Contribution Measures for Incentivizing Personalized Collaborative Learning

Frédéric Berdoz [1] **Martin Jaggi** [1] **Mary-Anne Hartley** [1]

Abstract

Federated and decentralized learning have become key building blocks for privacy-preserving machine learning. Participation in these opaque federations may be better incentivized by transparent communication of each user's contribution. For real-world applications with large numbers of heterogeneous participants, quantifying these contributions according to their impact on model quality remains challenging. We discuss the applicability various contribution measures with a particular focus on the personalized learning setting, where each participant has their own learning objective.

1. Introduction

In past decades, research in artificial intelligence has been driven primarily by the pursuit of predictive performance often at the cost of privacy or other sociological concerns coming from the interaction of users providing data or getting predictions from such systems. However, in the last few years, several incidents contributed to a change in public perception, notably large scale privacy breaches and mis-incentivized social network algorithms (Wheatley et al., 2015). In this light, current interest in privacy preserving techniques such as decentralized or federated learning is increasing. During training, these schemes keep the data local on the device of each user, for improved privacy and control over data access. The standard framework to learn a model under these circumstances is to have a central server that distributes the model and gathers/aggregates the users' updates. This is known as *Federated Learning* (FL) (McMahan et al., 2016). However, as pointed out by Lian et al. (2017b) and Vanhaesebrouck et al. (2017), the use of a central server creates a single point of failure that compromises trust, and might limit incentivization. Decentralized (or *Peer-to-Peer*) learning (P2PL) goes a step further since it does not depend on a central orchestrator, and therefore lets users interact between themselves without relying on global insights that may not represent each participant's interests. (Tsitsiklis et al., 1986; Lian et al., 2017a; Nadiradze et al., 2019; Koloskova et al., 2020).

The need for model personalization makes the decentralised setting particularly interesting. In such decentralized frameworks, users are likely to have heterogeneous data, i.e. data that is not identically and independently distributed (non-IID), and the performance of the shared model might thus vary across users. To mitigate this effect, each client can personalize its own model, i.e. by training in parallel a local model and dynamically averaging it with the shared model (Bellet et al., 2018; Mansour et al., 2020). Alternatively, the client might choose to modify, clean, or augment their local dataset, such as to better match the overall feature distribution of the other peers, and thus improving collaborative training. The goal of this study is to explore how to quantify each peer's contribution to the global (or personalized) training process, with the aim of incentivizing informed participation. For instance, to determine if a given user is profiting (or not) from the shared model, and then to potentially reward users that have high contribution. In the personalized setting, the contribution measure can be used by the client to select its collaborators, in order to maximize quality of the tailored model.

Contribution measurement and *reward allocation* constitute an important (but often overlooked) aspect of decentralized learning known as *Incentivization*, i.e. how to naturally guide participants' behavior towards a collective interest. Since one of the main difficulties of both decentralized or federated learning comes from the fact that data is non-IID, this article aims particularly at incentivizing collaboration between users in order to bring their own distribution closer to the joint distribution that yields best results for the global or personal task at hand.

Our work studies and empirically compares contribution measures (CM) for learning that can then be used for future non-monetary reward mechanisms, both for federated or decentralized learning. We specifically focus on the case of realistic heterogeneous data, for training global models or for personalized learning. In an example scenario of a medical application where both privacy and collaboration are of crucial importance, it could be hypothesized

[1]Machine Learning and Optimization Laboratory, EPFL, Lausanne, Switzerland. Correspondence to: Frédéric Berdoz <frederic.berdoz@epfl.ch>.

Preprint.

that peers with high contribution measures may be able to (willingly) share some information so that other users can adapt their datasets accordingly (for instance, collecting different features or performing specific preprocessing for improved interoperability). Thus, we propose that guiding users on how to improve their contribution for the reward of reciprocity should, in turn incentivize participation and improve the overall model performance.

2. Related work

2.1. Decentralized Learning

Although definitions may vary, decentralized learning usually refers to the setup in which the data is heterogeneously fragmented (non-IID) across multiple nodes, whereas distributed learning refers to the setup in which the distribution is homogeneous across nodes (Kairouz et al., 2021), i.e. when the datasets are too large to be stored on a single server. As mentioned in the Introduction, decentralized learning can be further divided into two categories:

Federated Learning: Although the term was first coined by McMahan et al. (2016), the idea of privacy preserving learning algorithms has been circulating over the past decade, where Agrawal & Srikant (2000), for instance, tried to build a decision tree classifier without having access to the real data. Nowadays, FL is widely used, notably by Google (Yang et al., 2018) and Apple. The typical FL learning pipeline is summarized below:

1. The server initializes the model
2. The server distributes the model to the clients (edge nodes)
3. The clients perform local learning on their private datasets
4. The server gathers and aggregates the local updates of the clients
5. The server updates the global model and repeats steps 2 to 5

Additionally, a distinction is usually made depending on the number of nodes and the size of their datasets (Kairouz et al., 2021). On one hand, when the number of nodes is relatively small and the datasets are large, it is referred to as *Cross-Silo* FL, and on the other hand, when number of nodes is large, it is referred to as *Cross-Device* FL.

Fully Decentralized (Peer-to-Peer) Learning: The idea to fully decentralize a computational task is not new, but became popular with the file-sharing software Napster (Saroiu et al., 2003). For machine learning, decentralized training as an example of this paradigm has received increased research attention recently (Tsitsiklis et al., 1986; Lian et al., 2017a; Wang & Joshi, 2018; He et al., 2018; Nadiradze et al., 2019; Koloskova et al., 2020; Vogels et al., 2020). However, even in fully decentralized learning, a helper server is often needed to hold metadata such as user IPs and task descriptions, but it does not participate directly in the learning of the model. Another difference with FL is that the global model is usually replaced by local models on each node, which is particularly suited for task personalization (Mansour et al., 2020). The collaborative learning updates are typically performed through model averaging with neighbouring nodes, also known as gossip communication on the models (Tsitsiklis et al., 1986; Koloskova et al., 2020; Vogels et al., 2020; Bouchra Pilet et al., 2020).

2.2. Barriers to Participation

Decentralized learning addresses a long standing problem concerning information sharing, particularly in medical applications. In the recent Ebola crisis, for instance, it was found that front line researchers where withholding their data despite the urgency of the situation. A WHO consultation (Goldacre et al., 2003) found diverse disincentives to data sharing such as the fear of *personal data re-identification* (i.e. privacy), concerns about *data ownership*, the lack of *reciprocity*, poor *interoperability* and the lack of a *regulatory/ethical framework*. A decentralized setup alleviates several of these concerns, notably the problem of data ownership, privacy and regulatory frameworks, since data is not directly shared. Reciprocity is also partly satisfied, since users are only able to use the model when they participate in the learning task. However, decentralized learning also has some drawbacks that can disincentivize participation:

Privacy: Even when the data is not directly shared, the model updates (gradients) still contain information about the data, and several studies have demonstrated that individual data samples could be fully recovered using only the gradient of a mini-batch (Zhu et al., 2019; Zhao et al., 2020; Geiping et al., 2020). To counter this, several techniques were developed:

- *Differential Privacy* (DP): The idea behind DP is to add noise to the data so that individual samples cannot be recovered. It has the advantage of being quantifiable but the noise also decreases the precision of the model. It is therefore seen as a trade-off between privacy and performance, which is not always desirable. Several FL frameworks are built around DP, e.g. Sherpa.ai by Rodríguez-Barroso et al. (2020).
- *Secure Multiparty Computation* (MPC): This refers

to a way of computing a previously agreed function whose arguments are dispatched among several entities (like the aggregation of the model updates in FL) but without revealing to each participant more than the result of the computation. The theory behind MPC was developed prior to machine learning (Yao, 1986) but is now widely used in FL. An example using a 2-server secure computation can be found in the work of Mohassel & Zhang (2017).

- *Homomorphic Encryption* (HE): HE is a way of encrypting the data so that certain computations yield the same results with the encrypted and decrypted data (Gentry, 2009). It can also be used to enable MPC, but the mathematical operations that are compatible with the encryption scheme are often very limited, which can be a problem for FL.

Performance Although decentralising the data in a non-IID fashion increases the privacy of the individual data providers, it usually also decreases the performance of the learning algorithm. The different types of distribution disparities can be categorized as follows, where $P_u(x)$ and $P_u(y)$ represents the probability of finding feature x and label y in the dataset of user u, respectively (Kairouz et al., 2021):

1. *Feature distribution skew*, i.e. $P_u(x) \neq P_v(x)$ for two different users u and v.
2. *Label distribution skew*, i.e. $P_u(y) \neq P_v(y)$.
3. *Same label, different features*: $P_u(x|y) \neq P_v(x|y)$.
4. *Same features, different label*: $P_u(y|x) \neq P_v(y|x)$.

