BC-FL k-means

A Consortium Blockchain for Federated Clustering

Leeuw, W.V.

Award date:
2022

Technische Universiteit
Eindhoven
University of Technology

Department of Mathematics and Computer Science
Architecture of Information Systems Research Group

BC-FL k-means: A Consortium Blockchain for Federated Clustering

Master's Thesis

W.V. Leeuw

Supervisors:

dr. ir. M. Alishahi,
dr. ir. N. Zannone

Assessment Committee:

dr. ir. M. Alishahi,
dr. ir. N. Zannone,
dr. ir. T. Ozcelebi

Eindhoven, June 2022

BC-FL k-means: A Consortium Blockchain for Federated Clustering

W.V. Leeuw

ABSTRACT

Research in privacy-preserving collaborative learning has seen great strides in the past years. With the introduction of federated learning, models can be trained locally on devices to subsequently share updates with a central server so that no raw data is required to be shared. In turn, the central server aggregates the updates to build a global model. However, traditional federated learning suffers from shortcomings such as requiring trust in a central server and thus having a single point of failure. Moreover, unsupervised learning has historically been underrepresented in federated learning research. This research presents a novel process to train clustering models collaboratively without compromising accuracy while accommodating privacy and security in a decentralized manner. To decentralize collaborative learning and remove the single point of failure, a consensus is instead built collaboratively to facilitate the federated learning task. We present a committee-based consensus method designed for blockchain-based federated learning that is more scalable than common consensus protocols used in blockchains. We present a prototype based on the aforementioned ideas and show that its performance is comparable with centralized k-means through the use of a reputation system, regardless of the distribution of data among devices.

1 INTRODUCTION

Machine learning has seen improvements throughout the past years, increasing the quality of models and their potential application areas. As such, many organizations are increasingly using machine learning models to perform tasks for (exploratory) data analysis using their local data for training purposes. Being able to use many organizations' data to train machine learning models improves model quality [35]. Due to privacy concerns, such collaborative training of models has not yet been fruitful, especially in light of the General Data Protection Regulation (GDPR) [4]. With the introduction of federated learning, however, models can be trained collaboratively while maintaining data security and privacy. To do this, each device participating in the collaborative learning trains its local model and shares its updates with a central server [39]. The central server then aggregates the received updates to train a global model, which is then commonly shared with all participating devices. Thus, federated learning is a solution that aims to remove the necessity to share raw data between organizations. Issues and concerns still remain, however, as federated learning may still leak information from the local updates, suffers from connectivity issues and is susceptible to attacks due to its centralized nature [30].

Traditional federated learning suffers from several shortcomings. These shortcomings include:

(1) being required to trust a central server,
(2) having a single point of failure in a central parameter server,
(3) having limited traceability as historical models are lost, and
(4) having been insufficiently applied to the setting of unsupervised learning.

Using a blockchain to facilitate decentralized federated learning can alleviate the first three shortcomings. Blockchains decentralize the collaborative learning process as no single entity is responsible for the global update. Consequently, the requirement of a central parameter server is removed. Moreover, blockchains provide full traceability of learning rounds as each participant maintains a local ledger [12]. Lastly, as unsupervised learning in the setting of federated learning has not been researched thoroughly, we present k-means clustering for federated learning.

The contribution of this research is thus two-fold: we present a blockchain-based federated learning system and use this system to apply our federated k-means clustering. We investigate the advantages, as well as disadvantages of using a blockchain in federated learning, pertaining to scalability, security, resilience, and traceability. Moreover, as blockchains have historically lacked in terms of communication and computation costs, possible ways to remedy this and thus improve the scalability of blockchain-based federated learning systems are investigated.

1.1 Research Questions

This section describes the research questions which are used as a starting point for the work done in this thesis. The research questions pertain both to the adaptation of a k-means clustering algorithm for federated learning as well as the usage of blockchain in federated learning systems. The questions are revisited in Section 8.

Q1 How can blockchains be adapted to be better suitable to the setting of federated learning?
Q2 What are the (dis-)advantages of using a blockchain in federated learning in terms of scalability, quality, security, resilience, traceability, and privacy?
Q3 How can k-means clustering be adopted in (blockchain-based) federated learning?

Research question **Q1** aims to find adaptations of the blockchain architecture to increase the scalability regarding throughput and the number of participating devices.

Research question **Q2** aims to look at both the advantages as well as disadvantages of using blockchains to facilitate decentralized federated learning in terms of scalability, quality, security, resilience to attacks, traceability, and privacy.

Research question **Q3** is posed as the typical application in federated learning is supervised learning. Unsupervised learning techniques, such as k-means clustering, have not been sufficiently researched within federated learning.

1.2 Contribution

This research presents k-means clustering for blockchain-based federated learning. We show both how k-means clustering models

can be trained in federated learning generally and how blockchains may help facilitate federated learning.

We designed a blockchain-based federated learning framework to perform k-means clustering, with current problems in mind that were put forward in the existing research on blockchain-based federated learning systems. For research question **Q1**, we investigated how some of these common problems can be remedied and present our design in Section 4.

To answer research question **Q2**, we assessed the aforementioned system in terms of security, traceability, resilience to attacks, and privacy preservation. This is done in both Section 5 and Section 7. Moreover, we implemented a prototype serving as a proof-of-concept to assess the scalability and the model quality by executing several simulations.

Finally, for research question **Q3**, we first identified how k-means clustering can be adopted to federated learning. Moreover, we looked at attacks that are commonly carried out in federated learning and analyzed them in the context of federated k-means clustering.

1.3 Structure

The remainder of this report is structured as follows: Section 2 provides the required background information on unsupervised learning, federated learning, and blockchains. Section 3 presents the problem in more detail and also describes the motivation behind designing a blockchain-based federated learning system for k-means clustering. Section 4 details the design of the proposed blockchain-based federated learning system. Section 5 presents a security analysis judging from the proposed design. Section 6 briefly describes the prototype that was built and the decisions that were made in its implementation. Section 7 puts forward a performance analysis of the proposed system. Section 8 presents the lessons learned in this research, referring back to the research questions. Section 9 discusses related work, including other blockchain-based federated learning frameworks as well as solutions explored within traditional federated learning. Lastly, the report concludes with a summary of the most important findings and directions for future research in Section 10.

2 PRELIMINARIES

This work builds on existing knowledge in the fields of unsupervised learning, federated learning, and blockchains. These concepts are shortly introduced in this chapter to provide the required understanding. Section 2.1 introduces unsupervised learning. Section 2.2 presents federated learning, both intuitively as well as formally, applied specifically to k-means clustering. Lastly, Section 2.3 introduces the concepts of blockchain that are relevant for this research.

2.1 Unsupervised Learning

Unsupervised learning aims to recognize patterns or structures in unlabeled data and is often referred to as a form of exploratory data analysis. It is fundamentally different from supervised learning that involves labeled data being used to train a model to, for instance, predict the label of a future record with respect to its features. Semi-supervised learning uses both labeled and unlabeled data to train machine learning models. Unsupervised learning includes many different types of models, but this research focuses first and foremost on k-means clustering.

Notation	Description
$d \in \mathcal{D}$	Any device with no specified role.
N_d	The total number of devices.
$l \in \mathcal{L}$	A device that performs a learning task.
$j \in \mathcal{J}$	A participant being considered for or performing a committee-related task.
$c \in C$	A committee member.
S	The sets into which data points are partitioned by k-means, indexed by i.
$train_d$	Private training set of device d.
M_t^l	Locally updated model of device l at round t.
w_t^l	The parameters of the local model at device l at round t.
u_t^l	The local update at device l at round t.
G_t	Global model constructed at the end of round t.
w_t^G	The parameters of the global model at round t.
u_t^G	The global update at round t, obtained after aggregation.
$C(l_t)$	The contribution value of a device l at round t.
$R(j)$	The reputation value of participant j, denoted in the form (s_j, f_j).
θ_{contr}	The threshold for $C(l_t)$ at which point device l is excluded.
θ_R	The threshold on the ratio of f_j to s_j at which point j is excluded.
$S_C(A, B)$	The cosine similarity between vectors A and B.
θ_C	The threshold on the cosine similarity for the exclusion of aggregates.

Table 1: List of notations and their definitions.

2.1.1 *k-means clustering.*

Clustering is a well-known unsupervised learning task often employed to recognize patterns in datasets. Formally, given a dataset of observations $x_1, x_2, ..., x_n$ where each observation is a d-dimensional real vector, k-means clustering attempts to partition the dataset into $k \leq n$ sets $S = \{S_1, S_2, ..., S_k\}$, typically minimizing the within-cluster sum of squares (WCSS) [45].

k-means clustering aims to find centroids such that the WCSS is minimized. In other words, the centroids converge to a position where the distances between the points and its corresponding centroid are minimized. This learning objective can be formalized as:

$$arg\,min_S \sum_{i=1}^{k} \sum_{x \in S_i} ||x - \mu_i||^2, \tag{1}$$

where μ_i is the mean of the points in the set S_i.

2.2 Federated Learning

Historically, collaborative learning of models, where multiple parties work together to train a collective model, has been done by sharing data with a central server which was then tasked with the training of a collective model. The motivation for the development of federated learning is that directly sharing data poses security and privacy risks even if done using techniques to preserve privacy [39]. Rather than sharing data, federated learning has each participant train a local model and share parameter updates. A central server then aggregates those parameter updates to train a global model. Consequently, no raw data is required to be shared and thus collaborative learning is made more secure and better preserves privacy.

The notion of federated learning was developed by researchers at Google [39]. Specifically, its first application was that of Gboard, the Google keyboard, which can predict the next word while typing on smartphones.

Federated learning is often characterized by three properties, namely the data is *massively distributed*, *non-IID* and *unbalanced* [39]. The number of devices can be much larger than the average number of records stored on each device, meaning that the data is massively distributed. Non-IID means that the data is not independently and identically distributed. In other words, datasets residing on different devices are specific to each user and thus no single dataset is representative of the entire population. Similarly, the size of the datasets residing on different devices varies greatly, which is what is signified by the unbalanced property. All of these properties must be taken into account in federated learning.

Typically, federated learning is applied in settings that deal with cross-device rather than cross-silo collaborative learning [39]. Cross-device refers to a setting in which many devices each have a small dataset (relative to the size of the total dataset). Cross-silo, on the other hand, refers to a setting in which a handful of organizations each have a relatively large dataset.

Moreover, the datasets that reside on each device can be thought of as being partitions of the global dataset. These partitions can be horizontal, meaning each dataset contains the same features but the records are split among datasets. Alternatively, the partitions can be vertical, meaning each dataset contains the same records but the features are split among datasets. In this research, horizontal partitions are dealt with.

2.2.1 Formalization. In this section, we formalize the k-means clustering algorithm in the setting of federated learning. To do so, we apply the generalized federated learning setting [39] to k-means. As such, we consider the typical client-server structure in this section.

During each learning round, each device l trains its own local k-means clustering model by minimizing the within-cluster sum of squares, as follows:

$$arg\,min_S \sum_{i=1}^{k} \sum_{x \in S_i} ||x - \mu_i||^2, \tag{2}$$

where k is the number of clusters, x represents a record and μ_i is the mean of the points in the set S_i. In fact, μ_i is the centroid corresponding to the set S_i. We can thus extract μ_i for each $i \in \{1, ..., k\}$ from each device l. The collection of centroids from device l at round t is denoted as w_l^t.

Aggregating the local centroids requires two steps. First, each local centroid produced in round t is matched with the nearest global centroid from round $t-1$. This matching is done based on Euclidean distance. For each cluster, the local centroids are then combined to produce an aggregate:

$$\mu_t^{i*} = \frac{1}{N} \sum_{l=1}^{N} \mu_t^{l,i} \tag{3}$$

where μ_t^{i*} is the aggregate for the i-th global centroid, N is the total number of devices selected to supply an update and $\mu_t^{l,i}$ is the centroid from device l that corresponded with the global centroid.

Afterward, the i-th global centroid is updated according to the following rule:

$$\mu_t^i = \gamma \cdot \mu_{t-1}^i + (1 - \gamma) \cdot \mu_t^{i*}, \tag{4}$$

where $\gamma \in (0, 1]$ is a coefficient to control the influence of the previous global centroids and the newly aggregated local centroids. In that sense, γ is similar to a learning rate in deep learning [19]. μ_t^i is the new i-th global centroid, and μ_{t-1}^i was the i-th global centroid during the previous round. The collection of all k global centroids at round t is denoted as w_t^G.

Often, federated averaging (FedAvg) [39] is used for aggregation of all received local updates. This is done to assign more weight to updates from devices that have more records than others. In the context of k-means clustering, federated averaging can be denoted as:

$$\mu_t^{i*} \leftarrow \frac{1}{N} \sum_{l=1}^{N} \frac{|train_l|}{train_{avg}} \mu_t^{l,i}, \tag{5}$$

where μ_t^{i*} is the aggregate for the i-th global centroid, N is the total number of devices selected to supply an update, $|train_l|$ is the number of records that device l used for training and $train_{avg}$ is the average number of records per device. Notably, the weight of the update should be scaled based on the total number of records at a device and not the number of records that are closest to the i-th cluster. This is done to avoid information leakage, as devices may not want to share how many records they have that correspond to a certain cluster.

It should be stressed that both the choice for the learning rate γ in Equation 4 and the way in which federated averaging is applied in Equation 5 touch on design decisions made in this research. We briefly thouch on these choices in Section 7, where we compare our prototype with an existing implementation of federated k-means [52].

2.3 Blockchain

In 2008, Nakamoto proposed a blockchain with its first application being Bitcoin [43], which is a decentralized currency. In essence, a blockchain is a distributed ledger that allows collaborators to maintain a record of transactions in a decentralized manner [12]. Notably, this means that no central party is responsible for the verification of transactions. Instead, a consensus is reached through a type of decentralized arbitration, where a so-called *miner* can verify a transaction if they win a cryptographic puzzle referred to as Proof-of-Work (PoW). There are several aspects and features of blockchains that are important to consider in this research. Blockchains allow to store any type of data, are append-only, ensure data integrity and allow parties involved to reach consensus in a decentralized manner [12]. For a block to be added to the chain, consensus on that block must first be reached between participants. As such, an actor in the network may attempt to add a block to the chain but if no consensus is reached, the block will not be added to the chain. Once a block is appended to the chain its contents cannot be changed or reverted, which ensures the integrity of the system.

To showcase the structure of a blockchain, we take a closer look at Bitcoin. Figure 1 shows that Bitcoin is a chronologically ordered list of blocks in which each block refers to the unique cryptographic hash of the previous block. Due to the difficulty of

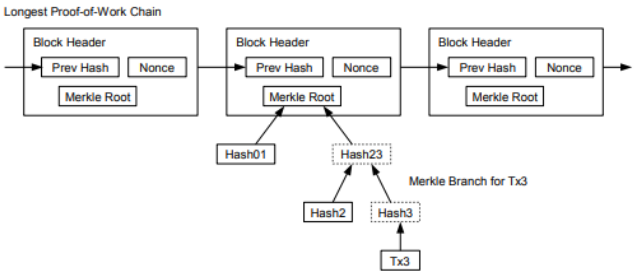


Figure 1: The structure of the Bitcoin blockchain [43].

the cryptographic puzzle, it is sufficiently unlikely that a single party succeeds in verifying subsequent transactions. Solving the puzzle and showing the resulting hash serve as proof that work was done. Typically, transactions are signed to ensure they are not changed using a mechanism called asymmetric encryption. In asymmetric encryption, each actor within a network has a key pair, namely a private and a public key. After a transaction is created, its creator can sign it using their private key, after which anyone seeing the transaction can verify that the creator signed it by verifying it using the creator's public key [16].

Much debate exists about the use of PoW as mining difficulty increases over time and it requires a relatively large amount of resources [21]. However, other types of consensus protocols exist which use fewer resources, such as Intel's proof-of-elapsed-time (PoET) [10] or election-based consensus protocols such as RAFT [46].

Typically, a blockchain system is either *private* or *public* [58]. In private blockchains, authentication is required and often more loose assumptions exist about the (malicious) nature of participants. In public blockchains, anyone can enter and exit the network as they wish. A third type of blockchain exists that is semi-private, also known as a *consortium* blockchain. A consortium blockchain only allows a group of pre-selected devices to participate. Within the context of federated learning, this pre-selection is especially important as participants should contribute positively to the global learning process. Thus, requiring potential participants to prove that they have data pertaining to the global learning task or are willing to pay a fee to join the learning process may already thwart potential attackers.

In this research, we consider a consortium blockchain where anyone can enter granted that they pay a fee that can be earned back if they positively contribute to the global learning process but few assumptions exist on the nature of participants.

