3rd IFC Workshop - Federated Learning Data Science in Central Banking

Aggregation strategies for Federated Learning

Erika Cerasti

- Benjamin Santos, Julian Templeton, Rafik Chemli, Saeid Molladavoudi (Statistics Canada)
- **Erika Cerasti**, Francesco Pugliese, Massimo De Cubellis (ISTAT)
- Matjaz Jug (Statistics Netherlands)

17-19 October 2023
Bank of Italy - Rome

International Context

PETs
***privacy-enhancing technologies*:**
Methodologies and approaches
to mitigate privacy risks when using
sensitive or confidential data

NSOs can build greater trust with the public and unlock new opportunities associated with more accurate and complete data collection.

https://unstats.un.org/bigdata/task-teams/privacy/index.cshtml

Are PETs highly reliable and can they be used for accessing sensitive data (health records, tax records or credit card data) by NSOs?

International Context

UNCEBD : UN Committee of Experts on Big Data and Data Science for Official Statistics

Elements to accelerate the adoption of PETs in the NSO community:

- Experimentation (PET Lab)
- Outreach & Training
- Support Services

UN PET Lab

Created to facilitate experimentation on pilot projects and effective collaboration on "real world" use case.

<u>Objectives</u>: Develop principles, policies and open standards for data sharing, taking full account of data privacy, confidentiality and security issues.