In addition to these disparities, some users might also be malicious or comprise a genuinely different or poorly interoperable distribution which is not of interest to the other participants. Here, the simple task of normalizing the dataset becomes difficult. Wang et al. (2020) proposed the idea of collectively creating a small dataset representing the joint distribution so that users can refer to it. This also have drawbacks since this this fictive dataset could be poisoned by malicious/divergent users. Concerning the learning algorithm, a (non-extensive) summary of the different options can be found in work of Kairouz et al. (2021). Some of these algorithms also address the communication constraints by compressing the gradients in a suitable way.

2.3. Incentivization

Incentivization can take multiple forms, for instance:

- Incentivization to participate (attract more users),
- Incentivization to participate efficiently (with accurate/sufficient data),
- Incentivization to participate constructively (instead of maliciously),
- Incentivization to participate collaboratively (for example by improving the datasets of others).

In any case, the incentive mechanism is composed of two distinctive steps: contribution measurement and reward allocation.

Contribution Measurement can be further categorized into three main categories (Huang et al., 2020):

- *Self-reported* based measurement, where each user reports some information about the performance of the model on its own dataset, and that information is used to compute the CM (Pandey et al., 2020). In this setup, some precautions must be taken to make sure that the self-reported information is accurate.
- *Marginal Loss* based measurement, where the CM is computed by measuring what is lost by excluding one given participant (also referred to as *leave-one-out* measures (Wang et al., 2019)). Song et al. (2019) propose two techniques, one where the CM is computed at each round, and a second one where the new measure is computed using the CM of previous rounds (with a forgetting parameter). In general, marginal loss CMs are based on Shapley values (Shapley, 2016), which are theoretical quantities that share several desirable properties.
- *Similarity* based measurements, where participants measure similarities between their respective gradients. For instance, Kang et al. (2019) use these similarities to create a reputation system. In other papers, Zhao et al. (2021) measure these similarities by training a data value estimator using reinforcement learning, whereas Wu & Wang (2020) use the angles between the gradients. Finally, Liu et al. (2020) use the similarities between the clients' model parameters and the global parameters to compute their proposed CM (named FedCM).

Reward Allocation has two main categories:

- *Monetary rewards*: The simplest way of incentivizing users is to reward them with money. This is particularly useful when the central entity (server) generates a profit using the global model and redistributes this profit to the edge users. In that scenario, the contribution measure needs to be particularly fair and robust

to malicious users. Another popular approach is to use a contract theory based framework so that users are held responsible if they degrade the model (Kang et al., 2019; Lim et al., 2020). This is a particular setup in which no explicit CM is required, since the rewards (or punishments) are defined by the contracts.

- *Non-monetary rewards*: Little to no research is done on non-monetary reward allocation. This is because in most scenarios, the model is designed to generate some sort of value. Since even a perfectly performing model could not generate an infinite amount of money, the learning task becomes a competition rather than a collaboration. This is particularly unwanted in medical applications, and the need for a non monetary reward is therefore obvious.

3. Setup

3.1. Context for Federated Learning

Let S denote the central server and $\mathcal{U}$ be the set of participant and define $N := |\mathcal{U}|$. Let $\mathbf{w}^r$ be the global model at round r and $\mathcal{D}_i$ the dataset of user u_i, separated as usual in a training and testing dataset $\mathcal{D}_i^{tr}$ and $\mathcal{D}_i^{te}$. Also, denote

$$\mathcal{D} := \bigcup_{u_i \in \mathcal{U}} \mathcal{D}_i, \quad \mathcal{D}^{tr} := \bigcup_{u_i \in \mathcal{U}} \mathcal{D}_i^{tr}, \quad \mathcal{D}^{te} := \bigcup_{u_i \in \mathcal{U}} \mathcal{D}_i^{te}.$$

Moreover, denote $\mathcal{D}_C := \bigcup_{u_j \in C} \mathcal{D}_j$, $\forall C \subseteq \mathcal{U}$. Let $\mathcal{A}^r$ be the server-side aggregation algorithm (taking as input the set of models to aggregate) and let $\mathcal{F}_i^r$ be the local update algorithm of user u_i at round r (taking as input the model to update and depending on $\mathcal{D}_i^{tr}$). With this notation, the most general FL pipeline can be represented by Algorithm 1, which aims at solving the problem

$$\text{Find } \mathbf{w}^\star = \arg\min_{\mathbf{w}} \mathcal{L}(\mathbf{w}, \mathcal{D}). \tag{1}$$

where $\mathcal{L}$ is the empirical risk (or loss) and can be expressed in term of the loss function l as follows:

$$\mathcal{L}(\mathbf{w}, \mathcal{D}) := \frac{1}{|\mathcal{D}|} \sum_{\mathbf{d} \in \mathcal{D}} l(\mathbf{w}, \mathbf{d}). \tag{2}$$

We focus on the following standard scenario of FL:

- The set of users is constant over time ($U^r = \mathcal{U}$ in Algorithm 1).
- The local update functions and the aggregation function do not depend on time ($\mathcal{F}_i^r \equiv \mathcal{F}_i$ and $\mathcal{A}^r \equiv \mathcal{A}$).

With these assumptions, the setting resembles to the one showed in Figure 1, where $N = 4$.

Algorithm 1: FEDERATED LEARNING

S initializes $\mathbf{w}^0$, $r \leftarrow 0$
while *training:*
  $r \leftarrow r + 1$
  S selects $U^r \subseteq \mathcal{U}$
  for $u_i \in U^r$*:*
    S sends $\mathbf{w}^{r-1}$ to u_i
    $\mathbf{w}_i^r \leftarrow \mathcal{F}_i^r(\mathbf{w}^{r-1})$
    u_i sends $\mathbf{w}_i^r$ to S
  $\mathbf{w}^r \leftarrow \mathcal{A}^r(\{\mathbf{w}_i^r\}_{1 \le i \le N})$
return $\mathbf{w}^r$

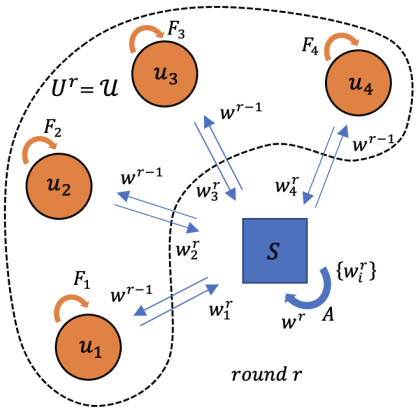

Figure 1. FL setup considered in this article, with $N = 4$.

3.2. Context for Decentralized Learning

Using the same notation as in the FL setting, apart from the fact that the aggregation function $\mathcal{A}_i^r$ can now depend on the user, the general peer-to-peer learning pipeline can be represented by Algorithm 2, which aims at solving the personalized learning problem

$$\text{Find } \{\mathbf{w}_i^\star\} \quad \text{s.t.} \quad \mathbf{w}_i^\star = \arg\min_{\mathbf{w}} \mathcal{L}(\mathbf{w}, \mathcal{D}_i). \tag{3}$$

A central helper server S can still be used to generate the communication graph but it does not take part in the training nor the aggregation. We focus on the following standard scenario of decentralized learning:

- The sets of collaborating users remain constant over time ($U_i^r \equiv U_i$)
- The local update functions do not depend on time ($\mathcal{F}_i^r \equiv \mathcal{F}_i$).
- The local aggregation function is identical for all users ($\mathcal{A}_i \equiv \mathcal{A}$).

