Hybrid topology construction for a federated learning framework using multi-objective squirrel search optimization based clustering

Vishakha Chichra[1,1], Shashikala Tapaswi[1,1], Neetesh Kumar[1,1]

Abstract

In this data driven era, computationally efficient and privacy-aware solutions for large scale machine learning problems become crucial, specially in the healthcare and e-commerce domain. Previous research has been focused on centralized algorithms, which assume the existence of a central data repository which stores and can process the data from all participants. Such an architecture, however, introduces scalability issues and single-point of failure risks which could compromise the integrity and privacy of the data. Federated learning is a recent advance in privacy protection in which a trusted curator aggregates parameters optimized in decentralized fashion by multiple clients. The resulting model is then distributed back to all clients, ultimately converging to a joint representative model without explicitly having to share the data. However, the protocol is vulnerable to differential attacks, which could originate from any party contributing during federated optimization. A completely decentralised federated architecture can enhance the security of the federated framework, but at the cost of increased communication latency and computation expenses. This research work presents the use of multi-objective squirrel search optimization algorithm for clustering the nodes of a federation, to construct a hybrid architecture which combines the speed of a centralised system with the security-enhancing qualities of a decentralised information exchange.

Keywords: Federated learning, big data, machine learning, nature-inspired, clustering, hybrid topology, differential privacy

1. Introduction

The fields of machine learning and artificial intelligence are witnessing path breaking development in the current age. The accuracy obtained for the machine learning applications is primarily dependent on the data quality and relevancy. Privately-held data is more informative as compared to publicly-accessible data. These private data may be stored in individual electronic devices such as smart-phones, tablets and computers and are not easily accessible. Federated learning provides a solution to all these constraints.

The idea behind federated learning is as conceptually simple as it its technologically complex. Traditional machine learning programs relied on a centralized model for training in which a group of servers run a specific model against training and validation data-sets. That centralized training approach can work very efficiently on many scenario but it also proven to be challenging in use cases involving a large number of endpoints using and improving the model.In those scenarios, each individual endpoint can contribute to the training of a machine learning model in its own autonomous way. In other words, knowledge is federated.

In a case of federated architecture, machine learning algorithm will be run on the devices without extracting the data from the devices, which will be kept within the devices. Only the machine learning result will be aggregated with outcomes generated from other devices to form an unbiased, comprehensive and accurate predictions.Through the concept of federated learning, both the private data and processing power for machine learning are decentralized as algorithms are run directly on individual devices by utilizing their idle processing power. Only the machine learning result will be aggregated with outcomes generated from other devices to form an unbiased, comprehensive and accurate predictions.

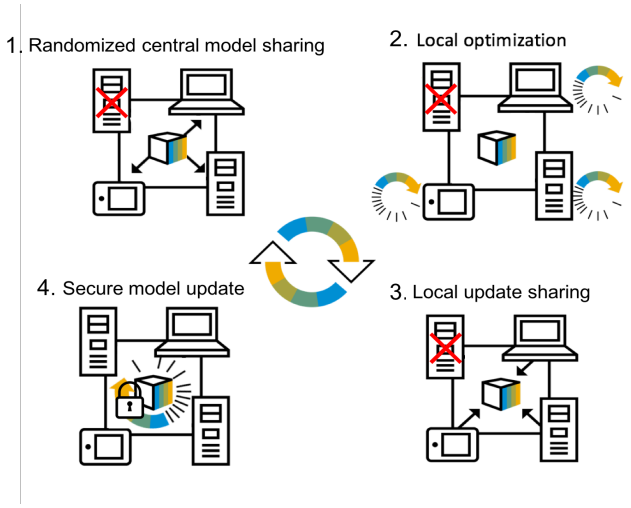

Figure 1: Functioning of a centralise federated learning system

1.1. Issues with centralised learning setup

Centralized AI is by far the most common architecture. However, by separating AI algorithms from users' devices, we are constantly moving colossal volumes of data. These repetitive transfers create serious constraints:

- Diminished privacy: the obligation to transfer our data and to have it stored on remote servers creates opportunities for hackers to intercept data and use it inappropriately.
- Incompatibility with many sectors: for confidentiality reasons several industries are not able to share their data and store it in the cloud (health sector, insurance, bank, military etc). These sectors cannot benefit from Centralized Artificial Intelligence.
- Latency problems that slow inference : centralized AI is inappropriate for many use-cases where AI needs to interact in real time with the real world (i.e autonomous cars).
- High transfer costs : It is due to the exploding amount of data that needs to be handled (an autonomous car generates 4000GB of data to infer every day).

1.2. Applications areas of federated learning

E-commerce domain utilises machine learning techniques to provide customers with personalized services, mainly including product recommendation and sales services. The data features involved in the smart retail business mainly include user purchasing power, user personal preference, and product characteristics. In practical applications, these three data features are likely to be scattered among three different departments or enterprises. For example, a user's purchasing power can be inferred from her bank savings and her personal preference can be analyzed from her social networks, while the characteristics of products are recorded by an e-shop. In this case, federated learning provides a good technical support for us to build a cross-enterprise, cross-data, and cross-domain ecosphere for big data and artificial intelligence.

In a finance application we are interested in detecting multiparty borrowing, which has been a major risk factor in the banking industry. This happens when certain users maliciously borrows from one bank to pay for the loan at another bank. Multi-party borrowing is a threat to financial stability as a large number of such illegal actions may cause the entire financial system to collapse. To find such users without exposing the user list to each other between banks A and B, we can exploit a federated learning framework. In particular, we can use the encryption mechanism of federated learning and encrypt the user list at each

party, and then take the intersection of the encrypted list in the federation. The decryption of the final result gives the list of multi-party borrowers, without exposing the other "good" users to the other party.

Smart healthcare is another domain which we expect will greatly benefit from the rising of federated learning techniques. Medical data such as disease symptoms, gene sequences, medical reports are very sensitive and private, yet medical data are difficult to collect and they exist in isolated medical centers and hospitals. We envisage that if all medical institutions are united and share their data to form a large medical dataset, then the performance of machine learning models trained on that large medical dataset would be significantly improved. Federated learning combining with transfer learning is the main way to achieve this vision.

1.3. Challenges associated with federated learning

A deep analysis of the work by Konecny et al. (2016) leads us to summarize the challenging aspects of federated learning as follows :

- Huge number of clients: Since machine learning generally requires a lot of data, the applications that use it have to have many users. Every one of these users could theoretically participate in federated learning, making it far more distributed than anything in a data center.
- Non-identical distributions: In a data center setting, it is possible to ensure that every machine has a representative set of data so that all updates look very similar. In Federated Learning, this cannot be guaranteed.While similar users might have similar local training data, two randomly picked users could produce very different weight updates.
- Unbalanced number of samples: Along the same lines, we cannot expect most users to have the same number of local training examples. There could be users with only a handful of data points, while others might have thousands.
- Slow and unstable communication: In a data center, it is expected that nodes can communicate comparatively quickly with each other and that it is ensured that messages do not get lost. Uploads are typically going to be much slower than downloads and, especially if the connection is from a cell phone, it might be extremely slow. These properties motivate why federated learning requires its own specialized algorithms.

1.4. Federated learning as an improvement to artificial intelligence

Most of the previous work in federated learning has been done in centralised algorithms, in which a central server tends to aggregate the machine learning updates from the participating nodes. A comparison between the centralised, decentralised and distributed architectures reveals the following points.

Analysis of the following facts leads to the conclusion that a hybrid architecture can enable us to eliminate the limitations of both the decentralised and centralised architecture and combining the beneficial qualities of both the topologies. This research paper proposes the use of multi-objective squirrel search algorithm for clustering, in order to construct a hybrid topology for a federated learning system.

2. Related Works

2.1. The need for more data in machine learning and research work by google employees

In the past few years, machine learning has led to major breakthroughs in various areas, such as natural language processing, computer vision and speech recognition. A work by LeCun, Y. et al. (2015) describes the need for data in deep learning. Thus the success is been based on collecting huge amounts of data. For

example, one of Facebook's latest Detectron models for object detection was trained on 3.5 billion images from Instagram.

