Fed-BioMed: A general open-source frontend framework for federated learning in healthcare

Santiago Silva[1], Andre Altmann[2], Boris Gutman[3], and Marco Lorenzi[1]

[1] Université Côte d'Azur, INRIA Sophia Antipolis, EPIONE Research Project, France
[2] COMBINE Lab, Centre for Medical Image Computing (CMIC), University College London, London, UK
[3] Department of Biomedical Engineering, Illinois Institute of Technology, Chicago, IL USA

Abstract. While data in healthcare is produced in quantities never imagined before, the feasibility of clinical studies is often hindered by the problem of data access and transfer, especially regarding privacy concerns. Federated learning allows privacy-preserving data analyses using decentralized optimization approaches keeping data securely decentralized. There are currently initiatives providing federated learning frameworks, which are however tailored to specific hardware and modeling approaches, and do not provide natively a deployable production-ready environment. To tackle this issue, herein we propose an open-source federated learning frontend framework with application in healthcare. Our framework is based on a general architecture accommodating for different models and optimization methods. We present software components for clients and central node, and we illustrate the workflow for deploying learning models. We finally provide a real-world application to the federated analysis of multi-centric brain imaging data.

Keywords: federated learning, healthcare, medical imaging

1 Introduction

The private and sensitive nature of healthcare information often hampers the use of analysis methods relying on the availability of data in a centralized location. Decentralized learning approaches, such as federated learning, represent today a key working paradigm to empower research while keeping data secure[11].

Initially conceived for mobile applications [8], federated learning allows to optimize machine learning models on datasets that are distributed across clients, such as multiple clinical centers in healthcare [4]. Two main actors play in the federated learning scenario: *clients*, represented for instance by clinical centers, and a *central node* that continuously communicate with the clients [23].

Federated learning methods must address three main issues: *security*, by preventing data leakages and respecting privacy policies such as the EU general data protection regulation (GDPR) [22], *communication efficiency*, by optimizing the

rounds of communication between clients and the central node, and *heterogeneity robustness*, by properly combining models while avoiding biases from the clients or adversarial attacks aiming to sabotage the models [2,1]. These issues are currently tackled through the definition of novel federated analysis paradigms, and by providing formal guarantees for the associated theoretical properties [12,13].

Besides the vigorous research activity around the theory of federated learning, applications of the federated paradigm to healthcare are emerging [6,7,10]. Nevertheless, the translation of federated learning to real-life scenarios still requires to face several challenges. Besides the bureaucratic burden, for federation still requires to establish formal collaboration agreements across partners, the implementation of a federated analysis framework requires to face important technical issues, among which the problem of data harmonization, and the setup of software infrastructures. In particular, from the software standpoint, the practical implementation of federated learning requires the availability of frontend frameworks that can adapt to general modeling approaches and application scenarios, providing researchers with a starting point overcoming problems of deployment, scalability and communication over the internet.

In this work we propose Fed-BioMed, an open-source production-ready framework for federated learning in healthcare. Fed-BioMed is Python-based and provides modules to deploy general models and optimization methods within federated analysis infrastructures. Besides enabling standard federated learning aggregation schemes, such as federated averaging (FedAVG) [8,14], Fed-BioMed allows the integration of new models and optimization approaches. It is also designed to enable the integration with currently available federated learning frameworks, while guaranteeing secure protocols for broadcasting.

We expect this framework to foster the application of federated learning to real-life analysis scenarios, easing the process of data access, and opening the door to the deployment of new approaches for modeling and federated optimization in healthcare. The code will be freely accessible from our repository page (https://gitlab.inria.fr/fedbiomed).

2 Related works

NVIDIA Clara is a large initiative focusing on the deployment of federated learning in healthcare [24], currently providing a service where users can deploy personalized models. The code of the project is not open, and it requires the use of specific hardware components. This may reduce the applicability of federated learning to general use-cases, where client's facilities may face restrictions in the use of proprietary technology.

