Seguridad, privacidad y aspectos legales

Técnicas de descentralización de datos: Aprendizaje Federado

Judith Sáinz-Pardo Díaz

Grupo de Computación Avanzada y e-Ciencia
Instituto de Física de Cantabria (IFCA) - CSIC-UC

Máster universitario en ciencia de datos / Master in Data Science

Distributed Machine Learning

Distributed Machine Learning

OBJETIVO: manejar grandes conjuntos de datos y desarrollar algoritmos eficientes y escalables.

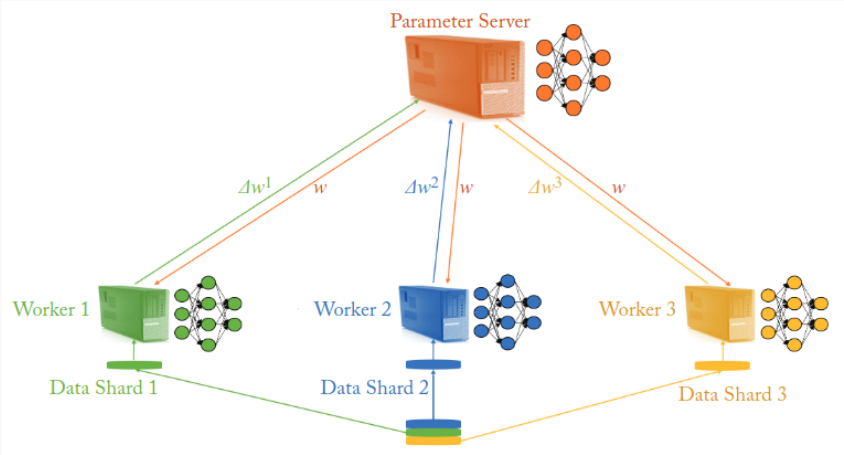
Definición

El DML o aprendizaje automático distribuido, se refiere a los algoritmos y sistemas de Machine Learning o de Deep Learning diseñados para mejorar el rendimiento, preservar la privacidad y escalar a más datos de entrenamiento y modelos más grandes.

Idea

Los datos de entrenamiento se dividen en fragmentos de datos disjuntos y se envían a disintos workers que realizan el entrenamiento localmente. Estos envían al servidor central los pesos o grandientes correspondientes a su entrenamiento, donde son agregados.

DML - Esquema



Esquema DML. Fuente: Yang, Qiang, et al. "Federated learning."Synthesis Lectures on Artificial Intelligence and Machine Learning 13.3 (2019): 1-207.Fuente: Yang, Qiang, et al. "Federated learning."Synthesis Lectures on Artificial Intelligence and Machine Learning 13.3 (2019): 1-207.

Federated Learning

Ejemplo: teclado predictivo

OBJETIVO: usar los datos de distintos terminales para entrenar modelos que predigan la próxima palabra.

- **IDEA 1:** se recopilan los datos de todos los terminales posibles y se entrena un modelo de ML en la nube. Se devuelve a cada dispositivo las predicciones. En este caso los datos tienen que salir del terminal. $\longrightarrow$ **ENFOQUE CENTRALIZADO**
- **IDEA 2:** cada dispositivo entrena en local un modelo con sus propios datos y hace las predicciones. Los datos no salen del terminal. $\longrightarrow$ **EDGE COMPUTING**

Enfoque clásico (centralizado)