In 2016, google researchers Konečný, J., McMahan, H.B., Yu, F.X., Richtárik, P., Suresh, A.T. and Bacon, D. devised an application of trying to predict the next word. While users type on their keyboards, the model tries to predict the next word. As soon as the user typed the next word, a new data point is created and the true label (the last word) is determined. The model can then automatically update itself without having to store the data permanently. In such a situation, Federated Learning is extremely powerful because models can be trained with a huge amount of data that is not stored and not directly shared with a server at all. We can thus make use of a lot of data that we could otherwise not have used without violating the users' privacy.

2.2. Application areas of federated learning

The work by Theodora S.Brisimi et al. (2018) described the use of SVMs for federated learning of health records. In principle, this idea can be applied to any model for which some notion of updates can be defined. This naturally includes everything based on gradient descent, which most of the popular models nowadays are. Linear regression,logistic regression, neural networks can all be used for Federated Learning by letting users compute gradients. There are other models that are not based on gradients but where it is possible to define updates. For k-means clustering, updates could correspond to moving the cluster centers. Similar averages can be used with the power iteration method to implement a distributed version of PCA. For some other models like decision trees, it can be much harder to think of a federated version that allows for continuous updates.

2.3. A study of the techniques used for improving communication efficiency

Neural networks commonly have millions of parameters nowadays. Sending updates for so many values to a server leads to huge communication costs with a growing number of users and iterations. The work by Konečný et al. (2016) states that specialized compression techniques for federated learning can be applied. It is acceptable that individual updates are compressed in a lossy manner, as long as the overall average does not change too much.

2.4. Multi-evolutionary federated learning

Compared with the traditional centralized approach, the federated setting consumes considerable communication resources of the clients, which is indispensable for updating global models and prevents this technique from being widely used. In their work Hangyu Zhu and Yaochu Jin (2018) tried to optimize the structure of the neural network models in federated learning using a multi-objective evolutionary algorithm to simultaneously minimize the communication costs and the global model test errors. A scalable method for encoding network connectivity is adapted to federated learning to enhance the efficiency in evolving deep neural networks.

2.5. Completely decentralised federated learning

Considering the drawbacks of a centralised federated learning setup, Anusha Lalitha .et al (2018) trained a machine learning model over a network of users in a fully decentralized framework. They proposed a distributed learning algorithm in which users update their belief by aggregate information from their one-hop neighbors to learn a model that best fits the observations over the entire network.

2.6. Hybrid topologies for large networks

In 2004, Chien-Chung Shen et al, described in their work that the topology of an network has a significant impact on its performance.Existing topology control algorithms utilize either a purely centralized or a

purely distributed approach. A centralized approach, although able to achieve strong connectivity, suffers from scalability problems. In contrast, a distributed approach, although scalable, lacks strong connectivity guarantees. So, they proposed a hybrid topology control framework, cluster-based topology control (CLTC) that achieves both scalability and strong connectivity.

2.7. Applications of nature-inspired algorithms to multiple domains

A paper by Satyasai Jagannath and Ganapati Panda published in 2014 embodies an up-to-date review of all major nature inspired metaheuristic algorithms employed till date for partitional clustering. Further, the paper discusses key issues involved during formulation of various metaheuristics as a clustering problem and major application areas.
These algorithms are broadly classified into Evolutionary Algorithms, Physical Algorithms, Swarm Intelligence, Bio-inspired Algorithms and others and are popular in various research fields due to their capability to cluster large datasets. These algorithms have been used in signal and image processing for image segmentation, in wireless sensor network for classifying the sensors to enhance lifetime and coverage in communication to design accurate blind equalizers, in robotics to efficiently classify the humans based upon their activities in computer science for web mining and pattern recognition, in medical sciences to identify diseases from a group of patient reports and genomic studies.

	Centralized	Decentralised	Distributed
Points of failure	Single	Infinite	Many
Fault tolerance	Very low	High	Moderate
Scalability	Low	Moderate	Infinite
Ease of development	High	Low	Low
Diversity	Low	High	High

Table 1: Comparison of the various topologies

3. Problem formulation

In this section, the problem of construction of a hybrid federated learning framework is formulated. The problem assumes the existence of a set V of N nodes, V=$\{V_1, V_2, ..., V_N\}$ which have formed a federation by having agreed to collaborate on training of a high quality machine learning model, by sharing their local weight updates. Initially, it is considered that no node is connected to other node and assumed that the federation of nodes in graph G=(V,E) where set of connections (edges) between nodes are empty, that is, E = and thus G is a null graph. The aim of this research work is to identify valid and efficiency edges between the nodes such that it forms a hybrid topology which promotes effective communication between collaborating clients, while minimizing privacy loss and enhancing accuracy of the model at the same time.

The problem of hybrid topology construction for a federated lea ring framework can be addressed as a clustering problem, where the set of nodes is split into a set of clusters, such that the set C = {c1, c2, ck} represents the cluster leaders of the k clusters. No two clusters should overlap. If a node $V_i \in$C$_i$ i, then $V_i \notin$C$_j$ and $C_i \cap$C$_j$ is ϕ.The ideal number of clusters k, is determined by using the algorithm named "Clustering by fast search and finding of density peaks" (CFSFDP). It is based on the assumption that cluster centres with higher density than their neighbours and are at a relatively larger distance from a point which has a higher density.

The clusters are a centralised federated learning setup in themselves. The cluster leaders act as the trusted curator of the updates which are send by the cluster member nodes. The cluster heads form a decentralised peer-to-peer network amongst themselves. they also need to share their aggregated weight updates with other cluster heads (leaders). The group of cluster leaders, which is a decentralised P2P network, communicates their weight updates using "Gossip Protocol".

Federated learning setup can as a multi-party setting in which the training data-set is distributed amongst the K parties P_1, P_2, ..., P_k so that k^{th} party P_k possesses a subset of data $D_k \subseteq$D. The goal is to learn a classifier from complete data-set D. However, each party wishes not to disclose any information about any individual data point. this is the concept of differential privacy in federated learning.

As more than one objective function are needed to optimise during routing, there is no single possible solution for such a problem. As one solution may dominate other solution with respect to another objective and other solutions may dominate with respect to other objectives.

A detail of all the objectives for the given problem are as follows:

1. **Topology construction error :** It is metric which is used to measure the effectiveness of the derived hybrid topology with respect to two parameters, quantization error and data-points difference error. Quantization error measures the distance between the nodes of network and tends to impact the communication costs. Another value, that is, the data-points difference error measures the variance in the average number of data sample per cluster. This value indicates the ease of weights aggregation. the lower this value is, the better is the distribution of data amongst the clusters.

   - Quantization error is a metric introduced by moving each point from its original position to it's associated quantum point. In clustering, it is often measured as the root-mean square of each point (moved to centroid of cluster).
   - Data-points difference error : This parameter is used for evaluating how uniformly the data is divided amongst the clusters.

