# Differentially Private Federated Learning for High-Accuracy Carbon Footprint Prediction that Protects Sensitive Industrial Data

**Hanna Jarlaczyńska**
Unit8 SA
Cracow, Poland

**Vijay K. Narasimhan**
EMD Electronics
San Jose, CA 95134
vijay.narasimhan@emdgroup.com

**Tingting Ou**
Columbia University
New York, NY

## Abstract

Life Cycle Impact Assessment (LCIA) often lacks accurate data owing to reluctance in industry to share proprietary production information. Here, we present a privacy-preserving framework that improves carbon footprint prediction using federated learning and differential privacy. Our method maintains data confidentiality while enhancing prediction accuracy and consistency. Experiments on public data show strong performance ($R^2 = 0.96$ at $\epsilon = 15$), comparable to standard and aggregated data models. This approach enables more reliable Scope 3 emissions assessments, supporting accurate and collaborative LCIA amid growing regulatory demands.

## 1 Introduction

Life Cycle Impact Assessment (LCIA) plays a central role in sustainable design by estimating environmental impacts across supply chains [11]. The LCIA process involves converting material and energy flows into impact indicators, such as global warming potential, using characterization factors. These factors are often sourced from databases such as ecoinvent (`https://ecoinvent.org/database/`), Sphera GaBi (`https://lcadatabase.sphera.com/`), and openLCA Nexus (`https://nexus.openlca.org/databases`), especially when direct supplier data is unavailable. However, existing databases contain only a fraction of the 204 million known materials [3], often leading to inconsistent assessments. For example, 100-year global warming potential values can vary by up to 600 percent between sources [4]. With new regulations like the EU's CSRD and California's Climate Corporate Data Accountability Act (HSC 38532), companies face growing pressure to report Scope 3 emissions, increasing the demand for reliable carbon footprint prediction frameworks.

We can treat carbon footprint and other LCIA calculations as a supervised deep learning problem: the inputs are the quantities of resources used by a process, and the model predicts its carbon footprint, trained to minimize prediction error. Training such models is technically straightforward but practically challenging: process and resource data are proprietary and often contain information that could be relevant to a company's competitive advantage, so organizations hesitate to share such data within their own supply chains and publicly through databases, leading to siloed, small datasets and models that do not generalize well. Prior work has improved life cycle assessment models using machine learning [2], decision trees [13], secure multi-party computation [8], and probabilistic methods [6]; integrating privacy-preserving techniques into generalized databases of characterization factors remains a promising direction to address key LCIA limitations.

Here, we propose a practical framework that integrates federated learning and differential privacy. Federated learning frameworks aggregate locally trained model updates at a central server without exposing raw data, and are widely accepted in privacy-sensitive environments such as healthcare, finance, and mobile applications [7]. However, even federated learning models are susceptible to

Tackling Climate Change with Machine Learning: workshop at NeurIPS 2025.

**Algorithm 1** Differentially Private Federated Learning (DPFL)

**Input:** Datasets $\{D_i\}_{i\in[m]}$ belonging to $m$ clients, initial baseline model $\theta_b$, target privacy budget $(\epsilon, \delta)$, number of local epochs $E$, learning rate $\eta$, the number of round $T$
**Output:** Differentially private global model $\bar{\theta}$
**for** each round $t = 1, 2, \ldots, T$ **do**
  **for** each client $i \in [m]$ in parallel **do**
    Receive global model $\theta_b$ from the server
    Compute noisy local gradient updates:
    $\tilde{g}_i = \mathsf{LocalTrain}(\theta_b, D_i, E, \eta, \frac{\epsilon}{Tm}, \frac{\delta}{Tm})$
    Send $\tilde{g}_i$ to the server
  **end for**
  Aggregate noisy gradients at the server:
  $\Delta = \frac{1}{m}\sum_{i\in\mathcal{S}} \tilde{g}_i$
  Update global model: $\theta_b \leftarrow \theta_b + \Delta$
**end for**
Return the final global model $\bar{\theta} = \theta_b$

privacy risks, as they can memorize and potentially reveal information about their training data. Through the injection of random noise into the model updates, differential privacy (DP) provides a formal guarantee that the inclusion or exclusion of a single individual's data does not significantly affect the outcome of any analysis [5]. By augmenting federated learning with differential privacy, we can protect sensitive information while enabling collaborative model training.