3 PROBLEM DESCRIPTION AND MOTIVATION

This section describes the problem and the motivation behind this research. Section 3.1 describes the problem in detail and provides a use case with which we illustrate the motivation behind this research. Section 3.2 describes the requirements that we defined for blockchain-based federated clustering based on existing research.

3.1 Detailed Problem Description

We consider the setting of healthcare where we want to properly diagnose patients. In this setting, we consider the participants of federated learning to be hospitals. We can utilize clustering to identify profiles of patients having similar symptoms. Specifically, each resulting cluster has distinguishable features that are likely to map onto clearly defined diagnoses that in turn help to diagnose a patient whose illness is difficult to assess. For the sake of this example, we refer to those clusters as profiles.

Data pertaining to a patient's symptoms are sensitive and can be misused if they are leaked to an adversary. As such, the data must be kept confidential and thus a privacy-preserving and secure solution is necessary to train such a model collaboratively.

Each hospital may have, either coincidentally or because of its location, varying sets of patients. Consequently, the data may show that they suffer from various symptoms as well. This showcases the need for a solution that takes non-IID data into account. Moreover, this may cause certain versions of the collaboratively trained model to perform better locally than a more recent version. Lastly, as there is no clear candidate to host a central server that performs the aggregation task in this setting, it makes sense to decentralize the learning process.

Currently, the only way to perform k-means clustering is to share raw data with a central server that then builds a collaborative model. As such, information may leak if such a solution was opted for. Moreover, traditional collaborative k-means clustering models do not take into account that data from different sources may be non-IID distributed. Instead, most operate under the assumption that each participant has data pertaining to each of the k clusters. Lastly, with a central server being tasked with the training of a collaborative model, it is often the case that only the most recent model resides at the central server. It may be the case that an earlier version of the global model works best for a specific participant, however. Accordingly, they should be able to retrieve that version.

Blockchain-based federated clustering can remedy some of these problems. Firstly, k-means clustering has not been investigated thoroughly in the setting of federated learning. Applying k-means clustering to the setting of federated learning already solves issues pertaining to privacy, as raw data no longer has to be shared. Moreover, by applying similarity measures and local validation we can identify potentially malicious participants who supply updates that aim to deteriorate the global model. These techniques have been investigated for solutions other than unsupervised learning [26]. Blockchain-based federated learning also removes the need for a central server as the aggregation task is decentralized.

3.2 Requirements

We identified several requirements which were used as a reference when designing the blockchain-based federated clustering system. These requirements are based on those put forward by the authors of [54] and are shown in Table 2.

We shortly describe the requirements and the motivations behind them:

- **Personalization:** The requirement of personalization was put forward as traditional federated learning does not allow participants to select a model depending on which version

ID	Requirement	Description
R1	Personalization	Participants share local updates to augment the global model and can extend the global model for local purposes.
R2	Decentralization	No single party owns all data nor is a single party responsible for the federated learning task.
R3	Trust	The system maintains the trust of participants, either through monitoring of local updates or behavior generally.
R4	Heterogeneity-awareness	Differences in data residing at devices are accounted for such that these differences are not marked as malicious nor are they disregarded because of their dissimilarity with other devices' data.
R5	Context-awareness	Differences between participating devices in terms of computation power and link speeds are accounted for by dividing computational and communicational load.
R6	Secure	Information leakage is kept at a minimum by only sharing data that is necessary to build a global model.
R7	Robust	Attacks aiming to deteriorate the global model quality are protected against.

Table 2: Requirements of (blockchain-based) federated learning systems with detailed descriptions.

performs best locally, but rather only allow them to use the most recent version [30].

- **Decentralization:** Decentralization is one of the main aims of using a blockchain with distributed consensus in the context of federated learning. By removing the need of a central server, no single party is responsible for the federated learning task and the global learning process can be made robust against connectivity issues.
- **Trust:** Using distributed consensus requires the maintenance of trust to prevent malicious participants from propagating bad updates in the global learning process. For this measure of trust, we are no longer exclusively interested in the quality of local model updates but also on the general behavior of participants, such as whether they respond and successfully produce updates at all.
- **Heterogeneity-awareness:** Heterogeneity is particularly relevant to consider in the context of federated learning as the local data residing at participants may differ greatly. Proper awareness of this fact is characterized by allowing honestly produced local updates to be very dissimilar while still being able to identify malicious updates.
- **Context-awareness:** Typically, federated learning is performed using devices that have limited hardware, meaning both computation and communication are slow [33]. Consequently, federated learning systems should be context-aware, meaning that devices having more resources should be chosen to have a higher workload during federated learning than other devices.
- **Secure:** The federated learning system should be secure in the sense that data sharing is kept at a minimum. This translates to only sharing data that is necessary to improve the global model or broadcast the global model to participants. In k-means, this translates to only sharing local updates with (partially) trusted participants and sharing the global centroids with all participants. This requirement is especially important in the use case that we defined, as patient data is very sensitive.
- **Robust:** Lastly, the federated learning system should protect against attacks that aim to deteriorate the global model quality. In our use case, deterioration of the quality of the global model can result in clusters that do not align with any of the patients seen by any hospital. Consequently, discovered clusters may become entirely useless if such attacks are not protected against.

In Sections 5 and 7 we briefly mention these requirements when evaluating the proposed system. Moreover, Section 9 provides an overview of existing solutions for federated learning along with an overview showing which requirements those solutions meet. Using that overview, we compare our proposed system with existing solutions.

4 SYSTEM DESIGN

This section describes the design of the system. We researched several existing frameworks, which included proof-of-stake (PoS) based approaches [13], committee-based approaches [36], and also frameworks that considered fairness and privacy preservation in great detail [38]. The design decisions made in those frameworks along with the requirements we defined in the previous section helped guide the design process of blockchain-based k-means clustering. In this section, we show how blockchains can be used to facilitate federated learning and how k-means clustering can be adopted in that setting.

Section 4.1 describes the architecture of the system. Section 4.2 describes the reputation management and utilization within the system. A global overview of the system architecture is provided in Figure 2.

4.1 Architecture

The blockchain-based federated clustering system (BC-FL k-means) can be divided into five separate processes: blockchain initialization, local training, validation and aggregation, consensus, and committee election. Except for blockchain initialization, each of these processes is performed during each learning round. Figure 2 shows a global overview of these processes.

This section describes in detail the different processes and steps that the system performs.

4.1.1 Entities. There are several roles that participants can fulfill in the learning process:

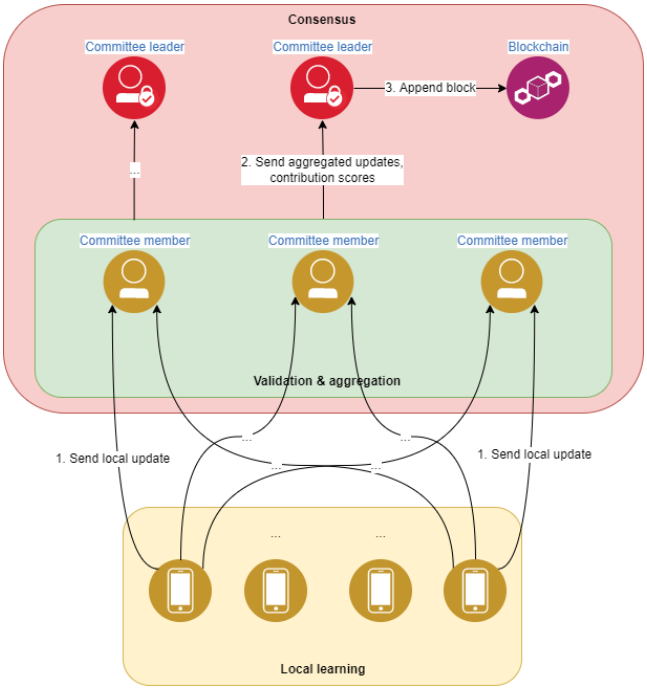


Figure 2: Global overview showing communication between different parts.

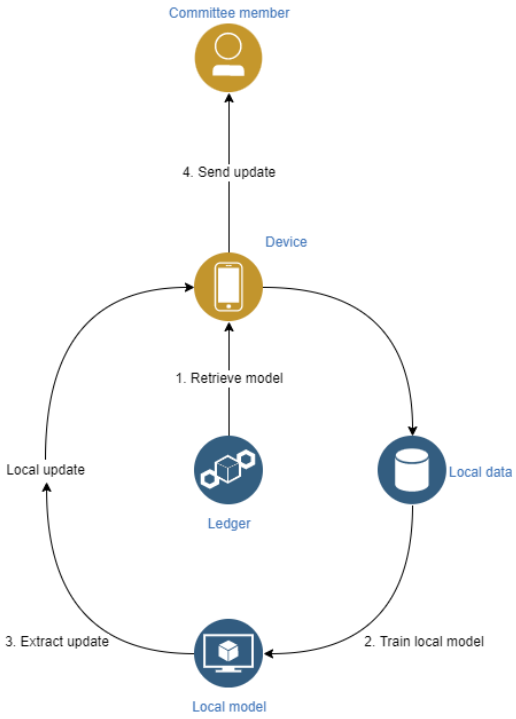


Figure 3: Local training on a device.

(1) Data owners: who perform the local learning step and send their updates to the committee members.
(2) Committee members: who validate received updates and aggregate them, after which they send the aggregated update to the committee leaders. They also vote on blocks that committee leaders propose.
(3) Committee leaders: who propose the addition of new blocks using the received aggregated updates to update the global model parameters.

4.1.2 Blockchain Initialization. At the start of the collaborative learning process, the genesis block must be created and the peer-to-peer communication channels must be established. Moreover, consensus is required on the initial parameters of the global model. The genesis block then forms the root of the blockchain, containing a timestamp and the initial model parameters. For the initial parameters, we assume the bounds of each feature in the dataset such that random initialization of k initial centroids is possible. Furthermore, a handful of devices must be randomly selected to be committee members and leaders. The other devices participating in the federated learning task can then start the local training step using the initial global model parameters.

4.1.3 Local Training. The local training process is shown in Figure 3. At the start of each learning round, each data owner selected to supply an update retrieves the most recent global model from their local ledger and performs a local training step using that global model as a starting point. Specifically, this means that each data owner matches their local clusters with the global clusters and then executes a single iteration of k-means. In doing so, we take into account the heterogeneity among participants by allowing data owners that have data describing fewer clusters than the global model to produce an update only for those clusters. After performing the local training each data owner extracts the update and sends it to the committee members with whom they were associated. By default, data owners send their updates to every committee member, but this can be tuned to reduce the load on committee members.

4.1.4 Validation and Aggregation. The validation step is shown in Figure 4. After having retrieved the local update from a data owner, the committee member validates it locally. This means that they retrieve the most recent global model from their local ledger, apply the retrieved update and then test the resulting model using their local data. Using this validation step, they assess the *contribution* of the data owner whose update they validated. This contribution is detailed in Section 4.2. After retrieving all of the updates or after a timeout, the committee member aggregates all validated updates and sends the result to the committee leader. This aggregation is computed using federated averaging [39].

4.1.5 Consensus. The consensus protocol is shown in Figure 5. After retrieval of aggregated updates from different committee members, the committee leaders can propose to append a block. It is important to note that there are multiple leaders such that if one disconnects, there is still one that can successfully produce and propose a block. They do this by constructing a block containing the updated global model parameters and trust values. The global model parameters are updated according to the update rule presented in Equation 4. In this sense, we deviate from simple k-means as we consider a parameter similar to a learning rate in deep learning

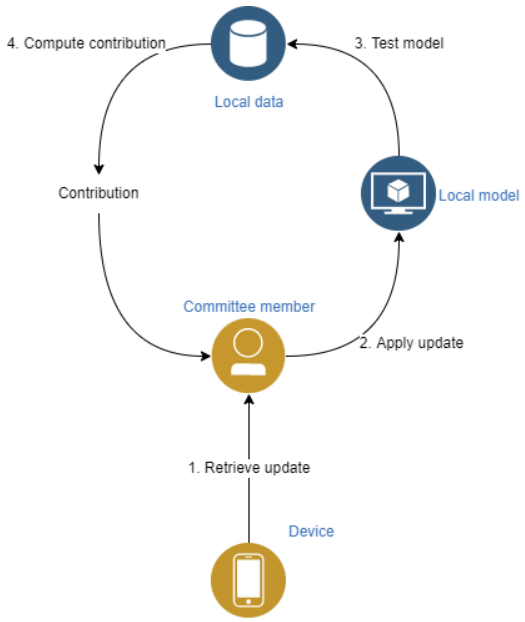


Figure 4: Validation step performed by a committee member.

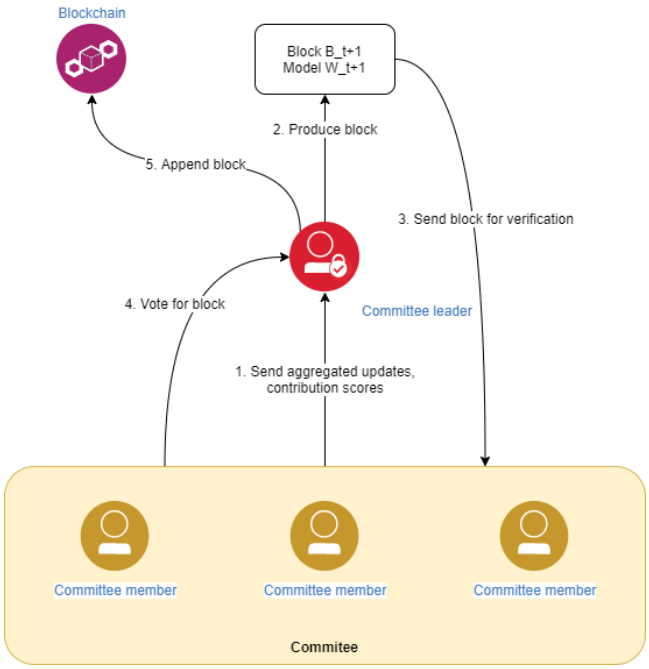


Figure 5: Consensus protocol of the system.

[19]. After block creation, leaders send their proposed block to the committee members. Specifically, the committee leader signs the block using their private key. The committee members can then verify the signature using the leader's public key. This step is performed to confirm the leader's identity. If the committee member agrees with the appendage of the proposed block, they produce a vote. This vote contains their signature on the proposed block, their identifier, their public key, and a timestamp at which the vote was cast. The committee members then send their vote to the committee leader. The committee leader can append their proposed block to the chain once they receive $\lfloor \frac{|C|}{2} \rfloor + 1$ number of votes, where $|C|$ is the number of committee members. To ensure enough votes were received by a committee leader, they serialize the votes into a byte-array of which the length can be checked by committee members to assess whether the length corresponds to a majority vote. Afterward, the winning leader can propagate the block to committee members who then propagate it throughout the network. If there is no winner after all votes are cast or after a timeout window, a new learning round starts.

It is worth noting that each winning block is appended to the chain such that every state of the global model can be accessed by all (honest) participants. This ensures that the global learning process is completely traceable and thus participants can choose the version of the global model that performs best for them locally.

4.1.6 Committee Election. After block propagation, a new committee is elected. The idea behind this election is that participants that behave honestly are selected to be part of the committee, either as member or leader. Other devices are selected to be data owners unless they have been flagged as having supplied malicious updates. The selection process is detailed in Section 4.2.2.

4.2 Trust

Trust is an important aspect of decentralized peer-to-peer communication networks. In situations where no central party records the reputation values of participants, methods aggregating local trust values are often used, as is the case in EigenTrust [31]. In these situations, peers maintain trust values for local peers and can share them if prompted to do so. However, in the case of blockchain-based systems, the values associated with trust, namely *reputation* and *contribution* in this research, can be stored on-chain. Reputation is associated with the behavior of the committee members and leaders while contribution depends on the quality of updates supplied by data owners. This means that both reputation and contribution values are maintained for each participant as they may change roles after each learning round.

4.2.1 Contribution. The contribution score is used to reflect the quality of the local updates based on the gain in model quality. Committee members validate local updates by applying them locally and then testing the resulting model on their local dataset.

Definition 4.1 (Contribution)**.** *The contribution of a device l is used to represent how their local updates affect global model performance and can be defined as:*

$$C(l_t) = \beta \cdot (\mathcal{S}(W_t^l) - \mathcal{S}(W_{t-1}^G)) + (1 - \beta) \cdot C(l_{t-1})$$

where $\beta \in (0, 1]$ is a coefficient to control the effect of the historical contribution and the contribution during round t on the eventual contribution score $C(l_t)$. $\mathcal{S}(\cdot)$ gives the average silhouette score as computed by a committee member. The silhouette score measures how similar records are to other records in the same cluster and how dissimilar they are to records associated with other clusters. The average silhouette score is given in the range of [-1, 1], meaning the range of

possible values of $C(l_t)$ is also [-1, 1]. Negative values of $C(l_t)$ imply that applying the local update from device l at time t deteriorates global model performance. W_t^l are the centroids of the local model at the current round and W_{t-1}^G are the centroids of the global model in the previous round. Moreover, the term $(1-\beta)\cdot C(l_{t-1})$ is used to consider the historical contribution of device l.

