Secure and efficient decentralized machine learning through group-based model aggregation

1st Brandon A. Mosqueda González
Department of Computer Science
INSA Lyon
Villeurbanne, France
brandon.mosqueda-gonzalez@insa-lyon.fr

2nd Omar Hasan
University of Lyon, CNRS
INSA Lyon, LIRIS, UMR5205, F-69621
France
omar.hasan@insa-lyon.fr

3rd Wisnu Uriawan
University of Lyon, CNRS
INSA Lyon, LIRIS, UMR5205, F-69621
France
wisnu.uriawan@insa-lyon.fr

4th Youakim Badr
Computer Science and Engineering Department
The Pennsylvania State University
Malvern (PA), USA
yb@psu.edu

5th Lionel Brunie
University of Lyon, CNRS
INSA Lyon, LIRIS, UMR5205, F-69621
France
lionel.brunie@insa-lyon.fr

***Abstract*—In the domain of decentralized machine learning, enhancing privacy often comes at the cost of reduced efficiency or utility, and vice versa. Striking a balance between privacy, efficiency, and utility remains a challenge. In this paper, we present the Secure Group-Based Model Aggregation (SGBMA) framework for decentralized learning. SGBMA introduces a novel approach by dividing the set of participants into small groups and employing an efficient secure multiparty computation protocol to aggregate models within the groups. The adoption of a balanced binary tree topology of groups facilitates the seamless combination of models computed in the groups into a unified global model. At each training round, SGBMA achieves equal participation from each user in the global model, equivalent to federated learning. The privacy-efficiency balance can be adjusted with the size of the groups with no impact on model utility. By leveraging SGBMA, decentralized learning can be executed while ensuring privacy and making it applicable to large-scale scenarios. Our experiments show that SGBMA produces higher model utility for Independent and Identically Distributed data (IID) and comparable results as federated learning in non-IID.**

***Index Terms*—Decentralized machine learning, artificial neural networks, privacy-preserving machine learning, secure multi-party computation**

I. Introduction

The vast amount of data that is generated every day has driven the adoption of machine learning techniques for solving complex real-world problems. However, concerns regarding user privacy have arisen alongside these advancements. To address these challenges, privacy-preserving machine learning (PPML) was proposed as a way to leverage data while respecting privacy.

Privacy is a wide term that has different definitions in different domains. In PPML context, privacy refers to the protection of sensitive and personally identifiable information during the process of training machine learning models. It involves implementing techniques and protocols that prevent unauthorized access or disclosure of private data while still allowing the model to learn from it. According to Xu et al. [1], a PPML solution asserts data-oriented privacy if an adversary cannot learn private information directly from input data samples or associate private information with a specific person's identification. Similarly, a PPML solution is said to provide a model-oriented privacy guarantee if and only if an adversary cannot derive any private information from a given model by querying it several times. In this work, we are interested in the model-oriented privacy guarantee because the studied PPML solutions do not share data samples.

One of the most popular PPML approaches is federated learning [2], a framework for training machine learning models directly on remote devices, enabling collaborative model development without sharing raw data. In federated learning, a central server distributes a global model to multiple devices, such as smartphones or edge devices, which then train the model using their local data. The devices only share with the central server the updated model parameters, not the raw data. These parameters are aggregated by the central server, which combines the knowledge from all the devices to create an improved global model. That new version of the global model is shared with the devices to train again with their local data. This process is repeated until a given number of iterations is reached or convergence is achieved.

Distributing the computational cost among all the devices and avoiding data sharing is possible by training on remote devices. Despite its advantages, federated learning relies on a central trusted server for model coordination and aggregation (Fig. 1a), leading to communication bottlenecks and potential privacy risks because of the influence the central entity has over all the devices [3].

The limitations of federated learning motivated the development of fully decentralized learning (or simply

decentralized learning), where there does not exist a central coordinator entity. Instead, each device communicates directly with a subset of devices, its neighbors, with whom it will share its model update and will receive theirs (Fig. 1b). In this way, not only the model update is distributed among the devices but also the aggregation task. For this reason, decentralized learning is more efficient than federated learning because it requires much less communication on the busiest node [4]. This approach offers potential advantages, such as scalability and increased privacy guarantees that arise from full decentralization [5], [6].

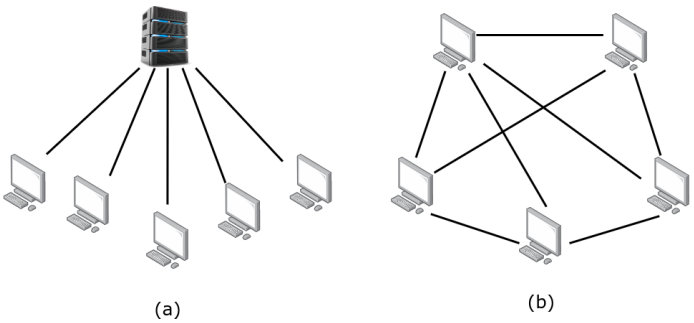

Fig. 1. (a) Federated learning architecture. (b) Decentralized learning architecture.

Recent studies by Geiping et al. [7], Boenisch et al. [3], Wang et al. [8], have shown that even though raw data is not shared in both federated and decentralized learning, it can be reconstructed from model updates that are shared with others. Local updates on the model's parameters can leak more information about the data used to train [9] which compromises model-oriented privacy.

The first state-of-the-art proposals in federated and decentralized learning were mainly intended to improve efficiency and/or utility [10]–[12], that is, the convergence in optimization is achieved with fewer iterations and the final model has high accuracy. When novel attacks on the local model updates were proposed, subsequent studies also focused on protecting model aggregation [6], [13], [14], but each time it is shown that there is a trade-off between privacy, utility, and efficiency.

Increasing privacy comes with a cost in efficiency or even in utility. That is the reason why Pasquini et al. [15], after extensive experiments of different novel attacks on modern decentralized frameworks, mention that current decentralized proposals do not offer any security advantage over federated learning. Strong privacy guarantees in decentralized learning would require denser connected networks, losing any practical advantage over the federated setup on large scales, and therefore completely defeating the objective of the decentralized approach.

In this paper, we propose a decentralized learning framework that uses an efficient Secure Multiparty Computation protocol [14] for model aggregation in small groups of nodes. This protocol enables model-oriented privacy with low computational costs. The groups are arranged in a balanced binary tree topology so that models of the groups can be combined in a global model efficiently. The balanced binary tree topology takes advantage of the parallelization nature of the problem and is used to combine the models from leaf nodes to root node. With this framework, decentralized learning does not require a dense topology to offer model-oriented privacy in the training process.