## 2 Algorithm

Our algorithm, Differentially Private Federated Learning (DPFL), adopts the federated averaging (FedAvg) framework [9] and assumes that there are $m$ clients, and each client owns a private dataset $D_j$, $j \in [m]$. Each client trains a local model $f_i : \mathcal{X} \to \mathcal{Y}$ via the differentially private stochastic gradient descent (DP-SGD) algorithm [1] to predict carbon footprints based on their local data. These local models $f_i, i \in [m]$, are then aggregated to produce a final global model $\bar{f}$. In the simplest case, this would be equivalent to aggregating a table of characterization factors from many small and potentially overlapping datasets, using noise to mask which factors came from which source.

The framework consists of two key components: a global server that coordinates model updates and multiple clients that independently train local models. At the start of each communication round, the server sends the global model to the clients (in the first round, the central server randomly initializes the global model). Each client then trains its local model using the LocalTrain procedure, which uses DP-SGD to learn in a privacy-preserving manner. We set an *overall* desired privacy budget in terms of $(\epsilon, \delta)$, where $\epsilon$ controls the privacy loss, with smaller values indicating stronger privacy guarantees, while $\delta$ accounts for a small probability of privacy violation. Then, if we run the algorithm for $m$ clients over $T$ rounds, by the basic composition rule in [5], we can enforce each LocalTrain subroutine to consume a privacy budget of $(\epsilon/Tm, \delta/Tm)$. Each client computes per-example gradients for mini-batches of its local dataset. These gradients are clipped to a predefined norm $C$ to limit the sensitivity of updates, ensuring that no single data point disproportionately influences the result. Gaussian noise, scaled by the sensitivity and the desired privacy level, is then added to the averaged gradients. The noisy gradients are sent to the server to refine the global model.

Our proposed DPFL algorithm extends and modifies existing approaches to federated learning with differential privacy. Traditional DP-FedAvg algorithms, such as the one in [10], typically add noise to the aggregated model updates at the central server, thereby achieving global differential privacy. However, this approach assumes trust in the central server and fails to provide privacy guarantees at the client level. In contrast, our algorithm applies noise at the client level using DP-SGD, ensuring local differential privacy and eliminating the need for a trusted server. This design choice aligns with the framework of DP-FedSGD, where noise is added at the client level. However, unlike DP-FedSGD, which sends noisy updates after training on a single batch of data, our method allows clients to train for multiple local epochs before communicating with the server. This modification reduces the communication overhead and improves the performance significantly.

**Algorithm 2** LocalTrain

**Input:** Initial model $\theta$, local dataset $D$, number of local epochs $E$, learning rate $\eta$, target privacy budget $(\epsilon, \delta)$
**Output:** Noisy gradient update $\tilde{g}$
Initialize gradient accumulator: $g = 0$
Compute clipping norm $C$ and noise scale $\sigma$ for the target privacy level $(\epsilon, \delta)$
**for** epoch $e = 1, \ldots, E$ **do**
  **for** each mini-batch $B_1, \ldots, B_k \subset D$ **do**
    Compute per-example gradients: $\{g_j\}_{j \in B} = \nabla \ell(\theta; B)$
    Clip the gradients: $g_j \leftarrow g_j / \max\left(1, \frac{\|g_j\|}{C}\right)$ for all $j \in B$
    Compute the batch average gradient: $g_B = \frac{1}{|B|} \sum_{j \in B} g_j$
    Accumulate gradients: $g \leftarrow g + g_B$
  **end for**
**end for**
Add Gaussian noise: $\tilde{g} = \frac{g}{Ek} + \mathcal{N}(0, \sigma^2 C^2 I)$
Return $\eta \tilde{g}$

