>
<th>Accuracy on training set</th>
<th>Accuracy on test set</th>
<th>Best attack accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>99.98%</td>
<td>83.2%</td>
<td>67.3%</td>
</tr>
<tr>
<td>MemGuard</td>
<td>99.98%</td>
<td>83.2%</td>
<td>65.8%</td>
</tr>
<tr>
<td>AdvReg</td>
<td>91.9%</td>
<td>78.5%</td>
<td>57.4%</td>
</tr>
<tr>
<td>Our SELENA</td>
<td>82.7%</td>
<td>79.3%</td>
<td>54.3%</td>
</tr>
</tbody>
</table>
Experiment: evaluation of utility privacy trade-off

Compared to adversarial regularization.

- Our SELENA achieves both higher classification accuracy and lower best MIA attack accuracy.

<table>
<thead>
<tr>
<th>Defense method</th>
<th>Accuracy on training set</th>
<th>Accuracy on test set</th>
<th>Best attack accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>99.98%</td>
<td>83.2%</td>
<td>67.3%</td>
</tr>
<tr>
<td>MemGuard</td>
<td>99.98%</td>
<td>83.2%</td>
<td>65.8%</td>
</tr>
<tr>
<td>AdvReg</td>
<td>91.9%</td>
<td>78.5%</td>
<td>57.4%</td>
</tr>
<tr>
<td>Our SELENA</td>
<td>82.7%</td>
<td>79.3%</td>
<td>54.3%</td>
</tr>
</tbody>
</table>
Conclusion

<table>
<thead>
<tr>
<th>Provable privacy</th>
<th>Low utility</th>
<th>Provable privacy</th>
<th>High Utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>DP-based: DP-SGD</td>
<td>Desired</td>
<td>Not considered</td>
<td>Desired</td>
</tr>
</tbody>
</table>

Adversarial Regularization (CCS 18)
MemGuard (CCS 19)
Our Work SELENA (Security 22)

Source code: [https://github.com/inspire-group/MIAdefenseSELENA](https://github.com/inspire-group/MIAdefenseSELENA)