A. Contributions

The main contributions of this work are summarized below:

- This paper is an extended version of our previous work published in the Blockchain Computing and Applications (BCCA) conference [16]. This paper extends the original paper by significantly more than 50% in contributions as well as length.
- A new decentralized learning framework flexible enough to configure the privacy-efficiency trade-off with no impact on model utility. This is a completely new contribution and considers the original contribution of the BCCA paper as a case study.
- We conduct a comprehensive privacy analysis of the proposed framework, assessing its ability to preserve data privacy and mitigate potential data leakage risks.
- We provide an example that demonstrates the variation of data leakage risk with different values of the only parameter in the framework. This analysis offers valuable insights into the framework's sensitivity to parameter settings and informs best practices for ensuring data security in decentralized settings.
- Experiments on a state-of-the-art benchmark dataset to evaluate the model utility of our approach over existing methods.

B. Outline

This paper is structured as follows: Section II introduces the related literature, Section III addresses the proposal, Section IV explains the experiments conducted to evaluate the proposal, Section V presents the results, Section VI examines a case of study, Section VII addresses the discussion, and finally, in Section VIII, the conclusions are presented.

II. Related work

One of the earliest works in privacy-preserving machine learning using deep neural networks was by McMahan et al. [2]. This paper introduced the concept of federated learning and proposed the FedAvg algorithm for training deep neural networks using decentralized data. In FedAvg, the aggregation of different deep neural network models is done by averaging

the synaptic weights of all models.

Similarly, Lian et al. [4] introduced the Decentralized Parallel Stochastic Gradient Descent (D-PSGD) algorithm, which preserves the main ideas of FedAvg but without the central coordinator entity. In D-PSGD, each node in the network aggregates the models received from its neighbors. The aggregation is also performed taking the average of the synaptic weights. Both federated and decentralized learning have been focused on training deep neural networks, but applications with different machine learning algorithms have been proposed too [17], [18]. We only consider deep neural networks in this work.

Even though data is not shared, private information can be reconstructed or inferred from model updates. Geiping et al. [7] show that is possible to faithfully reconstruct images at high resolution from the knowledge of their parameter gradients. They demonstrate that such a break of privacy is possible even for trained deep networks. Another popular attack is the membership inference attack by Shokri et al. [9]: given a data record and black-box access to a model, determine if the record was in the model's training dataset. The experiments show that even commercial models can be vulnerable to membership inference attacks. These two types of attacks can be considered passive attacks because the attacker does not interfere with the protocol, he only uses the information received from model updates. There are several study cases where active attackers are evaluated, for example, Boenisch et al. [3]. They study the case of an active malicious central server in federated learning who modifies the shared model weights before users compute model gradients. That allows the central server to recover complete user data and exploits the influence of the central server over all nodes.

As new vulnerabilities have been found, new protection mechanisms have been proposed. The vast majority of them try to preserve model-oriented privacy because no data is shared among participants by definition. In Tran et al. [14], for example, a secure aggregation protocol is proposed for decentralized learning based on Secure Multiparty Computation. The protocol solves the privacy problem for the honest-but-curious scenario even with $n - 2$ malicious users. However, it suffers from not being useful in large-scale scenarios because it practically emulates the topology of federated learning and overloads one of the nodes. They achieve privacy with high costs in efficiency.

Another popular approach involves the use of differential privacy which adds controlled noise to model updates so that the original data is masked and therefore protected. Guo et al. [19] uses differential privacy in a decentralized learning context. They enhance a stochastic gradient descent algorithm with differential privacy and topology-aware noise reduction integrating a time-aware noise decay technique. As with other differential privacy solutions, model-oriented privacy is achieved, but it comes with a high cost on model utility. The authors emphasize the importance of identifying the minimal amount of noise that can provide desired privacy protection while maintaining acceptable model performance.

Both differential privacy and secure multiparty computation provide model-oriented privacy in PPML, but differential privacy is most widely used due to its strong information-theoretic guarantees, algorithmic simplicity, and relatively small systems overhead [20]. Secure multiparty computation does not protect against membership inference attacks over the aggregated model, but differential privacy does. The main caveat with differential privacy is that it reduces model utility. The more noise is included in model updates, the more model utility is reduced.

In our proposal, we opted for the use of secure multiparty computation for ensuring model-oriented privacy. Even though this solution is vulnerable to membership attacks, attacks for reconstructing training data can only be performed on aggregated models, so data cannot be directly linked to users. We focus on the honest but curious scenario and leave the investigation of active malicious parties to future work. Our proposal provides privacy under these assumptions, with a configurable impact on efficiency, and an equivalent model utility as federated learning.

While secure aggregation techniques, including secure multiparty computation protocols, are applicable in both federated and decentralized learning settings, it is important to note that these protocols are particularly well-suited for small subsets of nodes. This choice helps mitigate potential increases in computational or communication costs, making them efficient solutions within such contexts. Moreover, our framework stands out by efficiently distributing computational aggregation costs, resulting in minimal propagation times. In contrast, these protocols can become prohibitively expensive in terms of computation and communication when applied to large-scale networks, as often encountered in federated learning scenarios.

III. Our proposal

A. Motivations

Previous decentralized learning solutions that include privacy mechanisms end up having a significant impact on efficiency and/or utility, which does not provide security advantages over more practical approaches such as federated learning [15]. Our proposal shows that decentralized learning can provide model-oriented privacy with a reasonable and adjustable impact on efficiency, and no reduction of model utility.

There have been studies [4], [21] that show that decentralized learning achieves faster convergence than federated learning, mainly because each node is connected with a small subset of neighbors and not to the whole network as federated learning. For this reason, we expect decentralized

learning topologies with nodes having low degrees to be more efficient. It is assumed the network consists of similar devices with similar computational power, so the node with the highest degree becomes the bottleneck in the network. The computational complexity of the privacy mechanisms for model aggregation has also an impact on the efficiency of the training process, so mechanisms with low computational complexity are preferred. Our proposal uses a low degree topology and an efficient model aggregation protocol based on secure multiparty computation [14].

In decentralized learning, when less dense topologies are used, the model updates of a node are more influenced by its local data and the updates from its neighbors. That brings the disadvantage of making more difficult model-oriented privacy, due to local generalization [15]. Additionally, the neighbors of an honest node could collude and learn its raw model updates, even if a secure multiparty computation protocol is used. The model utility could be also compromised because the influence of model updates of a node decreases the further away it is from other nodes [12]. The fewer direct neighbors it has, the more prone to local generalization is [15]. Our proposal involves dividing the nodes into groups of approximately equal size. This division aims to minimize the likelihood of colluded nodes becoming neighbors with honest nodes. We also include a mechanism that enables model aggregation equivalent to federated learning, so the updates of each node participate equally in the global model at each training round.

B. Secure Group-Based Model Aggregation (SGBMA)

Our proposal, Secure Group-Based Model Aggregation (SGBMA), is a decentralized learning framework that as its name suggests, performs model aggregation in small groups of nodes. In table I, we introduce the notation used in algorithm 1 to describe SGBMA.