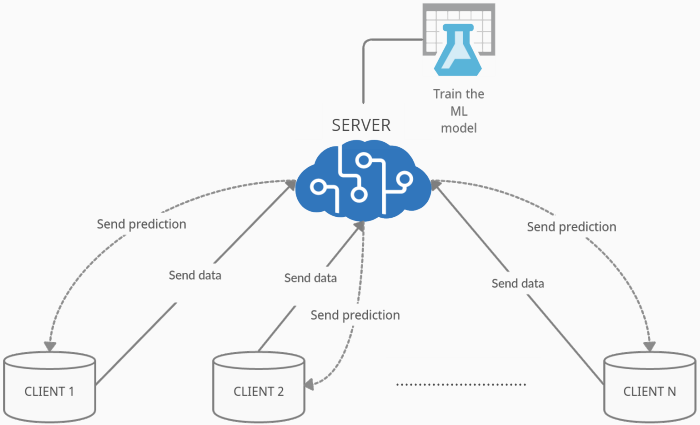


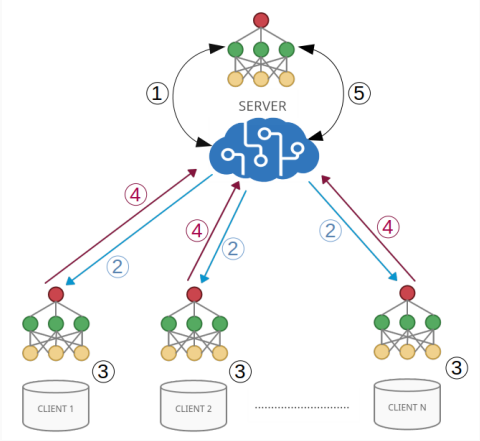
Figura: Enfoque centralizado.

Federated Learning - Idea

- **Descentralización** de datos.
- Los datos no salen del dispositivo/centro que los genera.
- Estructura **SERVIDOR-CLIENTE**. El servidor crea el modelo que entrenará cada cliente en local con sus propios datos.
- Los clientes sólo envían al servidor los parámetros obtenidos tras entrenar el modelo.
- El servidor agrega los pesos obtenidos para el modelo por cada cliente y actualiza el modelo inicial.
- Puede verse como un tipo especial de DML.
- Hay que identificar los potenciales clientes, y ver que tengan datos suficientes y de calidad.
- **Federated Learning: enfoque colaborativo y descentralizado de Machine Learning**.

Federated Learning - Esquema

1. **SERVIDOR**: crea el modelo que cada cliente entrenará en local.
2. **SERVIDOR**: transmite el modelo a cada cliente.
3. **CLIENTE**: cada uno entrena el modelo en local con sus datos.
4. **CLIENTE**: envía al servidor los pesos del modelo tras entrenarlo.
5. **SERVIDOR**: agrega los pesos de cada cliente y actualiza el modelo.
6. Repetir desde el paso 2.



Función de agregación

- Función que combina los pesos obtenidos tras entrenar cada cliente el modelo inicial.
- Conviene que tenga en cuenta el número de datos de cada cliente.
- *Ejemplo: media ponderada*. Sea N el número de clientes, n_i el número de datos del cliente i $\forall i \in \{1, \dots, N\}$. Sean x_i los pesos del modelo obtenidos tras entrenar con el cliente i $\forall i \in \{1, \dots, N\}$. Podemos agregar los pesos como sigue:

$$\sum_{i=1}^{N} w_i x_i, \text{ con } w_i = \frac{n_i}{\sum_{i=1}^{N} n_i} \quad \forall i \in \{1, \dots, N\}$$

Cross-device FL y Cross-silo FL

Cross-device FL

En este enfoque de Federated Learning, los clientes son un gran número de dispositivos que almacenan información sensible de distintas personas o entidades. *Ejemplo: teclados predictivos.*

Cross-silo FL

En este enfoque de Federated Learning, los clientes no son dispositivos, si no, por ejemplo, hospitales, bancos, universidades, instituciones gubernamentales, etc. De la misma forma, estas instituciones no quieren/pueden compartir sus datos entre ellos ni con un servidor central, luego se aplica FL. *Ejemplo: imagen médica.*

Ventajas respecto al enfoque clásico (centralizado)

- Se asegura la **seguridad** y la **privacidad**: los datos no "viajan".
- Reducción de los **costes de comunicación**:
  - Se transfieren los pesos en lugar de los datos.
  - Comprimir la matriz de números que definen un modelo $\longrightarrow$ ahorra ancho de banda
- Mayor **ahorro de energía**.
- Menor **coste computacional**.
- Menor **latencia**.

OJO: hay que tener en cuenta que los clientes pueden ser intermitentes (desaparecer algunos y entrar otros nuevos).

Posibles ataques

Ataques de reconstrucción (reconstruction attacks)

El adversario trata de reconstruir los datos originales interceptando información intermedia. En FL esta información intermedia podrían ser los pesos del modelo tras cada actualización. Puede interesar aplicar técnicas de seguridad para mantener esta información privada, como HE.