The Collaborative Informatics and Neuroimaging Suite Toolkit for Anonymous Computation (COINSTAC) [17] focuses on single-shot and iterative optimization schemes for decentralized multivariate models, including regression and latent variable analyses (e.g. principal/independent-component analysis, PCA-ICA, or canonical correlation analysis, CCA). This project essentially relies on distributed gradient-based optimization as aggregating paradigm. A frontend

distributed analysis framework for multivariate data modeling was also proposed by [20]. Similarly to COINSTAC, the framework focuses on the federation of multivariate analysis and dimensionality reduction methods. The `PySyft` initiative [18] provides an open-source wrapper enabling federated learning as well as encryption and differential privacy. At this moment however this framework focuses essentially on optimization schemes based on stochastic gradient descent, and does not provide natively a deployable production-ready framework. Fed-BioMed is complementary to this initiative and allows interoperability, for example by enabling unit testing based on `PySift` modules.

3 Proposed Framework

3.1 Architecture

Fed-BioMed is inspired by the 2018 "Guide for Architectural Framework and Application of Federated Machine Learning" (Federated Learning infrastructure and Application standard)[4] basing the architecture on a server-client paradigm with three types of instances: central node, clients and federators.

The clients are responsible for storing the datasets through a dedicated client application (Figure 1, right). Since not every client is required to store the same type of data, this enables studies with heterogeneous or missing features across participants. Moreover, depending on the target data source and on the analysis paradigm, centers can be either included or ignored from the study. Each client verifies the number of current jobs in queue with the central node, as well as data types and models associated to the running jobs.

The central node consists in a secured RESTful API interfacing the federators with the clients, by storing their jobs in queue and submissions. This instance also allows researchers to deploy jobs using predefined models and data.

The federators are in charge of combining the models submitted by the clients for each job, and sharing the global model back through the central node. For this initial stage, Federated Averaging is used as the default federating aggregator. However, as each federator instance is conceptualized as an isolated service, alternative federated aggregators can be included, such as based on distributed optimization via Alternating Direction Method of Multipliers (ADMM) [3]. To allow clients and federators to interact with the RESTful API, we provide a dedicated standard Python library (`fed-requests`). This eases the procedure for deploying new models or federators into the framework.

Communication scheme: Every instance is packaged and deployed in form of Docker containers [15] interacting between each other through HTTP requests. Containerized instances help to overcome software/hardware heterogeneity issues by creating an isolated virtualized environment with a predefined operating system (OS), thus improving reproducibility as every center run on the same software environment. This scheme also achieves scalability and modularity when

[4] https://standards.ieee.org/project/3652_1.html

dealing with large amounts of clients. In this case, multiple instances of the API can be created under a load balancer, while federator instances can be separately deployed on a dedicated computation infrastructure.

Security: Fed-BioMed addresses typical security issues regarding communication (e.g. man-in-the-middle and impersonation attacks) and access permissions as follows: 1) all requests are encrypted by using HTTP over SSL, 2) user authentication relies on a password-based scheme, and 3) reading/writing operations are restricted by role definitions. Protection from adversarial or poisoning attacks [2] is currently not in the scope of this work, but can be naturally integrated as part of the federator. In the future, malicious attacks will be also prevented by implementing certification protocols attesting the safety of the model source code before deployment [19].

Traceability: to allow transparency to the centers and for the sake of technical support, each instance leverages on a logging system that allows to keep track of every request made, shared data and of the current available jobs.

The architecture behind Fed-BioMed is illustrated in Figure 1, left. The common procedure involves the deployment of one or multiple jobs from the researchers. Each job must contain the architecture model to be trained and its initialized parameters for reproducibility, the number of rounds, and the federator instance to be used as optimizer.