   In the context of clustering, a single particle represents the N_c cluster centroid vectors. That is, each particle x_i is constructed as follows:

$$\mathbf{x}_i = (\mathbf{m}_{i1}, \cdots, \mathbf{m}_{ij}, \cdots, \mathbf{m}_{iN_c}) \tag{1}$$

   where m_{ij} refers to the j-th cluster centroid vector of the i-th particle in cluster C_{ij} . Therefore, a population represents a number of candidate clusterings for the current data vectors. The quantization error for the particles is calculated as

$$Q_e = \frac{\sum_{j=1}^{N_c} \left[\sum_{\forall \mathbf{z}_p \in C_{ij}} d\left(\mathbf{z}_p, \mathbf{m}_j\right) / \left|C_{ij}\right| \right]}{N_c} \tag{2}$$

   where d is defined as,

$$d\left(\mathbf{z}_p, \mathbf{m}_j\right) = \sqrt{\sum_{k=1}^{N_d} \left(z_{pk} - m_{jk}\right)^2} \tag{3}$$

and m_j is calculated as

$$\mathbf{m}_j = \frac{1}{n_j} \sum_{\forall \mathbf{z}_p \in C_j} \mathbf{z}_p \tag{4}$$

and $|C_{ij}|$ is the number of data vectors belonging to cluster C_{ij} , i.e. the frequency of that cluster.
The other parameter named difference value tends to evaluate how uniformly the data is divided amongst the clusters. It is calculated as

$$D_e = \sqrt{\sum_{i=1}^{k} (A - A_c)^2} \tag{5}$$

where A denotes the average number of data samples per node, and is calculated as

$$A = \frac{\sum_{i=1}^{N} p_i}{N} \tag{6}$$

here N denotes the total number of nodes in the federated setup
where A_c denotes the average number of data samples per node in cluster c and is calculated as

$$A_c = \frac{\sum_{i=1}^{N_c} p_i}{N_c} \tag{7}$$

here N_c denotes the total number of nodes in cluster c ans p_i denotes the number of data points possessed by node i
The total error, therefore is given as

$$T_e = \frac{Q_e + D_e}{2} \tag{8}$$

Therefore, the total error is defined as,
Minimize F(T) = T_e
Subject to

(a) $Q_e < Q_max$
(b) $D_e < D_max$

where Q_{max} and D_{max} represent the maximum limits of the respective constraints which need to be satisfied.

2. **Differential privacy** : The multiparty classification problem involves the cluster leaders in this case which gathers information from the individual parties and computes a private classifier. Specifically, each party (or cluster member) P_k and computes a local classifier $w_k \in \mathrm{R}_d$ by minimizing the local regularized ERM objective on its own data set

$$D_k = \left\{ \left(\mathbf{x}_1^k, y_1^k \right), \ldots, \left(\mathbf{x}_{N_k}^k, y_{N_k}^k \right) \right\} \tag{9}$$

given by

$$\nabla J\left(\mathbf{w}_t\right) = \frac{1}{N} \sum_{k=1}^{K} \nabla G_k\left(\mathbf{w}_t\right) + \lambda \mathbf{w}_t \tag{10}$$

The third party aims to aggregate the local classifiers wk through averaging and release an output perturbed version of the average, where η is a noise vector.

$$\mathbf{w}_{priv} = \frac{1}{K} \sum_{k=1}^{K} \mathbf{w}_k + \eta \tag{11}$$

A trusted curator that is the cluster leaders in this case, wants to minimize the overall multiparty objective J(w) by running a gradient descent algorithm. We will assume that the loss function ϕ is convex and differentiable, which will ensure that gradients exist and that the minimizer of the objective is unique. To run such an algorithm, all that the aggregator needs is the gradient information at any given w_t. The gradient of J(w) at wt is given by

$$J(\mathbf{w}) = \frac{1}{N} \sum_{i=1}^{N} \phi\left(y_i \mathbf{w}^\top \mathbf{x}_i\right) + \frac{\lambda}{2} \|\mathbf{w}\|_2^2 \tag{12}$$

which can equivalently be written as

$$\nabla J\left(\mathbf{w}_t\right) = \frac{1}{N} \sum_{i=1}^{N} \phi'\left(y_i \mathbf{w}_t^\top \mathbf{x}_i\right)\left(y_i \mathbf{x}_i\right) + \lambda \mathbf{w}_t \tag{13}$$

Clearly, the gradient of J(w) at any wt can be computed from pieces of information from the K different parties. In particular, if party P_k provides the gradient

$$\nabla G_k\left(\mathbf{w}_t\right) = \sum_{j=1}^{N_k} \phi'\left(y_j^k \mathbf{w}_t^\top \mathbf{x}_j^k\right)\left(y_j^k \mathbf{x}_j^k\right) \tag{14}$$

$$\nabla J\left(\mathbf{w}_t\right) = \frac{1}{N} \sum_{k=1}^{K} \sum_{j=1}^{N_k} \phi'\left(y_j^k \mathbf{w}_t^\top \mathbf{x}_j^k\right)\left(y_j^k \mathbf{x}_j^k\right) + \lambda \mathbf{w}_t \tag{15}$$

Each party P_k can return a perturbed version of the gradients, by adding a noise vector $\rho_t^k \in \mathbb{R}^d$.

$$\nabla \widehat{G}_k\left(\mathbf{w}_t\right) = \nabla G_k\left(\mathbf{w}_t\right) + \rho_t^k \tag{16}$$

The addition of noise tends to provide privacy, but can lead to too much distortion of the machine learning weight updates, thus making it useless. Thus, to keep a check on privacy factor, δ is calculated after each communication round. Our aim is to minimise δ, which is the probability of ϵ privacy being broken.

3. **Machine learning model accuracy** : Stochastic gradient descent can be applied naively to the federated optimization problem, where a single mini-batch gradient calculation is done per round of communication. This approach is computationally efficient, but requires very large numbers of rounds of training.
The amount of computation is controlled by three key parameters: C, the fraction of clients that perform computation on each round, E, then number of training passes each client perform over its local data-set on each round; and B, the mini-batch size used for the client updates. The algorithm selects a C-fraction of clients on each round, and computes the gradient of the loss over all the data held by these clients. Thus, in this algorithm C controls the global batch size, with $C = 1$ corresponding to full-batch (non-stochastic) gradient descent.

   A typical implementation of distributed gradient descent with a fixed learning rate η has each client k compute $g_k = \Delta\ F_k(w_t)$, the average gradient on its local data at the current model w_t, and the central server aggregates these gradients and applies the update

$$w_{t+1} \leftarrow w_t - \eta \sum_{k=1}^{K} \frac{n_k}{n} g_k \tag{17}$$

   since

$$\sum_{k=1}^{K} \frac{n_k}{n} g_k = \nabla f\left(w_t\right) \tag{18}$$

   This update is equivalent to :

$$\forall k, w_{t+1}^k \leftarrow w_t - \eta g_k \tag{19}$$

$$w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k \tag{20}$$

   When $B = \infty$ to indicate that the full local data-set is treated as a single mini-batch. gradients and applies the update.

   That is, each client locally takes one step of gradient descent on the current model using its local data, and the server then takes a weighted average of the resulting models. For a client with n_k local examples, the number of local updates per round is given by

$$u_k = E \frac{n_k}{B} \tag{21}$$

   After performing the schotastic gradient updates, the model is trained and the accuracy of the model is calculated as

$$Accuracy = \frac{c_p}{t_p} \tag{22}$$

   where c_p is total number of correct predictions and t_p is the total number of predictions made.

Above discussed functions help in the construction of clusters, with cluster leaders acting as the central server for their respective cluster nodes. Each metric plays an important role in the construction of topology. So, in order to satisfy all the parameters and constraints, this problem has been formulated as a multi-objective optimization problem.A set of MOPs is in practice and is call pareto-optimal set. Form -objectives MOP is formulated as

Optimise q = F(x) = {f_1(x),f_2(x) , ... f_k(x)}

where X = (x_1, x_2, .. x_n) is a n-dimensional decision variable vector, q is the decision variable space and Q denotes objective space.

The above problem will be solved using multi-objective squirrel search optimization.

4. Overview of squirrel search algorithm

In a recent work, Jain et al. proposed a meta-heuristic, i.e., Squirrel Search Algorithm (SSA) and claimed to overcome the shortcomings of the existing meta-heuristics such as GA, BAT, PSO and Firefly algorithms with improved performance. In SSA, flying squirrels glide (it takes lesser energy than flying and allows mammals to travel long distance rapidly and efficiently) from trees to trees. They modify lift and drag variables to control the force at right angle to the movement's direction and opposite of the direction of movement. The search process of SSA begins in the search of food (nuts) which includes two type of foods; hickories and acorn. Acorn trees meet the daily energy requirements of the squirrels while nuts from hickories trees are stored to help them for meeting energy requirements in winter. Each day of autumn, squirrels move forward to the direction of hickory tree from acorn tree to store the food for winters. At the end of winters, squirrels again actively start the old cycle of survival. Thus, SSA design objective is to update the position of flying squirrels such that they are able to fulfill their energy needs in autumn as well as in winters.In this iterative approach, based on fitness value, squirrels move in the direction of optimal position by updating their positions according to the nearest acorn and hickory trees. Basic mathematical model includes N number of squirrels (search agents) in K-dimensional search space, N_{AN} number of acorn trees and one hickory nut tree. It is also assumed that one squirrel is residing on individual tree. Thus, remaining (N(N_{AN} + 1)) trees are normal trees.
Basic steps of SSA are as follows,

1. N number of squirrels are initialized at random positions and represented by a vector of k dimensions where s_{ij} represents the position of i^{th} squirrel in j^{th} dimension. Position of each squirrel follows the boundary set for lower and upper bound.
2. Fitness of each squirrel is computed and squirrels on hickory, acorn and normal trees are initialized.
3. Location of the squirrels on acorn and normal trees are updated according to the location of hickory and acorn trees using different gliding parameters.
4. Termination criteria of SSA includes the maximum number of iterations or convergence of the agents to the same point.

