Security and Privacy in Machine Learning
Threat Models and Mitigation Measures

Raksha Ramakrishna

Joint work with György Dán

EECS/CS/NSE KTH
ML is expected to become ubiquitous

Communication networks
Smart grids
Healthcare
Transportation systems
Smart cities and buildings
Manufacturing
Training ML models with centralized data

Empirical loss function
\[ L(\theta) = \sum l(\theta; x_i, y_i) \]

Variety of loss functions available
- Cross-entropy
- Log loss
- Exponential loss
- Hinge loss
- Mean Square Error (MSE, l2 norm)
- Mean Absolute Error (MAE, l1 norm)
- Huber Loss

Training:
\[ \min_{\theta} L(\theta) \]
Federated Learning - Distributed Data

- Objective
  \[ \min_{\theta} L(\theta), \quad \text{where} \quad L(\theta) = \sum_{k=1}^{K} p_k L_k(\theta) \]

- Local objectives \( L_k(\theta) \)
  - Empirical loss function
    \[ L_k(\theta) = \sum_{i_k=1}^{N_k} l_k(\theta; x_{i_k}, y_{i_k}) \]

- Weighting of local objectives
  - Uniform \( p_k = \frac{1}{n} \)
  - Proportional \( p_k = \frac{n_k}{n} \), where \( N = \sum_k n_k \)

- Learning of global model
  - Gradient averaging
    \[ \theta^{t+1} = \theta^t - \eta t \sum_{k=1}^{n} p_k \nabla L_k(\theta^t) \]
  - Federated averaging
    \[ \theta^{t+1} = \sum_{k=1}^{n} p_k \theta_k \]

What could go wrong?

**TRAINING**

Local updates → Global model → Averaging

- Local training
  - Local data
  - \((x_{j1}, y_{j1})\)
  - \((x_{jn}, y_{jn})\)

**INFEERENCE**

- Input \(x\)
- Label \(y\)
## Taxonomy of threat models

- Attack surface and attack vector

<table>
<thead>
<tr>
<th>Data</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training time</td>
<td>Data poisoning</td>
</tr>
<tr>
<td></td>
<td>Backdoor</td>
</tr>
<tr>
<td>Inference time</td>
<td>Evasion (adversarial examples)</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Information availability

Black box

White box
Privacy Attacks: Threat Models

- Membership inference
- Property inference
- Model inversion

- Partial training data
- Auxiliary data
- No data

Privacy attacks in brief

- **Membership inference attack**
  - *Was this data record in or out of training dataset?*

- **Property inference attack**
  - *Is this property present or absent in the training dataset?*

- **Class-label distribution inference**
  - *What is the proportion of training data with label c?*

- **Model Inversion attack**
  - *Training data reconstructed using model predictions*

- **Model extraction**
  - *Model parameters or hyper-parameters extracted (reverse engineering)*
Membership Inference Attacks

• "Given a machine learning model and a record, determine whether this record was used as part of the model's training dataset or not."

• Method: Shadow training: Auxiliary data

Mitigation of Membership Inference Attacks

- Differentially-private training is by construction immune to membership inference attacks.
- Restrict prediction $F_\theta(x)$ to top $k$-classes.
- Round-off the prediction vector
- Increase entropy of prediction vector temperature $t$: $\frac{\exp\left(\frac{z_i}{t}\right)}{\sum_j \exp\left(\frac{z_j}{t}\right)}$
- Regularization $L_2: \lambda \|\theta\|_2$

