A Repository of Jupyter Notebooks on Unlearning in Federated Learning

First Presentation

FYP22002 @ HKUCS

Sheng “Victor” HUANG

Supervisor: Prof. S M YIU

18 Jan 2023
Introduction

• A Story
  • Alice lent her laptop to Bob yesterday
  • Bob googled “heart disease”
  • Bob wiped his search history and returned the laptop to Alice today
  • Alice finds heart disease medicine on various sites that use Google Ads
  • Now Alice thinks Bob might have a heart disease
  • Google thinks Alice might have a heart disease

• Why?
  • Search history used to train ML models in Ad recommendation system
  • Wiping search history ⇔ wiping data influence in ML models

• Need to erase data from ML models
Introduction

• Privacy – legal requirement
  • Right to be Forgotten (RtbF)

Note 1:
No standalone RtbF in HKSAR, as there is no such right mentioned in the PDPO. Data erasure is pursuant to Data Protection Principle 2 and section 26 of the PDPO.
(2020 Administrative Appeals Board Decision -> Appeal No. 15/2019)

Note 2:
No RtbF mentioned in PRC’s Personal Information Protection Law (PIPL).
Introduction

• Not just privacy
  • Security: data leakage, adversarial attacks (model stealing, membership inference, poisoning), ...
  • Usability: outdated/incorrect data in ML systems
  • Fidelity: human bias (biased data on gender, age, race, ...)

Machine Unlearning [CY15]

• Remove data (and its influence) from ML models
• ML models as black boxes [KL17]
• Naïve method -> to retrain on remaining data -> slow

• Challenges:
  • Stochasticity of training
    • Data influence difficult to track
  • Incrementality of training
    • Training with a data point affected by prior training, and will affect later training
  • Catastrophic unlearning
    • Unlearnt models generally perform worse than retrained models

[KL17] https://proceedings.mlr.press/v70/koh17a
Unlearning Framework

Current Data → Current model

Learning Algorithm
- SGD
- Regression
- Decision Trees
- ...

Removal request
- Item
- Feature
- Class
- ...

Unlearning Algorithm
- Model-agnostic
- Model-intrinsic
- Data-driven
- ...

Requirements
- Completeness
- Timeliness
- Guarantee
- ...

Evaluation Metrics
- Accuracy
- ZRF score
- Anamnesis Index
- ...

Verification
- Feature Injection Test
- Membership
- Inference Attack
- Forgetting Measuring
- ...

Retrain

Not satisfied?

Untraining Framework

- Unlearning requests
  - Item removal
  - Feature removal
  - Class removal
  - Task removal
  - Stream removal
  - ...
- Design requirements
  - Completeness (consistency)
  - Timeliness
  - Accuracy
  - Light-weight
  - Provable guarantees
- Model-agnostic
- Verifiability
- ...

- Unlearning verification
  - Membership inference attacks
  - Backdoor attacks
  - Slow-down attack
  - Feature injection
  - Forgetting measure
  - Information leakage
  - Cryptographic protocol
  - ...

Unlearning Methods

• Exact unlearning
  • Only guaranteed with retraining or variations of retraining
  • Slow, especially with complex models
  • Variations usually take snapshots during training [Bou+21, Ngu+22]

• Approximate unlearning
  • Fisher unlearning [Mar20]
    • Uses the remaining data, takes a corrective Newton step and injects optimal noise
  • Influence unlearning
    • Computes the influence of the target data and remove it from model
  • Gradient unlearning
    • Approximates SGD steps as if retraining was done, using remaining data

Federated Learning (FL) [McM+17]

- Central server $S$ initializes a model $M_0$;
- For training in iteration $i$, Central server $S$ sends $M_i$ to a group of $K$ decentralized clients $C = \{C_1, C_2, ..., C_K\}$;
- Each client $C_k \in C$ trains the model with its own data, generating an update $U^i_k$;
- Clients send updates $\{U^i_1, U^i_2, ..., U^i_K\}$ to $S$;
- $S$ averages all the updates $\{U^i_1, U^i_2, ..., U^i_K\}$ to $U^i$ and updates the model to $M_{i+1} = M_i + U^i$;
- When the termination criteria are met, $S$ stop training and output the model $M_{i+1}$ as $M$;
- If not terminated, $S$ sends $M_{i+1}$ to $C$ for iteration $i + 1$.