TABLE I
SGBMA ALGORITHM NOTATION

Notation	Description
n	number of nodes
k	size of the groups
t	total number of training rounds
d	depth of the tree
G	set of groups where each group is a set with k elements
V	set of all nodes
A	set of aggregation nodes
A_H	aggregation node of group H
r	root node in the tree
$L(i)$	set of nodes in level i in the tree
$\sigma_1(i)$, $\sigma_2(i)$	left and right children of node i, in the tree, respectively
X_i	local dataset of node i
Θ^i	vector of parameters of the global model at time i
Θ^i_v	vector of the model's parameters of node/group v at time i
η	learning rate

The idea behind group-based aggregation is to take advantage of the Efficient Secure Sum Protocol (ESSP), proposed by Tran et al. [14]. The main problem with the original use of ESSP is that at each round, it requires a central aggregation node and other k randomly selected nodes to update the global model. The k selected nodes update the model with their local data and after that, the aggregation node combines all k models for the next iteration. The aggregation node is overloaded at each iteration, so a low value of k is desirable, but in large-scale scenarios, a small value of k may slow convergence or impact the final model utility. Consequently, in SGBMA, the nodes are divided into groups of size k (algorithm 1, line 1), and an aggregation node within each group is selected (line 2). Instead of training one group at a time, all the groups work in parallel at each training round using the ESSP (lines 8-10), so aggregation within groups is performed privately. In this way, we can use small values of k even in large-scale scenarios. ESSP combines model updates using addition, not averaging, something useful to our framework that is discussed below.

Isolated training groups would not share the knowledge of their models, so aggregation nodes are randomly arranged in a balanced binary tree aggregation graph to ensure a global connected graph (see Fig. 2). When training nodes finish updating the models with their local data, they share the update with their associated aggregation node using ESSP (line 14). The aggregation node combines its model with the models of the rest of the nodes in the group (line 15), and then, the global aggregation phase begins (lines 20-27). Leaf nodes in the tree share the aggregated models, which include information from the nodes in the groups, with their parent nodes (line 23). Parent nodes then aggregate the models received from their two children with theirs (line 24) and share the resulting model with their parents, going up a level in the tree. Aggregation in intermediate levels is done using addition, without the ESSP protocol. This process finishes when the root node is reached and it makes the last combination by averaging (lines 30-31). That will be the model to optimize in the next training round. The global model is sent back among aggregation nodes using the balanced binary tree, but this time from up to bottom (lines 34-40). Aggregation nodes have the responsibility to share this global model with the nodes in their groups (line 37).

Thanks to this aggregation method, SGBMA achieves equal user participation in the global model update, similar to federated learning. It employs a balanced binary tree structure for aggregation, which leverages the inherent parallelization of the problem. Assuming that the time required to send one user update to another is constant, combining the models and propagating them takes only $2\log(n/k)$ units of time, where n is the number of participants and k the size of the groups. This is due to the presence of n/k groups and aggregation nodes in the tree. At each level l in the tree, 2^l participants simultaneously share their updates, aggregating only three models: two from their children and their own. The time needed for update propagation is

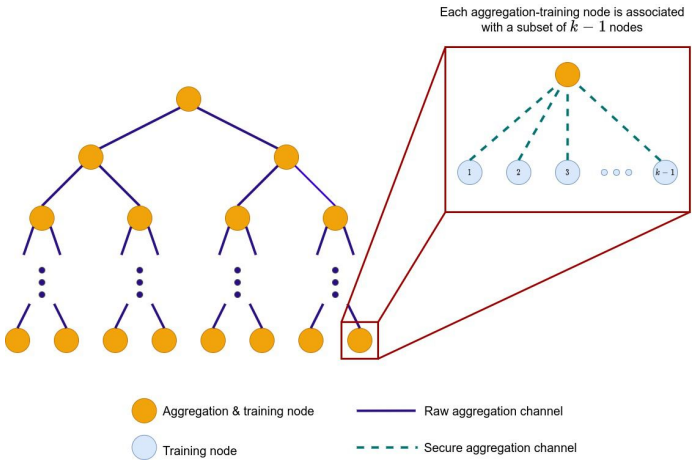

Fig. 2. Secure Group-Based Model Aggregation (SGBMA) diagram. Training nodes, in blue, only update the model with their local data and share their updates with the aggregation node in their group using ESSP. Aggregation & training nodes, in yellow, update the model with their local data and, receive and combine the models of their respective groups. They also receive the aggregated models from their two children in the binary tree and combine them with the model of their group. Then, they send this combined model to their parents in the tree.

equal to the depth of the balanced binary tree, which, as we know, is $\log(m)$, where m is the number of nodes. The propagation involves sending models twice, once for aggregation and once to share the updated global model, resulting in a total time of $2\log(n/k)$ units. Even in a case with approximately one billion aggregation nodes ($\approx 2^{30}$), only $2\log(2^{30}) = 2{\times}30 = 60$ units of time would be required.

Employing addition for combining models in intermediate levels avoids decreasing the influence of models of deeper nodes in the global model. ESSP is only used for aggregation within groups. In the balanced binary tree, the raw aggregated models are shared. Attacks on aggregated models do not leak information about specific users but optimal group sizes have to be investigated.

SGBMA sacrifices a little efficiency, giving more work to some nodes, to improve privacy. We can see this protocol as many small and secure federated learning problems that work together to improve a global model.

C. Privacy analysis

As mentioned early, SGBMA uses the Efficient Secure Sum Protocol (ESSP) [14], in the aggregation nodes to ensure model-oriented privacy. This protocol has a low computational complexity and is intended to be used in small subsets of participants to keep it low. Moreover, it ensures privacy even with $k-2$ parties colluding, where k is the number of nodes in a group.

Our framework uses ESSP in parallel in different and small instances of the problem, so in each of these subsets of nodes, data leakage may occur if all nodes but one

Algorithm 1 Secure Group-Based Model Aggregation

Randomly divide the nodes into groups of size k
Within each group randomly select one node as aggregator
Randomly arrange aggregation nodes in a balanced binary tree
Initialize the model parameters Θ^0

for each $i \in [1, 2, \ldots, t]$ **do**
  // Update of the global model with local data
  for each $v \in V$ **in parallel do**
    $\Theta_v^{i+\frac{1}{4}} = \Theta_v^i - \eta \nabla_{\Theta_v^i}(X_v, \Theta_v^i)$
  end for

  // Secure aggregation by groups
  for each $H \in G$ **in parallel do**
    Using ESSP, $\forall v \in H$ share $\Theta_v^{i+\frac{1}{4}}$ with A_H
    A_H aggregates models into $\Theta_H^{i+\frac{2}{4}}$
  end for