<table>
<thead>
<tr>
<th>Hospital dataset</th>
<th>Testing Accuracy</th>
<th>Attack Total Accuracy</th>
<th>Attack Precision</th>
<th>Attack Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Mitigation</td>
<td>0.55</td>
<td>0.83</td>
<td>0.77</td>
<td>0.95</td>
</tr>
<tr>
<td>Top $k = 3$</td>
<td>0.55</td>
<td>0.83</td>
<td>0.77</td>
<td>0.95</td>
</tr>
<tr>
<td>Top $k = 1$</td>
<td>0.55</td>
<td>0.82</td>
<td>0.76</td>
<td>0.95</td>
</tr>
<tr>
<td>Top $k = 1$ label</td>
<td>0.55</td>
<td>0.73</td>
<td>0.67</td>
<td>0.93</td>
</tr>
<tr>
<td>Rounding $d = 3$</td>
<td>0.55</td>
<td>0.83</td>
<td>0.77</td>
<td>0.95</td>
</tr>
<tr>
<td>Rounding $d = 1$</td>
<td>0.55</td>
<td>0.81</td>
<td>0.75</td>
<td>0.96</td>
</tr>
<tr>
<td>Temperature $t = 5$</td>
<td>0.55</td>
<td>0.79</td>
<td>0.77</td>
<td>0.83</td>
</tr>
<tr>
<td>Temperature $t = 20$</td>
<td>0.55</td>
<td>0.76</td>
<td>0.76</td>
<td>0.76</td>
</tr>
<tr>
<td>L2 $\lambda = 1e - 4$</td>
<td>0.56</td>
<td>0.80</td>
<td>0.74</td>
<td>0.92</td>
</tr>
<tr>
<td>L2 $\lambda = 5e - 4$</td>
<td>0.57</td>
<td>0.73</td>
<td>0.69</td>
<td>0.86</td>
</tr>
<tr>
<td>L2 $\lambda = 1e - 3$</td>
<td>0.56</td>
<td>0.66</td>
<td>0.64</td>
<td>0.73</td>
</tr>
<tr>
<td>L2 $\lambda = 5e - 3$</td>
<td>0.35</td>
<td>0.52</td>
<td>0.52</td>
<td>0.53</td>
</tr>
</tbody>
</table>

Property Inference Attacks (PIA)

- Machine Learning (ML) models unintentionally memorize properties of training data
- Extract global statistics about training data via access to trained model (black v/s white-box)
- Usually a binary classifier problem—presence/absence of a certain property.
- Constitute a privacy risk in many healthcare and industrial applications
- Online learning: when does a certain property appear

Class Label Distribution Inference
PIA on ML classifiers with C output classes: infer the class label distribution (categorical) of training data

Mitigation of Generic Property Inference Attacks

- Add noise to training data: flip labels, introduce adversarial examples
- Add noise to classifier output
- Encode arbitrary information while not compromising on model generalizability and performance.
- Although encoding or memorizing information is also an attack, it will bypass meta-classifiers learnt using shadow-training.

Class-label distribution \( p \in \Delta^{C-1} \) given labeled training data \( D \triangleq \{x_i, e_i\}_{i=1,2,3,...,N} \)

\[
p = \frac{1}{N} \sum_i e_i \triangleq \frac{N_c}{\sum_c N_c}
\]

A new type of PIA we introduced

- Training-time: Federated learning
- Inference: Meta-classifier

**New Attack!**

Federated Learning (FL)- distributed machine learning: server-client model

Training data: local and private to client

Server: unaware of potential class-imbalance:
  - Class-imbalance deteriorates accuracy-detection and mitigation important
  - Composition of client's data- privacy risk in many healthcare and industrial applications
Motivation: TECoSA partner

- Training data from clients: labeled as **anomalous**/non-anomalous (**fatigued**/non-fatigued)
- Each client wants to keep their training data and labels private
- **Goal**: To learn a classifier (supervised ML) to decide if there is fatigue or not. **Server**: Company.
- Can the server infer the fraction of training data labels that are anomalous or not using model parameter updates?
- Knowledge of this fraction:
  - Could provide competitive advantage
  - Idea about client’s profitability
Goals

- Develop methods for class-label distribution inference when parameter updates at every round \( t \) are available
- Identify conditions for exact inference
- Develop methods for non-exact inference (estimators)
Related Work

Class-label distribution inference studied as class-imbalance mitigation and as attack:

- To address class-imbalance in FL:
  - change loss-function
  - cluster clients
- As a property inference attack: preference profiling attacks (PPA)
- Gradients from last layer to extract label-proportion information
- Gradients to reconstruct training data