5. Squirrel search based hybrid federated learning topology

5.1. Determination of ideal number of clusters

Many measurements have been developed for solving clustering number in literature. These methods are often running clustering algorithms with different values of K, and the best value of K is then chosen based on a predefined criterion. In this paper, we have utilise an algorithm proposed by Jiali Wang, Yue Zhang, Xv Lan in their work Automatic Cluster Number Selection by Finding Density Peaks in 2016. While the traditional algorithms such as the Gap method and the elbow method can help find the appropriate cluster number to some extent, the time cost is quite high. Furthermore, only after deriving the results of clustering, can they determine the appropriate number of cluster by iterating this algorithm.

The basic idea of CFSFDP is on the assumption that cluster centers are with higher density than their neighbours and at a relatively larger distance from the point which has a higher density. The algorithm is based on finding the knee point of a new index named CS. It improves the conventional model selection

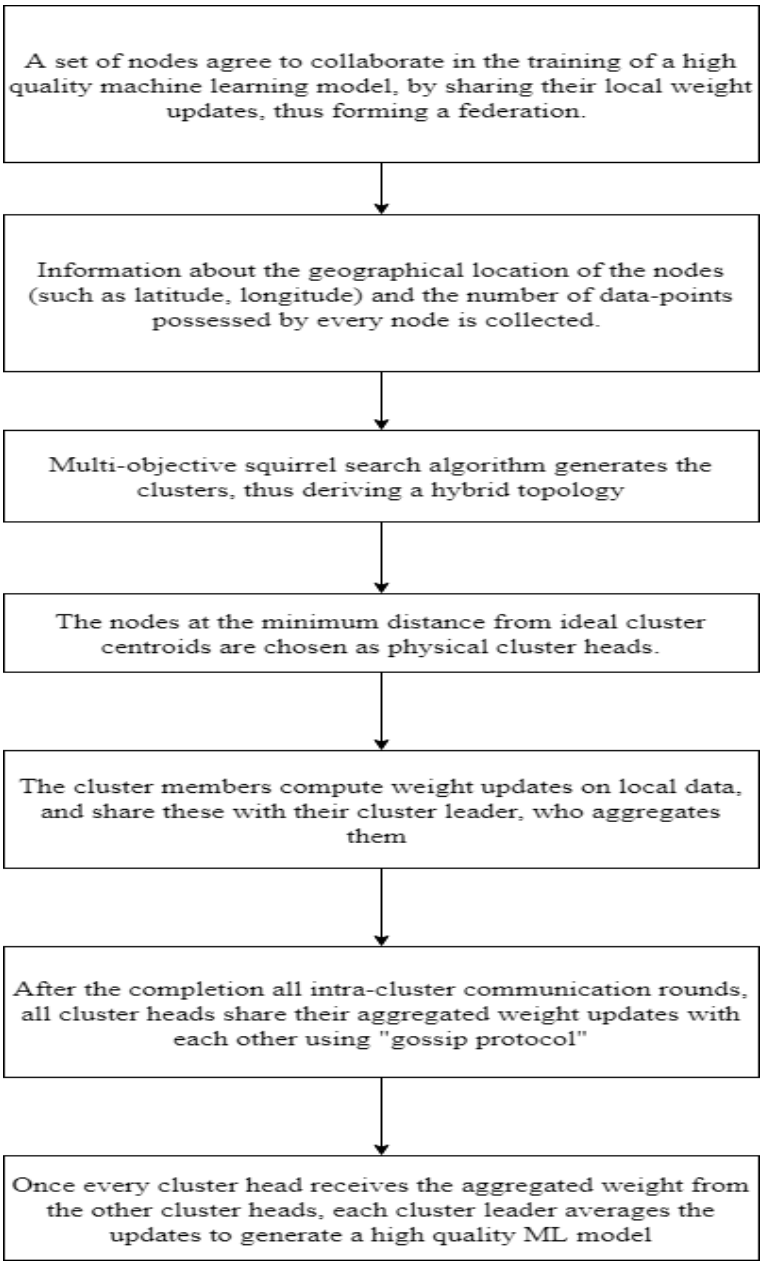

Figure 2: Construction and evolution of a hybrid topology for a federated learning setup

process by enhancing the precision of finding the correct cluster number. For each data point i, the two quantities determine whether the data point has potential to become the cluster center c are:

1. Local density ρ_i
2. The point's distance d_i from points from other points of higher density.

Definition 1 : ρ_i is the density of point d_i

$$\rho_i = \sum_j \chi\left(d_{ij} - d_c\right) \tag{23}$$

where $\chi(x)$=1 if x <0 and $\chi(x)$=0 otherwise. d_c is a cutoff distance, which is the only parameter determined by users, and d_{ij} indicates the distance between point i and point j. In other words, ρ_i equals to the number of points which are closer than d_c to point i

Definition 2 : d_i minimal distance from point i to the point with a higher local density.

$$\delta_i = \min_{j:p_j>p_i} \left(d_{ij}\right) \tag{24}$$

if point i has the highest density, we take δ_i=max$_j$ $\left(d_{ij}\right)$

5.2. Logical flow of squirrel search based hybrid federated topology construction

Construction of a hybrid topology utilising squirrel search has following major steps:

1. Once a couples of nodes agree to form a federation which would coordinate and contribute to the training of a machine learning model, information such as the geographical location of the contributing client in terms of latitude and longitude and the current number of data sample possessed by the nodes are collected. This information about a node is essential as these parameters are crucial for constructing a hybrid federated learning setup.
2. In order to ensure efficient clustering, the ideal number of cluster heads to be present in the hybrid topology is identified using clustering by fast search and finding of density peaks (CFSFDP) which determines the number of ideal number of clusters before the actual clustering takes place.
3. Each search agent represents solution in the form of a set of cluster leaders. Once, the appropriate number of cluster heads is identified using CFSFDP, randomly k nodes are chosen as the k cluster leaders.
4. It has been studied earlier in various work previously that nature inspired algorithms such as particle swarm optimization (PSO), and Squirrel Search Algorithm (SSA) are prone to getting trapped in local optima. In order to eliminate that scenario, K-means clustering algorithm is used to determine the recalculate the randomly allocated cluster centroids.
5. The initial centroids are evaluated for their fitness. Fitness function determines the effective of the clusters based on two parameters, that is, the distance between the nodes of a cluster should be as minimum as possible and moreover, the clusters should be constructed in such a manner that the average number of data samples per cluster is nearly constant.
6. After evaluation of the fitness of the initial centroids, pareto optimal solutions are generated. The best solution is determied utilsing the IPESA algorithm. This solution acts as the hickory squirrel. Other non-dominated solutions of the pareto front are considered as the acorn tree squirrels.

7. Centroids are updated in order to achieve fitter solutions utilising the updation criterias and methods as proposed by the squirrel search algorithm, iteratively.

8. During this updating process, it is assumed that all the squirrels tend to move towards the hickory tree. So, if no randomness is introduced, then all the centroids might converge to a single solution. To eliminate that issue, exploration is guaranteed by utilizing the concept of random selection of exploratory squirrels which conduct levy flight to explore new probably fitter solutions.