Secure Multi-Party Computation

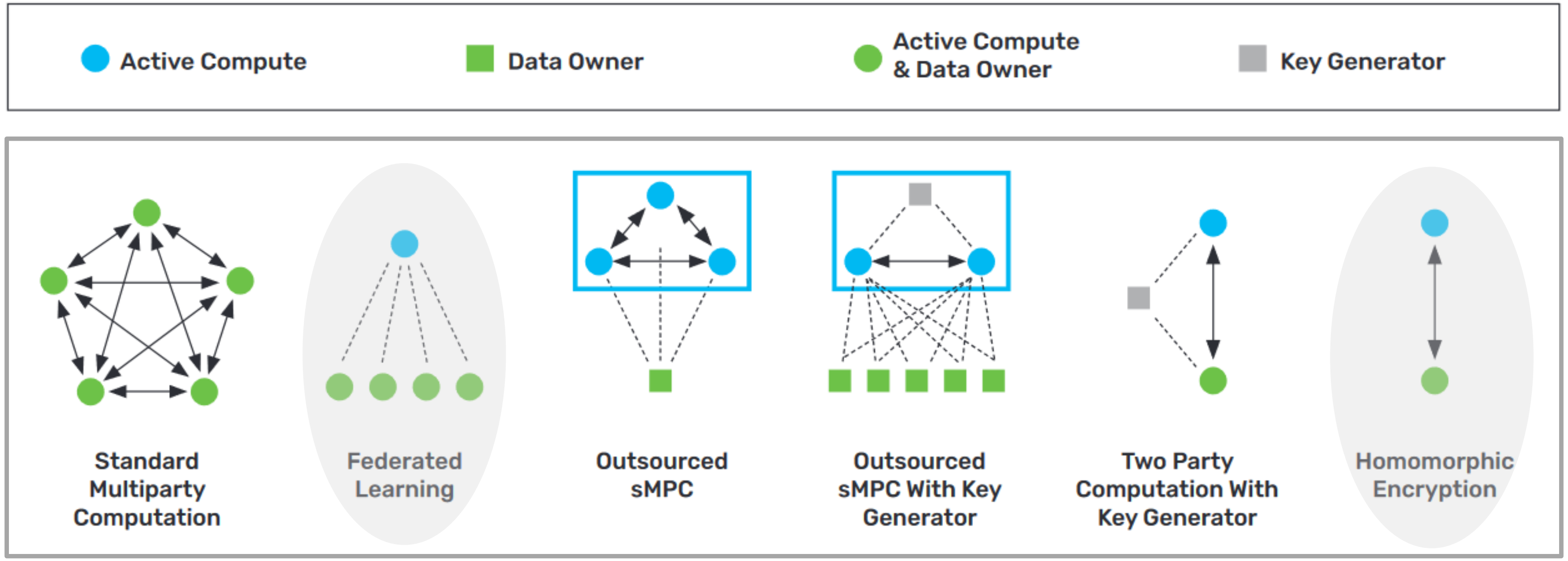
Secure multi-party computation – sMPC

Two or more (mutually distrusting) parties wish to compute an agreed-on function on data that they provide to that computation but are unwilling to disclose to others.

Federated Learning

sMPC protocols use frequent communication among the compute parties.

- available network bandwidth
- network latency between parties



Federated Learning

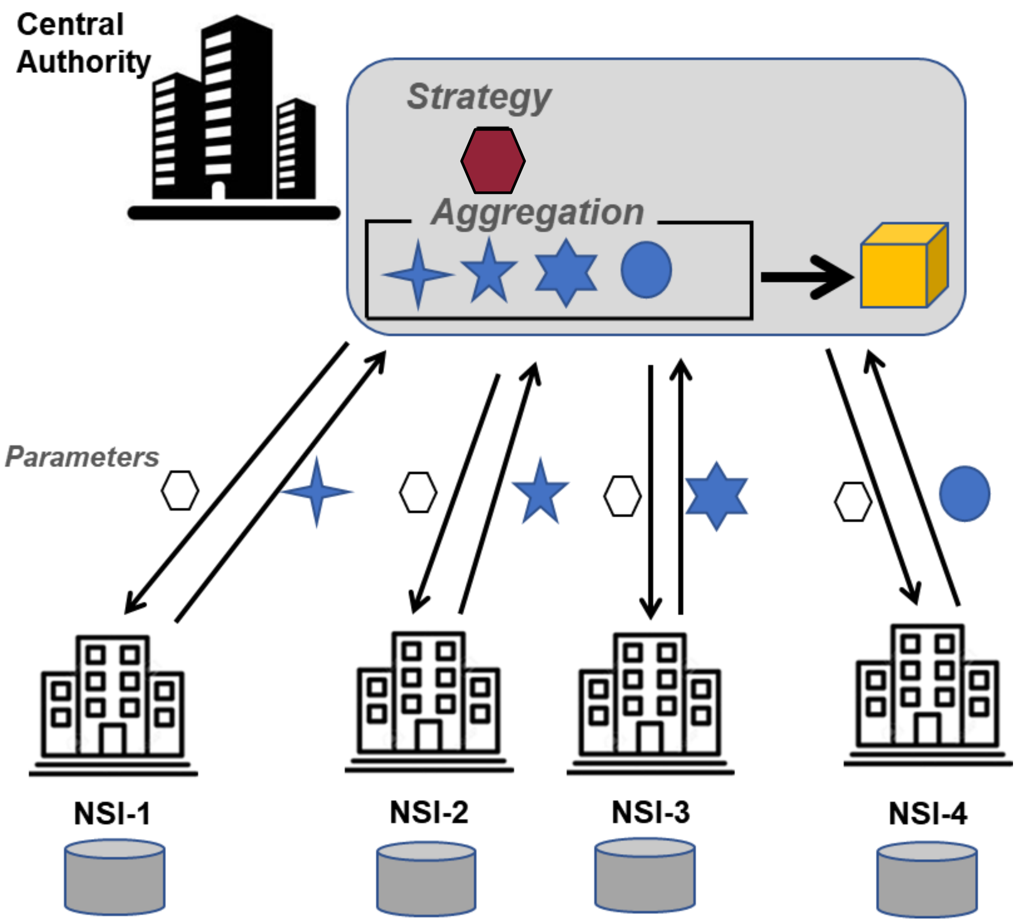
Decentralized / Distributed Computation

FL allows a centralized ML model to be trained on data residing on distributed client devices.
After training the model with data locally, a client will send the weights or gradients back to the server to be aggregated.