  // Aggregation of groups' aggregated models
  // by levels through the tree, from bottom to top
  $j \leftarrow d-1$
  while $j > 0$ **do**
    for each $v \in L(j)$ **in parallel do**
      Receive aggregated models from $\sigma_1(v)$, $\sigma_2(v)$
      $\Theta_v^{i+\frac{3}{4}} \leftarrow \Theta_v^{i+\frac{2}{4}} + \Theta_{\sigma_1(v)}^{i+\frac{3}{4}} + \Theta_{\sigma_2(v)}^{i+\frac{3}{4}}$
    end for
    $j--$
  end while

  // Aggregation with the average in root node
  Receive aggregated models from $\sigma_1(r)$, $\sigma_2(r)$
  $\Theta^{i+1} \leftarrow \frac{1}{n}\left(\Theta_r^{i+\frac{2}{4}} + \Theta_{\sigma_1(r)}^{i+\frac{3}{4}} + \Theta_{\sigma_2(r)}^{i+\frac{3}{4}}\right)$

  // Propagation of the global model, from top to bottom
  while $j \leq d$ **do**
    for each $v \in L(j)$ **in parallel do**
      v sends Θ^{i+1} to $\sigma_1(v)$, $\sigma_2(v)$
      v sends Θ^{i+1} to $\forall u \in G_v$
    end for
    $j++$
  end while
end for

return Θ^t

are colluding. In this framework, the risk of data leakage is a probabilistic problem subject to how groups are generated.

Let n be the total number of participants, and $k \ll n$, the number of nodes associated with each aggregation node (counting the aggregation node), k can be considered the only framework's parameter and controls the privacy-efficiency trade-off. A larger value of k overloads aggregation nodes reducing efficiency, but it reduces data leakage risk because reduces the probability that colluded nodes belong to the same aggregation group.

Since the groups and the balanced binary tree are generated at random, the probability that at least one user is at risk, that is, an honest user belongs to a group with exactly $k-1$ malicious participants is:

Let m be the number of malicious participants, where $k-1 \leq m \leq n-1$, h the number of honest users, where $n = h+m$ and, E_i the event that the i^{th} user is at risk, then:

$$P(E_1 \cup \cdots \cup E_h) = \sum_{i=1}^{h} (-1)^{i-1} \binom{h}{i} P(E_1 \cap \cdots \cap E_i) \quad (1)$$

To find the probability of at least one of the events in $E_1, E_2, \ldots, E_h$ occurring, we use the inclusion-exclusion principle to account for the overlapping nature of these events. According to this principle, in equation 1 we subtract the probabilities of odd i's and add the probabilities of even i's $(-1)^{i-1}$. We count the number of possible combinations of taking i honest users $\binom{h}{i}$, and then we multiply it by the probability that all i users are at risk. It is easy to see that the probability that all i users are at risk is:

$$P(E_1 \cap \cdots \cap E_i) = \frac{\binom{m}{k-1}}{\binom{n-1}{k-1}} \times \frac{\binom{m-(k-1)}{k-1}}{\binom{n-1-k}{k-1}} \times \cdots \times \frac{\binom{m-(i-1)(k-1)}{k-1}}{\binom{n-1-(i-1)k}{k-1}} \quad (2)$$

The probability that the first honest user is at risk is $\frac{\binom{m}{k-1}}{\binom{n-1}{k-1}}$. This is because there are $\binom{m}{k-1}$ ways to choose $k-1$ malicious users to be in the group with that honest user, out of a total of $\binom{n-1}{k-1}$ to choose $k-1$ other users to be in that group.

Conditional on the first honest user being at risk, the probability that the second honest user is at risk is $\frac{\binom{m-(k-1)}{k-1}}{\binom{n-1-k}{k-1}}$. This is because there are now only $m-(k-1)$ malicious users to choose from, and only $n-1-k$ other remaining in total, and so on. Expanding equation 1, we have:

$$\sum_{i=1}^{h} (-1)^{i-1} \binom{h}{i} \frac{\binom{m}{k-1}}{\binom{n-1}{k-1}} \times \frac{\binom{m-(k-1)}{k-1}}{\binom{n-1-k}{k-1}} \times \cdots \times \frac{\binom{m-(i-1)(k-1)}{k-1}}{\binom{n-1-(i-1)k}{k-1}} \quad (3)$$

Using equation 3, we can see the probabilities of risk with a small example. Let $n = 100$, in figure 3 we show the probability that at least one user is at risk for different numbers of malicious users and different group sizes (k).

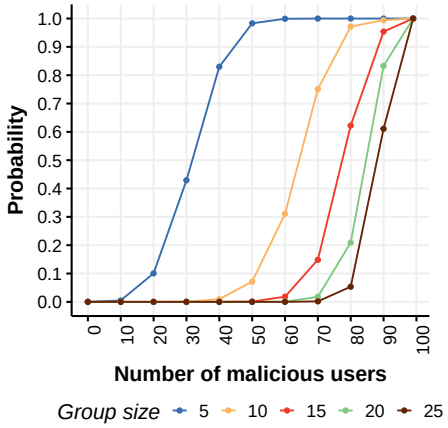

Fig. 3. Probability at least one user is at risk with $n = 100$ for different values of group size (k) and number of malicious users (m).

In figure 3, it can be appreciated that groups of size 5 have a high probability of data leakage. However, increasing the group size to 10 significantly lowers this probability. For example, even with 50 malicious nodes, the probability is only 0.0714. With a group size of 20, the probability of data leakage is only 0.000014 with 50 malicious nodes. This makes SGBMA feasible to be used in large real scenarios.

IV. Experiments

In this section, we present the experimental setup to evaluate the proposed decentralized learning framework, SGBMA. The objective is to compare the model utility, in terms of accuracy and the loss value on the testing set, with two different reference frameworks:

1) Federated learning, using the FedAvg algorithm [2], which employs a central server to orchestrate the training process. In this algorithm, during each round, the central coordinator dispatches the current global parameters to all participants. Subsequently, participants update their models locally and transmit these updates back to the coordinator, who aggregates them into a global model for the next iteration.
2) Conventional decentralized approach, using D-PSGD algorithm [4], which does not require a central coordinator. In this algorithm, every node establishes connections with a specific subset of nodes for sharing and receiving updates. During each round, after receiving these updates, each node aggregates the models from its connected neighbors and then proceeds to conduct local training using its own dataset. We evaluated D-PSGD with two different topologies: A ring graph (DecRing), where each node has two neighbors forming a ring, and a random regular graph of degree four (DecRegular4),

where each node has four randomly assigned neighbors maintaining a connected graph.

The four frameworks, SGBMA, FedAvg, DecRing and DecRegular4, were evaluated in similar scenarios with 100 nodes each.