Algorithm 2: PERSONALIZED DECENTRALIZED LEARNING

for $u_i \in \mathcal{U}$*:*
  u_i initializes its own model $\mathbf{w}_i^0$
$r \leftarrow 0$
while *training:*
  $r \leftarrow r + 1$
  for $u_i \in \mathcal{U}$*:*
    $\tilde{\mathbf{w}}_i^r \leftarrow \mathcal{F}_i^r(\mathbf{w}_i^{r-1})$
    u_i (or S) selects $U_i^r \subseteq \mathcal{U}$
    u_i sends $\tilde{\mathbf{w}}_i^r$ to u_j **for** $u_j \in U_i^r$
    $\mathbf{w}_i^r \leftarrow \mathcal{A}_i^r(\{\tilde{\mathbf{w}}_i^r\} \cup \{\tilde{\mathbf{w}}_j^r \text{ such that } u_i \in U_j^r\})$
return $\{\mathbf{w}_i^r\}_{1 \leq i \leq N}$

3.3. Shapley Values

Shapley values were first introduced by Shapley (2016) in the context of game theory, but are now commonly used in interpretable machine learning to quantify the contribution of each feature according to a prediction. In decentralized machine learning, they can also be used to compute the contributions of data owners to a trained model. Their formal definition of some of their main properties are provided below. Let $v : 2^{\mathcal{U}} \to \mathbb{R}$ be a value function (giving value to a certain coalition in $\mathcal{U}$). The Shapley value ϕ_i of user $u_i \in \mathcal{U}$ is definied as

$$\phi_i := \sum_{\substack{C \subseteq \mathcal{U} \\ u_i \in C}} \frac{1}{|C|} \binom{N}{|C|}^{-1} \left[v(C) - v(C \setminus \{u_i\}) \right]. \quad (4)$$

- **Efficiency**: $\sum_{u_i \in \mathcal{U}} \phi_i = v(\mathcal{U}) - v(\emptyset)$.
- **Symmetry**: $\phi_i = \phi_j$ if $v(C \cup \{u_i\}) = v(C \cup \{u_j\})$ $\forall C \subseteq \mathcal{U}$ such that $\{u_i, u_j\} \cap C = \emptyset$.
- **Linearity** with respect to the value function v.
- **Null player**: $\phi_i = 0$ if $v(C \cup \{u_i\}) = v(C)$ $\forall C \subseteq \mathcal{U}$ such that $u_i \notin C$.

For the moment, Shapley values are the only contribution measures that have all these four properties combined.

4. Methods

4.1. Datasets, Models and Training Parameters

Datasets. The benchmark dataset used in this article is the *Adult Income dataset*[1]. It consists of 8 categorical features, 6 numerical (continuous) features and one binary label y. The continuous features are normalized to have zero mean and unit variance. The dataset is composed of 48'842 samples and it is split randomly into a train (n=34'189) and a test dataset (n=14'651). Each categorical feature x_i is embedded in a space of dimension $\lceil \frac{k_i}{2} \rceil$, where k_i is the number of categories of feature x_i. We then test the approach on a unique real-world medical dataset from the 2014-16 West African Ebola epidemic [2]. It exemplifies the scenario of disincentivzed data sharing outlined in this article which is summarized in the aforementioned WHO consultation on data sharing in health emergencies (Goldacre et al., 2003). It comprises clinical tabular data on 8386 patients suspected of Ebola Virus Disease (EVD) that were collected at 12 independent Ebola treatment centers distributed across 3 countries (Sierra Leone, Guinea and Liberia). It is composed of 2 categorical features, 5 numerical features and 65 distinct binary clinical signs and symptoms at admission to the centre. The outcome is a binary categorization of diagnosis (EVD+ vs EVD-) by molecular blood test (RTPCR). The data was collated into a central public repository by the Infectious Disease Data Observatory (IDDO). This collation took several years to complete and several datasets are still not represented, thus showing the need for real time incentivized non-datasharing collaboration strategies.

Models and parameters. The model architecture used for these two binary prediction tasks is a simple fully connected neural network with three hidden layers of sizes $64, 16$ and 4 for the adult income dataset (activated with ReLU) and two hidden layers of sizes 128 and 64 for the Ebola dataset (activated with Tanh). For both tasks, respectively, the binary cross-entropy loss is used along with 1 and 5 epochs of mini-batch stochastic gradient descent as the local update functions $\mathcal{F}_i$ (batch size of 32). The outputs are classified as true if they are above a threshold τ. Finally, the aggregation function $\mathcal{A}$ is chosen to be a static average, and the training is stopped after 100 rounds of either FL or decentralized learning. This number of rounds is large enough so that all the CMs can converge and the models can reach an acceptable performance level.

4.2. Contribution Measures in FL

Four different contribution measures are tested for the federated learning setting. For scalability reasons, the metric used as the value function of a specific coalition $C \subseteq \mathcal{U}$ is the negative testing loss $v(C) = -\mathcal{L}(\mathbf{w}_C^r, \mathcal{D}^{te})$. Indeed, the empirical loss is the only metric that is present in all machine learning tasks (classification, regression, etc.). Let $\mathbf{w}_C^r$ denote the model at round r that was only trained and aggregated using users in coalition $u_i \in C \subseteq \mathcal{U}$ during rounds 0 to r. In other words, $\mathbf{w}_C^r$ is equivalent to $\mathbf{w}^r$ if $\mathcal{U} = C$. To avoid confusion, note the distinction between $\mathbf{w}_{u_i}^r$, which is the model who was trained only with $\mathcal{D}_i^{tr}$

[1] https://archive.ics.uci.edu/ml/datasets/adult.

[2] https://www.iddo.org/ebola/data-sharing/accessing-data.

Table 1. Ebola Dataset Statistics. N represents the number of patient, PR the Ebola positive rate and MR the fraction of male.

Location	N	PR	MR	Median Age
Donka	1975	0.379	0.578	29
Guéckédou	1517	0.900	0.477	31
Kalihun	1173	0.726	0.529	27
Makeni	848	0.207	0.518	28
Foya	564	0.798	0.512	30
Bong	529	0.317	0.531	31
Bo	519	0.848	0.511	36
Port-Loko	477	0.379	0.539	30
Kerry-Town	275	0.956	0.447	25
Kambia	217	0.216	0.516	29
Magburaka	155	0.290	0.581	30
Nzérékoré	137	0.577	0.489	28

during rounds 0 to r, and $\mathbf{w}_i^r$ which is the local update of $\mathbf{w}^{r-1}$ using $\mathcal{D}_i^{tr}$. If $C \subset \mathcal{U}$, the model $\mathbf{w}_C^r$ is trained in parallel by users in C. The four tested CMs are given below:

Shapley value with retraining (SV):

$$\phi_i^r = \sum_{\substack{C \subseteq \mathcal{U} \\ u_i \in C}} K_C \left[\mathcal{L}(\mathbf{w}_{C\setminus u_i}^r, \mathcal{D}^{te}) - \mathcal{L}(\mathbf{w}_C^r, \mathcal{D}^{te}) \right] \quad \text{(SV)}$$

Shapley value during aggregation (SVa):

$$\hat{\phi}_i^r = \sum_{\substack{C \subseteq \mathcal{U} \\ u_i \in C}} K_C \left[\mathcal{L}(\hat{\mathbf{w}}_{C\setminus u_i}^r, \mathcal{D}^{te}) - \mathcal{L}(\hat{\mathbf{w}}_C^r, \mathcal{D}^{te}) \right] \quad \text{(SVa)}$$

Marginal loss with retraining (ML):

$$\theta_i^r = \mathcal{L}(\mathbf{w}_{\mathcal{U}\setminus u_i}^r, \mathcal{D}^{te}) - \mathcal{L}(\mathbf{w}_{\mathcal{U}}^r, \mathcal{D}^{te}) \quad \text{(ML)}$$

Marginal loss during aggregation (MLa):

$$\hat{\theta}_i^r = \mathcal{L}(\hat{\mathbf{w}}_{\mathcal{U}\setminus u_i}^r, \mathcal{D}^{te}) - \mathcal{L}(\hat{\mathbf{w}}_{\mathcal{U}}^r, \mathcal{D}^{te}) \quad \text{(MLa)}$$

where $\hat{\mathbf{w}}_C^r := \mathcal{A}\left(\{\mathbf{w}_i^r\}_{u_i \in C}\right)$ and

$$K_C := \frac{1}{|C|} \binom{N}{|C|}^{-1}.$$

The main properties of these CMs are sumarized in Table 2, where N_{models} is the number of models that need to be trained per user, and N_{agg} the number of model aggregations on the server. A few practical remarks need to be made. Firstly, SV and ML are highly inefficient since they require the training of additional models, they are presented here for the sake of comparison and are not feasible in practice. The differences between the CMs that are computed at aggregation (MLa and SVa) and the full CMs (ML and SV) are explored in Section 5.3. Secondly, we assume in this work that the server S has access to the test datasets of all users so it can compute the losses $\mathcal{L}(\cdot, \mathcal{D}_C^{te})$ for all $C \subseteq \mathcal{U}$. In practice, however, the privacy of the test datasets could be preserved using Algorithm 3, where the server only needs to know the dataset sizes $\{|\mathcal{D}_i^{te}|\}_{u_i \in \mathcal{U}}$.