This allows analytics to be derived without collecting data.

Aggregation strategy

Performance of the trained models can reach similar performance of centralized approaches but a careful selection of the hyperparameters and the aggregation strategy is important.



Federated Learning and Homomorphic Encryption

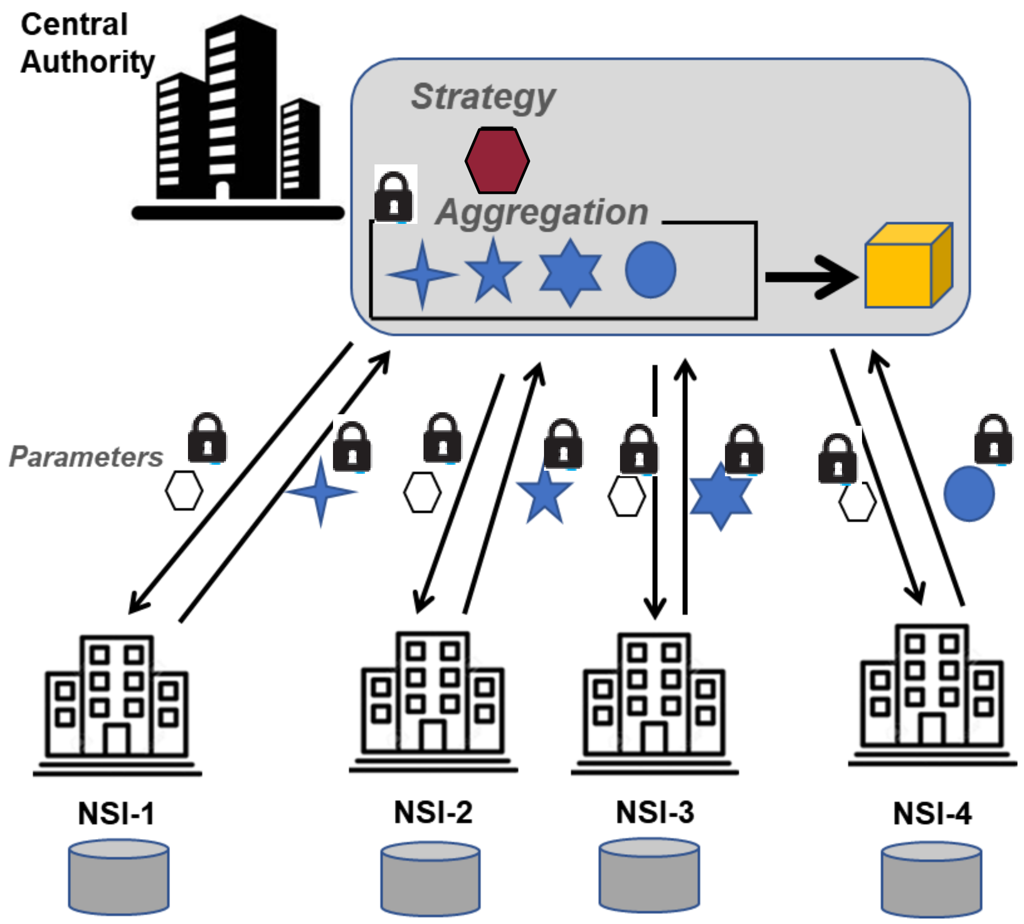
Decentralized / Distributed Computation

Locally trained client models can still be attacked, which can be better protected by using other PETs in conjunction with FL

- Example: Homomorphic Encryption (HE)

HE is a cryptographic technology allowing for the direct computation (addition and multiplication) on encrypted data.

It adds more computational complexity and can result in a high computational overhead.