It is worth noting that the contribution score of a particular device depends on the local data of the committee member providing feedback on the local update of the device. As such, it can be the case that a device is assigned a low contribution score even if their local update may be valuable to the global learning process. To prevent a device from being excluded completely, we still select poorly contributing devices to provide local updates with a 10% probability. The only exception to this is when those devices are flagged as malicious. This happens when their contribution is below a threshold $\theta_{contr} < 0.0$.

4.2.2 Reputation. The reputation score is used to reflect the behavior of participants when they fulfill the role of committee member. Specifically, their reputation reflects whether they successfully validate, aggregate, and then send the received local updates to the committee leaders.

Definition 4.2 (Beta distribution)**.** *A beta distribution is a family of continuous probability distributions described in the interval [0, 1], characterized by two parameters, α, $\beta \geq 1$. The PDF of a beta distribution with shape parameters α and β is:*

$$\beta(x;\alpha,\beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\int_0^1 y^{\alpha-1}(1-y)^{\beta-1}\,dy}.$$

We maintain, for each participant, a value for α and β such that we can take samples from their beta distributions. Let i and j both be participants in federated learning. Let α_i and β_i be the parameters of the beta distribution of i and let α_j and β_j be the parameters of the beta distribution of j. If $\alpha_i > \alpha_j$ and $\beta_i < \beta_j$, the expected value when sampling a random variable from participant i's beta distribution is larger than that of j. We use this notion in the selection of committee members and leaders.

In the Beta model of reputation, only two types of interactions exist, namely successful or unsuccessful [42]. In other words, if no fine-grained ratings are required or possible to be given, relying on beta distributions suffices. In this research, a beta model is considered for reputation both for its simplicity in general and its applicability to the role that committee members and leaders fulfill.

The reputation value for a participant j at time t, $R(j)$, is a pair, namely $R(j) = (s_j, f_j)$. We consider the parameters s_j, f_j to be the number of successful and unsuccessful interactions respectively. Initially, $s_j = 0$ and $f_j = 0$, then we require $\alpha_j = \beta_j = 1$ as otherwise, we have no valid shape parameters for the beta distribution. As such, we have $\alpha_j = s_j + 1$ and $\beta_j = f_j + 1$, meaning the beta distribution associated with participant j has shape parameters (α_j, β_j). Whenever a new participant k joins the network, they start with the shape parameters $\alpha_k = s_k + 1$ and $\beta_k = f_k + 1$. To allow them to catch up, we assign a committee member or leader role to devices having $\alpha_j = 1$ and $\beta_k = 1$ with a probability of 10%.

Definition 4.3 (Cosine similarity)**.** *The cosine similarity is a measure of similarity between two numeric sequences, returning a number in the range [-1, 1], where -1 indicates dissimilar sequences and 1 indicates equal sequences. The numerical sequences are viewed as vectors in an inner-product space, such that the cosine similarity is the cosine of the angle between them.*
The cosine similarity of two vectors A and B is:

$$S_C(A,B) = \frac{A\cdot B}{||A||\,||B||} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2}\sqrt{\sum_{i=1}^n B_i^2}},$$

where A_i and B_i are the components of A and B, respectively.

Whenever an interaction happens between a committee member j and a committee leader, the leader provides feedback to the committee member by incrementing either s_j or f_j, depending on whether the interaction was successful or not. Specifically, an interaction is deemed successful if an aggregate was sent and the cosine similarity between the leader's update and the aggregate is above a threshold of $\theta_C \geq 0.0$. Cosine similarity is used here as it helps to assess whether an aggregate helped move the updated global centroids toward the global learning goal [44]. Moreover, if the ratio of unsuccessful interactions (f_j) to successful interactions (s_j) reaches a threshold of θ_R, a device is no longer allowed to fulfill the role of committee member or leader. We evaluate the ratio of f_j to s_j to exclude malicious devices or otherwise unreliable devices. It should also be noted that committee leaders' reputation cannot be negatively affected.

Utilizing the Beta model of reputation, we can arrive at an estimate of the probability of success when trusting a participant j as well as at the expectation of trusting participant j [29]. Let m be the number of participants to be selected to be committee members or leaders, then we can sample from each participant's Beta distribution, described by $R(j) = (\alpha_j, \beta_j)$, and select the top-$m$ participants. From those participants, the leaders are randomly elected. Moreover, no participant can be a committee member or leader for subsequent learning rounds. In other words, it can never occur that a participant j is a committee member in round $t+1$ if they have been a committee member or leader in round t. This is done to limit information leakage, as this ensures no subsequent updates from the same device are sent to the same participant.

5 SECURITY ANALYSIS

In this section, we analyze the security and privacy-preservation of the proposed BC-FL k-means system. Specifically, the analysis focuses on the confidentiality of local data and the resilience to attacks aiming to deteriorate the quality of the global model. In Section 5.1 we describe the adversarial model that we consider with respect to poisoning attacks, the adversary's strategy, and defenses against them. We consider possible information leakage in BC-FL k-means in Section 5.2. Lastly, we discuss other attacks that are relevant to BC-FL k-means in Section 5.3.

5.1 Poisoning Attacks

For the evaluation of the resilience against poisoning attacks of BC-FL k-means, we consider an adversarial model typically considered in the context of federated learning [8] [26]. The adversary attempts to poison the global model, thus threatening the quality of the

global learning process. This is typically done through either data poisoning or model manipulation.

Model manipulation attacks involve training rule manipulation to produce updates aimed to deteriorate global model performance [8]. Data poisoning attacks involve injecting faulty data to update a (local) model with, such that the resulting update deteriorates the quality of the global model.

Typically, federated learning handles either type of poisoning attack through (robust) aggregation schemes [44]. These aggregation schemes often adopt some type of outlier detection [18], similarity measure [44], or pruning of models [37], [28].

Rather than relying solely on an aggregation scheme, we use similarity measures at leaders and validation at committee members to identify malicious updates. Specifically, the similarity measure, cosine similarity in the case of BC-FL k-means, is used to discern whether a committee member's aggregate positively contributes to the eventual global update produced by a (winning) leader. On top of that, we leverage the data storage capacity of blockchains to track the contribution and reputation of devices based on the feedback they are given.

5.1.1 Adversarial Behavior. The malicious behavior exhibited by the adversary in the context of poisoning attacks is twofold:

(1) Data poisoning, where an adversary alters or otherwise perturbs their training data to produce faulty local model updates.
(2) Model manipulation, where an adversary alters the updates produced by their local model, or alters the training rule by which their local model trains to produce a poisoned model.

We consider the majority of participants assigned to each role (data owner, committee member, and leader) to behave honestly. Moreover, we consider the adversary to not have access to real data which aligns with the global model. If they did have such data, they would benefit from the clustering model produced through collaborative learning. In other words, the adversary's motivation to deteriorate the quality of the global model relies on them not being able to contribute to it. We consider the adversary to be successful if they manage to inject a poisoned update such that the quality of the global model deteriorates across participants.

Lastly, the adversary knows the following:

(1) The update rule of the global model,
(2) θ_{contr}, the threshold on contribution such that devices having contribution below that threshold are excluded,
(3) θ_C, the threshold set on cosine similarity such that leaders only accept aggregates scoring above that threshold, and
(4) θ_R the threshold of the ratio of unsuccessful versus successful interactions such that devices are excluded if their ratio is above the threshold.

Given what the adversary knows, they are able to manipulate their model entirely. As this is a more direct approach to deteriorating global model quality, we limit our analysis to model manipulation.

In the context of BC-FL k-means, poisoning attacks can be performed in each of the roles:

(1) Data owner, in which role the adversary attempts to produce updates that deteriorate the global model such that their update moves the centroids in the opposite direction that they would move if only updates were accepted from honest participants. As the adversary has no data related to the global learning goal, they can produce this update based on previous updates of the global model or if no information is available entirely, they can produce random updates. Formally, this can also be viewed as the adversary attempting to maximize deterioration of the global model by producing an update u for which $S_C(u, w) = -1$, where w represents the aggregated update from honest participants.
(2) Committee member, in which role the adversary obtains local updates from all data owners. They attempt to assess the likely contribution of an update based on previous global model updates. From this, they produce an aggregate w from the positively contributing local updates. Afterward, they produce an aggregate v for which $S_C(v, w) = -1$ and send that to leaders.
(3) Leader, in which role the adversary obtains aggregates from committee members. From those aggregates, they attempt to produce a combined aggregate z. Afterward, they produce a global update u_t^G such that $S_C(u_t^G, z) = -1$, where z is the combined aggregate derived from honest committee members.

On top of adversaries performing poisoning attacks, they are also able to collude with other malicious participants if they are assigned the role of committee member or leader. This collusion is centered around the provision of positive feedback to other malicious participants:

(1) Committee member, in which role the adversary accepts updates that move away from the global learning goal and provides them with positive feedback. To identify these updates, they look at the previous changes of the global model and assess the likelihood with which an update deteriorates the global model quality.
(2) Leader, in which role the adversary accepts aggregates from committee members that they deem to be likely to deteriorate global model quality. To identify those aggregates, the adversary looks at previous changes of the global model and assesses the likelihood with which the aggregate moves away from the global learning goal. They then provide the committee members that sent those updates with positive feedback.

5.1.2 Defenses. There are three defenses in place against poisoning attacks, namely the validation of local updates by committee members, the checking of the cosine similarity of aggregates (with respect to the produced update) by leaders and the voting for proposed blocks done by committee members. We present per role how BC-FL k-means is robust against poisoning attacks:

(1) Data owner, in which role the adversary produces an update u for which $S_C(u, w) = -1$ and sends it to all committee members. Committee members apply the obtained update locally and validate its contribution by obtaining the average silhouette score. If $S_C(u, w) = -1$, it is very unlikely that the average silhouette score is larger than that of the global model at the previous round. Thus, the adversary's update will not be accepted by honest committee members and their

contribution is decreased accordingly. If, as a result, their contribution drops below $\theta_{contr} < 0.0$, they are excluded from the global learning process entirely.

(2) Committee member, in which role the adversary produces an aggregate v for which $S_C(v, w) = -1$ and sends it to all leaders. However, the aggregates from committee members are not accepted if the cosine similarity is less or equal to θ_C. Moreover, if a leader encounters an aggregate for which the cosine similarity is less or equal to θ_C, the leader will provide the committee member with negative feedback. If the ratio between unsuccessful interactions and successful interactions (i.e. r_j and s_j respectively for a participant j) becomes greater than θ_R, the device is excluded from the global learning process and no longer has access to new global model updates. Consequently, the adversary can only produce aggregates of which the cosine similarity is greater or equal to θ_C if they want leaders to accept it and provide them with positive feedback such that they maintain access to the global learning process. However, the adversary can still give positive feedback to data owners that supplied poisoned or otherwise poor updates and send that feedback to the leader. The leader combines this feedback with feedback from other committee members, causing the effect of an adversary to be small if the majority of committee members behave honestly.

(3) Committee leader, in which role the adversary is able to produce a global update that decreases the quality of the global model. However, due to the voting mechanism, a majority of committee members will vote for the global update that performs best for them locally or not vote at all if all produced global updates deteriorate their local performance. As such, the adversary is unable to have their update accepted . It is worth noting that the adversary can give positive feedback to committee members they control if they are a leader. However, this feedback is only propagated within the network if they produce the winning block.

5.1.3 Alternative Attack Strategy. Rather than maximizing the deterioration of the global model, an adversary can opt for a strategy that slows down the global learning process by producing updates such that they are within the thresholds of θ_{contr} and θ_C. This would allow the poisoned updates to be accepted by committee members and leaders. However, as this does not immediately deteriorate global model quality it is more of a nuisance to honest participants as convergence then requires more iterations. Moreover, attacks of this type can be partially mitigated by setting stricter thresholds for θ_{contr} and θ_C.

5.2 Information Leakage

Information leakage in the context of k-means clustering refers to any information or data that may be learned other than the centroids describing the global clustering model. Typically, research focuses on the disclosure of (raw) data and the assignment of specific records to centroids or the distances from those records to the centroids [27] [25]. Some research argues that information leaks even when the number of iterations or the number of records per cluster is disclosed [40] and several papers have investigated issues such as that thoroughly [24] [56]. However, attacks that typically occur in distributed learning regarding data privacy, such as inference attacks and reconstruction attacks, have not been researched in the context of clustering [47].

Within BC-FL k-means, intermediate centroids of local models are shared with committee members. Moreover, the intermediate centroids of the global model are sent to all participants. Other than that, no data is disclosed as the assignment of records to clusters and the computation of distances between records and centroids is performed locally.

The assertion can be made that disclosing the intermediate centroids of local models to committee members leaks information. However, each participant only sees the intermediate centroids of the local model for a fraction of the total number of iterations. On top of that, participants having relatively better reputation are more likely to be selected as a committee member and are more likely to behave honestly. Moreover, the centroids from a local model are heavily influenced by the global centroids from the previous round as only one iteration of k-means is performed locally by each data owner each round. All that can be inferred from the intermediate centroids of local models are the number of centroids for which participants have records. If this is unwanted, participants can be forced to train their local model with the same number of clusters as the global model. However, this may cause data owners to produce the same update for multiple global centroids as all of them map onto the same local centroid and thus decrease global model quality.

Lastly, we argue that disclosing intermediate global centroids every learning round does not leak information as any information that can be derived from them cannot be linked to an individual participant and therefore does not infringe on their privacy. Thus, we conclude that BC-FL k-means is secure, in that it meets the requirement presented in Table 2.

5.3 Other Attacks

A different attack type that is more general to networks is Sybil attacks [26]. Sybil attacks in the context of federated learning often constitute attacks where a single paticipant attempts to inject poisoned data, for instance, through the use of multiple devices. As these devices often do not carry a unique identifier, it tends to be difficult to identify them. A naive approach to dealing with Sybil attacks is to use similarity measures, as it is expected that updates from the same participant would be very similar, especially in the case of data poisoning [17]. A more widely adopted way of dealing with Sybils in the context of blockchain is to have participants put in a stake or fee to participate in the network [59]. Alternatively, blockchain-based systems may require participants to identify themselves through the signing of transactions [16]. We assume a fee must be paid to join the network in BC-FL k-means and mitigate Sybil attacks accordingly.

6 PROTOTYPE IMPLEMENTATION

In this section, the implementation of the prototype of BC-FL k-means, is described. This section describes the decisions made in the implementation of the prototype and where its usage differs from real-world distributed systems.

This prototype is meant as a proof-of-concept of BC-FL k-means, aimed to test the proposed committee-based consensus method as well as federated k-means. As such, it is not immediately deployable on real distributed systems. Instead, it is meant to showcase how committee consensus and clustering work in a blockchain-based federated learning setting and how it performs in a simulation, modeled such that it is comparable to how federated learning behaves in a real distributed system. We analyze the performance using this prototype in Section 7.

6.1 Implementation Details

The prototype was implemented using Python 3.8.8, NumPy 1.21.4 and Scikit Learn 0.24.1. The code can be found at https://github.com/WVLeeuw/BC_Unsupervised_FL. The prototype consists of three main components:

(1) Blockchain;
(2) Devices;
(3) Federated k-means algorithm.

The blockchain uses simple hashes to have each block point to the previous block. As proof-of-work is not used as a consensus algorithm, hashing does not become more difficult as the chain length increases. Instead, the committee consensus protocol is implemented and leaders' block appendages are verified by checking the length of the byte-array that leaders produce using the votes they obtained, also referred to as proof-of-vote (PoV) [34]. As currently existing blockchains do not support PoV consensus, the blockchain was built from scratch. Blockchains that support pluggable consensus algorithms exist, though no equivalent of PoV is currently available [5]. These may be viable candidates for actual real-world implementations of blockchain-based federated learning.

The devices can have three different roles, namely data owner, committee member or leader, or be excluded from participating entirely during a learning round.

The device class allows for parameters to be set regarding their computation power and link speeds. These parameters are used to simulate real distributed systems as these include devices having varying hardware.