Algorithm 3: FEDERATED LOSS COMPUTATION

Input: Dataset sizes $\{|\mathcal{D}_i^{te}|\}_{u_i \in \mathcal{U}}$ and model $\mathbf{w}$.
Output: Empirical loss $\mathcal{L}(\mathbf{w}, \mathcal{D}_C^{te})$.

for $u_i \in C$:
  S sends $\mathbf{w}$ to user u_i
  u_i computes $\hat{\mathcal{L}}_i := |\mathcal{D}_i^{te}| \cdot \mathcal{L}(\mathbf{w}, \mathcal{D}_i^{te})$
  u_i sends $\hat{\mathcal{L}}_i$ to S
S computes $\mathcal{L}(\mathbf{w}, \mathcal{D}_C^{te}) = \sum_{u_i \in c} \frac{1}{|\mathcal{D}_i^{te}|} \hat{\mathcal{L}}_i$
return $\mathcal{L}(\mathbf{w}, \mathcal{D}_C^{te})$

Table 2. Contribution Measures in FL, where *SV* denotes Shapley Value, *ML* denotes Marginal loss either during aggregation (*a*) or with retraining

Name	$\mathbf{N}_{models}$	$\mathbf{N}_{agg}$	Symbol
SV	2^{N-1}	$2^N - N - 1$	ϕ_i
SVa	1	$2^N - N - 1$	$\hat{\phi}_i$
ML	N	$N + 1$ (1 if $N = 2$)	θ_i
MLa	1	$N + 1$ (1 if $N = 2$)	$\hat{\theta}_i$

Lastly, neither of $\phi_i, \hat{\phi}_i, \theta_i$ or $\hat{\theta}_i$ is properly normalized. Their scale depends mainly on the loss function l which makes them less interpretable. Several normalization functions can be used to solve this problem, in particular L^2 normalization, *Softmax* normalization, *Standard* normalization and *Min-Max* normalization.

Evaluation Setup under Heterogeneous Data. There are many different ways to evaluate the contribution measures in the FL setting. In this article, we focus on their dependence on the heterogeneity of the data distributed over clients, in terms of three following properties:

- Local dataset sizes (SZ),
- Label noise (YN),
- Feature noise (XN).

For all baseline analyses in the Adult Income dataset, the number of users is set to $N = 4$ in order to facilitate the

comparison. A larger number of users ($N = 12$) are then tested in the Ebola dataset. The three experiment setups used to test the CMs are summarized in Table 3. (SZ) represents heterogeneous dataset size per client. For added label noise (YN), η_y represents the label accuracy (i.e. $\eta_y = 0.8$ represents a dataset where 20% of the labels have been randomly switched). For different levels of feature noise (XN) on each client, σ_x represents the standard deviation of the Gaussian noise that is added to all continuous features. In order to keep the data properly normalized (zero mean, unit variance), the noisy version of feature x^i_{cont} is obtained as follows:

$$\hat{x}^i_{cont} = \frac{x^i_{cont} + Z}{\sqrt{1+\sigma_x^2}}, \qquad Z \sim \mathcal{N}(0, \sigma_x^2).$$

Table 3. Overview of the experiments on the Adult Income dataset. Various heterogeneous data distributions are induced in terms of data size per participant (SZ), label noise (YN), and feature noise (XN).

ID		Users				Results	
		u_1	u_2	u_3	u_4	FL	decentr.
SZ	$\frac{\lvert\mathcal{D}_i\rvert}{\lvert\mathcal{D}\rvert}$	0.4	0.3	0.2	0.1	Fig. 3	Fig. 11 (Appendix)
	η_y	1	1	1	1		
	σ_x	0	0	0	0		
YN	$\frac{\lvert\mathcal{D}_i\rvert}{\lvert\mathcal{D}\rvert}$	0.25	0.25	0.25	0.25	Fig. 9 (Appendix)	Fig. 4
	η_y	1.0	0.98	0.96	0.94		
	σ_x	0	0	0	0		
XN	$\frac{\lvert\mathcal{D}_i\rvert}{\lvert\mathcal{D}\rvert}$	0.25	0.25	0.25	0.25	Fig. 10 (Appendix)	Fig. 12 (Appendix)
	η_y	1	1	1	1		
	σ_x	0	0.1	0.2	0.3		

4.3. Contribution Measures in Decentralized Training

A novel idea to adapt the CMs to a fully decentralized setting is to make each user act as a central server. As such, each peer will have its own personalized set of contribution measures and it will therefore be able to see which peer is of most value to them. However, since not all users are neighbors in the communication graph, it becomes totally impractical to train several models in parallel. This is why only SVa and MLa are tested for the decentralized setting. Let $\hat{\phi}^r_{ij}$ and $\hat{\theta}^r_{ij}$ be the CMs of user u_j from the point of view of user u_i, computed using Equations (5) and (6), respectively.

$$\hat{\phi}^r_{ij} := \sum_{\substack{c \subseteq U_i \cup u_i \\ u_j \in C}} \bar{K}_{C,i} \left[\mathcal{L}(\bar{\mathbf{w}}^r_{C \setminus u_j}, \mathcal{D}^{te}_i) - \mathcal{L}(\bar{\mathbf{w}}^r_C, \mathcal{D}^{te}_i) \right], \tag{5}$$

$$\hat{\theta}^r_{ij} := \mathcal{L}(\bar{\mathbf{w}}^r_{U_i \setminus u_j}, \mathcal{D}^{te}_i) - \mathcal{L}(\bar{\mathbf{w}}^r_{U_i}, \mathcal{D}^{te}_i), \tag{6}$$

where $\bar{\mathbf{w}}^r_C := \mathcal{A}\left(\{\tilde{\mathbf{w}}^r_j\}_{u_j \in C}\right)$ and

$$\bar{K}_{C,i} := \frac{1}{|C|} \binom{|U_i|+1}{|C|}^{-1}.$$

5. Results

5.1. Contribution Measures in FL

Figure 2 displays the history of the different contribution measures for the experiment that leverages the effect of the dataset sizes on the CMs. Several observations can be made:

- All CMs converge during training, but those that are computed using Shapley values (SV and SVa) tend to take more time to reach their limit. This is quite intuitive since in order to improve its SV or SVa, one must be beneficial to all coalitions, and some of them might have tasks that are substantially different, even in the case of uniform dataset splitting. On the other hand, to improve its ML or MLa, it is sufficient to be beneficial only to the grand coalition.
- As expected, the users with bigger datasets end up having larger contributions, but the scale of the differences depends on the CM. When using models that are retrained from scratch (SV and ML), the difference is more significant. This comes from the fact that for SVa and MLa, the models that are used to compute the marginal losses are not completely decoupled (see section 5.3). This makes it more sensitive to the noise generated by the randomness of SGD, and it is therefore the main drawback of using marginal aggregated models instead of marginal retrained models.
- Although it is sufficient to be beneficial only to the grand coalition for MLa and ML, it is easier to be beneficial to smaller ones, but these small contributions are neglected when Shapley Values are not used. This explains why the CMs based on marginal losses rather than Shapley Values tend to be negative.

Since in practice, only the CMs computed at aggregation (SVa and MLa) will be used (due to the computational overhead resulting from training several models), and since these two are sensitive to noise, the use of a low-pass filter over several rounds can be beneficial (i.e. moving average, exponential moving average, etc.). Moreover, if a normalizing function f is used, one can also consider the cumulative CMs:

$$\bar{\xi}^r_i = f\left(\sum_{t=1}^{r} \xi^r_i\right),$$

where ξ is either $\phi, \hat{\phi}, \theta$ or $\hat{\theta}$.

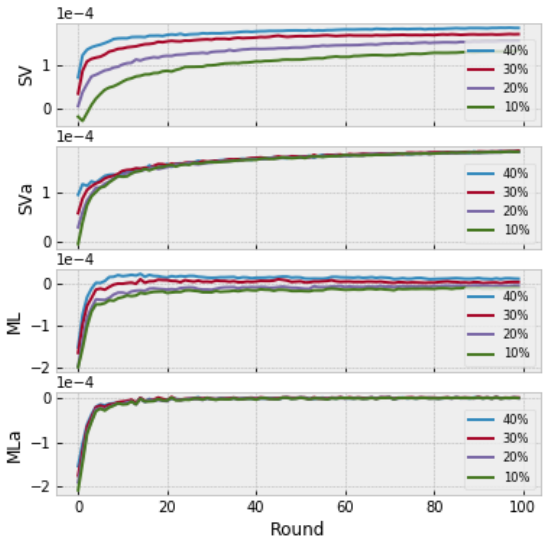

Figure 2. Different CMs (unnormalized) during training obtained in the dataset size experiment (SZ) as defined in Table 3. Each graph represents a contribution measure outlined in Table 2. Lines in each graph represent the four participants with dataset sizes of 10%, 20%, 30%, and 40% (= 100%).