Aggregation strategies

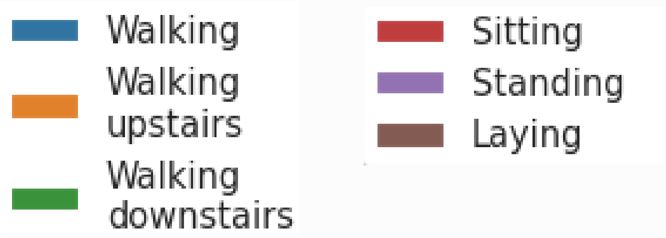
Human Activity Recognition public DATASET

30 volunteers (19-48 years) - 6 class of human activities

Recordings:

Accelerometer and gyroscope data collected by smartphones.
(3-axial linear acceleration and 3-axial angular velocity, rate of 50Hz)

<u>Human activity classes</u>



D. Anguita et al ,ESANN 2013

Work objectives

- Explore different federated aggregation strategies
- Explore the role of dataset **heterogeneity**
- Apply HE to model weights

<u>Federated aggregation Strategies</u>

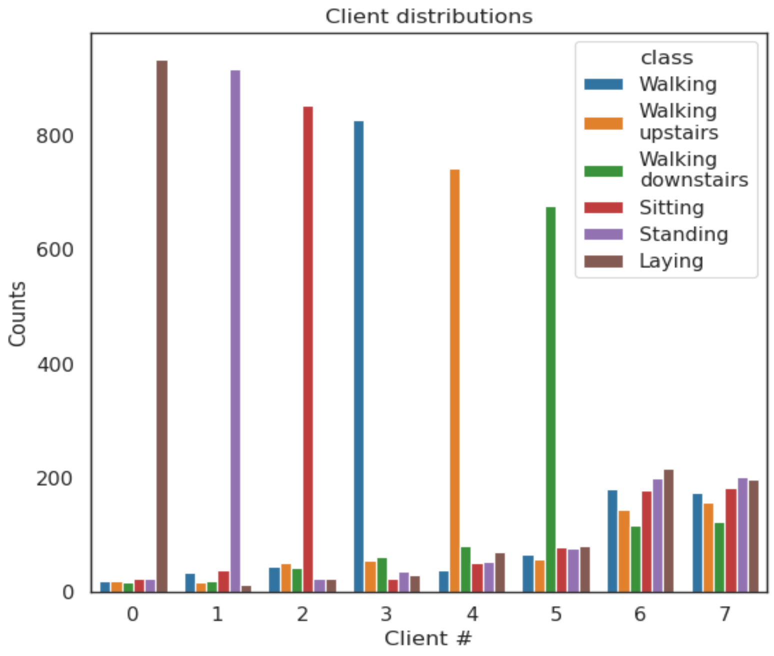
- FedAvg
- **Weighted FedAvg (WFedAvg)**
- FedAdagrad
- FedAdam
- FedYogi

Data Splitting Methodologies

Splitting methods - heterogeneous datasets – 8 clients

- ***Random***: Samples are randomly distributed among clients
- ***Majority even:*** Each client has one majority class, same number of records
- ***Majority:*** Each client has one majority class, different number of records
- ***Pick two***: Each client has two majority classes, same number of records

Note that each class can only be assigned as a majority class once, where remaining clients without a majority class are given a distribution of all classes.



Weighted Federated Averaging Strategy

- Index i runs on clients, i.e, 1 to 8 clients
- Index j runs on classes, i.e, 1 to 6 classes
- Properties on i are related to clients: weights w_i, Gini coefficient G_i, entropy H_i
- Properties on j are related to classes: probabilities for picking particular class j for client i.

A vector of probabilities p_i is local for a client i: $\vec{p}_i = p_j^i \to \{p_1^i, \ldots, p_6^i\}$ with $p_j^i = \frac{\text{count class } j \text{ client } i}{N_i}$ with N_i the total number of samples for client i. p_j^i is normalized.