The local learning part of the federated k-means algorithm uses the implementation of k-means clustering from Scikit Learn [2]. Scikit Learn's implementation uses state-of-the-art techniques to improve the convergence of clustering models and is thus sufficient. Moreover, it allows for pre-defined initial centers to be passed, which was a necessity as we require each device to have the same starting point. Globally, to perform the aggregation at the committee members and leaders, NumPy was used as it deals efficiently with large-scale arrays as it is memory-efficient and leverages C code to achieve high speeds [22]. Moreover, NumPy simplifies the code base as it is vectorized, removing complicated and inefficient loops. The local learning step in combination with the aggregation steps at the committee members and leaders can be perceived as comprising the federated k-means algorithm in its totality.

6.2 Limitations

The payment of a fee and how it can be earned back through the positive contribution to the global learning process is not part of the prototype. In an actual distributed system, a reward pool would be instantiated for each learning round and devices can earn a portion of that pool depending on their contribution and reputation during that learning round.

The prototype only includes the basic implementation of k-means clustering as provided by Scikit Learn [2]. The current implementation may thus not be sufficiently equipped to deal with non-convex sets. Moreover, dimensionality reduction techniques are not included but may be promising for efficient clustering of datasets having numerous features.

7 PERFORMANCE EVALUATION

In order to evaluate the general performance of the proposed BC-FL k-means prototype in the previous section, we ran several experiments. These experiments are run using the prototype, simulating devices and their behavior. From those simulations, we draw conclusions with respect to the performance of BC-FL k-means. This performance includes model quality, convergence and time spent learning.

We briefly describe the evaluation environment in Section 7.1. The experiments that were run, and the parameters being changed, are described in Section 7.2. Lastly, the results of the experiments are evaluated in terms of model quality, convergence and runtime in Section 7.3.

7.1 Evaluation Environment

To measure overall performance, we perform BC-FL k-means on a synthetic dataset as well as several toy datasets [3]. To account for large datasets, a dataset describing forest cover types [1] was used. Table 3 describes all datasets that were used to evaluate the performance. The simulation was run on a laptop with an Intel i7 CPU having a clock rate of 2.30 GHz and 16 cores.

7.1.1 Parameter Settings. We briefly describe some of the parameters which were set for the simulations. These were kept equal across all simulations.

- $\theta_{contr} = -0.2$, the threshold on contribution at which a device is no longer allowed to participate. Moreover, devices are not considered to be data owners if they are in the bottom 25% of contribution values (if negative contribution values exist).
- $\theta_C = 0.25$, the threshold of the cosine similarity of an aggregate produced by a committee member at which point it is not considered by the leader.
- $\theta_R = 3$, the threshold on the ratio of successful (s_j) to unsuccessful interactions (f_j). A participant j is no longer considered to be a committee member or leader when $f_j > 3 \cdot s_j$. Setting this threshold at 3 allows for (coincidentally) failed connections or poor aggregates.
- $\epsilon = 0.05$, the stopping condition defined as a proportion of the minimum distance between the initial centroids such that it is independent of the dataset being trained on. Let δ be the minimum distance between initial centroids, then the global learning process has reached its stop condition if the average Euclidean distance that centroids move during a learning round is less than $\epsilon \cdot \delta$.
- Network stability was kept at 90% such that whenever there is an interaction between devices, either device has a 10%

Name	No. records	Features	Description
Blobs	500	2	Synthetic data having a distribution such that it lends itself well to clustering.
Breast cancer Wisconsin	569	30	Dataset describing features computed from a digitized image of a fine needle aspirate (FNA) of a breast mass.
Forest cover types	581012	54	Dataset describing soil cover measurements and the accompanying forest cover type.

Table 3: Datasets used in experiments.

chance to disconnect. Next time they are prompted to interact with a device, they have a 90% chance to reconnect.
- The transmission speed of devices is sampled from a uniform distribution ranging from 1 to 70,000 kb/s.
- The computation power of devices is sampled from a uniform distribution ranging from 0.1 to 4.0. To obtain the time spent on an action by a (simulated) device, the actual time spent is divided by their computation power.

7.2 Simulations

This section describes the experiments carried out to assess the performance of BC-FL k-means using the built prototype. Specifically, seven different experiments were carried out to observe the effects on model quality, convergence, and runtime. Moreover, we compared the model quality of centralized k-means, traditional federated k-means, and BC-FL k-means.

The experiments carried out can be summarized as follows:

E1 Comparison between federated and centralized k-means.
E2 Differing number of global clusters.
E3 Differing number of features in the dataset.
E4 Differing number of records in the dataset.
E5 Influence of the reputation system with differing proportions of malicious devices.
E6 Scalability of blockchain.
E7 Scalability of BC-FL k-means.

Experiment **E1** was carried out to investigate whether federated k-means maintains the same clustering quality as centralized k-means. If this is the case, that would mean that it is viable to perform k-means clustering in the setting of federated learning.

Experiments **E2** to **E4** were posed to investigate the scalability of BC-FL k-means with respect to number of global clusters, number of features in the dataset, and number of records in the dataset. The time complexity of k-means is $O(t \cdot k \cdot n \cdot d)$, where t is the number of iterations, k is the number of clusters, n is the number of records and d is the number of features in the dataset [23]. This suggests that changing the number of clusters, records or features has an equally large effect on runtime. Moreover, in BC-FL k-means specifically, the validation step at the committee members often forms the bottleneck, as its time complexity is $O(u \cdot k \cdot n \cdot d)$, where u is the number of updates obtained by a committee member. t is always equal to 1 at committee members as they are only required to perform one step of k-means to validate an update. We want to investigate whether, in practice, the runtime grows according to the presented time complexity.

Experiment **E5** was carried out to investigate how well the reputation system is able to investigate malicious devices and guarantee convergence. These malicious devices attempt to poison the global model. As data owners and committee members, they produce entirely random updates. In the role of leader, however, they produce a global update based on the aggregates they obtain from committee members. They essentially flip that global update such that the global centroids move in the opposite direction. Moreover, we are interested in whether the use of the reputation system negatively affects model quality if no malicious devices are present.

Experiment **E6** aims to investigate the effect of the number of blocks on the runtime of BC-FL k-means. If BC-FL k-means is unaffected by the number of blocks, this would suggest that its consensus protocol is already more scalable than proof-of-work consensus. To investigate this, we start the global learning process using an existing blockchain.

Experiment **E7** was posed to investigate the scalability of BC-FL k-means broadly. To this end, we investigate the runtime when using a dataset having many records, namely the forest cover type dataset [1].

In the evaluation, we distinguish between cases dealing with IID and non-IID distributed data. It is worth noting that we treat non-IID distributed data as each device having data only belonging to a single cluster. This is a strict interpretation of non-IID distributed data, as in practice it is more likely that non-IID distributions translate to devices having data that does not describe all clusters [30].

The simulations are run first and foremost to evaluate the performance of BC-FL k-means. On top of that, we investigate the requirement of heterogeneity and context-awareness, as described in Table 2. To account for the random initialization of centroids, each simulation setting was run 100 times unless specified otherwise. Moreover, the default setting for the number of devices was 20 having a role division of 12, 5, and 3 for the number of data owners, committee members, and leaders respectively. The time spent is computed as if the devices ran concurrently, meaning the time spent at each step is computed by taking the maximum time spent by a device fulfilling that step.

7.3 Performance Analysis

Performance was evaluated in terms of quality and convergence of the global k-means clustering model and time spent. The quality is assessed in terms of average silhouette score and Davies-Bouldin index, combining the data from each device to create a global dataset used for evaluation. The average silhouette score is defined for the interval [-1, 1] and a higher value corresponds with a better clustering. The Davies-Bouldin index, on the other hand, is defined for the interval [0, 1] and a lower value corresponds with a better

clustering. Using these measures, BC-FL k-means is comparable with centralized k-means as well as federated k-means [52]. Moreover, the time spent was computed by adding the maximum time spent per step for each step during every learning round. This is done to estimate the time spent if the devices ran in parallel.

The remainder of this section describes a comparison with traditional federated and centralized k-means and further results in terms of quality, convergence and time spent separately. Each subsection presents several interesting results based on the simulations that were run as described in Section 7.2.

7.3.1 Comparison between Federated and Centralized k-means. An important assessment of the viability of BC-FL k-means is to compare its resulting model quality with traditional federated and centralized k-means. For the centralized k-means algorithm to compare with, we used Scikit Learn's implementation [2] without partitioning the data. For the federated k-means algorithm, we use an existing implementation [52] that is configured to use its own (NumPy-generated) synthetic data. That dataset is different from the Blobs presented in Table 3, but we consider it because the existing implementation may have been optimized with that synthetic dataset in mind.

The existing implementation presented by the authors of [52] differs from our federated k-means algorithm in two important ways:

(1) Number of local epochs, which are the number of local iterations of k-means carried out by data owners each global learning round. The number of local epochs is set to be equal to 5 by the authors of [52] while we set it at 1.
(2) Cluster size sharing, where the number of records belonging to a particular cluster are disclosed. In the context of potential information leakage, we want to limit the sharing of data. The authors of [52], however, chose to share the number of records belonging to each cluster for each device in their federated k-means algorithm. This leaks information about the data residing at each device. On the other hand, the existing implementation is able to use this information to update the global centroids more meaningfully.

We ran centralized k-means, federated k-means and BC-FL k-means using the synthetic dataset generated by [52] to compare them. This dataset is unlabelled. Because of this, we are unable to split it such that we end up with a non-IID distribution of data. We also ran centralized k-means, federated k-means and BC-FL k-means on the breast cancer dataset. As that dataset is labelled, we were able to distribute the data both IID and non-IID.

For BC-FL k-means, we only consider the setting where no malicious devices exist. We list the results in terms of the silhouette average and Davies-Bouldin index in Table 4. These results are averaged scores across multiple runs. The results for the breast cancer dataset are shown in Table 5. The resulting clusterings on the synthetic dataset are shown in Appendix A. The resulting clusterings on the breast cancer dataset are shown in Appendix B.

The time spent shown in Table 4 is the average time spent until convergence. The scores suggest that BC-FL k-means performs similar to the existing federated k-means implementation. Moreover, the scores achieved by BC-FL k-means are very similar to that of centralized k-means. However, federated k-means is almost 20 times faster than BC-FL k-means. If we exclude role assignment, BC-FL k-means spends 11.18 seconds on the actual learning process, which is still considerably slower than federated k-means.

The results for the breast cancer dataset, shown in Table 5, also suggest that BC-FL k-means performs similar to centralized k-means. However, this is no longer the case if the data is non-IID distributed. This is likely due to committee members providing feedback to data owners that have data that is very dissimilar to that of the committee members. Consequently, honest data owners are excluded from the global learning process. Comparing BC-FL k-means with federated k-means, on the other hand, we find that BC-FL k-means performs better with IID and non-IID distributed data. Notably, the difference between the runtime of federated k-means and BC-FL k-means is smaller for the breast cancer data than for the dataset synthesized by the authors of [52]. For the breast cancer dataset, federated k-means is 5 times faster than BC-FL k-means but achieves slightly worse results.

7.3.2 Number of Global Clusters. The number of global clusters is important to consider as having more (global) clusters can impact the time spent learning drastically, affecting the scalability of BC-FL k-means. To account for this, we already allow devices to match the global clusters with their local clusters.

Table 7 shows the time spent and number of iterations for several different numbers of global clusters on the Blobs dataset. For each of the settings, we averaged the time spent per iteration across 20 runs. As the required number of iterations increases in order to reach convergence as the number of global clusters increases, we conclude that the number of global clusters has a significant impact on runtime.

7.3.3 Number of Features. Using synthetic datasets to test BC-FL k-means, we are able to change the number of features without changing their variability. We investigate the effect of the number of features on runtime as this shows whether BC-FL k-means is well equipped to deal with datasets having many dimensions.

Table 6 shows the time spent and number of iterations on average for several different numbers of features on the Blobs dataset. For each of the settings, we averaged the time spent per iteration across 20 runs. The results suggest that the number of features does not significantly impact runtime.

7.3.4 Number of Records. Similar to the number of (global) clusters and features, we can also change the amount of records for the synthetic dataset. We were interested in whether this has a larger effect on runtime than either increasing the number of features or the number of clusters.

Table 8 shows the time spent on average and the number of iterations for several different amounts of (total) records on the Blobs dataset. We averaged the time spent across 20 runs. The results suggest that the number of records influences the runtime considerably, even though the number of iterations required to reach convergence remains more or less unaffected.

7.3.5 Influence of Reputation System. The influence of the reputation system should be assessed in cases when malicious devices exist within the network as well as when there are no malicious

k-means	Silhouette average	Davies-Bouldin index	Time spent (s)
Centralized	0.449	0.789	0.030
Federated	0.434	0.815	3.088
BC-FL	0.443	0.799	60.056

Table 4: Quality comparison between centralized, federated and BC-FL k-means ran on synthetic data from [52].

k-means	Silhouette average	Davies-Bouldin index	Time spent (s)
Centralized	0.673	0.554	0.104
Federated (IID)	0.547	0.628	13.80
Federated (non-IID)	0.443	0.797	15.82
BC-FL (IID)	0.673	0.535	74.94
BC-FL (non-IID)	0.590	0.596	70.12

Table 5: Quality comparison between centralized and BC-FL k-means ran on the breast cancer dataset.

No. features	Time spent (s)	no. iterations
2	15.07	12
6	12.08	6
10	8.90	6
60	12.99	10

Table 6: Time spent and number of iterations of BC-FL k-means ran on the Blobs dataset having different number of features.

No. global clusters	Time spent (s)	no. iterations
3	15.07	12
6	52.15	43
9	84.46	81

Table 7: Time spent and number of iterations on average of BC-FL k-means ran on the Blobs dataset having different number of global clusters.

devices. It is essential to know whether the presence of the reputation system slows down convergence or limits the quality of the global model as this is unwanted. To this end, we ran simulations both with and without malicious devices present and observed the results in terms of convergence, average silhouette score and Davies-Bouldin index of the global model. Figures describing the cumulative convergence of runs for proportions of malicious devices that are not shown here are shown in Appendix C. The role assignment of devices (distinguishing between malicious and honest) are shown in Appendix D.

Figure 6a shows the cumulative converged proportion of runs at every iteration when no malicious devices are present and data is IID distributed. Figure 6b shows the same setting with non-IID distributed data.

When data is IID distributed and no malicious devices exist, the global learning process converges in approximately 85% of runs when the reputation system is used. If it is not used, it converges in approximately 70% of runs. From Figure 6a, we also conclude that runs converge faster when the reputation system is used.

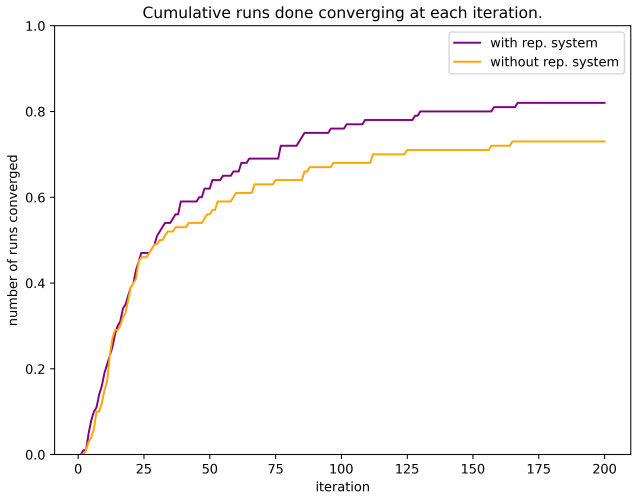
When data is non-IID distributed and no malicious devices exist, the global learning process converges in approximately 75% of runs when the reputation system is not used. If it is used, it converges in approximately 60% of runs. In other words, the reputation system negatively impacts convergence in this setting. This is likely due to feedback not properly reflecting whether a local update contributes poorly. However, many runs that do converge in case the reputation system is used, converge earlier (i.e. require less iterations) than runs do when the reputation system is not used.

Figure 7a shows the cumulative converged proportion of runs at every iteration when 40% of devices are malicious (8 out of 20) and data is IID distributed. The figure suggests that the reputation system has little effect in this setting as approximately 35% of runs converge with and without the reputation system. As the proportion of malicious devices increases, they affect feedback more. Consequently, the reputation system can no longer distinguish properly between malicious and honest devices. This would explain why we observe the same percentage of runs converging with and without the reputation system.