Contribution measures for FL in data size variation (SZ). Figure 3 displays the normalized cumulative CMs obtained for the experiment that concerns the dataset sizes. The violin plots represent the distribution of the values over 100 rounds.

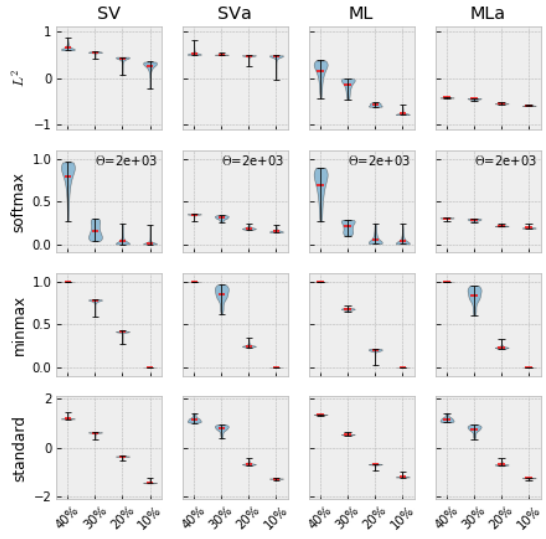

Figure 3. Different normalization strategies (rows) of the cumulative contribution measures (columns, and as detailed in Table 3) obtained in the dataset size experiment (SZ), where the four points in each graph represent the four participants with dataset sizes of 10%, 20%, 30%, and 40% (= 100%)

One can observe that the cumulative CMs yield different results with certain normalization strategies, which may be more intuitive to differentiate the expected contributions between participants. For instance, with the Min-Max, SVa and MLa tend to switch the order of contribution between u_1 and u_2. Indeed, after a certain number of rounds, the noise becomes dominant. Then, user u_2 is more often than u_1 the maximum of all contributions, even if it is by a negligible margin. Additionally, the cumulative contribution ensures that the work of the peers is not lost over time.

Contribution measures for FL in label (YN) and feature noise (XN). The results for (YN) in Figure 9 are in total accordance with intuition. However, the experiment with the feature noise (XN) yielded results that look random at first (see Figure 10). This is probably due to the task at hand (i.e. predicting if a individual earns more that 50K/year). Here, the categorical features such as occupation and workclass are more informative than continuous features, and were not altered in the perturbation.

5.2. Contribution Measures in Personalized Decentralized Learning

For the fully decentralized setting, only the complete communication graph is considered, where each user is connected to every other peer. This is because only the personalized CMs SVa and MLa are of interest in this work (i.e. from the point of view of a specific user). As differences in normalisation were already assessed in the FL setting (and do not behave differently in the decentralized setting) only standard normalization is shown here to reduce redundancy.

Figure 5 shows the temporal development of different contribution measures over rounds that are computed by user u_1 (i.e. the user that has the best dataset in all three experiments and shown cumulatively in the first column of Figure 4). Convergence is more or less reached in all scenarios, but some experiments have more noise than others.

When the CMs are normalized, as observed in Figure 4, the results obtained in the experiment with label noise (YN) are in accordance with expectations, suggesting that the adaptation to the fully decentralized setting does not alter the main properties of the CMs. However, this adaptation comes with an inevitable cost. Indeed, the value function that is used to compute MLa and SVa now depends on the users' datasets. One can observe, for instance, that the peer with most label noise (label accuracy of 94%) is least capable of distinctively ordering the other peers' contributions. Nevertheless, it is still able to detect that the user with 100% label accuracy contributes more to its personalized model than itself, which is remarkable, especially given that this is performed on unseen data.

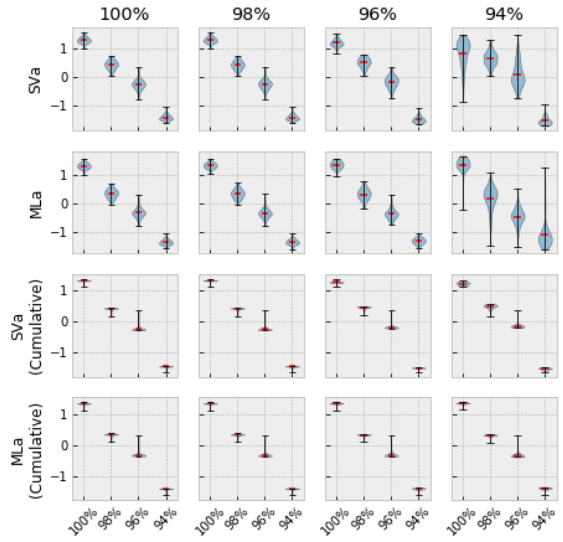

Figure 4. Different decentralized contribution measures (CMs, rows) obtained in the label noise experiment (YN). Each subplot represents proportional contributions of the four participants which differ by percentage label noise. Column i represents the CMs that are computed according to u_i. All CMs are displayed with standard normalization.

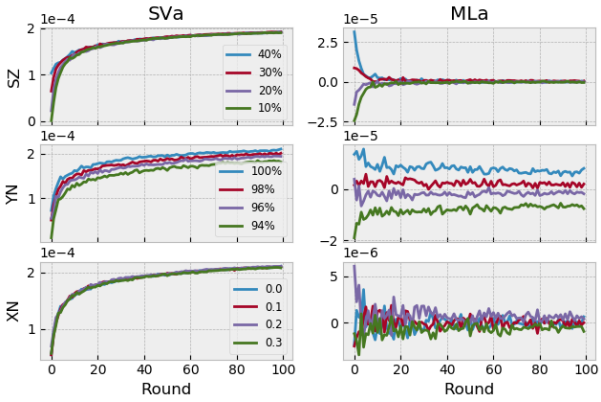

Figure 5. Training history of the different decentralized (unnormalized) CMs obtained in the different experiments.

The experiment with the dataset sizes (SZ) yields similar results for which the same conclusions can be drawn. However, as in the FL setting, the experiment where noise is added to the continuous features (XN) is not conclusive, probably for the same reasons as in the FL setting. For simplicity, this work does not address categorical feature shifts.

5.3. CMs at Aggregation vs CMs with Retraining

The marginal losses and Shapley values at aggragation (MLa and SVa) do not represent an exact marginal loss. Indeed, the model $\hat{\mathbf{w}}^r_{c\setminus u_i}$ contains some information about the dataset $\mathcal{D}_i$ due to the local updates in rounds 0 to $r-1$. In order to understand the difference between the model obtained by retraining without one specific user and the model obtained by excluding the same user at aggregation, consider the following federated learning study case where the random data points (X, Y) are in $\mathbb{R}^2$. Consider the two users scenario ($\mathcal{U} = \{u_1, u_2\}$) and let the data of users u_1 and u_2 be distributed as follows:

$$(X,Y)_{u_i} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \sigma_x^2 & (-1)^i\rho\sigma_x\sigma_y \\ (-1)^i\rho\sigma_x\sigma_y & \sigma_y^2 \end{bmatrix}\right), \quad (7)$$

where $\rho \in]0,1[$. Using the mean-squared error loss (MSE) $l(y,\hat{y}) = (y-\hat{y})^2$, the problem of user u_i can be expressed as follows:

$$\text{Find } g_i^\star(x) = \arg\min_g \mathbb{E}_{u_i}[l(Y, g(X))|X = x],$$

which has the elegant and intuitive solution

$$g_i^\star(x) = \mathbb{E}_{u_i}[Y|X = x].$$

Using the distributions (7), the optimal estimators are the simple linear functions

$$g_i^\star(x) = (-1)^i\rho\frac{\sigma_y}{\sigma_x}x, \qquad i = 1, 2.$$

Hence, by considering linear models of the form $g(x) = \alpha x + \beta$, the optimal parameters are given by

$$\mathbf{w}_i^\star := (\alpha_i^\star, \beta_i^\star) = ((-1)^i\varepsilon, 0),$$

where $\varepsilon := \rho\frac{\sigma_y}{\sigma_x}$ quantifies the distances between the task of u_1 and u_2. Suppose that the features are normalized (i.e. $\sigma_x = 1$). The expected value of the loss is therefore given by

$$\mathbb{E}_{u_i}\left[l(Y, g(X; \mathbf{w}))\right] = \alpha^2 - 2(-1)^i\varepsilon\alpha + \beta^2 + \sigma_y^2.$$