# 3 Experiments

## 3.1 Data and tools

To evaluate our approach, we use real-world processes from TianGong (`https://www.tiangong.earth/`), an open-source database for Life Cycle Assessment that includes over 10,000 unit processes and inventories. We match materials from TianGong processes to entries in the open emission factor database collected from `https://nexus.openlca.org/databases`. This table comprises carbon emission factors measured in kgCO2 equivalent per unit for various inputs, including reagents, transportation, materials, processes, and energy. We perform this matching to ensure that different processes maintain the same dimensionality ($n = 557$), allowing for consistent training, testing, and analyses. We selected those processes in which all inputs align with our table of materials and their carbon emission factors. This results in 508 processes with specified input material quantities. A total of 97 materials from the factor table are identified as inputs in these real processes. We calculate the total carbon footprint of each process using the emission factor table. For instance, if material 1 and 2 have input quantities $x$ and $y$ with emission factors $a$ and $b$, the ground truth label is $ax + by$. When units differ between sources (e.g., mass vs. energy), we use appropriate conversions. We split the dataset as follows: 80% of the data (407 samples) is used for training within the federated learning framework (split evenly across the $m = 3$ clients), while the remaining 20% (101 samples) is reserved for testing.

We used Google Colab's runtime with an Intel(R) Xeon(R) CPU @ 2.20GHz. The Python environment was based on Python 3.10 with PyTorch 2.5.1. Since the experiment is performed as a proof of concept to demonstrate the validity of the proposed DPFL algorithm, we only ran $T = 1$ round. In the LocalTrain subroutine, each client constructs a multilayer perceptron network with identical architecture for conducting regression on their respective input datasets. The fully-connected network comprises three layers, wherein the hidden layer is structured with a dimension of 500. Subsequent to local training by each client, the model parameter updates are aggregated and averaged to obtain the final model. The Opacus package [12] was used to determine the clipping norm $C$ and noise scale $\sigma$ under a fixed privacy budget for each client.

## 3.2 Results

After obtaining our final model which aggregates updates from 3 locally trained models, we then evaluate the final model on the reserved test data, measuring its performance using the $R^2$ score:

$$R^2 = 1 - \frac{\sum_i (y_i - f(x_i))^2}{\sum_i (y_i - \bar{y})^2}$$

where $i$ indexes the samples in the test dataset, $f(x_i)$ is the predicted emission value and $\bar{y}$ represents the mean of the true emission values $\{y_i\}$. This $R^2$ score quantifies how well our predicted emission

Table 1: **Model performance comparison for varying $\epsilon$ values with fixed $\delta = 10^{-4}$.** The first three columns represent the $R^2$ scores of the local models for clients 1, 2, and 3, each trained with a privacy budget of $\epsilon/3$ and $\delta = \frac{1}{3} \cdot 10^{-4}$. The fourth column presents the $R^2$ score of the aggregated model obtained via DPFL, using a total privacy budget of $\epsilon$. The last column presents the $R^2$ score of the baseline model, trained on the fully aggregated data with the same overall privacy budget of $\epsilon$.

| $\epsilon$ | $R^2(1)$ | $R^2(2)$ | $R^2(3)$ | $R^2(agg)$ | $R^2(baseline)$ |
|---|---|---|---|---|---|
| 1.5 | 0.9775 | 0.8150 | 0.4637 | 0.7709 | 0.9039 |
| 3 | 0.6969 | 0.7240 | 0.8734 | 0.8976 | 0.9852 |
| 15 | 0.9952 | 0.9464 | 0.9917 | 0.9608 | 0.9947 |
| 30 | 0.9990 | 0.9853 | 0.9910 | 0.9789 | 0.9954 |

outputs correlate with the actual emission outputs. We compare the performance of DPFL against both the individual models and a baseline model trained on the full dataset — an approach that is typically impractical in practice due to the need for raw data aggregation. Note that both the DPFL model and the baseline model are allocated the same privacy budget to ensure a fair comparison. Our results are presented in Table 1.