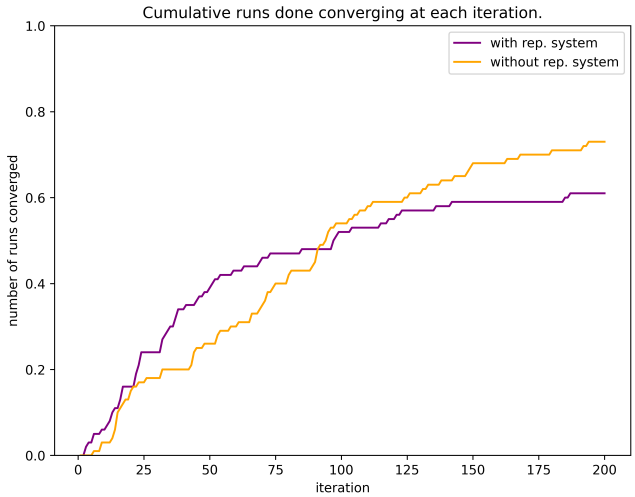
Figure 7b shows the cumulative converged proportion of runs at every iteration when 40% of devices are malicious with non-IID distributed data. When the reputation system is used, approximately 40% of runs converge. When it is not used, only approximately 25% of runs converge. This suggests that the reputation system can still discern random (malicious) updates from updates sent by

No. records (in total)	Time spent (s)	no. iterations
500	15.07	12
5000	18.14	14
50000	75.71	15

Table 8: Time spent and number of iterations on average of BC-FL k-means ran on the Blobs dataset having different number of records.

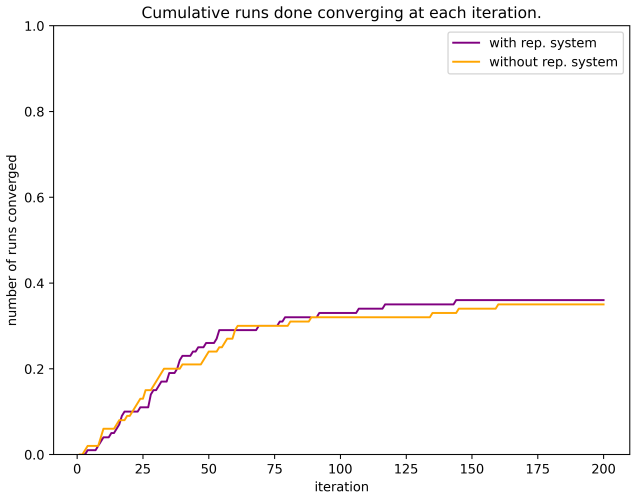


(a) Cumulative runs done converging with and without the reputation system with IID distributed data.

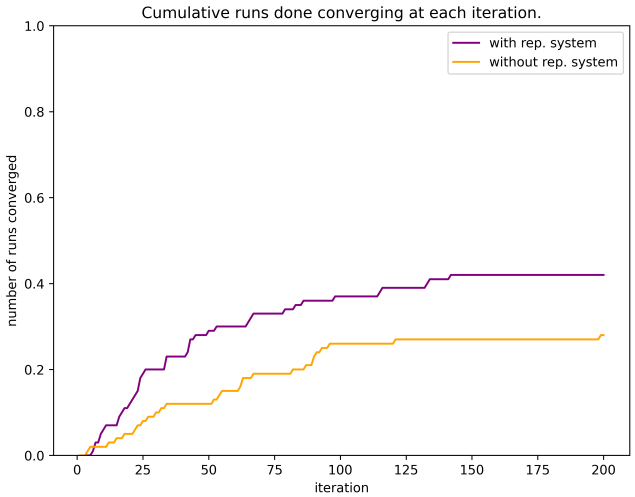


(b) Cumulative runs done converging with and without the reputation system with non-IID distributed data.

Figure 6: Cumulative runs done converging without malicious devices, with IID and non-IID distribution of data.



(a) Cumulative runs done converging with and without the reputation system with IID distributed data.

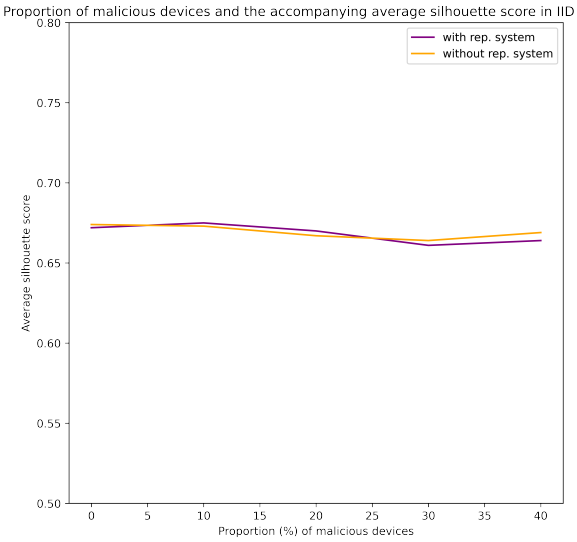


(b) Cumulative runs done converging with and without the reputation system with non-IID distributed data.

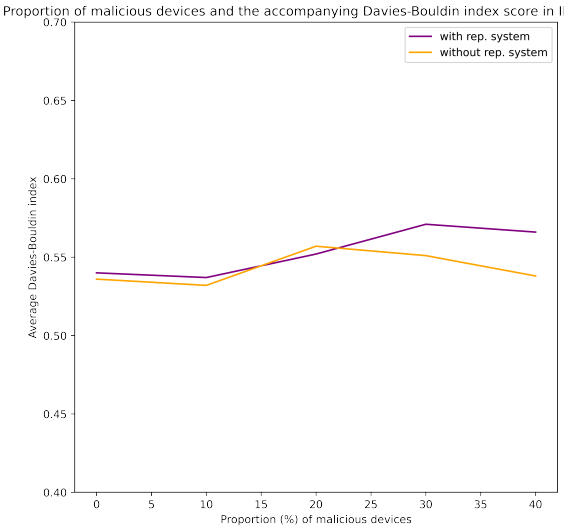
Figure 7: Cumulative runs done converging with 40% malicious devices, with IID and non-IID distribution of data.

honest data owners in non-IID setting. Notably, this means that the reputation system makes BC-FL k-means robust against malicious devices even in case data is non-IID distributed.

Figure 8a shows the average silhouette score and Figure 8b shows the Davies-Bouldin index for each proportion of malicious devices with IID distributed data. For IID distributed data, there is no significant observable difference between model quality and different

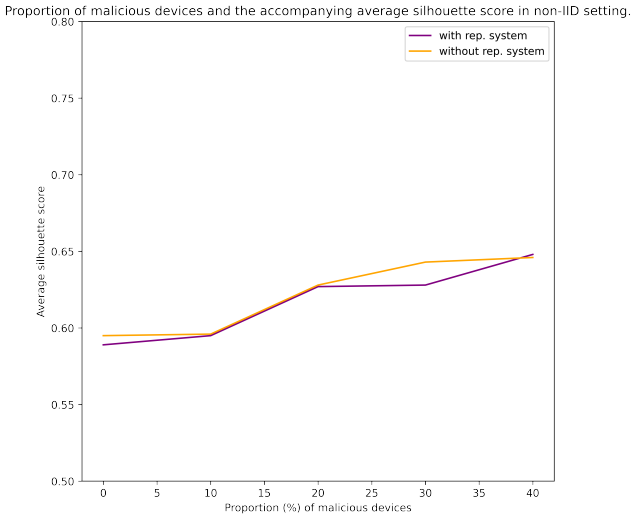


(a) Silhouette average per proportion of malicious devices.

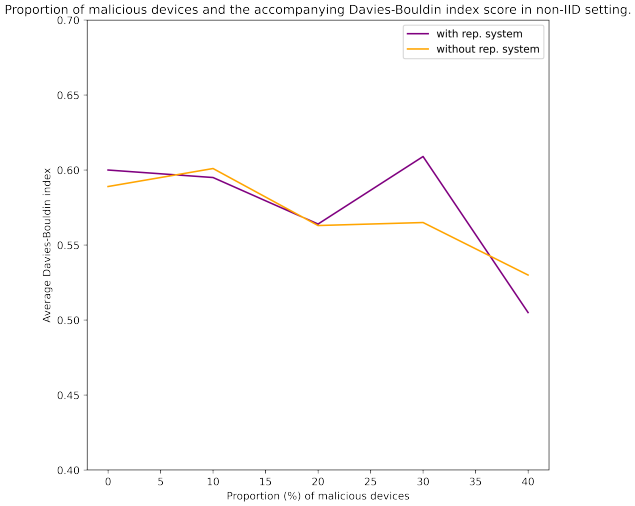


(b) Davies-Bouldin index per proportion of malicious devices.

Figure 8: Quality of k-means models per proportion of malicious devices with IID distribution of data for converged runs.



(a) Silhouette average per proportion of malicious devices.



(b) Davies-Bouldin index per proportion of malicious devices.

Figure 9: Quality of k-means models per proportion of malicious devices with non-IID distribution of data for converged runs.

proportion of malicious devices. This implies that in IID setting, the reputation system is always able to discern malicious updates from honest ones. As the data residing at each participant is sufficiently alike, it tends to be the case that a participant's update is not flagged as malicious if they are behaving honestly. In other words, the only updates being flagged as malicious are those from participants trying to poison the global model.

Figure 9a shows the average silhouette score and Figure 9b shows the Davies-Bouldin index for each proportion of malicious devices with non-IID distributed data. For non-IID distributed data, we observe that model quality increases if more malicious devices are present. As malicious devices try to poison the model by submitting random updates, this suggests that the reputation system allows BC-FL k-means to perform better because it can discern between random updates and genuinely poor updates. An update would be seen as poor if a committee member has data that is very dissimilar to that of a data owner from whom they receive an update. They would provide negative feedback to that data owner, but a device submitting random updates receives more negative feedback still.

7.3.6 Scalability of Blockchain. We investigated what would happen if we started the learning process with an existing chain already composed of many blocks. This is relevant as in many applications

No. blocks	Time spent per iteration (s)
0	1.02
100	1.20
500	1.34
1000	1.10

Table 9: Time spent on average per iteration of BC-FL k-means ran on the breast cancer dataset when starting with a blockchain having different numbers of blocks.

Reputation system	Devices per role	Time spent per iteration (s)
Used	80, 15, 5	103.41
Not used	80, 15, 5	97.80
Used	160, 30, 10	88.66
Not used	160, 30, 10	131.07

Table 10: Time spent on average per iteration of BC-FL k-means ran on the forest cover type dataset with and without the reputation system.

using blockchain the appendage of new blocks takes longer as mining difficulty increases. Moreover, each time a block is appended, the entire chain must be validated.

The time spent per iteration per number of blocks on the chain when starting the global learning process is displayed in Table 9. As the number of iterations required to converge depends on the centroids recorded in the latest block of the chain which was used as a starting point, we omitted it from the table. The results presented in the table suggest that the number of blocks does not significantly impact time spent. This is in line with our expectation as more blocks only results in having to check whether each block properly refers to the previous block. Moreover, during execution the number of blocks in BC-FL k-means is at most equal to the total number of iterations. The chain can then be appended to collaboratively if multiple participants want to continue the federated learning process, or the parameters can be extracted from a block and a local model can then be built.

7.3.7 Scalability of BC-FL k-means. A dataset describing forest cover types [1] was used to test the scalability of BC-FL k-means. The number of global clusters considered for this dataset was equal to 7, as that is the amount of forest cover types described in the datasest. We consider both the effect of the number of devices and the number of committee members validating each update in terms of time spent learning. In all simulations run, each committee member saw updates from half of all data owners. This was done to reduce learning times without drastically influencing global model quality.

Table 10 shows the time spent per setting and whether the reputation system was used. For the settings with 100 devices in total, we ran 100 iterations. For the settings with 200 devices in total, we ran 10 iterations and based the average time spent per iteration only on those 10 iterations. From these results, we conclude the size of the dataset has more of an effect on the time spent rather than the number of devices participating in the federated learning process. Moreover, most time was spent by committee members validating local updates. Lastly, it should be noted that the forest cover type dataset [1] has 581,012 records in total and has 54 features, as described in Table 3. On average, an iteration on the breast cancer dataset took 1.12 seconds. Running BC-FL k-means on the forest cover type dataset took 105.24 seconds, averaged across the four settings. Even though the dataset is more than 1,000 times larger than the breast cancer dataset, the runtime is only approximately 100 times larger.

8 DISCUSSION

In this section, the lessons learned from this research are presented and evaluated. We structure this section according to the research questions posed in Section 1 and evaluate the lessons learned given the requirements that were defined for federated learning systems in Table 2. This is done in Section 8.1. Moreover, certain limitations are presented in Section 8.2 that are worth considering when evaluating the applicability of blockchain and other methods introduced in this research to federated learning.

8.1 Lessons Learned

We reflect on the research questions we put forward in Section 1.1. Specifically, we reflect on our attempt to change the blockchain architecture to make it more suitable to federated learning in Section 8.1.1. We assess the (dis-)advantages of using a blockchain in federated learning in terms of scalability, privacy, security, resilience to attacks, and traceability in Section 8.1.2. Lastly, we reflect on our k-means clustering algorithm and unsupervised federated learning generally in Section 8.1.3.

8.1.1 Adaptations of Blockchain. Leveraging the collective knowledge of participants for calculating reputation (through validation) and building consensus works well with blockchain. Reputation values can be stored on-chain and retrieved if necessary. Maintaining these reputation values helps in making federated learning robust against poisoning attacks. A voting-based consensus protocol leverages the knowledge of participants about the global learning task to select the best global update to be propagated. Research into blockchain-based federated learning should keep this in mind,

rather than build further upon poorly scalable consensus protocols such as proof-of-work.

8.1.2 (Dis-)advantages of Blockchain. Blockchain aids in making traceable historical global models and reputation values while negatively impacting runtime. Moreover, using blockchain to facilitate federated learning has an insignificant effect on privacy, security, and resilience to attacks.

We argue that traceability of global model versions makes personalization of models easier, as participants can copy the version that performs best for them locally. In terms of runtime, it is worth noting that blockchain decentralizes the aggregation task. If a specific application of federated learning requires this, it may be worthwhile to use blockchain. Within BC-FL k-means, privacy is preserved by limiting how much data participants see. However, it can still be argued that additional techniques must be used to preserve privacy. Lastly, we made BC-FL k-means secure and resilient to attacks through means that are typically used in federated learning [30].

8.1.3 Unsupervised Federated Learning. Our implementation of federated k-means is able to achieve a performance similar to centralized k-means, meaning k-means has promising results within the setting of federated learning. Our federated k-means algorithm struggles to achieve a good quality clustering in the case of non-IID distributed data, but this is an ongoing problem in federated learning that has yet to be solved [30]. However, through the use of a reputation system, BC-FL k-means manages to produce clustering models that have a higher quality than those produced by the authors of [52], against which we compared our performance. As such, BC-FL k-means is sufficiently aware of the heterogeneity of data. Lastly, the proposed algorithm may be generalized to fit traditional federated learning by substituting the committee with a central server.

The merit of performing blockchain-facilitated unsupervised federated learning generally, however, remains an open case. k-means is a relatively simple algorithm whereas other unsupervised models are more difficult to execute in a blockchain-based framework. To exemplify this, FREPD [20] proposes a robust federated learning framework for variational autoencoders using probability distribution functions to determine whether an update is malicious. If this process would be decentralized, every participant that can be selected to validate updates would be required to maintain these probability distribution functions. Depending on the specific implementation, this would not only increase runtime but also have a decreased accuracy in identifying malicious updates.

8.2 Limitations

The simulations run using the prototype do not translate immediately to a real-world distributed system, especially considering that the simulations were run on a single machine. As the parameter settings per device regarding computation power and link speeds are randomized within a range, the simulation comes reasonably close. On the other hand, however, there is a single percentage with respect to connection stability. Thus, we do not encapsulate fully that devices may have connections with varying stability.

In the simulations, we assume we know the bounds on the values that the features take in the given datasets. However, this may not be the case in practice. To this end, a secure protocol to (randomly) select k initial centroids with multiple parties must be adopted [11].

We have not investigated the trade-off between the number of updates obtained by a committee member and runtime exactly. As committee members see more updates, more feedback is given to each data owner. As such, it becomes easier to discern malicious participants from honest ones. However, if many participants supply updates, this may produce a bottleneck at the committee members as they are tasked with the validation of every single one.

Due to limited available research on blockchain-based federated learning, we are unable to compare the scalability of our consensus protocol with existing solutions. Many existing solutions pertain to neural networks and thus tend to require more time to be built collaboratively due to the number of parameters. As such, it is difficult to assess the scalability of our consensus protocol.

Although information leakage is limited through the use of a committee-based consensus protocol, where only a limited amount of participants see local updates, it can still be argued that this is unwanted. To ensure information leakage is kept to a minimum, techniques aiming to preserve privacy, such as homomorphic encryption [6], (local) differential privacy [53] or secure multi-party computation [48] must be adopted.