5.3. Encoding of squirrel search agents

SSA based clustering begins by generating random agents in multidimensional search space. Length of each agent is equal to each agent is a k*3 matrix , where k denotes the number of clusters required and the 3 columns are the latitude, longitude and the number of data points possessed by the node respectively. The range of geographical location of the nodes, that is, the latitude and the longitude can be quite wide. In order to compress the search space, we initialized the particle for the squirrel search algorithm as follows.

$$s_{ij}(0) = s_{\min} + U(0,1) \times (s_{\max} - s_{\min}) \tag{25}$$

where $s_{\min}$ and $s_{\max}$ denote the lower and upper bounds for the position of i_{th} agent in j_{th} dimension respectively. In our case, j ranges from 0 to 2, and denotes latitude, longitude and the data samples residing at the node respectively. U(0,1) denotes a uniform random number distributor following [0,1] limit.

5.4. Fitness function evaluation

After the generation of valid search agents, each agent is evaluated to measure its quality via fitness function in order to explore the best agents. Considering the requirements mentioned above, fitness function in this case is an average of the quantization error and difference value.

In the context of clustering, a single particle represents the N_c cluster centroid vectors. That is, each particle x_i is constructed as follows:

$$\mathbf{x}_i = (\mathbf{m}_{i1}, \cdots, \mathbf{m}_{ij}, \cdots, \mathbf{m}_{iN_c}) \tag{26}$$

where m_{ij} refers to the j-th cluster centroid vector of the i-th particle in cluster C_{ij} . Therefore, a population represents a number of candidate clusterings for the current data vectors.

1. **Calculation of topology construction error**: Topology construction error is calculated as the mean of error values, quantization error and the data-points difference error. The procedure of the calculation of both the values is given in problem formulation section. Lesser error value implies better topology.

2. **Calculation of differential privacy** : Differential Privacy can be defined in terms of the application-specific concept of adjacent databases. Suppose, for adjacent databases where each training dataset contains a set of image-label pairs, we say that two of these sets are adjacent if one image-label pair is present in one training set while absent in the other.

   A randomized mechanism M:D $\rightarrow$ R with domain D and range R satisfies (ϵ,δ)-differential privacy if for any two adjacent inputs d, $d \in D$ and for any subset of outputs S $\subseteq$ R it holds that

$$Pr[\mathcal{M}(d) \in \mathcal{S}] \leq e^{\epsilon} Pr\left[\mathcal{M}\left(d'\right) \in \mathcal{S}\right] + \delta \tag{27}$$

Authors used Dwork et al. privacy definition of allowing the possibility that plain ϵ-differential privacy is broken with probability δ in their work.

The distortion due to Gaussian noise should not exceed a certain limit. Otherwise too much information from the sub-sampled average is destroyed by the added noise and there will not be any learning progress. GM and random sub-sampling are both randomized mechanisms.However, there it is used for gradient averaging, hiding a single data point's gradient at every iteration. Thus, σ and m also define the privacy loss incurred when the randomized mechanism provides an average approximation. In order to keep track of this privacy loss, we make use of the moments accountant as proposed by Abadi et al. (2016). This accounting method provides much tighter bounds on the incurred privacy loss than the standard composition theorem (3.14 in Dwork Roth (2014)). Each time the curator allocates a new model, the accountant evaluates given, σ and m. Thus, the value of δ attained till the completion of the machine learning model convergence is used as parameter to measure the fitness. The lesser the value, the better the topology.

3. **Calculation of machine learning model accuracy :** After the completion of all the communication rounds, all the cluster leaders locally compute the final machine learning model weights. These converged machine learning models are redistributed to the cluster members. The cluster member nodes reevaluate the accuracy of the final model, on their local training data and send the results back to their respective cluster leaders. The clusters leaders also perform the same operation of evaluating the model performance on their data. A weighted average of the accuracy by the various cluster members is calculated according to the number of data-points possessed by the node, by the cluster leader. The aggregated accuracy of all the clusters is used as parameter to determine the final accuracy which the collaboratively optimised machine learning model can attain, for the current hybrid topology.

5.5. Selection of optimal solutions

Due to the conflicting nature of the objectives in our problem statement, there is no single optimal solution but rather a set of Pareto optimal solutions (or called Pareto front in the objective space).PESA-II follows the standard principles of an EA, maintaining two populations: an internal population of fixed size, and an external population (i.e.,archive set). IPESA-II: Improved PESA-II is of non-fixed but limited size. The internal population stores the new solutions generated from the archive set by variation operations, and the archive set only contains the non-dominated solutions discovered during the search. A grid division of the objective space is introduced to maintain diversity in the algorithm. The number of solutions within a hyperbox is referred to as the density of the hyperbox, and is used to distinguish solutions in two crucial processes of an EMO algorithm: mating selection and environmental selection.

Unlike most EMO algorithms (including its predecessor PESA), the mating selection process of PESA-II is implemented in a region-based manner rather than in an individual-based manner. That is, a hyperbox is first selected and then the resulting individual for genetic operations is randomly chosen from the selected hyperbox—thus highly crowded hyperboxes do not contribute more individuals than less crowded ones.

In the environmental selection process, the candidate individuals in the internal population are inserted into the archive set one by one, thus the grid environment updated step by step. A candidate may enter the archive if it is non-dominated within the internal population, and is not dominated by any current member of the archive. Once a candidate has entered the archive, corresponding adjustment of the archive and grid environment will be implemented. Firstly, the members in the archive which the candidate dominates are removed to ensure that only non-dominated individuals exist in the archive. Secondly, the grid environment is checked to see whether its boundaries have changed with the addition or removal of individuals in the archive1. Finally, if the addition of a candidate renders the archive overfull, an arbitrary individual in the most crowded hyperbox will be removed.

THus, IPESA-II is proposed as an enhanced version of PESA-II that introduces three simple but effective improvements in the algorithm's environmental selection. The best solution is chosen as the hickory squirrel and all the other non-dominated solutions of the pareto-optimal set as considered as acorn squirrels.

5.6. Updating Rules of SSA based federated learning

Respective position of each agent is updated in the subsequent iteration. The position of an agent at hickory tree X_{ht} motivates the agents on acorn tree X_{at} and agents on normal tree X to update their respective positions. Similarly, position of agent on acorn tree motivates the position of agent on normal tree. Thus, SSA includes three cases; Case 1, Case 2 and Case 3 for the position updating of agents. In SSA, agent with best fitness is selected as a hickory agent, that works as the leader. Next, N_{AN} best agents are selected as acorn agents excluding x_{ht}.

Case 1: With the information of X_{ht}, acorn agents update their positions during the course of iterations as:

$$x_{at}(t+1) = \begin{cases} x_{at}(t) + d_g \times G_c \times (x_{ht}(t) - x_{at}(t)) & if R_1 >= P_{dp} \\ \text{Random location} & otherwise \end{cases} \tag{28}$$

where d_g represents the random gliding distance in [0.5, 1.11] , R1 represents the random number in range [0,1] and t represents the iteration number. Variable P_{dp} represents the probability of the presence of predator, which can attack the squirrels, and G_c represents the gliding constant.

Case 2: The positions of agents on normal trees are updated on the basis of acorn and hickory tree. If agent on normal tree needs to fulfill its daily energy requirement, then its location is updated using.

$$x_{nt}(t+1) = \begin{cases} x_{nt}(t) + d_g \times G_c \times (x_{at}(t) - x_{nt}(t)) & if R_2 >= P_{dp} \\ \text{Random location} & otherwise \end{cases} \tag{29}$$

Case 3: Otherwise The position of agents on normal trees is updated using where R2 and R3 are random numbers in the range of [0,1]

$$x_{nt}(t+1) = \begin{cases} x_{nt}(t) + d_g \times G_c \times (x_{ht}(t) - x_{nt}(t)) & if R_3 >= P_{dp} \\ \text{Random location} & otherwise \end{cases} \tag{30}$$

5.7. Seasonal constant updation

Generally, G_c balances the trade-off between the exploration and exploitation, but it is further improved by the incorporation of seasoning constant. Behavior of flying squirrel gets affected by the seasonal changes as they become less active during winters compared to autumn season. This constant plays a major role in balancing exploration and exploitation in SSA and helps to unfold the problem of getting trapped in local minima.

$$Se_c(t) = \sqrt{\sum_{j=1}^{k} x_{at}^j(t) - x_{ht}^j} \tag{31}$$

This seasoning constant affects the flow of SSA by accounting the monitoring condition, minimum value constant, denoted as S_{min}. This constant is responsible for true balancing in exploration and exploitation. The condition is given as Se c $<S_{min}$, where S_{min} is tuned as,

$$S_{\min} = \frac{\log(t)}{t_{\max}^{\beta}} \tag{32}$$

where t and t_{max} represent the current iteration and the maximum allowed iterations in the algorithm respectively, and controls the ratio. Large value of S_{min} supports exploration while small value stands for exploitation. If the condition is true, random relocation of the agents is performed.