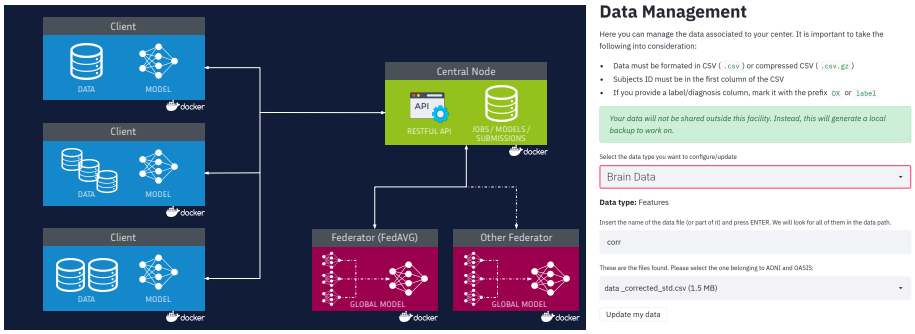

Fig. 1. Left: Clients providing different data sources share their local model parameters with the central node. The central node creates the jobs that will be run by the clients, and transmits the initialization parameters for the models in training. The federator gathers the collected parameters and combines then into a global model that is subsequently shared with the clients for the next training round. As instances are isolated in containers, new instances, such as a new federators (dashed line), can be introduced without altering the behavior of the infrastructure. **Right:** Screenshot of the "Manage Data" section in the client application.

3.2 Workflow

Deploying a new model. Fed-BioMed relies on a common convention for deploying a new model. A model must be defined as a PyTorch [16] module class, containing the common methods for any torch module:

- `__init__(**kwargs)` method: defines the model initialization parameters;
- `forward()` method: provides instructions for computing local model updates.

```python
class MyModel(torch.nn.Module):
    def __init__(**kwargs):
        # Define model

    @staticmethod
    def loss_function(**args):
        # Define a personalized
        # loss function if needed

    def forward(self, **kwargs):
        # Compute the local updates:
        # parameters to aggregate

    # You can also define
    # other functions
    # (e.g. encode, decode)
```

```python
from fed_requests import fedbionet_api as api

# Get list of running Jobs
jobs = api.get_jobs(
    params={
        'status': 'Running',
        'federator': 'MY_METHOD'
    }
)
for job in jobs:
    # Get submissions made by clients
    job_url = job['url']
    current_round = job['current_round']
    submissions = api.get_submissions(
        job_url=job_url,
        current_round=current_round
    )

    # Aggregate submitted models
```

Fig. 2. Examples of model declaration (left) and creation of a new federator (right).

Once the new class is defined, it can be integrated in the model zoo for both clients and federators (Figure 2, left).

Deploying a new federator or aggregation function in the backend is obtained by creating a containerized service that queries the API for the necessary submissions at each round, and subsequently aggregates the submitted local updates (Figure 2, right).

4 Experiments

This experimental section illustrates our framework on two different applications: 1) an analysis on the MNIST dataset [9] involving 25 clients, and 2) a multi-centric analysis of brain imaging data involving four research partners based on different geographical locations.

MNIST analysis: The 60000 MNIST images were equally split among 25 centers. Each center was synthetically emulated and setup in order to interact with the centralized API. The model was represented by a variational autoencoder (VAE) implementing non-linear encoding and decoding functions composed by three layers respectively, with a 2-dimensional associated latent space. For training we used a learning rate of $l = 1 \times 10^{-3}$, 10 local epochs for 30 optimization rounds, while the federated aggregating scheme was FedAVG.

Brain imaging data analysis: In this experiment we use our framework to perform dimensionality reduction in multi-centric structural brain imaging data (MRI) across datasets from different geographical locations, providing cohorts of healthy individuals and patients affected by cognitive impairment.

Four centers participated to the study and were based geographically as follows: two centers in France (Center 1 and Center 4), one in the UK (Center 3), and a last one in the US (Center 2). The central node and the federator were also located in France. Each center was running on different OS and the clients were not 100% online during the day allowing to test the robustness of the framework in resuming the optimization in real-life conditions.

For each center, data use permission was obtained through formal data use agreements. Data characteristics across clients are reported in Table 1. A total of 4670 participants were part of this study, and we note that the data distribution is heterogeneous with respect to age, range and clinical status. 92 features subcortical volumes and cortical thickness were computed using FreeSurfer [5] and linearly corrected by sex, age and intra-cranial volume (ICV) at each center.