9 RELATED WORK

This section explores existing research within the field of federated learning. We distinguish between research on traditional federated learning and research on blockchain-based federated learning. We also list the requirements, which were first presented in Table 2, met by the related solutions. We aim to showcase how different techniques can help meet the requirements put forward and thus illustrate how solutions differ.

Firstly, we summarize how our work meets the requirements in Section 9.1. The remainder of this section differentiates between traditional federated learning solutions, described in Section 9.2, and blockchain-based federated learning solutions, described in Section 9.3.

9.1 Our Work

Our work (BC-FL k-means) meets requirement **R1** by allowing each participant to select the version of the global model that performs best for them locally and continue training it locally. Moreover, through the committee-based consensus protocol we decentralize the aggregation task, thus meeting requirement **R2**. We meet requirement **R3** by maintaining reputation values based on local update quality and the reliability with which devices communicate. Through the selection of devices that are more reliable in their communication, BC-FL k-means takes into account the context in which federated learning tends to take place (**R5**). We have shown that our work still performs well in case of non-IID distributed data, thus we meet requirement **R4**. Through the committee-based consensus protocol, we argue that we limit information leakage sufficiently by limiting the number of participants that see local updates (**R6**). Lastly, we have shown that through the maintenance of contribution values, the validation step at committee members

ID	Requirement	[51]	[32]	[48]	[15]	Our work
R1	Personalization	×	×	×	✓	✓
R2	Decentralization	×	×	o	×	✓
R3	Trust	✓	✓	×	×	✓
R4	Heterogeneity-awareness	×	×	×	✓	✓
R5	Context-awareness	✓	✓	✓	×	✓
R6	Secure	×	×	✓	✓	✓
R7	Robust	✓	✓	×	×	✓

Table 11: Requirements fulfilled by related solutions within traditional federated learning. '✓' means a requirement is fulfilled. '×' means it is not fulfilled. 'o' means that a requirement is partially fulfilled.

and the checking of the cosine similarity of aggregates with respect to the global updates produced by leaders, BC-FL k-means is sufficiently robust against poisoning attacks (**R7**).

9.2 Traditional Federated Learning

We researched several existing frameworks within traditional federated learning that aim to remedy existing problems in federated learning. To ensure that the comparison between existing solutions remains meaningful, we focus on solutions that consider trust and security. Several proposed solutions are discussed in this section, with a summary of the requirements met by each of them showcased in Table 11.

The work done in [51] focuses on the setting of Internet-of-Things by taking into account limited resource availability. To do so, the proposed solution adopts a Beta reputation model that, together with a measure of connection reliability, is used in its scheduling policy. Maintaining these reputation values meets requirement **R3** and using them in the scheduling policy meets requirement **R5**. Moreover, as devices are excluded if they contribute poorly, this protects against poisoning attacks, meeting requirement **R7**. However, the proposed solution uses a Beta reputation model for the assessment of local updates, which produces a binary assessment with respect to update quality. This is unwanted, as it leads to situations where participants barely contributing positively and participants contributing very positively cannot be distinguished. Instead, a reputation model allowing fine-grained ratings for update quality assessment should be adopted. Moreover, the proposed solution only provides the most recent version of the global model to participants, thus failing to meet requirement **R1**. Lastly, they require a central server (**R2**), only consider uniform data distributions (**R4**) and do not consider information leakage at all (**R6**).

Another solution focusing on the setting of IoT is presented by Kang et. al. [32]. They consider a selection scheme for participants to supply updates. For this, they calculate the reputation of a device based on a (multi-weight) subjective logic model and store the resulting reputation on a blockchain. On top of this, they apply a poisoning attack detection scheme, specifically Reject on Negative Influence [49] and FoolsGold [17]. As such, requirements **R3** and **R7** are met by the presented solution. Moreover, as interaction frequency and reliability are considered in their subjective logic model, the proposed solution meets requirement **R5**. However, the model is not provided to all participants, thus failing to meet requirement **R1**. A central aggregator is still used, which means the provided solution fails to meet requirement **R2**. Lastly, the authors only consider a uniform distribution of data (**R4**) and do not consider information leakage at all (**R6**).

POSEIDON [48] focuses on privacy-preserving federated neural network learning. Specifically, POSEIDON is the first system that enables quantum-resistant distributed learning. It uses multi-party homomorphic encryption (MPHE) to preserve privacy, thus meeting requirement **R6**. Notably, they achieve partial decentralization of the aggregation task (**R2**) through the use of MPHE as they require multiple servers that together securely aggregate updates and produce the global update. Through the application of optimization techniques, it is sufficiently context-aware (**R5)** as its time complexity scales linearly rather than quadratically as a result, which is typically the case for solutions adopting multi-party computation [41]. Although POSEIDON's contribution in terms of privacy preservation and security is promising, it does have several shortcomings. POSEIDON does not provide all participants with the trained neural network (**R1**). On top of that, they do not maintain the reputation of participants (**R3**) and do not consider them having dissimilar data (**R4**). Lastly, they do not consider poisoning attacks (**R7**).

FedHealth [15] describes a novel approach to federated learning for neural networks by showcasing how transfer learning can be done in a distributed setting. Specifically, they focus on the setting of wearable healthcare devices for activity recognition. Their motivation is that activity recognition warrants the use of personalized models for individuals and a global model can aid in improving these personalized models. Through transfer learning, knowledge attained by the global model can be transferred to a personalized model in case that knowledge is general enough to be descriptive of the entire population. As such, FedHealth meets requirement **R1**. However, the effect of transfer learning within federated learning depends on the application area and thus the presented approach may not be generalizable. By limiting the layers of the neural networks that are used for the federated learning part, it is sufficiently aware of heterogeneity in data (**R4**). Through the use of homomorphic encryption, FedHealth meets requirement **R6**. As FedHealth still requires a central aggregator, requirement **R2** is not met. Moreover, FedHealth does not maintain reputation values (**R3**) and do not consider the context sufficiently (wearable healthcare devices) such that the federated learning can be optimized with respect to runtime (**R5**). Lastly, FedHealth does not consider poisoning attacks (**R7**).

ID	Requirement	[57]	[14]	[38]	[7]	Our work
R1	Personalization	×	✓	×	o	✓
R2	Decentralization	✓	✓	×	✓	✓
R3	Trust	✓	×	✓	✓	✓
R4	Heterogeneity-awareness	×	×	×	✓	✓
R5	Context-awareness	o	o	×	✓	✓
R6	Secure	✓	✓	✓	o	✓
R7	Robust	✓	✓	✓	✓	✓
n.a.	**Blockchain**	Consortium	Public	Private	Consortium	Consortium

Table 12: Requirements fulfilled by related solutions within blockchain-based federated learning. '✓' means a requirement is fulfilled. '×' means it is not fulfilled. 'o' means that a requirement is partially fulfilled.

9.3 Blockchain-based Federated Learning

Besides existing solutions within traditional federated learning, we also researched existing blockchain-based federated learning solutions. These solutions vary considerably as each deploys a different consensus protocol to validate or verify global updates. Different consensus protocols can lead to varying results in terms of scalability, resilience to attacks, and runtime. Several proposed solutions are laid out in this section, with a summary of the requirements each of them meets showcased in Table 12.

The authors of [57] present a solution focusing specifically on IoT. They differentiate between *manufacturers* and *customers*. The manufacturers can raise requests for a machine learning model to be built to encapsulate customers' (device) behavior. As customers do not gain access to the global model, we conclude that requirement **R1** is not met. They use differential privacy to minimize information leakage, thus meeting requirement **R6**. Moreover, they use the Multi-KRUM aggregation scheme [9] which deals effectively with poisoning attacks (**R7**). Using this aggregation scheme, they also assess customers' reputation (**R3**). The proposed solution only takes into account context partially by having fog servers execute most computations, regardless of whether a device is able to perform that computation locally or not. Moreover, the proposed solution does not consider non-IID distribution of data, thus it does not meet requirement **R4**. Lastly, they use Algorand's consensus protocol which is based on Proof-of-Stake and Byzantine fault tolerance (BFT). It is a committee-based consensus protocol that performs relatively well in the context of federated learning and decentralizes the aggregation step (**R2**).

LearningChain [14] presents one of the earliest applications of blockchain to the setting of federated learning. They use local differential privacy to minimize information leakage (**R6**) and use an l-nearest aggregation scheme to limit the impact of poisoning attacks (**R7**). Moreover, as each participant keeps a local copy of the ledger, they are able to retrieve any historical version of the global model, thus meeting **R1**. LearningChain decentralizes the aggregation task by having each *computing node* perform the aggregation of local updates from a subset of participants (**R2**). However, they do not maintain reputation values (**R3**) and do not take into account non-IID distribution of data (**R4**). Moreover, they only partially consider the context in which federated learning occurs by having devices compete according to the Proof-of-Work consensus protocol (**R5**). The winner of PoW can compute the global update and append a block to the chain.

Lyu et al. [38] describe a federated learning framework that focuses on fairness. They define fairness as rewarding participants proportional to model quality improvement. Notably, rather than a monetary reward, they provide a worse version of the global model to participants contributing poorly to global learning. This means that they do not meet the requirement of personalization (**R1**). They use local differential privacy to limit information leakage, thus meeting requirement **R6**. However, they assess the model quality improvement of participants based on the supplied update after perturbation according to differential privacy. This may result in a distorted view of the contribution of a participant and thus sabotage the fairness of the proposed solution. They maintain these values as a participant's credibility, which can be seen as their reputation (**R3**). However, they do not meet requirement **R4** as they are quite strict to participants having dissimilar data. This strictness does entail that they meet requirement **R7** as poisoning attacks are thwarted. They decentralize the aggregation task by using multi-party homomorphic encryption, thus meeting requirement **R2**. Lastly, the proposed solution is very computationally heavy through the use of differential privacy and multi-party homomorphic encryption. As such, they insufficiently take into account the context in which federated learning is usually performed, thus not meeting requirement **R5**.

Behera et al. [7] showcase how smart contracts may be used in an Ethereum-based blockchain to incentivize participants to join the federated learning process. In the paper, *federated contribution* is defined as a measure of reputation (**R3**) combining the number of records and the similarity of a local update compared to the global model update. By using federated contribution in the selection of local updates to be used for the aggregation, they meet requirement **R7**. As smart contracts make the federated contribution calculation and the accompanying rewarding scheme entirely transparent, this helps incentivize participants to join the federated learning process. They only achieve partial security (**R6**) as they use RSA cryptography [50] to securely send local model updates. They then fail to consider that the receiver of those local updates may be compromised themselves. Requirement **R2** is not met because they use a central server for the aggregation task and computation of federated contribution through smart contracts. Moreover, requirement **R1** is met partially through the central server sending each

participant the newest version of the global model. It can also be stated that they manage to take the context in which federated learning typically occurs into account by outsourcing the aggregation task to a central server (**R5**). Lastly, Behera et al. show that their proposed solution takes into account the heterogeneity of data by showing that they manage to obtain accurate models when data is non-IID distributed.

10 CONCLUSIONS AND FUTURE WORK

We presented a blockchain-based framework for federated learning in which we applied our federated k-means clustering algorithm. The framework included a committee-based consensus protocol designed with collaborative learning in mind and a reputation system to improve convergence and global model quality.

We have shown that BC-FL k-means performs similarly to centralized k-means if data is IID distributed. Moreover, the reputation system helps reach convergence in many cases, even when data is non-IID distributed and the proportion of malicious devices is high. The use of blockchain alongside federated learning, on the other hand, comes with an increase in runtime compared to traditional federated learning and only marginally helps in alleviating issues in traditional federated learning.

Future research should be concerned with arbitrarily partitioned datasets. In this research, we have only considered horizontally partitioned datasets. However, a generic solution to k-means clustering in the setting of federated learning should also work for vertically and arbitrarily partitioned datasets. Such solutions exist, but do not operate in the framework of federated learning [55] [25].

Existing blockchains currently developing the possibility to adopt pluggable consensus protocols may help in the application of decentralized machine learning. Currently, these blockchains, such as Hyperledger Fabric and Sawtooth [5], allow a limited number of consensus protocols to be plugged. As more development is done with respect to blockchains such as those, however, they may allow completely tuneable consensus protocols and thus offer a more scalable solution for blockchain-based federated learning and decentralized machine learning generally.

Lastly, smart contracts should be investigated to make transparent the computation of trust values, aggregates, and global updates such that participants are more likely to join the federated learning process as they gain confidence from the information they have been provided.

REFERENCES

[1] Covertype data set. https://archive.ics.uci.edu/ml/datasets/covertype. Accessed: 2022-02-01.

[2] Scikit-learn k-means clustering. https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html. Accessed: 2022-01-01.

[3] Scikit-learn toy datasets. https://scikit-learn.org/stable/datasets/toy_dataset.html. Accessed: 2022-02-01.

[4] General data protection regulation (gdpr), regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec, 2016.

[5] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, Srinivasan Muralidharan, Chet Murthy, Binh Nguyen, Manish Sethi, Gari Singh, Keith Smith, Alessandro Sorniotti, Chrysoula Stathakopoulou, Marko Vukolic, and Jason Yellick. Hyperledger fabric: A distributed operating system for permissioned blockchains. 01 2018.

[6] Muhammad Asad, Ahmed Moustafa, and Takayuki Ito. Fedopt: Towards communication efficiency and privacy preservation in federated learning. *Applied Sciences*, 10:1–17, April 2020.

[7] Monik Raj Behera, Sudhir Upadhyay, and Suresh Shetty. Federated learning using smart contracts on blockchains, based on reward driven approach. *ArXiv*, 2021.

[8] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin B. Calo. Analyzing federated learning through an adversarial lens. *CoRR*, abs/1811.12470, 2018.

[9] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[10] Mic Bowman, Debajyoti Das, Avradip Mandal, and Hart William Montgomery. On elapsed time consensus protocols. *IACR Cryptol. ePrint Arch.*, 2021:1–47, 2021.

[11] Paul Bunn and Rafail Ostrovsky. Secure two-party k-means clustering. In *Proceedings of the 14th ACM Conference on Computer and Communications Security*, CCS '07, page 486–497, New York, NY, USA, 2007. Association for Computing Machinery.

[12] Thiago Candido. A technical introduction to blockchain, October 2021.

[13] Hang Chen, Syed Ali Asif, Jihong Park, Chien-Chung Shen, and Mehdi Bennis. Robust blockchained federated learning with model validation and proof-of-stake inspired consensus. *AAAI 2021 Workshop - Towards Robust, Secure and Efficient Machine Learning*, 01 2021.

[14] Xuhui Chen, Jinlong Ji, Changqing Luo, Weixian Liao, and Pan Li. When machine learning meets blockchain: A decentralized, privacy-preserving and secure design. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 1178–1187, 2018.

[15] Yiqiang Chen, Xin Qin, Jindong Wang, Chaohui Yu, and Wen Gao. Fedhealth: A federated transfer learning framework for wearable healthcare. *IEEE Intelligent Systems*, 35(4):83–93, 2020.

[16] Ali Dorri and Raja Jurdak. Tree-chain: A fast lightweight consensus algorithm for iot applications. *CoRR*, abs/2005.09443, 2020.

[17] Clement Fung, Chris J. M. Yoon, and Ivan Beschastnikh. Mitigating sybils in federated learning poisoning. *CoRR*, abs/1808.04866, 2018.

[18] Avishek Ghosh, Justin Hong, Dong Yin, and Kannan Ramchandran. Robust federated learning in a heterogeneous environment. *arXiv*, 06 2019.

[19] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[20] Zhipin Gu, Liangzhong He, Peiyan Li, Peng Sun, Jiangyong Shi, and Yuexiang Yang. Frepd: A robust federated learning framework on variational autoencoder. *Computer Systems Science and Engineering*, 39:307–320, 01 2021.

[21] Runchao Han. On the performance of distributed ledgers for internet of things. *Internet of Things*, 10:100087, 08 2019.

[22] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[23] J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

[24] Geetha Jagannathan, Krishnan Pillaipakkamnatt, and Rebecca Wright. A new privacy-preserving distributed k -clustering algorithm. volume 2006, 04 2006.

[25] Geetha Jagannathan and Rebecca N. Wright. Privacy-preserving distributed k-means clustering over arbitrarily partitioned data. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, KDD '05, page 593–599, New York, NY, USA, 2005. Association for Computing Machinery.

[26] Malhar Jere, Tyler Farnan, and Farinaz Koushanfar. A taxonomy of attacks on federated learning. *IEEE Security & Privacy*, PP, 12 2020.

[27] Somesh Jha, Louis Kruger, and Patrick McDaniel. Privacy preserving clustering. volume 3679, pages 397–417, 09 2005.