A. Experimental data

We used the well-studied MNIST dataset [22]. This dataset consists of images of handwritten digits from 0 to 9. It is composed of two main parts, a training set and a testing set. The training set contains 60,000 images, while the testing set contains 10,000 images. Each image is a grayscale, 28x28-pixel square, resulting in a total of 784 pixels per image. We respected the partitioning of the MNIST dataset, consequently, we used the 60,000 images as the training set. The 10,000 images of the testing set were only used to monitor the loss and accuracy at each training round.

MNIST is widely used as a benchmark dataset in federated and decentralized learning [2], [4], [14], [15], [23], and therefore suitable to evaluate our framework.

We conducted experiments for the four frameworks with two different methods of partitioning data:

1) Independent and Identically Distributed (IID), where data is distributed uniformly and independently among all nodes. Each node possesses a similar data distribution, mimicking a homogeneous setup. MNIST training data set was randomly divided into 100 chunks with 600 images each, 60 images of each digit from 0 to 9.
2) Non-Independent and Identically Distributed (non-IID), where data is distributed unevenly among nodes, leading to varying data distributions across different nodes. This setup emulates a more realistic and heterogeneous scenario where nodes may have different data characteristics. MNIST training set was divided into 200 chunks with 300 images of the same digit each, then two chunks were assigned to each user so they only have data of two different digits.

B. Setup

We conducted experiments to compare the model utility of the aforementioned frameworks for the classification problem of the MNIST dataset. Given a 28x28 pixels image, we want to know which number represents. Based on the results reported by McMahan et al. [2], we used the same multilayer-perceptron to evaluate the four frameworks with the hyperparameters presented in table II.

Finally, for the evaluation of SGBMA, we considered groups (k) with 5 nodes, so the balanced binary tree has $100/5 = 20$ nodes and $\log(20) \approx 3$ depth.

TABLE II
EVALUATED MULTILAYER-PERCEPTRON HYPERPARAMETERS

Hyperparameter	Value
Number of hidden layers	2
Number of units in hidden layers	200
Activation function in hidden layers	ReLU
Loss function	Sparse categorical cross entropy
Number of training rounds	250
Learning rate	1.47
Batch size	10
Number of epochs of local training	5
Number of training rounds	250

C. Experimental environment

Our experiments were conducted on a single computer with an AMD A12-9700P Radeon 4-core processor (up to 2500 MHz) and 16GB of RAM. We implemented the four frameworks in the R [24] programming language using the Keras library [25]. Source code can be consulted on GitHub in this link.

V. RESULTS

We evaluated model utility in terms of accuracy and the loss value (for the sparse categorical cross entropy function). At the end of each training round, we compute both metrics over the testing set using the global model. Figure 4 shows the accuracy and loss values obtained by the four evaluated settings.

As expected, the four frameworks perform well on the IID partitioning. They reach their highest accuracy in a few training rounds, but SGBMA has a slightly higher accuracy than the other three. In terms of the loss value, all but SGBMA seems to overfit data after round 100 increasing the loss value in the testing set. Just SGBMA does not increase. On the other hand, for the non-IID partitioning, the performance of the four frameworks was affected. We can appreciate a clear negative impact in both metrics in the DecRing setting, especially in the first training rounds.

SGBMA and FedAvg outperform the two decentralized settings evaluated and they are practically overlapped during the 250 training rounds. We can also see that there is a big improvement in the conventional decentralized approach just by increasing the number of neighbors of each node. DecRegular4 produces noticeably higher accuracy than DecRing and lower loss values.

VI. A CASE STUDY

The Secure Group-Based Model Aggregation (SGBMA) framework, originally designed for decentralized machine learning scenarios, can potentially be adapted to address other kinds of decentralized problems as well. The adaptability of SGBMA to different domains relies on the fundamental principles and techniques it employs. In this section, we present a case study, outside of machine learning, where the ideas of decentralization of SGBMA can be applied to.

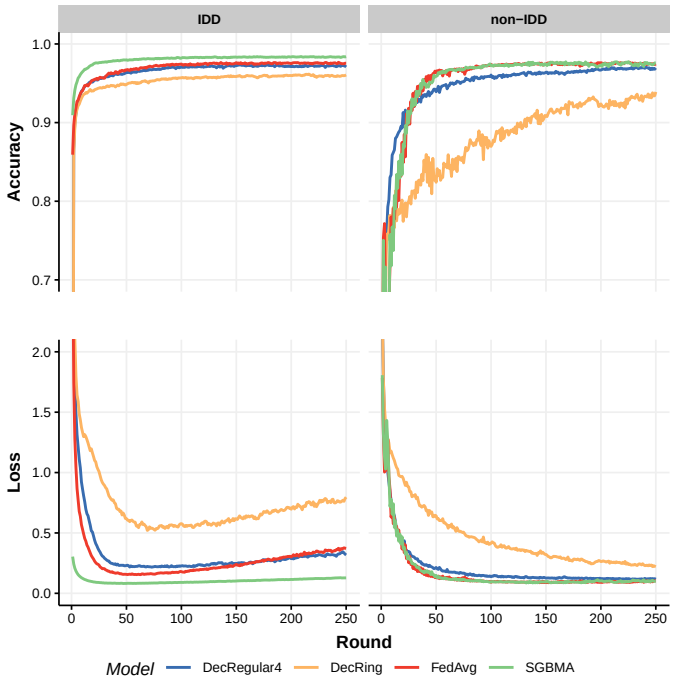

Fig. 4. Accuracy and loss values over the 250 training rounds with the four evaluated settings: decentralized with ring topology (DecRing), decentralized with random regular topology (DecRegular4), federated learning (FedAvg), and Secure Group-Based Models Aggregation (SGBMA). On the left side are the results for the Independent and Identically Distributed (IID) partitioning, and on the right side are the results for non-IID.

A. Background

In our previous work [16], we presented the personal lending platform model based on the borrower's trustworthiness score called LAPS (Loan Risk score, Activity score, Profile score, and Social Recommendation score).

Today, banks and financial institutions provide loans with terms and conditions that are not easy for borrowers to fulfill these conditions [26]. Banks or Lending marketplaces offer loans and still require collateral to guarantee that borrowers return their loans. Collateral can be in the form of assets that are easier to become money. A guarantor is a person who gives some guarantees to borrowers while applying for some loans. Types of debt financing and approval percentages are shown in Table III below. Borrowers' approval rates in Cash Advance Lenders (90%) are higher because of fast processing about 1-3 days approval, next followed by Alternative Lenders reach 70% loan processing environs 5-7 days, Traditional Banks about 45%, and Large Banks about 25%. The time processing of these two last options is around 14-30 days. Table III shows the scale of ratio and time processing impacts borrowers' proposal of some loans [27]. Table III also describes that it is still difficult to obtain some loans from traditional lending systems. The percentage approval was assumed from 100 borrowers. Of the Large Banks, 25 were approved, and 75 are denied in proposed loans.