This analysis involved data standardization with respect to the global mean and standard deviation computed across centers, and dimensionality reduction was performed via a VAE implementing a linear embedding into a 5-dimensional latent space. Federated data standardization was implemented as in [20], while VAE's parameters aggregation was performed through FedAVG. Federated learning was performed by specifying a pre-defined budget of 30 rounds of client-server iterations in total with 15 epochs/client-round at a learning rate of $l = 1 \times 10^{-3}$.

Table 1. Demographics for each of the centers sharing brain-imaging data. MCI: Mild Cognitive Impairment; AD: Alzheimer's Disease.

	Center 1	**Center 2**	**Center 3**	**Center 4**
No. of participants (M/F)	448/353	454/362	1070/930	573/780
Clinical status				
No. healthy	175	816	2000	695
No. MCI and AD	621	0	0	358
Age ± sd (range) [years]	73.74 ± 7.23	28.72 ± 3.70	63.93 ± 7.49	67.58 ± 10.04
Age range [years]	54 - 91	22 - 37	47 - 81	43 - 97

5 Results

The evolution of the models during training can be assessed by analyzing the weights' norm across iterations (Figure 3 and supplementary material for the complete set of weights). MNIST parameters evolution is shown in the top panel, while the related test set projected onto the latent space is shown in Figure 4, left panel, describing a meaningful variability across digits and samples.

Concerning the brain imaging data analysis experiment, the evolution of the encoding parameters throughout the 30 optimization rounds is shown in

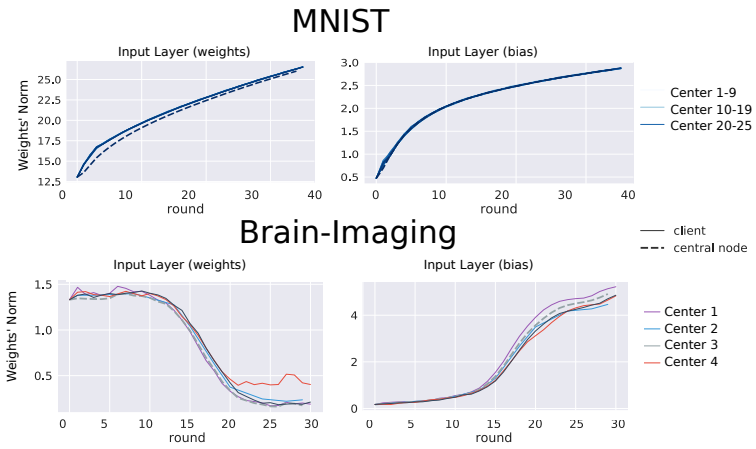

Fig. 3. Illustration of parameter evolution for VAE parameters (input layer). The federated model closely follows the clients' weights distribution. **Top:** MNIST across 25 centers with equally distributed data. **Bottom:** brain-imaging heterogeneous dataset across 4 centers with unevenly distributed data. Continuous lines: clients weights' norm. Dashed lines: federated model weights' norm.

Figure 3, bottom panel. For this real-world application we also collected each client's elapsed time to report its local updates to the central node, as well as the average time per round in the best scenario (Figure 5). As expected, the clients' time varies depending on the geographical proximity with the central node, as well as on the local upload/download speed. The model was further investigated by inspecting the latent space on the subset of the training data available to the coordinating Center 1. The right panel of Figure 4 shows that although most of the training data for the VAE comes from healthy and young participants, the model is also able to capture the pathological variability related to cognitive impairment in aging. The latent variables associated to the observations of Center 1 indeed show significantly different distributions across different clinical groups, from healthy controls (CN), to subjects with mild cognitive impairment (MCI), and patients with Alzheimer's disease (AD).