The first three columns show the $R^2$ values for models trained locally on small datasets with a privacy budget of $\epsilon/3$, where performance can be highly variable. Notably, at $\epsilon = 1.5$, one local model achieves a particularly poor $R^2$ of 0.4637, suggesting that individual local models may struggle due to limited data and privacy constraints. However, our federated learning approach mitigates this issue by aggregating the local models, as reflected in the fourth column ($R^2(agg)$), where the final aggregated model via DPFL achieves a much-improved $R^2$ of 0.7709. This result illustrates that federated learning helps correct for poor individual models, leading to a more robust overall model. Moreover, at $\epsilon = 3$ and above, the aggregated models show high fidelity and utility, with $R^2$ of 0.9 or higher.

Finally, aggregating models inherently loses some accuracy compared to aggregating raw data before training, as the latter allows for better optimization and learning directly from a unified dataset. However, at $\epsilon = 15$, a value similar to that used by private smart keyboard and census applications, $R^2(agg)$ is within 5% of $R^2(baseline)$. This result suggests that while federated learning sacrifices some accuracy for privacy and decentralization, it still achieves strong overall performance, making it a promising privacy-preserving alternative.

## Conclusion and impact

This work leverages federated learning and differential privacy to enable privacy-preserving collaboration for building large models used in carbon footprint analysis and Life Cycle Impact Assessment (LCIA). Results on real-world processes show strong predictive performance, with an epsilon value of 15 striking a balance between privacy and accuracy, achieving less than 5% degradation compared to non-private models. By safeguarding sensitive data while fostering global collaboration using algorithms with relatively little computational overhead, this approach supports more accurate and representative environmental insights. The potential societal impact includes advancing sustainable practices by scaling data availability, enabling data-driven decision-making for hot spot identification and product life cycle sustainability while ensuring data privacy and security.

## References


[1] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, page 308–318, New York, NY, USA, 2016. Association for Computing Machinery.

[2] Mikaela Algren, Wendy Fisher, and Amy E. Landis. Chapter 8 - machine learning in life cycle assessment. In Jennifer Dunn and Prasanna Balaprakash, editors, *Data Science Applied to Sustainability Analysis*, pages 167–190. Elsevier, 2021.

[3] CAS. The world's largest collection of chemistry insights, 2025.

[4] Xiaoju Chen, H. Scott Matthews, and W. Michael Griffin. Uncertainty caused by life cycle impact assessment methods: Case studies in process-based lci databases. *Resources, Conservation and Recycling*, 172:105678, 2021.

[5] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[6] Joshua Hester, T Reed Miller, Jeremy Gregory, and Randolph Kirchain. Actionable insights with less data: guiding early building design decisions with streamlined probabilistic life cycle assessment. *The International Journal of Life Cycle Assessment*, 23:1903–1915, 2018.

[7] P. Kairouz, H. B. McMahan, Brendan Avent, A. Bellet, M. Bennis, A. Bhagoji, Keith Bonawitz, Zachary B. Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, S. E. Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, M. Gruteser, Zaïd Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, T. Javidi, Gauri Joshi, M. Khodak, Jakub Konecný, A. Korolova, F. Koushanfar, Oluwasanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, R. Raskar, D. Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, A. Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning. *Found. Trends Mach. Learn.*, 14:1–210, 2019.

[8] Brandon Kuczenski, Cetin Sahin, and Amr El Abbadi. Privacy-preserving aggregation in life cycle assessment. *Environment Systems and Decisions*, 37:13–21, 2017.

[9] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[10] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations*, 2018.

[11] D.W. Pennington, J. Potting, G. Finnveden, E. Lindeijer, O. Jolliet, T. Rydberg, and G. Rebitzer. Life cycle assessment part 2: Current impact assessment practice. *Environment International*, 30(5):721–739, 2004.

[12] Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, Graham Cormode, and Ilya Mironov. Opacus: User-friendly differential privacy library in pytorch. *CoRR*, abs/2109.12298, 2021.

[13] Bu Zhao, Chenyang Shuai, Ping Hou, Shen Qu, and Ming Xu. Estimation of unit process data for life cycle assessment using a decision tree-based approach. *Environmental Science & Technology*, 55(12):8439–8446, 2021. PMID: 34053219.