5.8. Termination criteria

The stopping condition of multi-objective squirrel search optimization can be based on the following conditions:

1. When the maximum number of iterations have been completed.
2. Convergence point of all agents is same.

5.9. Localisation of cluster leaders

Once the squirrel search algorithm converges, we have the coordinates of the most effective cluster leaders in our compressed search space. But, since the exploration of the squirrel search agents is random and is conducted in continuous state space, the presence of nodes at these geographical coordinates, cannot be guaranteed. Thus, in order to choose the physically present nodes as the cluster heads, distance of every actual nodes is calculated from every centroid point. The data centre nearest to the centroid is chosen as the physical representative of the obtained centroid. These coordinates are then mapped back to their actual geographical coordinates and thus appropriate cluster leaders are obtained.

6. Experimental study

This sections consists of the details of the experiments conducted and the results obtained thereafter along with its analysis. This section is divided into three sections. The first section evaluates the performance of behaviour of MO-SSA is analysed through a set of extensive experiments. Various experiments are conducted which evaluate the multi-objective optimization problems (M0-PSO, MO-Cuckoo, MO-ACO, NSGA-II). Thus the first section is dedicated to analyse the efficiency of various multi-objective clustering algorithms. In order to ensure fairness in experimental study, all the algorithms are evaluated on same metrics. The second section compares the effectiveness of the hybrid topologies obtained by various multi-objective optimization methods. The comparison of the effectiveness of the topologies is done comparing the federated learning metrics such as the accuracy of the final machine learning model, computation time, communication costs, number of communication rounds required for convergence, differential privacy parameter, on all the network scenarios. The third section compares the performance of the centralised , decentralised and hybrid topologies for federated learning setup. The comparison is done on the metrics which govern the efficiency of a secure and privacy preserving federated learning setup. The evaluation parameters include final accuracy of the shared machine learning model, communication costs, number of communication rounds before convergence, differential privacy parameter (δ).

In our experiments, we consider ten network scenarios. The number of nodes in the federation range from 10 to 500. Each nodes has three properties as mentioned in the earlier sections, namely latitude, longitude and number of data samples. The location of the nodes in 2-D space (latitude and longitude) is generated randomly. The number of data points per node are also generated randomly. The ideal number of clusters for a particular network scenario are calculated using CFSDPF.

Network	Number of nodes in federation	Number of clusters
Net-1	10	2
Net-2	20	3
Net-3	30	3
Net-4	40	4
Net-5	50	4
Net-6	100	9
Net-7	200	17
Net-8	300	23
Net-9	400	27
Net-10	500	33

Table 2: Network scenarios considered for experimental study

6.1. MO-SSA performance metrics

To analyze the results of the state of the art algorithms, the widely used performance metrics are given as follows:

- **Overall Nondominated Vector Generation (ONVG)**: This metric indicates the total number of (nondominated vectors) solutions in the resultant pareto optimal front (PF). It is given by,

$$ONVG \triangleq |PF| \tag{33}$$

- **Spacing**: To measure the diversity of the obtained non-dominated solu- tions, spacing metric i.e., S is used. It calculates the relative distance between any two consecutive non-dominated solutions in the obtained pareto-front that is given by,

$$S \triangleq \sqrt{\frac{1}{|PF|}\sum_{j=1}^{|PF|}(o_j - \overline{o})^2} \tag{34}$$

where, o_j and o denote the distance metric of the j^{th} element of PF and mean value of the distance respectively which are formulated as:

$$o_j = \min_{k \in PF \wedge k \neq j} \sum_{n=1}^{m} \left| f_n^j - f_n^k \right| \tag{35}$$

$$\overline{o} = \sum_{j=1}^{|PF|} \frac{o_j}{|PF|} \tag{36}$$

- **Genrational distance (GD):** This metric is used to measure the close- ness of reference pareto front P Fref and PF which is given by,

$$GD \triangleq \frac{\left(\sum_{j=1}^{n} o_j^{\gamma}\right)^{1/\gamma}}{|PF|} \tag{37}$$

where, $= 2$ and oj represent the euclidean distance between each element j of PF and the closest element of P Fref respectively. If the GD value is smaller then it indicates that PF is closer to P Fref .

- **Inverted Generational Distance (IGD):** This metric is used to mea- sure the convergence and diversity. It is defined as the average distance

  from each point to its nearest counter-part in P Fref and X represents the total number of non-dominated solutions in approximation front, P Fknown.

$$IGD \triangleq \frac{\left(\sum_{v \in |PF|} d(v, X)\right)}{|PF|} \tag{38}$$

- **Hypervolume :** The HV of pareto-optimal set is the total size of the space that is dominated by other solutions (single or more) in it. The HV of a set is measured relative to a reference point.

The performance of the MOSSA is evaluated on ONVG, GD, IGD, HV and spacing metrics. The mean, standard deviation and confidence interval are accounted for each algorithm at the significance level of 0.05. In order to determine the effective values for various MOPSO test parameters, a set of training experiments is also conducted.

6.2. Experimental results

6.2.1. MOSSA metrics

In order to evaluate the overall performance of MOSSA based clustering algorithm a set of experiments are conducted, and results are also compared with other multi-objective optimization clustering algorithms i.e., MOACO, MOCuckoo, MOPSO and NSGA-II. For these set of experiments, the maximum number of iterations are fixed to 2000. The comparative statistics of ONVG, GD,IGD, HV and spacing is presented in Tables 6 - 8 respectively.

Number of nondominated solutions obtained by MOSSA are shown in Table 4 corresponding to ten network scenarios. From the results (Table 4), It is observed that the mean ONVG for MOSSA is higher than MOACO, MOCuckoo, MOCSA, MOPSO and NSGA-II. It has been that the accuracy obtained from the topology obtained from the MOSSA algorithm is much better than it's counterpart algorithms.

The hybrid topology obtains a value of model accuarcy which is comaprable

The reason for this improvement is that the the data point difference heuristic which plays an important role during position updation process. This limits the exploration of MOSSA towards the set of possible topologies, which have uniform distribution of data. This way, the performance of the algorithm is improved in terms of the cost of search process. As each generated particle has improved data samples distribution which supports easier aggregation, therefore, the values of the objective functions are better in comparison to other state of the arts.

Similarly, from Table 5, it is inferred that the mean value of GD metric is smaller than other algorithms for all twelve different network scenarios as it is desired. It attributes to the property that the algorithm converges rapidly towards the true PF. From the experimental results, it can be seen that the nondominated solutions obtained by MOSSA are closest to the true pareto front. Whereas, Table 6 presents the results of IGD parameter which are fairly compared with other state of the art algorithms. From the results, it can be seen that the value of IGD for MOSSA is less than other algorithms as the distance measure of the reference particle is lesser with the other PFs. Table 7 shows the result for the mean of HV on different network scenarios. It can be observed from the results that the value of HV is larger for MOSSA than other algorithms. It shows that the proposed algorithm has better diversity and convergence in comparison to other state of the art algorithms. Further, from Table 8, it is analysed that the use of heuristics in MOSSA helps to obtain smallest mean value for Spacing metric in comparison to other state of the art solutions.

Further, it also indicates that the generated solutions follow uniform distribution. Use of DE heuristic into turns the squirrel search agents towards true PF in an effective way. Therefore, a good trade-off between exploration and exploitation is maintained with the better diversity.