Ataques de inversión del modelo (model inversion attacks)

El objetivo del adversario es extraer alguna característica de los datos de entrenamiento del modelo. Sea M un modelo de ML que tiene como entrada un vector $(x_1, \dots, x_n)$ de un espacio n-dimensional (p.e. de $\mathbb{R}^n$), y sea y la salida obtenida al predecir: $y = M(x_1, \dots, x_n)$. El atacante accede al modelo M para extraer alguna característica x_i, $i \in \{1, \dots, n\}$, dado algún conocimiento previo del resto de características y de y.

Posibles ataques

Ataques de inferencia de miembros (membership inference attacks)

En este caso, dado un modelo de ML, el atacante quiere inferir si un cierto individuo a participado o no en el entrenamiento. Para tratar de prevenir este ataque suelen usarse algoritmos de Privacidad Diferencial (DP).

Ataques de envenenamiento de modelos (model poisoning attacks)

La meta del atacante es afectar el aprendizaje del modelo durante el entrenamiento. En FL, puede ocurrir cuando un cliente introduce un etiquetado erróneo de los datos para dañar la precisión global del modelo. La manera más simple y efectiva de prevenir este ataque es permitir que participen sólo clientes para los que se ha pre-establecido un cierto nivel de confianza.

Librerías para FL

- **TensorFlow Federated.**
- **PySyft.**
- **IBM Federated Learning.**
- **FedML.**
- **Flower.**
- **FATE.**
- **FedJAX.**
- **Sherpa.ai FL.**

Otras técnicas de descentralización de datos

Split Learning

La idea más simple consiste en que cada cliente entrena una red neuronal hasta una capa específica (capa de corte), y la salida de esta se envía al servidor que completa el resto del entrenamiento sin ver los datos en bruto.

Gossip Learning

Variación de FL en la que no se necesita un servidor central. En su lugar, los clientes comparten directamente las actualizaciones de sus modelos y la agregación tiene lugar de forma distribuida.

Práctica Federated Learning

1. Carga el dataset MNIST usando keras.
2. Divide el conjunto de train en 3 clientes (no importa como los crees mientras sean disjuntos).
3. Crea un modelo basado en redes convolucionales para MNIST.
4. Entrena el modelo de manera individual para cada uno de los 3 clientes, y guarda los pesos obtenidos en cada caso.
5. Agrega los pesos obtenidos con los 3 clientes (p.e. usa la media ponderada).
6. Lo pesos obtenidos al agregar los de los 3 clientes, serán ahora los nuevos pesos del modelo. Guarda dicho modelo.
7. Repite otras 2 veces los pasos 4-6.
8. Usa los 3 modelos que has guardado para entrenar sobre el conjunto de test. Compara los resultados.

Algunos recursos interesantes

- Rodríguez-Barroso, Nuria, et al. "Federated Learning and Differential Privacy: Software tools analysis, the Sherpa. ai FL framework and methodological guidelines for preserving data privacy." Information Fusion 64 (2020): 270-292. https://www.sciencedirect.com/science/article/pii/S1566253520303213.
- Li, Tian, et al. "Federated learning: Challenges, methods, and future directions." IEEE Signal Processing Magazine 37.3 (2020): 50-60. https://ieeexplore.ieee.org/abstract/document/9084352.
- Wahab, Omar Abdel, et al. "Federated machine learning: Survey, multi-level classification, desirable criteria and future directions in communication and networking systems." IEEE Communications Surveys & Tutorials 23.2 (2021): 1342-1397. https://ieeexplore.ieee.org/abstract/document/9352033.
- Yang, Qiang, et al. "Federated learning." Synthesis Lectures on Artificial Intelligence and Machine Learning 13.3 (2019): 1-207.

Algunos recursos interesantes

- Federated Learning: Collaborative Machine Learning without Centralized Training Data. Google AI Blog. https://ai.googleblog.com/2017/04/federated-learning-collaborative.html.
- Federated Learning Open Mined Blog. https://blog.openmined.org/tag/federated-learning.
- Tensorflow Federated (TFF). https://www.tensorflow.org/federated/federated_learning.
- PySyft. https://github.com/OpenMined/PySyft.