TABLE III
APPROVAL RATES

Type of Debt	Percentage (%)
Traditional Banks	45
Cash Advance Lenders	90
Alternative Lenders	70
Large Banks	25

Source:https://gudcapital.com/types-of-business-loans/

We provide the LAPS formula as a solution to the problem above. It contains the Loan Risk, Activity, Profile, and Social Recommendation score as a borrower's trustworthiness score. The borrower has confidence after receiving the trustworthiness score, with this score, the borrower does not need any more collateral. From the lender/investor side, they get assurance that the borrower can repay the loan, and the recommender bridges the gap between the borrower and the lender. With LAPS, all users will get incentives that can be applied safely.

Some existing solutions can be further improved. For example, the BLockchain-Enabled Social credits System (BLESS) applied in the system leverages the decentralized architecture of the blockchain network, which allows grassroots individuals to participate in the rating process of a social credit system (SCS) and provides tamper-proof of transaction data in the trustless network environment. The anonymity in blockchain records also protects individuals from being targeted in the fight against powerful enterprises. A smart contract-enabled authentication and authorization strategy prevent unauthorized entities from accessing the credit system. The BLESS scheme offers a secure, transparent, and decentralized SCS. However, they have difficulty implementing technology in social aspects such as public acceptance and mass adoption [28].

Conversely, other solutions such as [29], use machine learning models to estimate a credit score. They trained a model using logistic regression and a multivariate discriminant on the Moroccan Financial Institutions (MFIs). The model combines behavioral and descriptive data related to the borrowers (age, activity, level of education, number of unpaid debts, number of loans, etc.) and (amount of credit, duration of credit, number of concluded loans per portfolio manager, etc.). The weaknesses are required a more extensive data sample, a deep enough history of the behavior of the customer, and also more information about variables related to the client's activity and its performance to predict the default better.

B. LAPS Splitting Formula

We define the trustworthiness score in terms of four variables [30], namely *LAPS (Loan Risk, Activity, Profile, and Social Recommendation)* as borrower trustworthiness score [31], see Equation 7.

1) *Loan Risk score* is the component for measuring if the borrower candidate has another loan such as housing, car, etc. If there is any other loan it is risky to allow getting another loan. That decreases the trustworthiness score, see Equation 4.

$$Loan\ Risk\ score = \sum_{i=1}^{n}(w_i * L_i) \tag{4}$$

where:

w = Weight for each variable $\{w \in \mathbb{R} \mid w \leq 1\}$, that able to be defined by user.
i = Sequence of weight and variable.
L = Variables (loan, housing), where $\{L \in \mathbb{Z} \mid L \leq 100\}$, and scale of values are between 0 to 100.

2) *Activity score* describes the borrower's occupation such as a job or business activity. It is used to measure the ability to pay and to consider the credit limit that corresponds with his activity, if a borrower candidate has a good occupation they will get the highest value of *Activity score*, see Equation 5.

$$Activity\ score = \sum(A) \tag{5}$$

where:

A = Variable (Job activity), where $\{A \in \mathbb{Z} \mid A \leq 100\}$, and scale of values are between 0 to 100.

3) *Profile score* is the personal data of borrower candidates such as age, education level, and marital status. These variables support the trustworthiness score. For example, the borrower should be older than 18 years old and 88 years old maximum [32], have an education level, and have marital status to consider the family dependent. All the variables are summarised as *Profile score*. The formula to get the *Profile score* is shown in Equation 6:

$$Profile\ score = \sum_{i=1}^{n}(w_i * P_i) \tag{6}$$

where:

w = Weight for each variable $\{w \in \mathbb{R} \mid w \leq 1\}$, that able to be defined by user.
i = Sequence of weight and variable.
P = Variables (age, education, marital), where $\{P\ in\ \mathbb{Z} \mid 0 \leq P \leq 100\}$.

4) *The social recommendation score* is the primary variable the borrower gets support directly from the other users to add recommendation value. This value act as a guarantor for borrowers to get some loans from lenders/investors through the lending platform, see Equations 7 and 8. *Social Recommendation score* = variables S (Social Recommendation), where $\{S \in \mathbb{Z} \mid 0 \leq S \leq 100\}$

The borrowers' trustworthiness score has a default value for the first time. It will change with the borrowers' activity in the lending platform. The recommenders can give excellent recommendations to borrowers who propose a loan. An essential part of the personal lending simulation is a recommendation that aims to reduce dependence on collateral. The borrowers' trustworthiness score is computed as:

$$\begin{aligned} Trustworthiness\ Score\ = & Loan\ Risk\ score + \\ & Activity\ score + \\ & Profile\ score + \\ & Social\ Recommendation\ score \end{aligned} \tag{7}$$

where:

- *Trustworthiness Score*: Borrower trustworthiness score.
- *Loan Risk score*: Information of the record from another loan of Borrower.
- *Activity score*: Business activity or job information of Borrower.
- *Profile score*: Personal information of Borrower.
- *Social Recommendation score*: The recommendation value of Borrowers from Recommender.

The LAPS formula is a commitment between borrowers, lenders/investors, and recommenders set by the smart contracts management so that all parties understand each other's obligations and risks that will be accepted. We add a positive weight for each variable to make LAPS flexible to adjust the importance of each score in the final value as expressed in the following equation:

$$\begin{aligned} LAPS\ = & (w_l * Loan\ Risk\ score) + \\ & (w_a * Activity\ score) + \\ & (w_p * Profile\ score) + \\ & (w_s * Social\ Recommendation\ score) \end{aligned} \tag{8}$$

where $\{w \in \mathbb{R} \mid 0 \leq w \leq 1,\ w_l + w_a + w_p + w_s = 1\}$. The weights of the trustworthiness attributes are predetermined based on their priority value that can modify by consensus. For example, using $w_l = 0.25,\ w_a = 0.2,\ w_p = 0.25,\ w_s = 0.3$, the social recommendation is given the highest percentage, and activity is given the lowest value. The social recommendation is the priority to measure the eligible borrower candidate. Equation 8 is the complete formula for the trustworthiness score.

C. LAPS with SGBMA

By adapting the privacy foundations of SGBMA to improve LAPS, the computation of a trustworthiness score can benefit from enhanced privacy protection while also leveraging a decentralized network to make more informed decisions about borrowers' creditworthiness. Below it is described how SGBMA can be applied to enhance LAPS.

Integration of machine learning models

- **Comprehensive trustworthiness assessment:** LAPS can leverage the decentralized network structure introduced by SGBMA to enhance trustworthiness assessment. This involves training a machine learning model on data from all participants, including their trustworthiness scores and historical interactions within the network.
- **Incorporating historical context:** The machine learning model, when trained on the combined data, can provide a more comprehensive assessment of a borrower's creditworthiness. It considers not only trustworthiness scores but also historical behaviors and relationships within the network.