[28] Yuang Jiang, Shiqiang Wang, Bong Jun Ko, Wei-Han Lee, and Leandros Tassiulas. Model pruning enables efficient federated learning on edge devices. *CoRR*, abs/1909.12326, 2019.

[29] Audun Jøsang. Artificial reasoning with subjective logic. In *Proceedings of the 2nd Australian Workshop on Commonsense Reasoning*, 2008.

[30] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawit, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konecný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo,

Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Hang Qi, Daniel Ramage, Ramesh Raskar, Mariana Raykova, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 4(1):1–121, 2021.

[31] Sepandar Kamvar, Mario Schlosser, and Hector Garcia-molina. The eigentrust algorithm for reputation management in p2p networks. *The EigenTrust Algorithm for Reputation Management in P2P Networks*, April 2003.

[32] Jiawen Kang, Zehui Xiong, Dusit Tao Niyato, Yuze Zou, Yang Zhang, and Mohsen Guizani. Reliable federated learning for mobile networks. *IEEE Wireless Communications*, 27:72–80, 2020.

[33] Latif U. Khan, Walid Saad, Zhu Han, Ekram Hossain, and Choong Seon Hong. Federated learning for internet of things: Recent advances, taxonomy, and open challenges. *CoRR*, abs/2009.13012, 2020.

[34] Kejiao Li, Hui Li, Hanbing Wang, Hui yao An, Ping Lu, Peng Yi, and Fusheng Zhu. Pov: An efficient voting-based consensus algorithm for consortium blockchains. In *Frontiers in Genetics*, 2020.

[35] Tian Li, Anit Kumar Sahu, Ameet S. Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37:50–60, 2020.

[36] Yuzheng Li, Chuan Chen, Nan Liu, Huawei Huang, Zibin Zheng, and Qiang Yan. A blockchain-based decentralized federated learning framework with committee consensus. *IEEE Network*, 35(1):234–241, 2021.

[37] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. *CoRR*, abs/1805.12185, 2018.

[38] L. Lyu, J. Yu, K. Nandakumar, Y. Li, X. Ma, J. Jin, H. Yu, and K. Ng. Towards fair and privacy-preserving federated deep models. *IEEE Transactions on Parallel & Distributed Systems*, 31(11):2524–2541, November 2020.

[39] H.B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Agüera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

[40] Fatima Meskine and Safia Nait-Bahloul. Privacy preserving k-means clustering: A survey research. *International Arab Journal of Information Technology*, 9, 03 2012.

[41] Christian Mouchet, Juan Troncoso-Pastoriza, Jean-Philippe Bossuat, and Jean-Pierre Hubaux. Multiparty homomorphic encryption from ring-learning-with-errors. Cryptology ePrint Archive, Paper 2020/304, 2020. https://eprint.iacr.org/2020/304.

[42] Tim Muller and Patrick Schweitzer. On beta models with trust chains. In *IFIPTM*, 2013.

[43] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Cryptography Mailing list at https://metzdowd.com*, 03 2009.

[44] Thien Duc Nguyen, Phillip Rieger, Hossein Yalame, Helen Möllering, Hossein Fereidooni, Samuel Marchal, Markus Miettinen, Azalia Mirhoseini, Ahmad-Reza Sadeghi, T. Schneider, and Shaza Zeitouni. Flguard: Secure and private federated learning. *IACR Cryptol. ePrint Arch.*, 2021:25, 2021.

[45] Frank Nielsen. *Introduction to HPC with MPI for Data Science*, pages 185–213. Springer, February 2016.

[46] Diego Ongaro and John Ousterhout. In search of an understandable consensus algorithm. In *Proceedings of the 2014 USENIX Conference on USENIX Annual Technical Conference*, USENIX ATC'14, pages 305—-320, USA, 2014. USENIX Association.

[47] Maria Rigaki and Sebastian Garcia. A survey of privacy attacks in machine learning. *CoRR*, abs/2007.07646, 2020.

[48] Sinem Sav, Apostolos Pyrgelis, Juan Ramón Troncoso-Pastoriza, David Froelicher, Jean-Philippe Bossuat, Joao Sousa, and Jean-Pierre Hubaux. Poseidon:privacy-preserving federated neural network learning. *Network and Distributed Systems Symposium*, pages 1–24, September 2020.

[49] Muhammad Shayan, Clement Fung, Chris J. M. Yoon, and Ivan Beschastnikh. Biscotti: A ledger for private and secure peer-to-peer machine learning. *CoRR*, abs/1811.09904, 2018.

[50] Gurpreet Singh and Supriya Kinger. A study of encryption algorithms (rsa, des, 3des and aes) for information security. *International Journal of Computer Applications*, 67:33–38, 04 2013.

[51] Zhendong Song, Hongguang Sun, Howard H. Yang, Xijun Wang, Yan Zhang, and Tony Q. S. Quek. Reputation-based federated learning for secure wireless networks. *IEEE Internet of Things Journal*, pages 1–15, 2021.

[52] Oskar J. Triebe and Ram Rajagopal. Federated k-means clustering algorithm. https://github.com/ourownstory/federated_kmeans. Accessed: 2022-01-01.

[53] Stacey Truex, Ling Liu, Ka Ho Chow, Mehmet Emre Gursoy, and Wenqi Wei. Ldp-fed: Federated learning with local differential privacy. *CoRR*, abs/2006.03637, 2020.

[54] Muhammad Habib ur Rehman, Khaled Salah, Ernesto Damiani, and Davor Svetinovic. Towards blockchain-based reputation-aware federated learning. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 183–188, 2020.

[55] Jaideep Vaidya and Chris Clifton. Privacy-preserving k-means clustering over vertically partitioned data. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, page 206–215, New York, NY, USA, 2003. Association for Computing Machinery.

[56] Chang Xia, Jingyu Hua, Wei Tong, and Sheng Zhong. Distributed k-means clustering guaranteeing local differential privacy. *Computers & Security*, 90:101699, 12 2019.

[57] Yang Zhao, Jun Zhao, Linshan Jiang, Rui Tan, Dusit Niyato, Zengxiang Li, Lingjuan Lyu, and Yingbo Liu. Privacy-preserving blockchain-based federated learning for iot devices. *IEEE Internet of Things Journal*, 8(3):1817–1829, 2021.

[58] Zibin Zheng, Shaoan Xie, Hong-Ning Dai, Xiangping Chen, and Huaimin Wang. An overview of blockchain technology: Architecture, consensus, and future trends. 06 2017.

[59] Zibin Zheng, Shaoan Xie, Hong-Ning Dai, Xiangping Chen, and Huaimin Wang. Blockchain challenges and opportunities: A survey. *International Journal of Web and Grid Services*, 14:352, 10 2018.

APPENDIX

A CLUSTERING RESULTS SYNTHETIC DATA

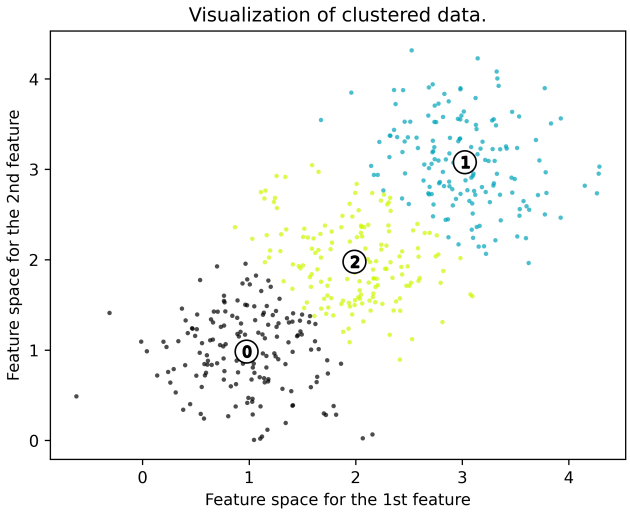


Figure 10: Clustering result of centralized clustering on synthetic dataset from [52].

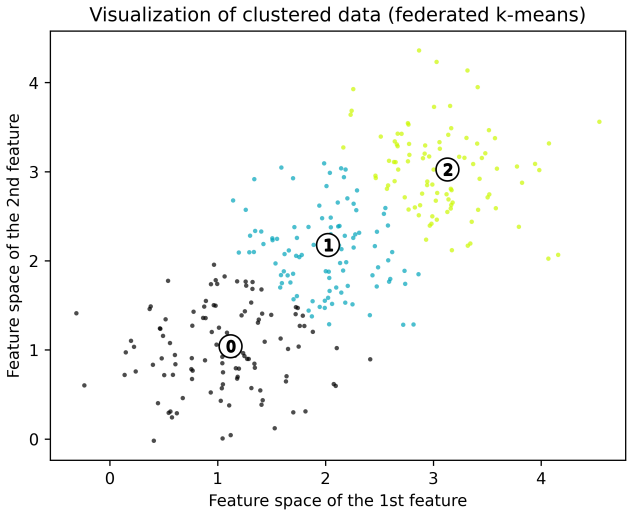


Figure 11: Clustering result of federated k-means on synthetic dataset from [52].

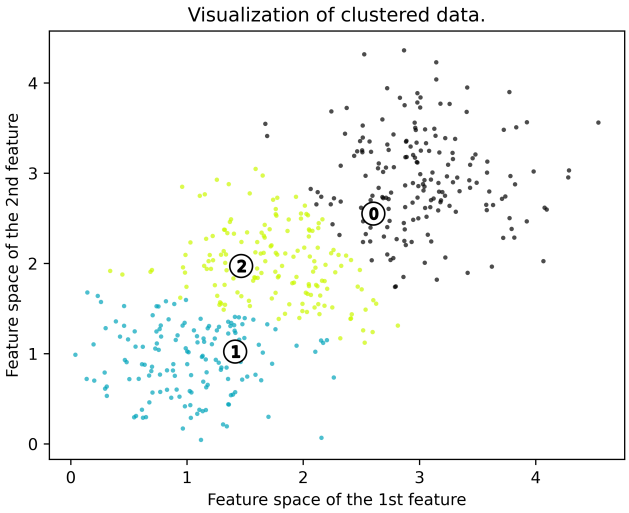


Figure 12: Clustering result of BC-FL k-means on synthetic dataset from [52].

B CLUSTERING RESULTS BREAST CANCER DATA

Figure 13: Clustering result of centralized k-means on the breast cancer dataset.

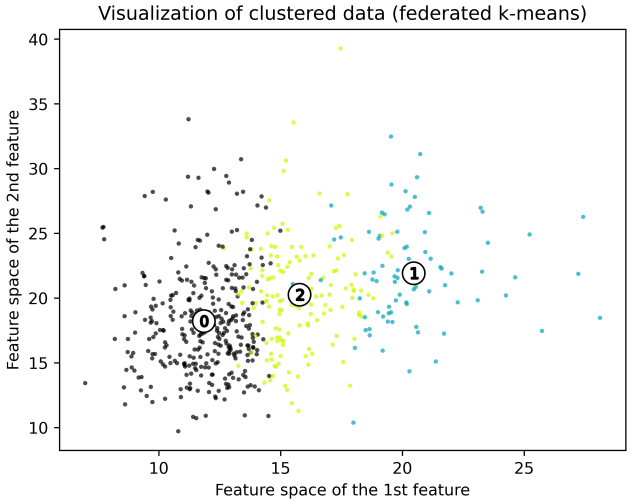
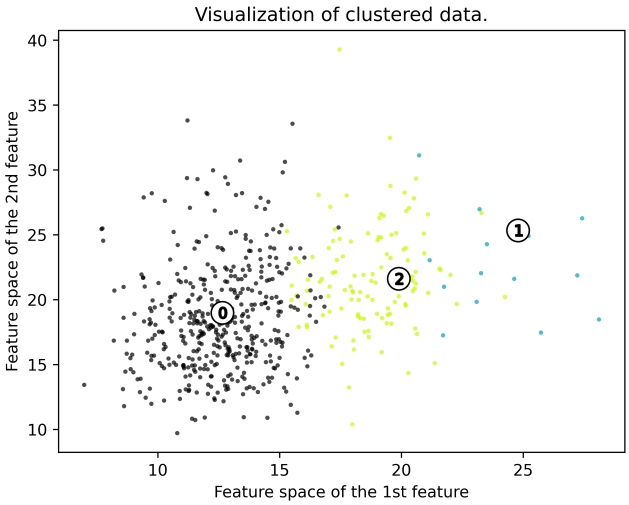


Figure 14: Clustering result of federated k-means on the breast cancer dataset.

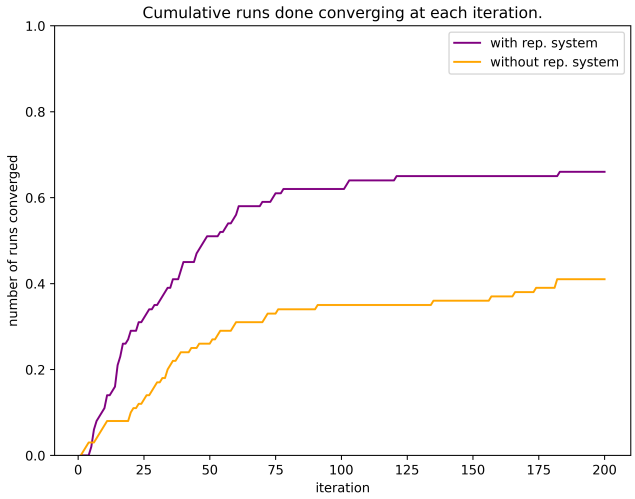


(a) Clustering result of BC-FL k-means using IID distributed data.

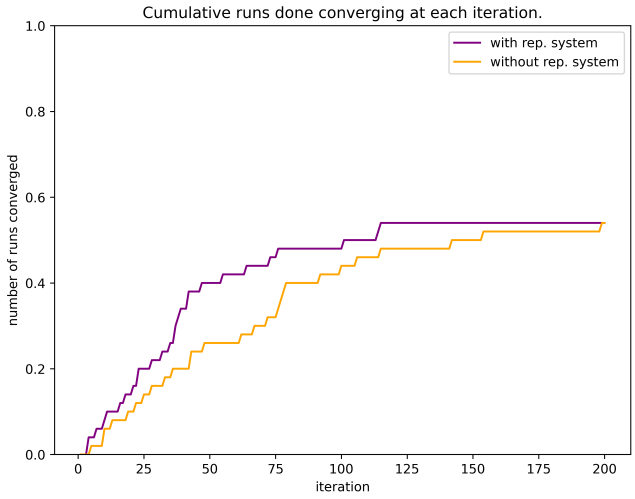
(b) Clustering result of BC-FL k-means using non-IID distributed data.

Figure 15: Clustering results of BC-FL k-means on the breast cancer dataset with IID and non-IID distributed data.

C CUMULATIVE CONVERGENCE OF RUNS WITH DIFFERING PROPORTIONS OF MALICIOUS DEVICES

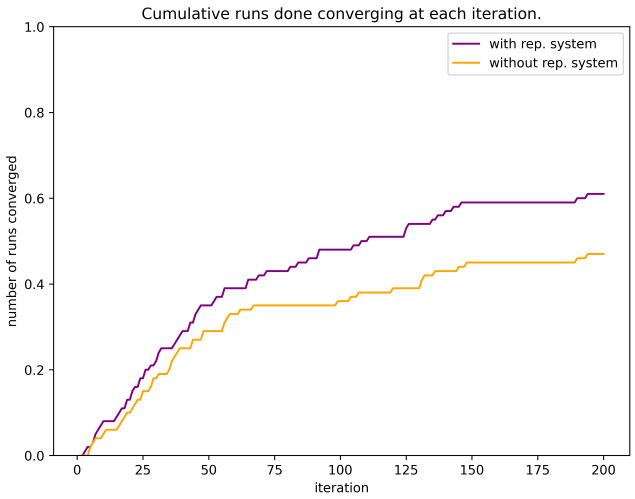


(a) Cumulative convergence of runs with and without reputation system with IID distributed data.

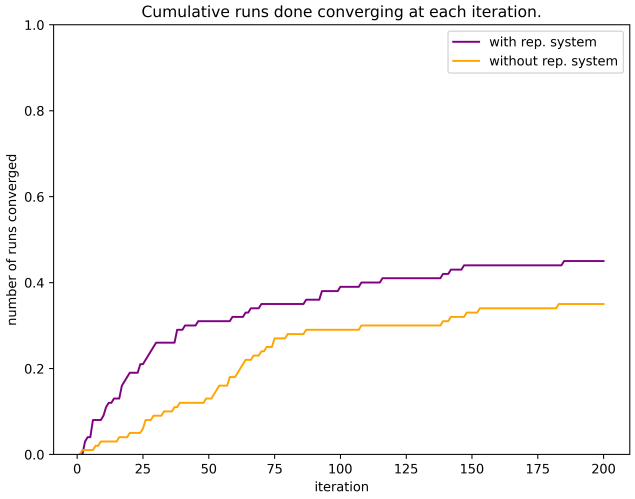


(b) Cumulative convergence of runs with and without reputation system with non-IID distributed data.