6 Conclusions

This work presents an open-source framework for deploying general federated learning studies in healthcare, providing a production ready reference to deploy new studies based on federated models and optimization algorithms. Our experimental results show that the framework is stable in communication, while being robust to handle clients going temporarily out of the grid. Scalability is obtained thanks to the use of containerized services. The ability to handle client-authentication and the use of secured broadcasting protocols are also appealing security features of Fed-BioMed. While the experiments mostly focused on VAE

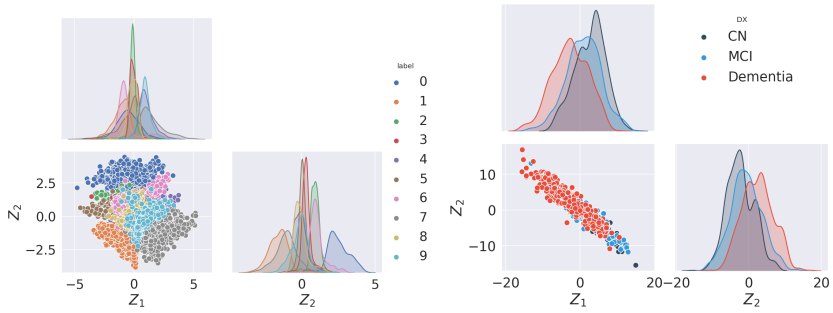

Fig. 4. Left: MNIST pixel data projected onto the latent space. **Right:** Brain features of Center 1 projected onto the first 2 components in the latent space. Although the model was trained with unbalanced data, it is still able to capture pathological variability. CN: healthy controls; MCI: mild cognitive impairment; Dementia: dementia due to with Alzheimer's disease.

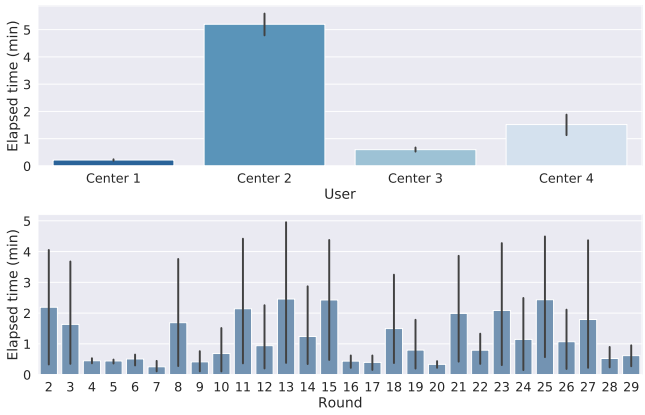

Fig. 5. Top: User average elapsed time per round (since a new version of the model is made available and each user to submit its local update). Geographical and data distribution: Center 1 (FR) , Center 2 (US), Center 3 (UK) and Center 4 (FR). **Bottom:** Averaged elapsed time across centers per round of updates.

and Federated Averaging as aggregating paradigm, our framework is completely extensible to other distributed optimization approaches. Future work will therefore integrate additional models for the analysis of different data modalities and bias, as well as enhanced secure P2P encryption. Concerning the clinical experimental setup, the brain application was chosen to provide a demonstration of our framework to a real-life analysis scenario, and it is not aimed to address a specific clinical question. In the future, the proposed work Fed-BioMed will be a key component for clinical studies tailored to address challenging research questions, such as the analysis of imaging-genetics relationships in current meta-analysis initiatives [21].

7 Acknowledgements

This work has been supported by the French government, through the 3IA Côte d'Azur Investments in the Future project managed by the National Research Agency (ANR) with the reference number ANR-19-P3IA-0002, and by the ANR JCJC project Fed-BioMed 19-CE45-0006-01. The project was also supported by the Inria Sophia Antipolis - Meediterraneee, "NEF" computation cluster.