Federated Learning (FL) [McM+17]

• FedAvg – most important algorithm in FL

```
Algorithm 1 FederatedAveraging. The $K$ clients are indexed by $k$; $B$ is the local minibatch size, $E$ is the number of local epochs, and $\eta$ is the learning rate.

Server executes:
- initialize $w_0$
- for each round $t = 1, 2, \ldots$ do
  - $m \leftarrow \max(C \cdot K, 1)$
  - $S_t \leftarrow$ (random set of $m$ clients)
  - for each client $k \in S_t$ in parallel do
    - $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
    - $w_{t+1} \leftarrow \sum_{k=1}^K \frac{n_k}{n} w_{t+1}^k$

ClientUpdate($k, w$):  // Run on client $k$
- $B \leftarrow$ (split $\mathcal{P}_k$ into batches of size $B$)
- for each local epoch $i$ from 1 to $E$ do
  - for batch $b \in B$ do
    - $w \leftarrow w - \eta \nabla \ell(w; b)$
- return $w$ to server
```

Federated Unlearning

Additional Challenges

• Limited data access
  • Server doesn’t have access to training data used at the client side
  • This makes unlearning methods for centralized ML difficult to implement for FL

• Limited client participation
  • Client goes offline and can’t participate in unlearning process

• Complicated relationship between data and clients
  • Unlearning all data of one client of just part of it?
  • What if there is data overlap on other clients?

• Data partition
  • Horizontal FL, vertical FL and federated transfer learning

• Statistical heterogeneity
  • IID (Independent and Identically Distributed)

• Adversarial model
  • Untruthful server/clients

• …

Remember challenges for centralized unlearning:

• Stochasticity of training
  • Data influence difficult to track

• Incrementality of training
  • Training with a data point affected by prior training, and will affect later training

• Catastrophic unlearning
  • Unlearnt models generally perform worse than retrained models
FedEraser [Liu+21a]

- First attempt
- For each remaining client
  - Calibration (local) training with its own data
  - Sends the new update to the server
- For the server
  - Calibrate saved updates
  - Aggregate the calibrated updates
  - Update the model

FedEraser [Liu+21a]

How much should the model change?

In what direction should the model change?

\[
\tilde{U}_{t_j}^{k_c} = \frac{U_{t_j}^{k_c}}{\hat{U}_{t_j}^{k_c}}
\]

FedEraser [Liu+21a]

Advantages
• Minor change to FL architecture
• Much faster than retraining

Limitations
• Require client participation
• Doesn’t work well with complex models
• Calibration’s effectiveness under-explored


Membership inference attacks show similar data erasure performance as retraining.

While performance doesn’t drop too much.
Federated Unlearning with Knowledge Distillation [WZM22]

\[ M_F = M_1 + \sum_{t=1}^{F-1} \Delta M_t \]

\[ \Delta M_t = \frac{1}{N} \sum_{i=1}^{N} \Delta M^i_t = \frac{1}{N} \sum_{i=1}^{N-1} \Delta M^i_t + \frac{1}{N} \Delta M^N_t \]

\[ \Delta M'_t = \frac{1}{N-1} \sum_{i=1}^{N-1} \Delta M^i_t = \frac{N}{N-1} \Delta M_t - \frac{1}{N-1} \Delta M^N_t \]

\[ M'_F = M_1 + \frac{N}{N-1} \sum_{t=1}^{F'-1} \Delta M_t - \frac{1}{N-1} \sum_{t=1}^{F'-1} \Delta M^N_t + \sum_{t=1}^{F'-1} c_t \]

\[ \Delta M'_t = \frac{1}{N} \sum_{i=1}^{N-1} \Delta M^i_t = \Delta M_t - \frac{1}{N} \Delta M^N_t \]

Assume client \( N \) participated but generated update is 0

Federated Unlearning with Knowledge Distillation [WZM22]

\[
M'_F = M_1 + \sum_{t=1}^{F-1} \Delta M'_t + \sum_{t=1}^{F-1} \epsilon_t
\]
\[
= M_1 + \sum_{t=1}^{F-1} \Delta M_t - \frac{1}{N} \sum_{t=1}^{F-1} \Delta M_t^N + \sum_{t=1}^{F-1} \epsilon_t
\]
\[
= M_F - \frac{1}{N} \sum_{t=1}^{F-1} \Delta M_t^N + \sum_{t=1}^{F-1} \epsilon_t
\]