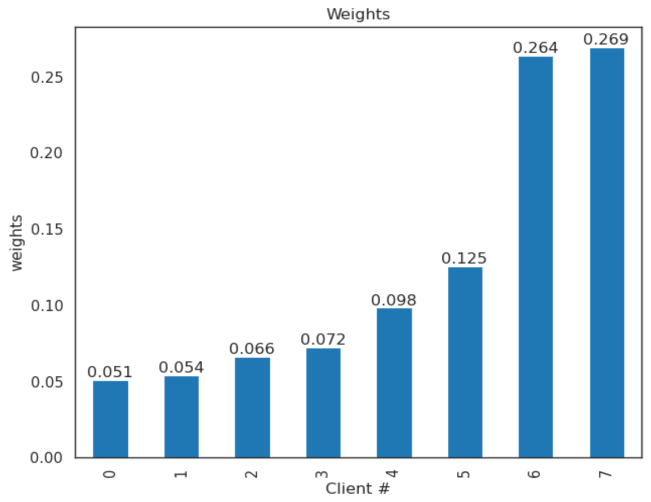
The vector P_j is global (common to all clients) $\vec{P} = P_j \to \{P_1, \ldots, P_6\}$ with $P_j = \frac{\text{count class j for all clients}}{N}$ with $N = \sum_i N_i$ the total number of samples for all clients. P_j is normalized.

A set of weights could be constructed by multiplying a vector of a local property times a set of local vectors for each client. For instance, let $\tilde{P}$ be the normalized product of the probabilities of picking a class j in the joint dataset times the square probability for a client i to have this class j.

$$\tilde{P}_i = \frac{\sum_j P_j \cdot (p_i^j)^2}{\sum_i \sum_j P_j \cdot (p_i^j)^2}$$

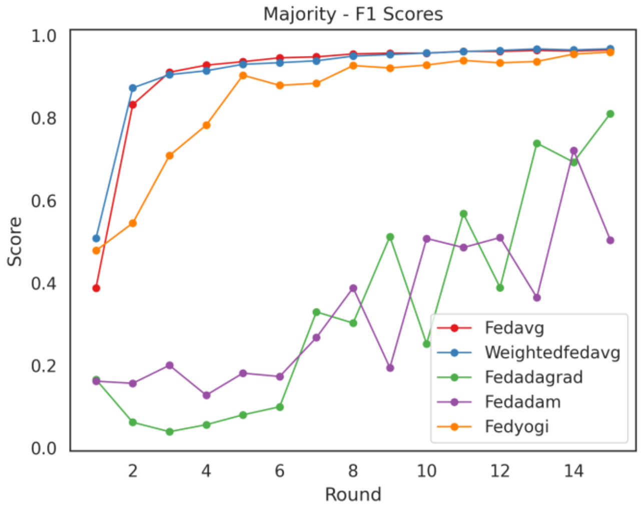
Setting this squared probabilities adds more importance to majority classes respect to the minority classes. Then the normalized weights can be computed as the inverse of this value:

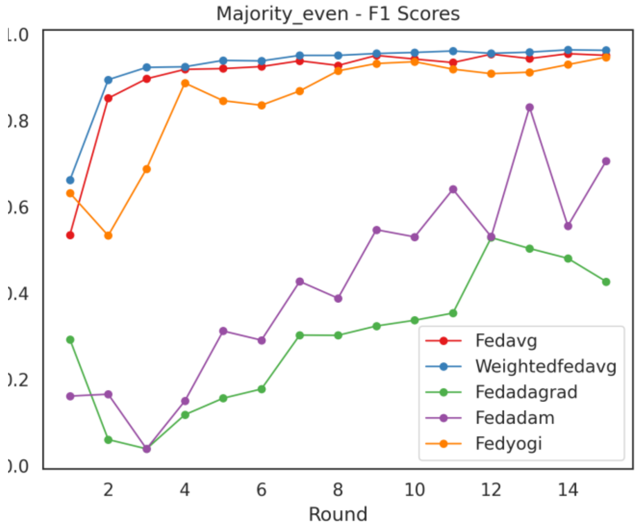
$$w_i = \frac{1/\tilde{P}_i}{\sum_i 1/\tilde{P}_i}$$