References

1. Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., Shmatikov, V.: How to backdoor federated learning. In: International Conference on Artificial Intelligence and Statistics. pp. 2938–2948 (2020)
2. Bhagoji, A.N., Chakraborty, S., Mittal, P., Calo, S.: Analyzing federated learning through an adversarial lens. In: International Conference on Machine Learning. pp. 634–643 (2019)
3. Boyd, S., Parikh, N., Chu, E.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Now Publishers Inc (2011)
4. Brisimi, T.S., Chen, R., Mela, T., Olshevsky, A., Paschalidis, I.C., Shi, W.: Federated learning of predictive models from federated electronic health records. International journal of medical informatics **112**, 59–67 (2018)
5. Fischl, B.: Freesurfer. Neuroimage **62**(2), 774–781 (2012)
6. Gupta, O., Raskar, R.: Distributed learning of deep neural network over multiple agents. Journal of Network and Computer Applications **116**, 1–8 (2018)
7. Kim, Y., Sun, J., Yu, H., Jiang, X.: Federated tensor factorization for computational phenotyping. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 887–895 (2017)
8. Konečnỳ, J., McMahan, H.B., Ramage, D., Richtárik, P.: Federated optimization: Distributed machine learning for on-device intelligence. arXiv preprint arXiv:1610.02527 (2016)
9. LeCun, Y., Cortes, C.: MNIST handwritten digit database (2010), http://yann.lecun.com/exdb/mnist/
10. Lee, J., Sun, J., Wang, F., Wang, S., Jun, C.H., Jiang, X.: Privacy-preserving patient similarity learning in a federated environment: development and analysis. JMIR medical informatics **6**(2), e20 (2018)
11. Li, T., Sahu, A.K., Talwalkar, A., Smith, V.: Federated learning: Challenges, methods, and future directions. IEEE Signal Processing Magazine **37**(3), 50–60 (2020)
12. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. arXiv preprint arXiv:1812.06127 (2018)
13. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smithy, V.: Feddane: A federated newton-type method. In: 2019 53rd Asilomar Conference on Signals, Systems, and Computers. pp. 1227–1231. IEEE (2019)
14. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics. pp. 1273–1282 (2017)
15. Merkel, D.: Docker: lightweight linux containers for consistent development and deployment. Linux journal **2014**(239), 2 (2014)

16. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. In: Advances in neural information processing systems. pp. 8026–8037 (2019)
17. Plis, S.M., Sarwate, A.D., Wood, D., Dieringer, C., Landis, D., Reed, C., Panta, S.R., Turner, J.A., Shoemaker, J.M., Carter, K.W., et al.: Coinstac: a privacy enabled model and prototype for leveraging and processing decentralized brain imaging data. Frontiers in neuroscience **10**, 365 (2016)
18. Ryffel, T., Trask, A., Dahl, M., Wagner, B., Mancuso, J., Rueckert, D., Passerat-Palmbach, J.: A generic framework for privacy preserving deep learning. arXiv preprint arXiv:1811.04017 (2018)
19. Shen, S., Tople, S., Saxena, P.: Auror: Defending against poisoning attacks in collaborative deep learning systems. In: Proceedings of the 32nd Annual Conference on Computer Security Applications. pp. 508–519 (2016)
20. Silva, S., Gutman, B.A., Romero, E., Thompson, P.M., Altmann, A., Lorenzi, M.: Federated learning in distributed medical databases: Meta-analysis of large-scale subcortical brain data. In: 2019 IEEE 16th international symposium on biomedical imaging (ISBI 2019). pp. 270–274. IEEE (2019)
21. Thompson, P.M., Stein, J.L., Medland, S.E., Hibar, D.P., Vasquez, A.A., Renteria, M.E., Toro, R., Jahanshad, N., Schumann, G., Franke, B., et al.: The enigma consortium: large-scale collaborative analyses of neuroimaging and genetic data. Brain imaging and behavior **8**(2), 153–182 (2014)
22. Voigt, P., Von dem Bussche, A.: The eu general data protection regulation (gdpr). A Practical Guide, 1st Ed., Cham: Springer International Publishing (2017)
23. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology (TIST) **10**(2), 1–19 (2019)
24. Yuhong, W., Wenqi, L., Holger, R., Prerna, D.: Federated Learning powered by NVIDIA Clara (2019), https://devblogs.nvidia.com/federated-learning-clara/