Improved decision-making

- **Enhanced credit assessments:** The trained machine learning model equips lenders with better tools for making loan approval decisions. It allows lenders to assess borrowers' trustworthiness scores in the context of their social connections and historical interactions, resulting in more accurate credit assessments.

Data privacy and confidentiality

- **Privacy foundations:** Adapting SGBMA's privacy principles to LAPS ensures that sensitive borrower data is securely processed and aggregated. This minimizes the risk of data exposure or privacy breaches during the computation and aggregation processes.
- **Group-based privacy:** Users and recommenders can be grouped based on commonalities like social connections or geographic location. Within these groups, secure multiparty computation protocols, such as the Efficient Secure Sum Protocol (ESSP), can be also applied to calculate the social recommendation score without exposing individual user data.
- **User confidence:** By employing decentralized computation techniques, LAPS can build trust among its participants. Users can have confidence that their data remains confidential throughout the assessment process.

By applying the SGBMA framework to LAPS in these ways, the platform can enhance privacy protection, provide more accurate credit assessments, and improve operational efficiency. This, in turn, empowers lenders to make better-informed loan approval decisions, ultimately benefiting both borrowers and lenders within the lending marketplace. LAPS is suitable as application problem for the proposed SGBMA framework for several compelling reasons:

Decentralization of data: SGBMA's core principle of decentralized learning aligns seamlessly with the LAPS framework. In LAPS, borrower data, trustworthiness scores, and social connections are inherently distributed among users and recommenders. SGBMA allows this decentralized data to be harnessed effectively, without the need for centralization or data pooling.

Privacy-preserving approach: LAPS places a high premium on user privacy, especially in the context of social recommendation scores. SGBMA's focus on privacy through secure multiparty computation protocols ensures that individual user data remains confidential during both local training and the aggregation process. This approach is well-suited to protect sensitive borrower information in LAPS.

Local training and model building: SGBMA encourages local training and model building within smaller groups of users and recommenders. This concept is highly applicable to LAPS, where trustworthiness scores and historical interactions are user-specific. Local training allows for the consideration of user-level nuances and ensures that the resulting machine learning models are tailored to the unique characteristics of each group.

Scalability and efficiency: LAPS, like any lending platform, may have a large user base. SGBMA's decentralized approach enhances scalability and efficiency by breaking down computations into smaller, manageable groups. This reduces the computational burden and ensures that the integration of machine learning remains efficient even as the platform scales.

Integral creditworthiness assessment: The integration of machine learning using SGBMA allows LAPS to provide a more integral assessment of borrowers' creditworthiness. By considering a wide range of features, including trustworthiness scores, social connections, and historical interactions, the resulting models can offer lenders a more comprehensive view, leading to more informed loan approval decisions.

VII. Discussion

Overall, the SGBMA framework yields the highest accuracy and the lowest loss value, the same as FedAvg in the non-IID partitioning. The results showed that these frameworks reach the highest accuracy with fewer training rounds, but decentralized learning can achieve similar results with denser topologies. More extensive experiments are required to evaluate the model utility of SGBMA compared with other PPML solutions. New datasets have to be used to compare the performances on different machine learning tasks.

While the current evaluation provides valuable insights into the model utility of the four frameworks under consideration,

it is important to acknowledge the limitations of our study and identify avenues for further research. First and foremost, the experiments were conducted on relatively modest datasets, which might not fully reflect the challenges and complexities of large-scale scenarios. Therefore, more extensive experiments should be conducted with significantly larger datasets to assess SGBMA's performance under real-world, large-scale conditions. Scaling up the dataset size can provide a better understanding of SGBMA's ability to handle more substantial data volumes and its potential advantages in terms of scalability and efficiency.

Furthermore, exploring SGBMA's performance in a parallel environment could be a promising direction for future investigations. By leveraging parallel processing capabilities, SGBMA might demonstrate enhanced efficiency in handling distributed data and computation, leading to reduced training times and improved overall performance. Conducting experiments in a parallel computing environment would allow us to evaluate the full potential of SGBMA and better understand its advantages in scenarios with a high degree of parallelism.

In addition to the scalability and efficiency aspects, it would be beneficial to explore SGBMA's robustness and generalization capabilities across various domains and datasets. Evaluating its performance on diverse datasets from different sources and domains can provide insights into its adaptability and reliability in a broader range of applications.

In the context of privacy-preserving techniques, it is noteworthy to consider the potential adaptability of SGBMA to incorporate well-established methodologies such as differential privacy. The principles of differential privacy can be readily integrated into the SGBMA framework to enhance privacy protection during the training process. By introducing noise or perturbations to the model updates on the group aggregated models, SGBMA can achieve a level of privacy guarantee, ensuring that no individual participant's data can be inferred from the shared model. This adaptation would enable SGBMA to strike a delicate balance between collaborative learning and data privacy, making it an appealing option for applications that require strict privacy protection while capitalizing on the advantages of decentralized learning.

Similar to SGBMA, federated learning frameworks that distribute model aggregation have been proposed. Indeed, Bonawitz et al. [33] address the distribution of computational operations in federated learning. The authors proposed an architecture involving various actors to distribute computationally expensive tasks such as coordination and aggregation. While this approach offers some advantages, it still relies on a central orchestrator entity to handle these tasks. As the number of participants increases, the computational demands on the orchestrator entity become substantial, and accommodating additional actors (servers) incurs additional costs. Scalability is limited to the amount of resources available. The novelty in our proposal lies in the fully decentralized nature of aggregation and model sharing that is carried out by the participants themselves. Once the parameter k, representing the size of aggregation groups, is defined, our framework can effortlessly scale by adding more nodes to the aggregation tree. This scalability feature prevents overburdening specific nodes, ensuring a balanced distribution of computational tasks. Moreover, our balanced binary tree aggregation structure mitigates the impact on model propagation time as the tree grows. The logarithmic growth of propagation time with the number of aggregation nodes ensures efficient and scalable decentralized learning.

VIII. Conclusion

The results of this study show that SGBMA is an effective framework for decentralized learning. It has the best performance in terms of accuracy and loss value on both IID and non-IID partitioning, compared to the benchmarked approaches. Contrary to DecRing and DecRegular4 settings, SGBMA's final model utility is not affected by the number of neighbors each node has.

While the present study has provided valuable findings regarding the performance of different frameworks under certain conditions, there is still significant room for improvement and exploration. More extensive experiments, involving larger datasets and parallel computing environments, are essential to fully assess SGBMA's potential in large-scale scenarios and its efficiency advantages. Nevertheless, SGBMA is a promising framework for decentralized learning that can be used to train accurate and robust models in a distributed setting.

The case of study described in Section VI, is one of the many areas where decentralized learning can be successfully applied. In an increasingly interconnected world, decentralized learning is gaining importance due to robustness, scalability, and privacy guarantees, making it particularly suitable for resource-constrained settings like edge computing and the Internet of Things (IoT).