Finally, assume that both u_1 and u_2 have enough data points to compute the exact expected gradients

$$\mathbb{E}_{u_i}\left[\nabla_{\mathbf{w}} l(Y, g(X; \mathbf{w}))\right] = 2\left(\mathbf{w} - \mathbf{w}_i^\star\right).$$

Consider the scenario in which the central server initializes the model at $\mathbf{w}^0 = (0, \beta^0)$ (which is already a good guess since it is equidistant from the optimal models $\mathbf{w}_1^\star$ and $\mathbf{w}_2^\star$), and in which the local updates $\mathcal{F}_i$ are given by the simple gradient descent

$$\mathcal{F}_i(\mathbf{w}) = \mathbf{w} - \gamma E_{u_i}\left[\nabla_{\mathbf{w}} l(y, g(x; \mathbf{w}))\right],$$

With these considerations, the learning algorithm is graphically represented in Figure 6. The exact expressions of the

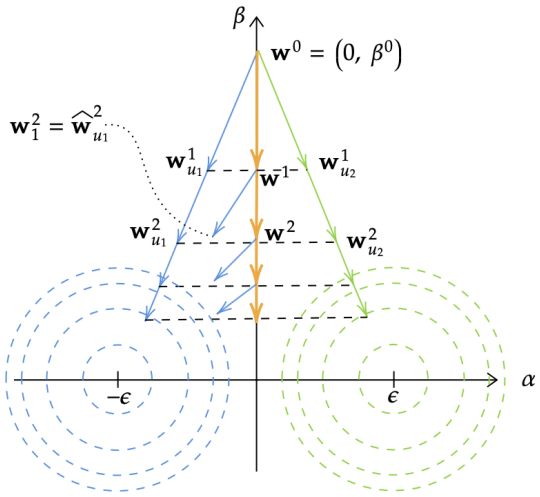

Figure 6. Gradient descent of both users and aggregation at the central server. The objective functions of user u_1 and u_2 are displayed in blue and green, respectively.

different models at each rounds are given by the recursive formulas:

$$\begin{aligned}\mathbf{w}_{u_i}^r &= \mathbf{w}^0 - 2\gamma \sum_{t=0}^{r-1} (\mathbf{w}_{u_i}^t - \mathbf{w}_i^\star),\\ \hat{\mathbf{w}}_{u_i}^r &= \mathbf{w}^{r-1} - 2\gamma(\mathbf{w}^{r-1} - \mathbf{w}_i^\star)\\ \mathbf{w}^r &= \frac{1}{2}(\hat{\mathbf{w}}_{u_1}^r + \hat{\mathbf{w}}_{u_2}^r)\end{aligned}$$

which are derived by a graphical analysis of Figure 6. Assuming that the probabilities that a data point comes from user u_1 or user u_2 are the same, the joint objective function can be expressed as

$$\begin{aligned}\mathcal{L}(\mathbf{w}) &:= \mathbb{E}_{u_1,u_2}[l(Y, g(X;\mathbf{w})]\\ &= \frac{1}{2}(\mathbb{E}_{u_1}[l(Y, g(X;\mathbf{w})] + \mathbb{E}_{u_2}[l(Y, g(X;\mathbf{w})])\\ &= \alpha^2 + \beta^2 + \sigma_y^2.\end{aligned}$$

which is optimized by the model $\mathbf{w}^\star = (0, 0)$. Since the learning algorithm is invariant to scaling (linear gradient), it can be characterised by ε, the step size γ and the ratio $\frac{\beta^0}{\varepsilon}$. In addition, the loss function needs the value of the absolute correlation $\rho \in]0, 1[$ to be fully characterised.

From there, several observations can be made:

- Assuming that $\gamma < 1$ (i.e. the algorithm converges):

$$\begin{aligned}\lim_{r\to\infty} \frac{\theta_i^r}{\hat{\theta}_i^r} &= \lim_{r\to\infty} \frac{\mathcal{L}(\mathbf{w}_{u_j}^r) - \mathcal{L}(\mathbf{w}^r)}{\mathcal{L}(\hat{\mathbf{w}}_{u_j}^r) - \mathcal{L}(\mathbf{w}^r)}\\ &= \frac{(\varepsilon^2 + \sigma_y^2) - \sigma_y^2}{(4\gamma^2\varepsilon^2 + \sigma_y^2) - \sigma_y^2} = \frac{1}{4\gamma^2}\end{aligned}$$

  This means that when γ is small, the ratio between the marginal loss with retraining and the marginal loss at aggregation explodes. This makes it more sensitive to the random nature of the stochastic gradient descent, as observed in Figure 2.

- At convergence, the value of the marginal loss with retraining does not depend on γ:

$$\frac{\partial}{\partial\gamma}(\lim_{r\to\infty} \theta_i^r) = 0.$$

  However, its rate of convergence remains γ dependent. This means in particular that MLa and SVa might behave unexpectedly when using adaptive gradient descents like Adadelta, Adagrad or Adam, among others. This is in part why only the standard SGD is used in this work.

- Instead of looking at the difference between $\mathcal{L}(\mathbf{w}_{u_j}^r)$ (or $\mathcal{L}(\hat{\mathbf{w}}_{u_j}^r)$) and $\mathcal{L}(\mathbf{w}^r)$, one could also look at their ratio. In this simple study case, these ratios would converge to

$$\begin{aligned}\lim_{r\to\infty} \frac{\mathcal{L}(\mathbf{w}_{u_j}^r)}{\mathcal{L}(\mathbf{w}^r)} &= \frac{\varepsilon^2 + \sigma_y^2}{\sigma_y^2} = \rho^2 + 1,\\ \lim_{r\to\infty} \frac{\mathcal{L}(\hat{\mathbf{w}}_{u_j}^r)}{\mathcal{L}(\mathbf{w}^r)} &= \frac{4\gamma^2\varepsilon^2 + \sigma_y^2}{\sigma_y^2} = 4\gamma^2\rho^2 + 1.\end{aligned}$$

  The scale difference between the two ratios is therefore greatly reduced due to the additional term $+1$, but the value of the second ratio remains γ dependent. However, since the absolute correlation ρ could be estimated using simple statistics on the datasets, γ could be tuned to minimize this scale difference.

One must nonetheless remain aware that these remarks are specific to this study case and may not be valid for other loss functions like maximum absolute error (MAE) or binary cross-entropy (BCE).

5.4. Ebola Dataset

We conclude with an example of how contribution measures may be visualised and interpreted in a unique real world example that inspired the assumptions of our setting. The Ebola dataset is naturally fragmented across collection sites which differ in geography, epidemiology, patient demographics, and data collection practices. It was collated retrospectively and semantic alignment was performed by a third party several years after acquisition. Thus, these datasets have a high risk of hidden label and feature biases along with the known differences in data size that are detailed in Table 1. We split the datasets according to these real world divisions and build a set of personalised P2PL

models that discriminate the diagnosis of Ebola virus disease. Figure 7 shows the model performance as area under the receiver operating curve (AUROC) for each participant, revealing clear differences in the predictive potential of each dataset.

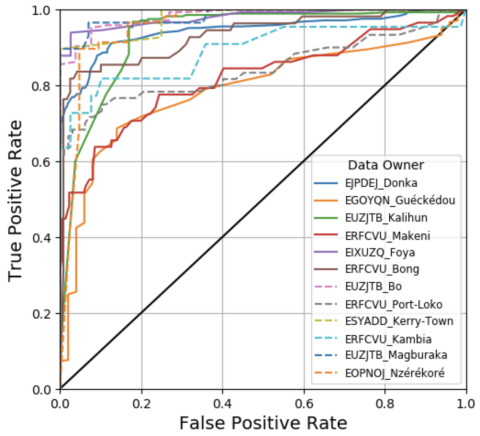

Figure 7. Area under the Receiver operating characteristic (AUROC) curve of each of the 12 data owners in the Ebola data repository of the Infectious Disease Data Observatory (IDDO).

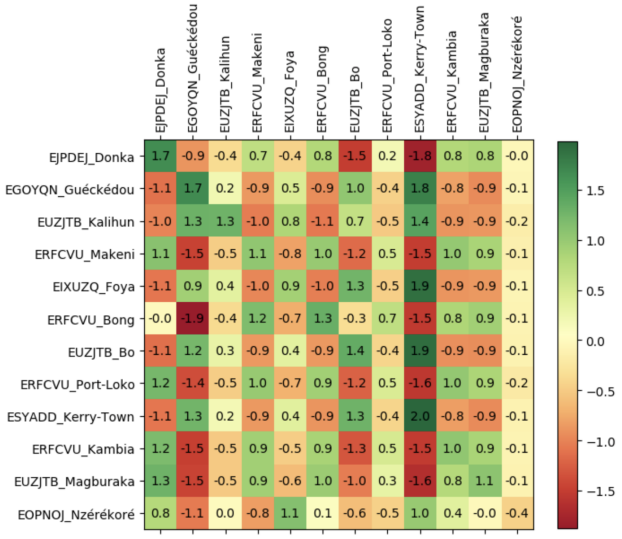

Figure 8. Cumulative MLa with standard normalization of the different data owners in the Ebola experiment. One row corresponds to a personalized set of CMs and one column represents the contribution of one particular user to everyone (including itself, visible on the diagonal).