Figure 16: Cumulative convergence of runs with IID and non-IID data distribution when 10% devices are malicious.

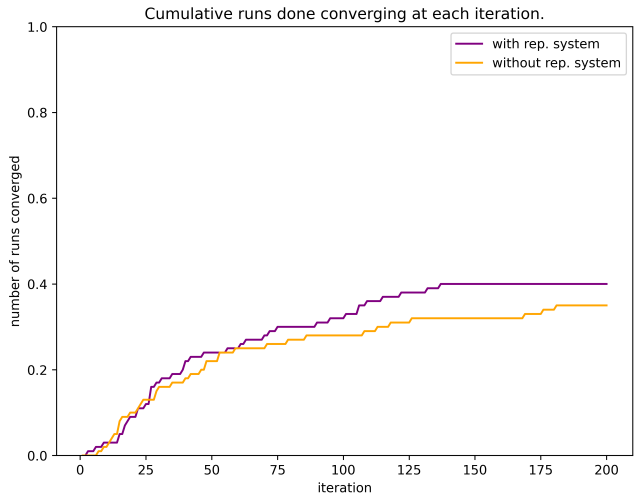


(a) Cumulative convergence of runs with and without reputation system with IID distributed data.

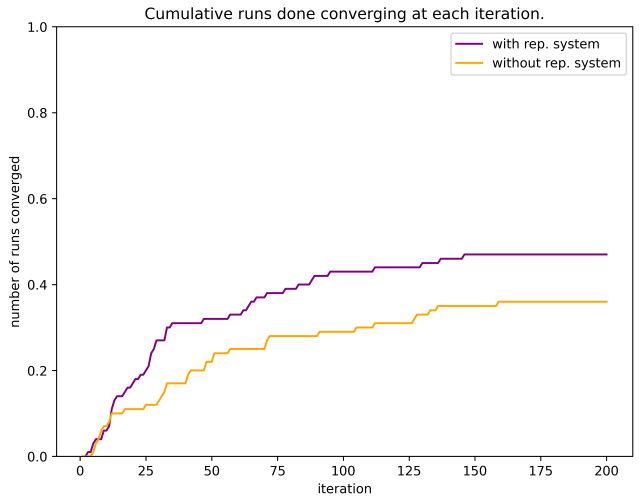


(b) Cumulative convergence of runs with and without reputation system with non-IID distributed data.

Figure 17: Cumulative convergence of runs with IID and non-IID data distribution when 20% devices are malicious.



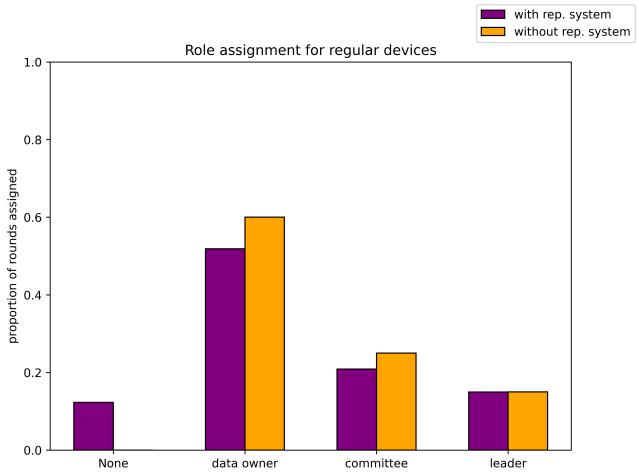
(a) Cumulative convergence of runs with and without reputation system with IID distributed data.



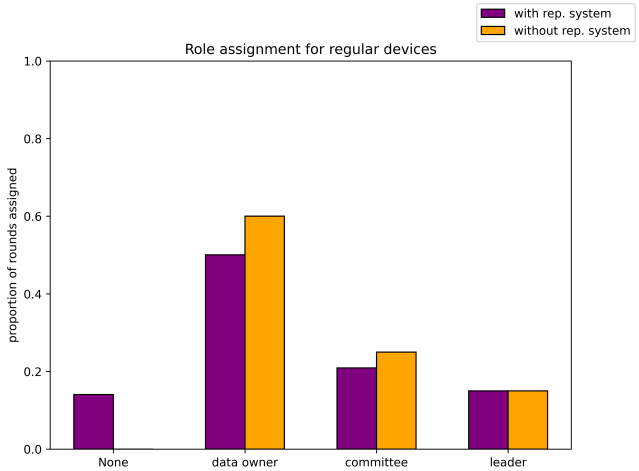
(b) Cumulative convergence of runs with and without reputation system with non-IID distributed data.

Figure 18: Cumulative convergence of runs with IID and non-IID data distribution when 30% devices are malicious.

D ROLE ASSIGNMENT OF REGULAR AND MALICIOUS WITH DIFFERING PROPORTIONS OF MALICIOUS DEVICES PRESENT

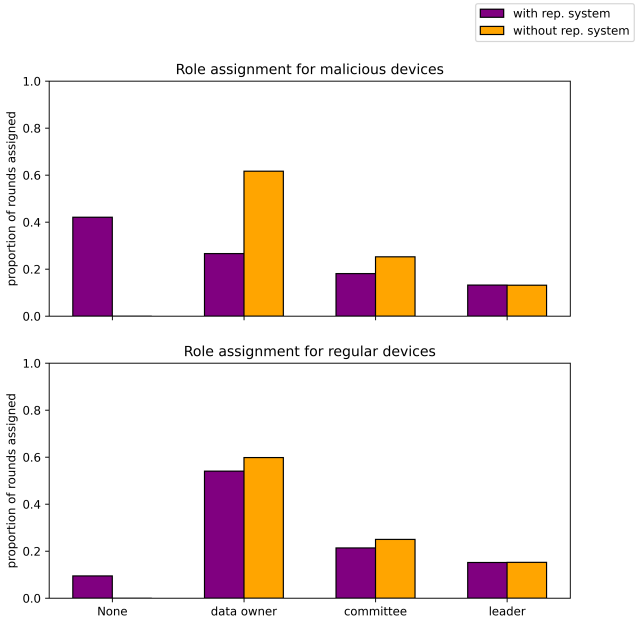


(a) Role assignment of devices with and without the reputation system when data is IID distributed.

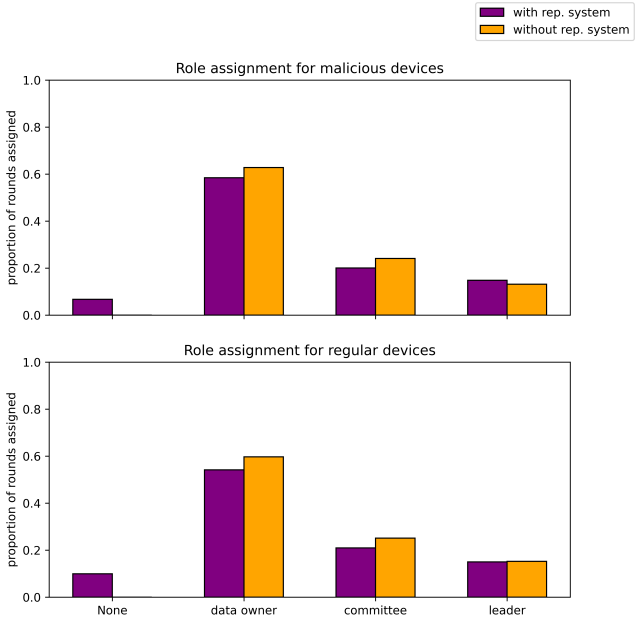


(b) Role assignment of devices with and without the reputation system when data is non-IID distributed.

Figure 19: Role assignment with IID and non-IID data distribution when no malicious devices are present.

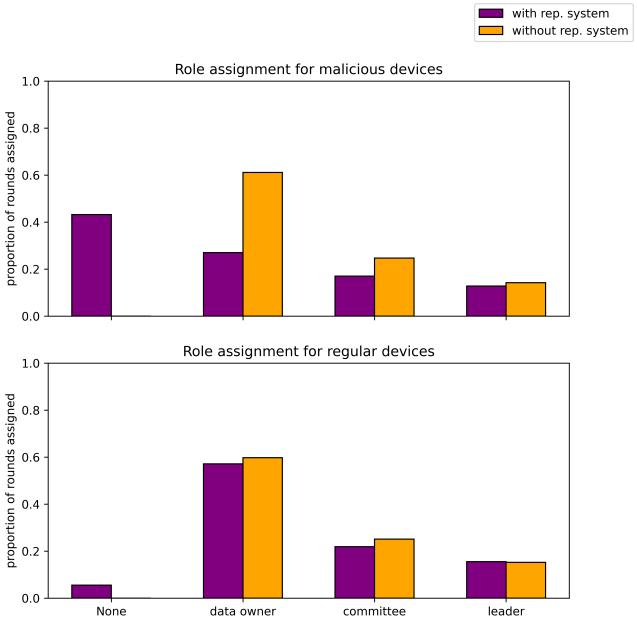


(a) Role assignment of devices with and without the reputation system when data is IID distributed.

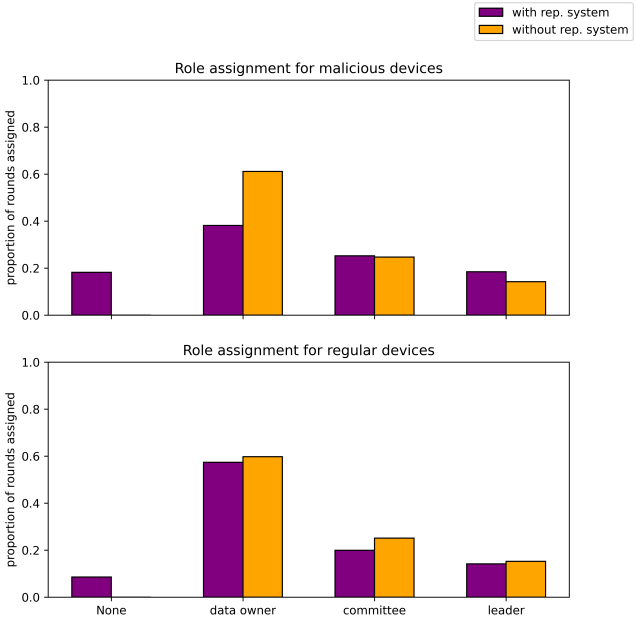


(b) Role assignment of devices with and without the reputation system when data is non-IID distributed.

Figure 20: Role assignment with IID and non-IID data distribution with 10% of devices being malicious.

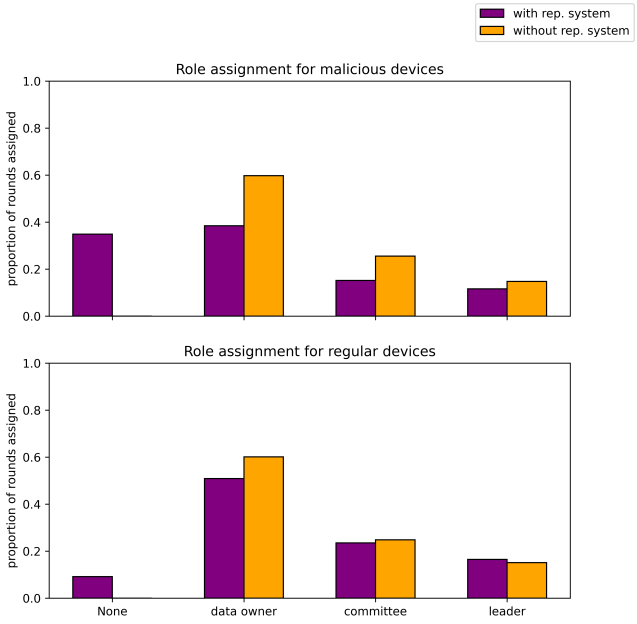


(a) Role assignment of devices with and without the reputation system when data is IID distributed.

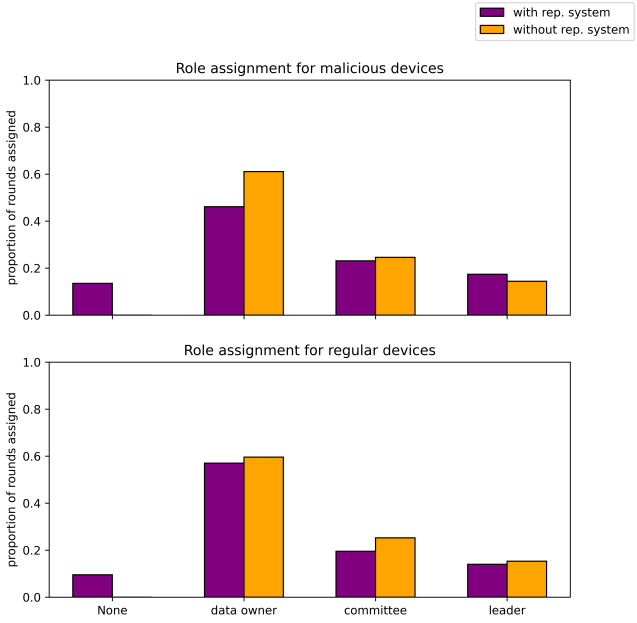


(b) Role assignment of devices with and without the reputation system when data is non-IID distributed.

Figure 21: Role assignment with IID and non-IID data distribution with 20% of devices being malicious.

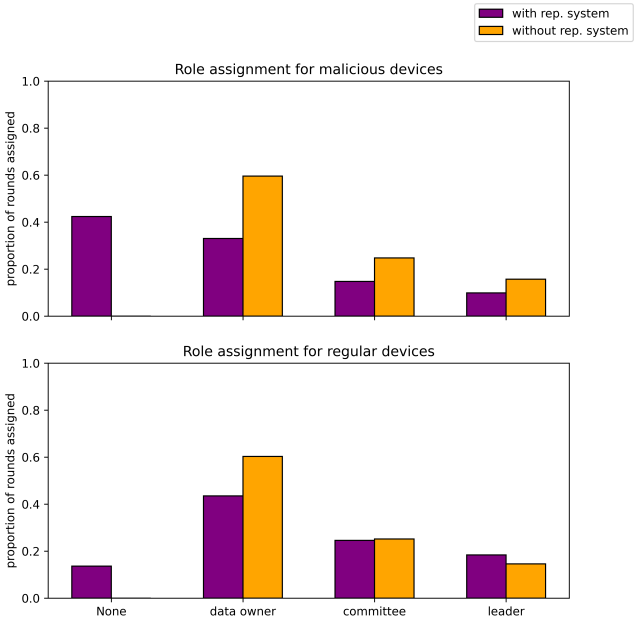


(a) Role assignment of devices with and without the reputation system when data is IID distributed.

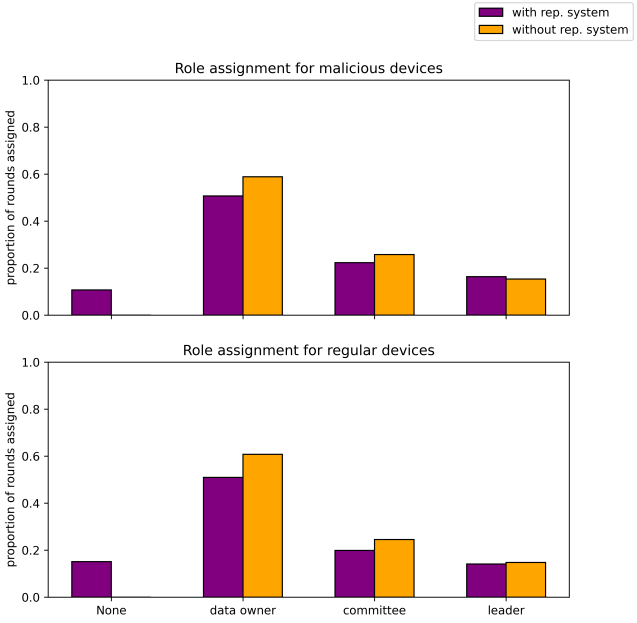


(b) Role assignment of devices with and without the reputation system when data is non-IID distributed.

Figure 22: Role assignment with IID and non-IID data distribution with 30% of devices being malicious.



(a) Role assignment of devices with and without the reputation system when data is IID distributed.



(b) Role assignment of devices with and without the reputation system when data is non-IID distributed.

Figure 23: Role assignment with IID and non-IID data distribution with 40% of devices being malicious.