References

[1] R. Xu, N. Baracaldo, and J. Joshi, "Privacy-preserving machine learning: Methods, challenges and directions," *arXiv preprint arXiv:2108.04417*, 2021.

[2] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized data," 2016. [Online]. Available: https://arxiv.org/abs/1602.05629

[3] F. Boenisch, A. Dziedzic, R. Schuster, A. S. Shamsabadi, I. Shumailov, and N. Papernot, "When the curious abandon honesty: Federated learning is not private," 2021. [Online]. Available: https://arxiv.org/abs/2112.02918

[4] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," 2017. [Online]. Available: https://arxiv.org/abs/1705.09056

[5] E. Cyffers and A. Bellet, "Privacy amplification by decentralization," 2020. [Online]. Available: https://arxiv.org/abs/2012.05326

[6] A. Cheu, A. Smith, J. Ullman, D. Zeber, and M. Zhilyaev, "Distributed differential privacy via shuffling," 2018. [Online]. Available: https://arxiv.org/abs/1808.01394

[7] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients – how easy is it to break privacy in federated learning?" 2020. [Online]. Available: https://arxiv.org/abs/2003.14053

[8] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," 2018. [Online]. Available: https://arxiv.org/abs/1812.00535

[9] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," 2016. [Online]. Available: https://arxiv.org/abs/1610.05820

[10] Y. Wang, "Co-op: Cooperative machine learning from mobile devices," 2017. [Online]. Available: https://era.library.ualberta.ca/items/7d680f04-7987-45c5-b9cd-4fe43c87606f

[11] J. Konečný, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016. [Online]. Available: https://arxiv.org/abs/1610.02527

[12] T. Vogels, H. Hendrikx, and M. Jaggi, "Beyond spectral gap (extended): The role of the topology in decentralized learning," 2023. [Online]. Available: https://arxiv.org/abs/2301.02151

[13] T. Ryffel, A. Trask, M. Dahl, B. Wagner, J. Mancuso, D. Rueckert, and J. Passerat-Palmbach, "A generic framework for privacy preserving deep learning," 2018. [Online]. Available: https://arxiv.org/abs/1811.04017

[14] A.-T. Tran, T.-D. Luong, J. Karnjana, and V.-N. Huynh, "An efficient approach for privacy preserving decentralized deep learning models based on secure multi-party computation," *Neurocomputing*, vol. 422, pp. 245–262, 2021. [Online]. Available: https://doi.org/10.1016/j.neucom.2020.10.014

[15] D. Pasquini, M. Raynal, and C. Troncoso, "On the privacy of decentralized machine learning," 2022. [Online]. Available: https://arxiv.org/abs/2205.08443

[16] W. Uriawan, O. Hasan, Y. Badr, and L. Brunie, "LAPS: Computing loan default risk from user activity, profile, and recommendations," in *2022 Fourth International Conference on Blockchain Computing and Applications (BCCA)*. IEEE, Sep. 2022. [Online]. Available: https://doi.org/10.1109/bcca55292.2022.9922034

[17] V. Hartmann, K. Modi, J. M. Pujol, and R. West, "Privacy-preserving classification with secret vector machines," 2019. [Online]. Available: https://arxiv.org/abs/1907.03373

[18] Y. Wu, S. Cai, X. Xiao, G. Chen, and B. C. Ooi, "Privacy preserving vertical federated learning for tree-based models," *Proceedings of the VLDB Endowment*, vol. 13, no. 12, pp. 2090–2103, Aug. 2020. [Online]. Available: https://doi.org/10.14778/3407790.3407811

[19] S. Guo, T. Zhang, G. Xu, H. Yu, T. Xiang, and Y. Liu, "Topology-aware differential privacy for decentralized image classification," 2020. [Online]. Available: https://arxiv.org/abs/2006.07817

[20] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, may 2020. [Online]. Available: https://doi.org/10.1109/MSP.2020.2975749

[21] I. Hegedűs, G. Danner, and M. Jelasity, "Decentralized learning works: An empirical comparison of gossip learning and federated learning," *Journal of Parallel and Distributed Computing*, vol. 148, pp. 109–124, Feb. 2021. [Online]. Available: https://doi.org/10.1016/j.jpdc.2020.10.006

[22] Y. LeCun, C. Cortes, and C. J. Burges, "The mnist database of handwritten digits," *Pattern Recognition*, vol. 26, no. 4, pp. 741–774, 1993. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[23] A. Nilsson, S. Smith, G. Ulm, E. Gustavsson, and M. Jirstrand, "A performance evaluation of federated learning algorithms," in *Proceedings of the Second Workshop on Distributed Infrastructures for Deep Learning*. ACM, Dec. 2018. [Online]. Available: https://doi.org/10.1145/3286490.3286559

[24] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2023. [Online]. Available: https://www.R-project.org/

[25] J. Allaire and F. Chollet, *keras: R Interface to 'Keras'*, 2023, r package version 2.11.1. [Online]. Available: https://tensorflow.rstudio.com/

[26] D. Roman and G. Stefano, "Towards a reference architecture for trusted data marketplaces: The credit scoring perspective," *Proceedings - 2016 2nd International Conference on Open and Big Data, OBD 2016*, pp. 95–101, 2016.

[27] G. Capital, "Unsecured Business Loans Without Collateral," 2018. [Online]. Available: https://gudcapital.com/unsecured-business-loans-without-collateral/

[28] R. Xu, X. Lin, Q. Dong, and Y. Chen, "Constructing trustworthy and safe communities on a blockchain-enabled social credits system," *ACM International Conference Proceeding Series*, pp. 449–453, 2018.

[29] G. Bennouna and M. Tkiouat, "Scoring in microfinance: Credit risk management tool -Case of Morocco-," *Procedia Computer Science*, vol. 148, no. Icds 2018, pp. 522–531, 2019.

[30] W. Uriawan., O. Hasan., Y. Badr., and L. Brunie., "Collateral-free trustworthiness-based personal lending on a decentralized application (dapp)," in *Proceedings of the 18th International Conference on Security and Cryptography - SECRYPT,*, INSTICC. SciTePress, 2021, pp. 839–844.

[31] D. Martens, M. De Backer, R. Haesen, J. Vanthienen, M. Snoeck, and B. Baesens, "Classification with ant colony optimization," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 5, pp. 651–665, Oct 2007.

[32] S. Kellison and R. London, *Risk Models and Their Estimation*, ser. ACTEX academic series. ACTEX Publications, 2011. [Online]. Available: https://books.google.fr/books?id=JvmZTcRTQ6cC

[33] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H. B. McMahan, T. Van Overveldt, D. Petrou, D. Ramage, and J. Roselander, "Towards federated learning at scale: System design," 2019. [Online]. Available: https://arxiv.org/abs/1902.01046