As discussed, one of the best feature of MOSSA is the use of data point difference error in particle selection during the clustering process. This evaluates the validity of the cluster combinations based on the

data sample distributions and FZVPG converts the invalid particles (generated during the particle updation) into valid particles. This process includes the DPE function which calculates the average sample difference from the mean value for a cluster and then replaces the nodes from one cluster to another, in order to generate a proper solution. in forwarding direction of destination. Doing so, the overhead occurred due to the inclusion of irrelevant cluster formations is reduced as only relevant cluster formations are searched.

Table 9 also presents the comparative statics of CPU time of MOSSA with MOACO [1], MOCuckoo [41], MOCSA [29], MOPSO [7] and NSGA-II [6] for all the results in minimum delay jitter is also shown in Figure 5d. As the communi- cation is getting completed with reduced delay and power, it means that the packets follow the path which is less congested. Pareto-fronts obtained for the given 12 network scenarios is given in Figure 6. Analysis of the results shows that the solutions obtained using MOSSA are better than the other state of the art algorithms. It is more closer to the ideal pareto front and acquire more effective solutions for different network scenarios (Net-1 to Net-12). It can be concluded from Figure 6 that MOSSA achieves better performance for all network scenarios in terms of quality of solutions.

To demonstrate the effectiveness of MOSSA, the p-values for all performance metrics such as ONVG, GD, IGD and Spacing is compared by using Student's t-test as shown in Table 10. The statistical results are obtained by one-tailed t-test at 0.05 level of significance. If p-value is less than the significance level then the dataset of 1 (obtained using FZMOPSO) is significantly better than dataset 2 (obtained using other algorithms MOACO [1], MOCuckoo [41], MOCSA [29], MOPSO [7] and NSGA-II [6]), otherwise it is worse. It is evident from Table 10 that the p-value of MOSSA is significantly better than other compared state of the art algorithms i.e., MOACO [1], MOCuckoo [41], MOCSA [29], MOPSO [7] and NSGA-II [6] for the several performance metrics: ONVG, GD, IGD and Spacing. The anomaly in just two p-values out of 42 p-values are due to false negative.

6.3. Federated Learning Metrics

6.3.1. Comparison of the state-of-the-art algorithms

In this subsection, a set of experiments are conducted to analyse the comparative performance behavior of MOSSA in terms of differential privacy preservation , communication rounds, communication costs, accuracy of the obtained machine learning model.

Differential privacy parameter (δ) indicates the probability of the privacy being breached. The other parameters such as accuracy measures the performance of final machine learning model after all the weights update sharing and aggregation amongst the cluster heads. One communication round is set to completed when all the cluster members have shared their updates to their respective cluster leaders, who have then aggregated it and then cluster leaders have also shared their aggregated weight updates amongst themselves to generate a final aggregation. Since, the nodes use stochastic gradient descent, it generally many communication rounds to evaluate updates on all the data samples. The communication costs based on the distance between the nodes, between which the transfer of weights takes place.

From the results, it can be observed that on increasing the network size and number of communication rounds, communication costs increases. This varies the performance of the algorithms accordingly. Figure 5a shows that on increasing network size, the power consumption and computation time also increases. However, MOSSA outperforms other state of the art due to the integration of data point difference error heuristic with MOSSA which truncates the search space and evaluates only proper cluster formations. This reduced search space helps the algorithm to attain the good quality of solution by the fact that the distribution of same number of particles in small region with same computational efforts significantly improves the performance.

6.3.2. Comparison of the topologies

This section compares the federated learning metrics for the three topologies, centralised , decentralised and the final hybrid topology obtained utilising the MOSSA clustering algorithm. It can be clearly observed that the number of communication rounds required are least in centralised topology, moderate in hybrid topology and the maximum in case of decentralised system. Similarly, the cost of communication also vary

Approach	Metric	Net-1	Net-2	Net-3	Net-4	Net-5	Net-6	Net-7	Net-8	Net-9	Net-10
MOSSA	Mean	7.5	6.4	6.7	6.8	7	7.4	9.4	8	3	8.9
	Std	3.100	1.837	2.057	1.619	2.624	2.221	2.368	2.503	1.632	1.911
	CI	2.218	1.315	1.472	1.158	1.878	1.589	1.695	1.713	1.168	1.368
MOACO	Mean	7.4	6.6	4.0	4.7	4.9	5.4	4.8	5.5	3.9	5.6
	Std	1.646	2.118	1.333	1.702	2.183	1.776	2.043	2.549	0.994	1.475
	CI	1.178	1.516	0.954	1.218	1.562	1.271	1.462	1.824	0.711	1.056
MOCSA	Mean	4.3	5.1	4.7	3.8	4.6	4.1	5.7	3.7	4.5	4.8
	Std	1.702	2.960	2.750	1.398	3.098	1.523	2.366	2.263	1.766	1.649
	CI	1.218	2.119	1.968	1.000	2.217	1.090	1.693	1.264	1.180	1.645
MO-Cuckoo	Mean	6.3	6.4	6.5	5.2	3.9	4.3	5.2	4.2	4.6	3.1
	Std	1.702	3.373	2.321	1.873	1.286	2.406	2.273	1.988	2.149	2.183
	CI	1.218	2.414	1.661	1.341	0.920	1.721	1.626	1.423	1.538	0.966
MOPSO	Mean	6.4	6.9	4.4	5.8	4.7	4.6	3.8	4.8	4.3	4.6
	Std	2.503	2.514	1.429	2.233	2.347	1.888	2.065	1.398	1.766	1.577
	CI	1.791	1.799	1.023	1.598	1.680	1.351	1.478	1.000	1.253	1.129
NSGA-II	Mean	6.6	6.5	4.3	4.8	5.2	3.7	4.7	5.7	3.8	4.4
	Std	3.717	1.779	1.946	2.403	2.299	2.002	1.251	2.496	2.250	2.011
	CI	2.660	1.273	1.393	1.462	1.645	1.433	0.895	1.786	1.610	1.118

Table 3: Comparison of ONVG metric for different algorithms

from lowest to highest from centralised to decentralised, while the hybrid construction depicts moderate costs. For the same levels of differential privacy, the accuracy attained for hybrid topology is comparable to the centralised federated learning architecture, while the decentralised architecture has certain lower value.

7. Conclusion

This paper addressed the problem of multi-objective clustering for the formation of hybrid topology for a federated learning setup. We introduced a way to hybridize the concept of Federated learning into MOSSA to solve the problem of ideal topology formation. We utilised the concept of uniform data distribution amongst the nodes for the formation of clusters and used data point difference error as a heuristic for the same. The resulting topology ensured the proper data distribution which fastened the process of machine learning weights aggregation. The hybrid topology thus formed accumulated the benefits of both the centralised and decentralised topologies, thus providing with a faster and privacy preserving framework simultaneously. Various other multi-objective optimization methods were also checked for the clustering problem, but the addition of data point error heuristic made the search space smaller, thus leading to faster convergence in case of MOSSA algorithm. Moreover, the levy flights in MOSSA, provided for better exploration and thus better diversity of solutions. The resulting clusters were compared on various federated learning metrics such as model accuracy, differential privacy parameter, communication costs and number of communication costs. Extensive set of experiments evidenced the effectiveness of MOSSA. Results are compared with other meta-heuristic techniques i.e., MOACO [1], MOCuckoo [4], MOCSA [9], MOPSO [7] and NSGA-II [6] in terms of multi-objective optimization metrics such as ONVG, Generational distance (GD), IGD, HV and Spacing. Results evidenced that MOSSA generates a hybrid topology for a federation of nodes which performs well on the federated learning metrics.

References

[1] LeCun, Y., Bengio, Y. and Hinton, G., 2015. Deep learning. nature, 521(7553), p.436.