Figure 8 shows the contribution measures (MLa) with standard normalization of the twelve data owners with respect to one another. Looking at the diagonal, we can compare the self-contributions of each user, where some (e.g. Kerrytown) strongly favor a local model, whereas Nzérékoré benefits more from the coalition than its own data. Another example is Kalihun, which seems to systematically differ from other datasets, as seen by the low contribution measures (yellow/red hues) down its column and it thus has a low contribution measure to all other models. Indeed, this dataset seems to be extreme in its large size and high percentage of Ebola cases as seen in Table 1 and thus, these contribution measures may reflect its outlier status.

6. Discussion

6.1. Limitations

This work proposes several methods to extract and visualize the individual contributions of a users' data to the performance of a collaborative model learned in federated and P2P settings. It also explores contribution measures when learning personalised decentralised models, which could have the useful application of being measures of similarity.

For clarity, this proof of concept study is limited to a single scenario inspired by (and tested on) a real world scenario of collaborative model building in public health emergencies. This work could be greatly expanded in future studies to better generalize the results to other learning scenarios, data types and distributions. Some examples include changing the number of users, altering the categorical features, feature-wise perturbations (one feature at a time), exploring different decentralized topologies and aggregation functions, as well as varying the update and loss functions).

A more specific aspect that can be improved is the normalization of the CMs. Indeed, the four functions that are presented in this article are fairly general and and it may be interesting to test how the normalisation strategies align with perceived contributions and other measures of similarity. Normalization strategies are easily interchangeable and can be selected differently for each task.

6.2. Future work

In addition to the suggested explorations listed above, future work should include the development of a system that allows users to willingly share some statistics about their data so as to guide other users in their data collection and preprocessing practices towards more impactful contributions. For example, consider a scenario where several users participate in a collaborative learning task that aims to recognize hand-written digits (like the MNIST dataset), but each user has its own handwriting (i.e. disparities of type 3). Suppose now that one user u has a significantly higher contribution than others, which suggests that its handwriting is easily readable. The goal would be to create a system that enables all peers to ask user u some statistic about its data so that they can try to adapt their handwriting and thus improve their own contribution and their own model.

The contribution measure could also be used in the aggregation functions $\mathcal{A}_i$ as a way to dynamically average the models, but care should be taken with to avoid the creation of an infinite learning loop. Indeed, as of now, the CMs presented in this work depend heavily on $\mathcal{A}_i$. It is therefore probably better to keep the CMs separate from the learning algorithm.

6.3. Conclusion

This work attempts to create an incentivization to participate constructively in collaborative learning by creating transparency on the individual contributions of users in a coalition. We hypothesize that visualising this information provides guidance on how users can increase their access to collaborative insights which has reciprocal benefits for other users in the collaboration.

Acknowledgements

This Research includes data provided by the Ebola Data Platform hosted by the Infectious Diseases Data Observatory, and the following data contributors, Alliance for International Medical Action (ALIMA), International Medical Corps (IMC), Institute of Tropical Medicine Antwerp (ITM), Médecins Sans Frontières (MSF), Oxford University, and Save the Children (SCI), who had no role in the production of these research outputs. The preprocessing of the Ebola dataset was a separate project that was done by David Roshewitz from the EPFL Machine Learning and Optimization laboratory.

References

Agrawal, R. and Srikant, R. Privacy-preserving data mining. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, SIGMOD '00, pp. 439–450, New York, NY, USA, 2000. Association for Computing Machinery. ISBN 1581132174. doi: 10.1145/342009.335438. URL https://doi.org/10.1145/342009.335438.

Bellet, A., Guerraoui, R., Taziki, M., and Tommasi, M. Personalized and private peer-to-peer machine learning, 2018.

Bouchra Pilet, A., Frey, D., and Taïani, F. Simple, Efficient and Convenient Decentralized Multi-Task Learning for Neural Networks. working paper or preprint, November 2020. URL https://hal.archives-ouvertes.fr/hal-02373338.

Geiping, J., Bauermeister, H., Dröge, H., and Moeller, M. Inverting gradients – how easy is it to break privacy in federated learning?, 2020.

Gentry, C. Fully homomorphic encryption using ideal lattices. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, pp. 169–178, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585062. doi: 10.1145/1536414.1536440. URL https://doi.org/10.1145/1536414.1536440.

Goldacre, B., Harrison, S., Mahtani, K. R., and Heneghan, C. Background briefing for who consultation on data and results sharing during public health emergencies, 2003. URL https://www.who.int/medicines/ebola-treatment/background_briefing_on_data_results_sharing_during_phes.pdf.

He, L., Bian, A., and Jaggi, M. COLA: decentralized linear learning. In *NeurIPS 2018 - Advances in Neural Information Processing Systems*, pp. 4541–4551, 2018.

Huang, J., Talbi, R., Zhao, Z., Boucchenak, S., Chen, L. Y., and Roos, S. An exploratory analysis on users' contributions in federated learning, 2020.

Infectious Disease Data Observatory (IDDO). Ebola Data Platform. URL https://www.iddo.org/research-themes/ebola.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Eichner, H., Rouayheb, S. E., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., Gruteser, M., Harchaoui, Z., He, C., He, L., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Joshi, G., Khodak, M., Konečný, J., Korolova, A., Koushanfar, F., Koyejo, S., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock, R., Özgür, A., Pagh, R., Raykova, M., Qi, H., Ramage, D., Raskar, R., Song, D., Song, W., Stich, S. U., Sun, Z., Suresh, A. T., Tramèr, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, F. X., Yu, H., and Zhao, S. Advances and open problems in federated learning, 2021.

Kang, J., Xiong, Z., Niyato, D., Xie, S., and Zhang, J. Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory. *IEEE Internet of Things Journal*, 6(6): 10700–10714, 2019. doi: 10.1109/JIOT.2019.2940820.

Kang, J., Xiong, Z., Niyato, D., Yu, H., Liang, Y.-C., and Kim, D. I. Incentive design for efficient federated learning in mobile networks: A contract theory approach, 2019.

Koloskova, A., Loizou, N., Boreiri, S., Jaggi, M., and Stich, S. U. A unified theory of decentralized SGD with changing topology and local updates. In *International Conference on Machine Learning (ICML)*, 2020.

Lian, X., Zhang, C., Zhang, H., Hsieh, C.-J., Zhang, W., and Liu, J. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017a.

Lian, X., Zhang, C., Zhang, H., Hsieh, C.-J., Zhang, W., and Liu, J. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017b. URL https://proceedings.neurips.cc/paper/2017/file/f75526659f31040afeb61cb7133e4e6d-Paper.pdf.

Lim, W. Y. B., Xiong, Z., Miao, C., Niyato, D., Yang, Q., Leung, C., and Poor, H. V. Hierarchical incentive mechanism design for federated machine learning in mobile networks. *IEEE Internet of Things Journal*, 7(10):9575–9588, 2020. doi: 10.1109/JIOT.2020.2985694.

Liu, B., Yan, B., Zhou, Y., Wang, J., Liu, L., Zhang, Y., and Nie, X. A real-time contribution measurement method for participants in federated learning. *CoRR*, abs/2009.03510, 2020. URL https://arxiv.org/abs/2009.03510.

Mansour, Y., Mohri, M., Ro, J., and Suresh, A. T. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

McMahan, H. B., Moore, E., Ramage, D., and y Arcas, B. A. Federated learning of deep networks using model averaging. *CoRR*, abs/1602.05629, 2016. URL http://arxiv.org/abs/1602.05629.

Mohassel, P. and Zhang, Y. Secureml: A system for scalable privacy-preserving machine learning. *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 19–38, 2017.

Nadiradze, G., Amirmojtaba Sabour, Peter Davies, I. M. S. L., and Alistarh, D. Decentralized SGD with asynchronous, local and quantized updates. *arXiv preprint arXiv:1910.12308*, 2019.