Algorithm 1 Federated Unlearning

**Input:** Global model \( M_F \), Total number of clients \( N \)

**Input:** Historical updates \( \Delta M_t^A \) of target client \( A \) at round \( t \)

**Input:** Outsourced unlabelled dataset \( X \)

**Parameter:** Distillation epoch \( k \), Temperature \( T \)

**Output:** The unlearning model \( M'_F \)

1: \( M'_F \leftarrow M_F - \frac{1}{N} \sum_{t=1}^{F-1} \Delta M_t^A \)
2: **for** \( \text{epoch} = 1, 2, \ldots, k \) **do**
3: \( y_{\text{teacher}} \leftarrow M_F(X), T \)
4: \( y_{\text{student}} \leftarrow M'_F(X), T \)
5: Calculate loss\(_{\text{distillation}}\) of \( y_{\text{teacher}} \) and \( y_{\text{student}} \)
6: **end** **for**
7: **return** unlearning model \( M'_F \)

No good method to calculate \( \epsilon_t \). Use distillation to estimate.

Federated Unlearning with Knowledge Distillation [WZM22]

Knowledge Distillation [HVD15]

• Use the prediction results of class probabilities produced by a teacher model to train a student model

• Knowledge acquired by model is not only encoded in weight parameters but also the class probability prediction output.

Federated Unlearning with Knowledge Distillation [WZM22]

Advantages
• No client participation needed
• Works well with complex models such as DNNs

Limitations
• Process is costly and approximate
• Original model may still be at client side
• Verification proposed in paper (backdoor attack) is not ideal in real world situations (intentionally introduces a backdoor)

Other Methods

• [Liu+21b] designed a novel framework of revocable (vertical) federated random forest (RevFRF). RevFRF uses a suite of homomorphic encryption (HE) based secure protocols to implement RF, enabling model construction, prediction and participant revocation, even with untruthful server and clients.

• [GSK21] studied federated learning and unlearning in a decentralized network within a Bayesian framework. It developed federated variational inference (VI) solutions based on the decentralized solution of local free energy minimization problems within exponential-family models and on local gossip-driven communication.

Other Methods

• [Wan+22] proposed a method for scrubbing information from CNNs in FL when an entire class need to be forgotten (class-level unlearning), using Term Frequency Inverse Document Frequency (TF-IDF) to quantize the class discrimination of channels.

• [Liu+22] proposed a smart retraining method for federated unlearning without communication protocols. The approach uses the L-BFGS algorithm to efficient solve a Hessian approximation with historical parameter updates for global model retraining.

Comparison

• [Liu+21a] and [WZM22] have more general methods that require little change to the existing FL framework. [Liu+21a] doesn’t work well with complex models, while [WZM22] don’t need client participation (both an advantage and a limitation).

• [Liu+21b], [GSK21] and [Wan+22] all designed unlearning methods that are very specific to unlearning requests, models, FL data partition, learning algorithms, etc.

• [Liu+22] proposed a rapid retraining method. However, it stores snapshots, which can bring burden to storage and creates privacy risk. And it also don’t work well with complex models.

<table>
<thead>
<tr>
<th>Paper</th>
<th>Date of Publication / Conference / Preprint Last Edit</th>
<th>Venue</th>
</tr>
</thead>
<tbody>
<tr>
<td>[WZM22]</td>
<td>24 Jan 2022</td>
<td>arXiv</td>
</tr>
<tr>
<td>[Gon+22]</td>
<td>22-23 May 2022</td>
<td>2022 IEEE Data Science and Learning Workshop (DSLW)</td>
</tr>
<tr>
<td>[Hal+22]</td>
<td>9 Sep 2022</td>
<td>arXiv</td>
</tr>
<tr>
<td>[Fra+22]</td>
<td>21 Nov 2022</td>
<td>arXiv</td>
</tr>
<tr>
<td>[Wu+22]</td>
<td>25 Nov 2022</td>
<td>arXiv</td>
</tr>
<tr>
<td>[Yua+22]</td>
<td>03 Dec 2022</td>
<td>arXiv</td>
</tr>
</tbody>
</table>
Some Takeaways

• FL has many variations. It’s difficult to find an unlearning method that both performs well and fits that many variations.

• FL is still in its early stages and not a practical privacy-enhancing technology (PET) yet [Boe+23], so there will be a lot of federated learning and unlearning research work done in the future.

• Some trends and open questions on unlearning: defining success of unlearning (verification / data auditing); unified unlearning requirements; unified unlearning benchmarking; adversarial machine unlearning; interpretable machine unlearning; …

Thank you!

• What have you learnt about federated unlearning?
  • I forgot... Maybe ask my clients.