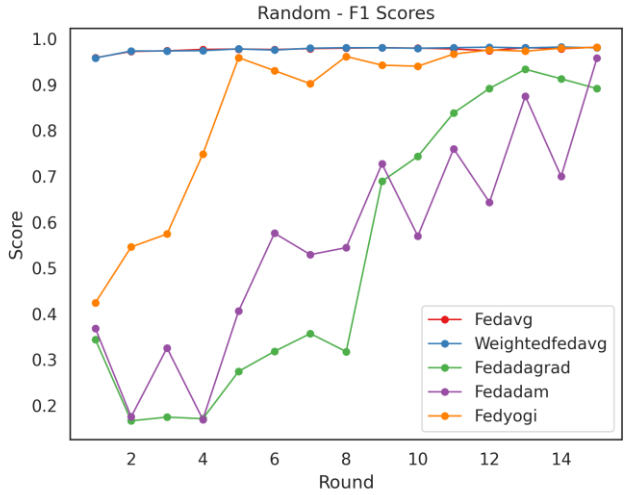


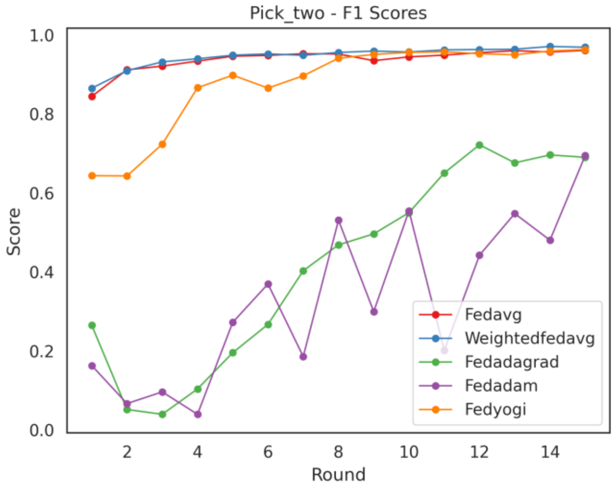
Experiment Results

- FedAvg, WFedAvg and FedYogi perform better than Fedadagrad and Fedadam
    - Fedadam and Fedadagrad would do better if properly optimized
- FedYogi seem to reach convergence a bit later than Fedavg and WFedAvg
- FedAvg and WFedAvg show a very similar behavior (for random split they are expected to be identical)
- The convergence is reached after few rounds