Class-Label Distribution Exact Inference

- Exact inference for client $k$ at global iteration $t$ possible
- Given: bias at server $b^t$ and updated at client $k$ $b_k^{t+1}$
- Conditions for exact inference:
  - Learning rate $\eta$
  - Data size $N$
  - Full-batch gradient descent
  - Single epoch update by the client
  - Weight matrix set to zero by server at iteration $t$
- Conditions not met: approximations used: 4 estimators for non-exact inference
- Use Auxiliary dataset containing $N_{aux}$ samples from each class

Generic NN classifier

Bias from last fully-connected layer used
Known parameters:
Learning rate $\eta$, training data size $N$, epochs $E$

Collect global parameter $W_t, b_t$ and update for client $k$ $W_{t+1}^k, b_{t+1}^k$

Is $W_t = 0$, $E = 1$, batch size=N? (yes/no)

- **Exact Inference**
- **Non-Exact Inference methods**

**Auxiliary data**: $N_{aux}$ samples from all $C$ classes

**Construct matrices**: $\Sigma_{aux}, B_{aux}, W_{aux}$

<table>
<thead>
<tr>
<th>Initialized bias</th>
<th>Soft label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approximation via bias value</td>
<td>Approximate the softmax output from the client</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Aux-grad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approximate the gradient wrt bias</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Aux-weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approximate gradient wrt weight matrix of last layer</td>
</tr>
</tbody>
</table>
Numerical Results

- Comparison with state of the art: Wang et.al
- UCI Census Income Dataset-binary classification

Numerical Results

CIFAR-10: 10 class image classification

Random oversampling as Attack Mitigation

- Can we mitigate class-label distribution inference attack?
- Random oversampling: sample with replacement for minority classes
- Makes class distribution ‘balanced’ (uniform distribution)
- Proposed methods fail to estimate which implies effective countermeasure

Imbalanced dataset

Imbalanced datasets balanced via random oversampling
Addressing class-imbalance in FL

- Class-imbalance in FL: slow convergence, low accuracy
- Problem mitigated by grouping clients based on class-label distribution (known)
- Remove clients with estimated class-imbalance
- US Census Income dataset:
  \[ p \in [0,0.2] \cup [0.8,1] \]
- Client removal: improved accuracy (AUC: area under the ROC curve) and faster convergence.

Area under the ROC curve (AUC) at iteration \( t \) for cases with and without client removal. US Census Income dataset

Class-Label Distribution Inference: Trained ML Models

- Fully-trained models, attack at inference time
- ML model parameters (after training) are available
- Target classifier architecture: fully connected neural networks
- Shadow training methodology: Meta-Classifier
- Challenge: multi-dimensional sampling for multi-class classifiers

Accuracy Augmented Meta-Classifier Attack

- Generate shadow training data sets to train shadow classifiers
- Meta-classifier architecture: permutation invariant
- Use parameters and accuracy to train meta-classifier
Numerical Results

UCI Census Income Classification (≥ 50k)

- Binary Classification
- Accurate estimates for most values of p

- Performance of Meta-Classifier (KL Divergence)
- Shows improvement over baseline:
  - Architectural changes
  - Accuracy augmentation

Robustness to random oversampling

- The imbalance of class labels is addressed by random oversampling of the minority class
- Makes class label distribution `balanced' (discrete uniform)
- Meta-classifier can still estimate original distribution!
- Further training the meta-classifier on oversampled shadow-training datasets improves performance
Summary and Conclusions

• Privacy Attacks:
  – Membership inference
  – Property inference

• Class-label distribution inference:
  – In FL via model updates
  – In trained ML models via meta-classifiers

• Random oversampling as countermeasure:
  – In FL works as a mitigation measure
  – Meta-classifiers seem robust to it
Ongoing and Future work

• Test meta-classifier based attacks in the FL setting

• Efficient online and adaptive methods of shadow training dataset sampling for higher dimensions (multi-class classifiers)

• Mitigation scheme against meta-classifier-based property inference attacks

• Meta-classifier attacks for other target classifier architectures
Thank you!