Pandey, S. R., Tran, N. H., Bennis, M., Tun, Y. K., Manzoor, A., and Hong, C. S. A crowdsourcing framework for on-device federated learning. *IEEE Transactions on Wireless Communications*, 19(5):3241–3256, 2020. doi: 10.1109/TWC.2020.2971981.

Rodríguez-Barroso, N., Stipcich, G., Jiménez-López, D., Ruiz-Millán, J. A., Martínez-Cámara, E., González-Seco, G., Luzón, M. V., Veganzones, M. A., and Herrera, F. Federated learning and differential privacy: Software tools analysis, the sherpa.ai fl framework and methodological guidelines for preserving data privacy. *Information Fusion*, 64:270–292, Dec 2020. ISSN 1566-2535. doi: 10.1016/j.inffus.2020.07.009. URL http://dx.doi.org/10.1016/j.inffus.2020.07.009.

Saroiu, S., Gummadi, K. P., and Gribble, S. Measuring and analyzing the characteristics of napster and gnutella hosts. *Multimedia Syst.*, 9:170–184, 08 2003. doi: 10.1007/s00530-003-0088-1.

Shapley, L. S. *17. A Value for n-Person Games*, pp. 307–318. Princeton University Press, 2016. doi: doi:10.1515/9781400881970-018. URL https://doi.org/10.1515/9781400881970-018.

Song, T., Tong, Y., and Wei, S. Profit allocation for federated learning. In *2019 IEEE International Conference on Big Data (Big Data)*, pp. 2577–2586, 2019. doi: 10.1109/BigData47090.2019.9006327.

Tsitsiklis, J., Bertsekas, D., and Athans, M. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31(9):803–812, 1986. doi: 10.1109/TAC.1986.1104412.

Vanhaesebrouck, P., Bellet, A., and Tommasi, M. Decentralized collaborative learning of personalized models over networks, 2017.

Vogels, T., Karimireddy, S. P., and Jaggi, M. Powergossip: Practical low-rank communication compression in decentralized deep learning, 2020.

Wang, G., Dang, C. X., and Zhou, Z. Measure contribution of participants in federated learning, 2019.

Wang, J. and Joshi, G. Cooperative SGD: A unified framework for the design and analysis of communication-efficient SGD algorithms. *arXiv preprint arXiv:1808.07576*, 2018.

Wang, T., Zhu, J.-Y., Torralba, A., and Efros, A. A. Dataset distillation, 2020.

Wheatley, S., Maillart, T., and Sornette, D. The extreme risk of personal data breaches & the erosion of privacy. *The European Physical Journal B*, 89, 05 2015. doi: 10.1140/epjb/e2015-60754-4.

Wu, H. and Wang, P. Fast-convergent federated learning with adaptive weighting. *CoRR*, abs/2012.00661, 2020. URL https://arxiv.org/abs/2012.00661.

Yang, T., Andrew, G., Eichner, H., Sun, H., Li, W., Kong, N., Ramage, D., and Beaufays, F. Applied federated learning: Improving google keyboard query suggestions. *CoRR*, abs/1812.02903, 2018. URL http://arxiv.org/abs/1812.02903.

Yao, A. C. How to generate and exchange secrets. In *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*, pp. 162–167, 1986. doi: 10.1109/SFCS.1986.25.

Zhao, B., Mopuri, K. R., and Bilen, H. idlg: Improved deep leakage from gradients, 2020.

Zhao, J., Zhu, X., Wang, J., and Xiao, J. Efficient client contribution evaluation for horizontal federated learning. *CoRR*, abs/2102.13314, 2021. URL https://arxiv.org/abs/2102.13314.

Zhu, L., Liu, Z., and Han, S. Deep leakage from gradients, 2019.

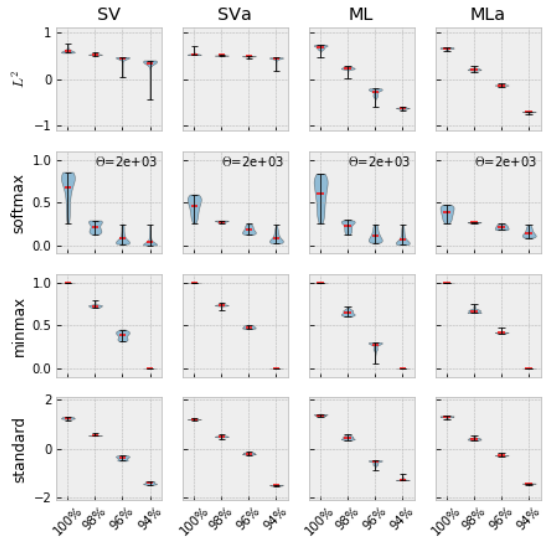

Figure 9. Different normalization strategies (rows) of the cumulative contribution measures (columns, and as detailed in Table 3) obtained in the label noise experiment (YN), where the four points in each graph represent the four participants with label accuracy of 100%, 98%, 96%, and 94%.

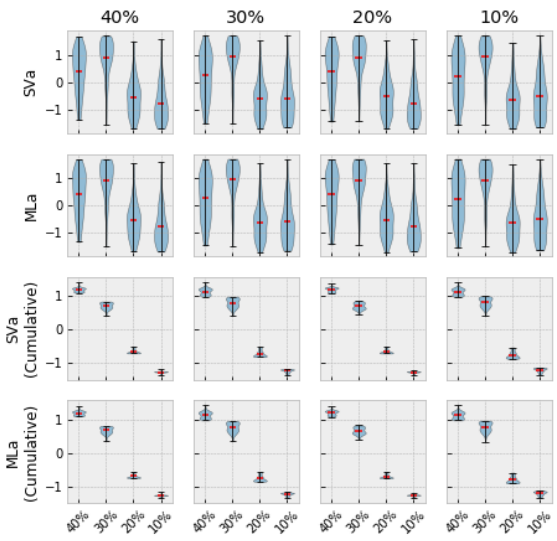

Figure 11. Different decentralized contribution measures (CMs, rows) obtained in the dataset size experiment (SZ). Each subplot represents proportional contributions of the four participants which differ by dataset size. Column i represents the CMs that are computed according to u_i. All CMs are displayed with standard normalization.

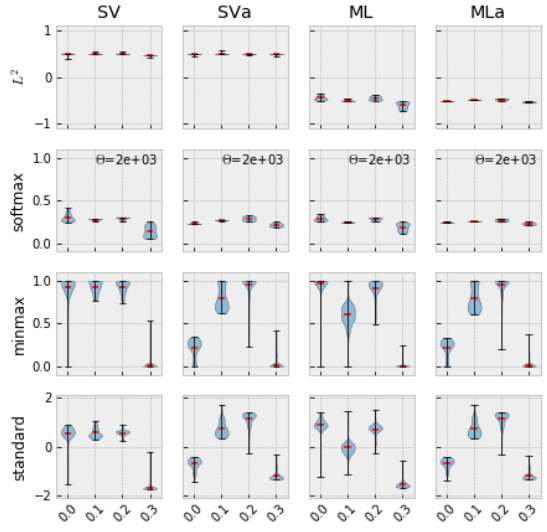

Figure 10. Different normalization strategies (rows) of the cumulative contribution measures (columns, and as detailed in Table 3) obtained in the label accuracy experiment (XN), where the four points in each graph represent the four participants whose features have been altered with white noise of standard deviation 0.0, 0.1, 0.2, and 0.3.

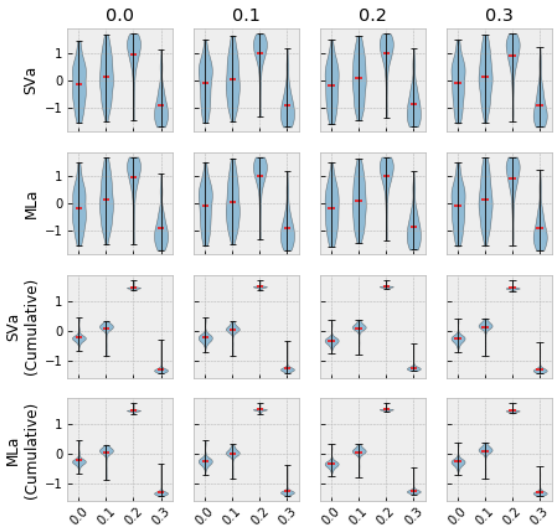

Figure 12. Different decentralized contribution measures (CMs, rows) obtained in the feature noise experiment (XN). Each subplot represents proportional contributions of the four participants which differ by noise variance. Column i represents the CMs that are computed according to u_i. All CMs are displayed with standard normalization.