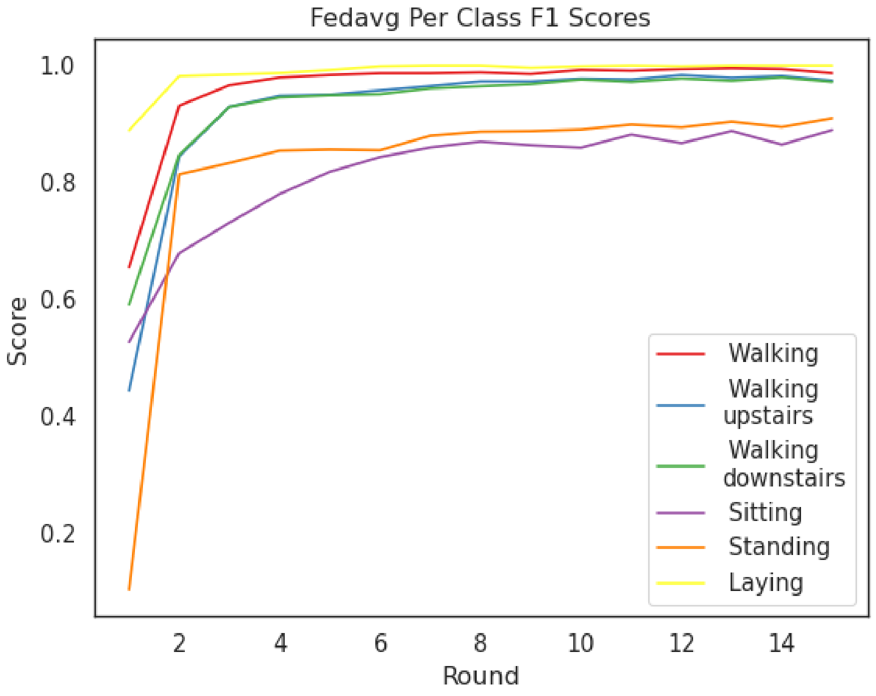


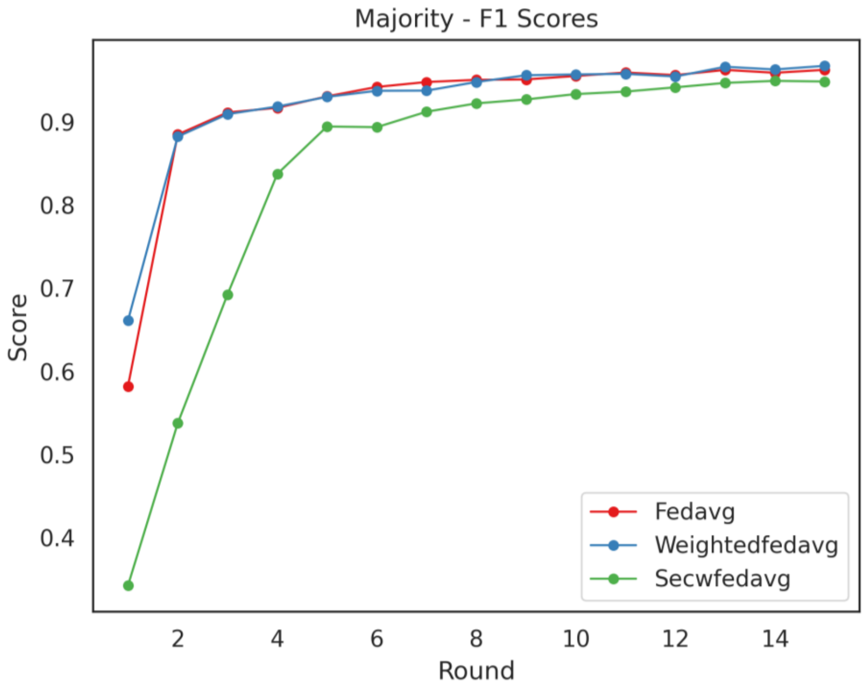






Experiment Results





- FedAvg Performance for each activity class
- For two classes learning is slower

- Performance of FedAvg and WFedAvg compared to FedAvg with Homomorphic Encritption
- HE makes the convergence slower

Conclusions

- Homomorphic Encryption can add significant time and communication costs, scaling with the amount of encrypted weights/gradients
- Heterogeneity of the training dataset seems to not affect performance in this example:
  - The best models are FedAvg and WFedAvg, which are very similar
  - Adaptive methods (FedYogi) do not show a better performance with heterogenous data
  - It seems that the simpler aggregation technique (FedAvg) work better for this dataset

These results are preliminary

- The task is too easy with the selected dataset. Convergence is reached too quickly
- Next steps are to explore different hyperparameters and use a more complex dataset

Preliminary results indicate faster overall training and faster learning of different classes when using Weighed Federated Averaging (can lead to less communication cost if learning is quicker)

UN PET repository: **https://unstats.un.org/wiki/display/UGTTOPPT/Case+study+repository**

Istat

Thanks for your attention

For further information please write to *erika.cerasti@istat.it*