Approach	Metric	Net-1	Net-2	Net-3	Net-4	Net-5	Net-6	Net-7	Net-8	Net-9	Net-10
MOSSA	Mean	0.171	0.226	0.241	0.236	0.182	0.192	0.258	0.195	0.218	0.179
	Std	0.60	0.076	0.110	0.092	0.052	0.043	0.092	0.074	0.059	0.047
	CI	0.043	0.055	0.079	0.066	0.037	0.031	0.065	0.052	0.042	0.034
MOACO	Mean	0.211	0.235	0.268	0.249	0.272	0.238	0.305	0.301	0.362	0.208
	Std	0.070	0.086	0.110	0.080	0.097	0.066	0.108	0.107	0.109	0.031
	CI	0.050	0.061	0.079	0.057	0.069	0.047	0.077	0.076	0.078	0.022
MOCSA	Mean	0.351	0.263	0.247	0.297	0.331	0.305	0.250	0.409	0.448	0.386
	Std	0.107	0.052	0.054	0.145	0.126	0.109	0.079	0.081	0.156	0.169
	CI	0.076	0.037	0.038	0.104	0.090	0.078	0.057	0.058	0.112	0.121
MO-Cuckoo	Mean	0.236	0.215	0.257	0.304	0.299	0.313	0.350	0.245	0.372	0.218
	Std	0.087	0.088	0.075	0.126	0.092	0.136	0.146	0.060	0.108	0.079
	CI	0.062	0.063	0.054	0.090	0.066	0.097	0.104	0.043	0.077	0.057
MOPSO	Mean	0.300	0.229	0.325	0.243	0.292	0.267	0.288	0.364	0.316	0.268
	Std	0.167	0.080	0.079	0.065	0.216	0.051	0.106	0.129	0.120	0.105
	CI	0.062	0.063	0.054	0.090	0.066	0.097	0.104	0.043	0.077	0.057
NSGA-II	Mean	0.307	0.222	0.260	0.297	0.284	0.369	0.251	0.281	0.430	0.283
	Std	0.315	0.061	0.107	0.104	0.118	0.226	0.034	0.119	0.166	0.059
	CI	0.225	0.044	0.0769	0.074	0.084	0.161	0.024	0.085	0.118	0.042

Table 4: Comparison of GD metric for different algorithms

Approach	Metric	Net-1	Net-2	Net-3	Net-4	Net-5	Net-6	Net-7	Net-8	Net-9	Net-10
MOSSA	Mean	0.070	0.226	0.241	0.236	0.182	0.192	0.258	0.195	0.218	0.179
	Std	0.060	0.076	0.110	0.092	0.052	0.043	0.092	0.074	0.059	0.047
	CI	0.043	0.055	0.079	0.066	0.037	0.031	0.065	0.052	0.042	0.034
MOACO	Mean	0.211	0.235	0.268	0.269	0.272	0.238	0.305	0.301	0.362	0.208
	Std	0.070	0.086	0.110	0.080	0.097	0.066	0.108	0.107	0.109	0.031
	CI	0.050	0.061	0.079	0.057	0.069	0.047	0.077	0.076	0.078	0.022
MOCSA	Mean	0.351	0.263	0.247	0.297	0.331	0.305	0.250	0.409	0.448	0.386
	Std	0.107	0.052	0.054	0.145	0.126	0.109	0.079	0.081	0.156	0.169
	CI	0.076	0.037	0.038	0.104	0.090	0.078	0.057	0.058	0.112	0.066
MO-Cuckoo	Mean	0.236	0.215	0.257	0.304	0.299	0.313	0.350	0.245	0.372	0.218
	Std	0.087	0.088	0.075	0.126	0.092	0.136	0.146	0.060	0.108	0.079
	CI	0.062	0.063	0.054	0.090	0.066	0.097	0.104	0.043	0.077	0.057
MOPSO	Mean	0.300	0.229	0.325	0.243	0.292	0.267	0.288	0.364	0.316	0.268
	Std	0.167	0.080	0.080	0.065	0.216	0.051	0.106	0.129	0.120	0.105
	CI	0.119	0.057	0.047	0.155	0.037	0.076	0.092	0.086	0.075	0.039
NSGA-II	Mean	0.307	0.222	0.260	0.297	0.284	0.359	0.251	0.281	0.430	0.283
	Std	0.315	0.061	0.107	0.104	0.118	0.226	0.034	0.119	0.166	0.059
	CI	0.225	0.044	0.0769	0.074	0.084	0.161	0.024	0.085	0.118	0.042

Table 5: Comparison of IGD metric for different algorithms

Approach	Metric	Net-1	Net-2	Net-3	Net-4	Net-5	Net-6	Net-7	Net-8	Net-9	Net-10
MOSSA	Mean	0.020	0.026	0.030	0.047	0.067	0.059	0.105	0.115	0.178	0.307
	Std	0.007	0.006	0.055	0.030	0.033	0.018	0.060	0.058	0.131	0.156
	CI	0.005	0.004	0.039	0.022	0.024	0.013	0.043	0.041	0.094	0.111
MOACO	Mean	1.490	0.108	0.042	0.150	0.174	0.181	0.164	0.185	0.159	0.335
	Std	0.880	0.066	0.027	0.038	0.029	0.060	0.087	0.054	0.090	0.273
	CI	0.629	0.047	0.027	0.021	0.043	0.062	0.039	0.064	0.195	0.145
MOCSA	Mean	0.167	0.199	0.216	0.220	0.299	0.266	0.685	0.449	2.090	2.935
	Std	0.062	0.091	0.134	0.145	0.141	0.165	0.212	0.402	0.331	0.943
	CI	0.044	0.065	0.096	0.104	0.101	0.118	0.152	0.288	0.237	0.675
MO-Cuckoo	Mean	0.045	0.028	0.052	0.057	0.104	0.059	0.142	0.180	0.200	0.622
	Std	0.026	0.015	0.027	0.037	0.113	0.040	0.065	0.105	0.053	0.268
	CI	0.019	0.011	0.019	0.026	0.081	0.029	0.047	0.075	0.038	0.192
MOPSO	Mean	1.588	2.110	3.722	4.235	5.516	6.270	11.886	13.237	15.265	18.296
	Std	0.932	1.619	2.182	3.454	4.274	3.488	6.532	7.985	5.584	13.867
	CI	0.667	1.158	1.562	2.400	3.058	2.496	4.675	5.714	3.996	9.924
NSGA-II	Mean	2.135	2.537	3.109	2.744	4.619	7.169	7.578	12.134	16.395	14.186
	Std	1.031	1.321	2.094	1.601	3.718	4.362	2.969	5.997	12.435	6.903
	CI	0.945	1.498	1.146	2.661	3.121	2.215	4.292	8.900	4.940	9.947

Table 6: Comparison of spacing metric for different algorithms

Approach	Net-1	Net-2	Net-3	Net-4	Net-5	Net-6	Net-7	Net-8	Net-9	Net-10
MOSSA	2.861	2.970	2.982	3.101	3.124	3.431	3.475	3.460	3.501	3.524
MOACO	2.079	2.003	2.479	2.632	2.638	2.604	2.637	2.951	2.879	2.628
MOCSA	2.133	2.662	2.515	2.571	2.657	2.884	2.590	2.531	2.807	2.474
MO-Cuk	2.122	2.136	2.256	2.686	2.603	2.884	2.805	2.945	2.503	2.650
MOPSO	2.144	2.655	2.481	2.149	2.538	2.713	2.726	2.508	2.474	2.635
NSGA-II	2.528	2.704	2.713	2.764	2.478	2.687	2.740	2.861	2.635	2.943

Table 7: Comparison of hypervolume metric for different algorithms

Approach	Net-1	Net-2	Net-3	Net-4	Net-5	Net-6	Net-7	Net-8	Net-9	Net-10
MOSSA	2.098	3.970	5.982	7.101	10.124	12.431	16.475	19.460	23.501	31.524
MOACO	3.079	4.003	6.479	7.632	10.638	12.604	17.637	20.951	22.879	32.628
MOCSA	2.133	2.962	5.515	7.571	9.657	12.884	15.590	20.531	26.807	33.474
MO-Cuk	5.122	7.136	8.256	9.686	12.603	15.884	20.805	24.945	27.503	37.650
MOPSO	3.123	4.655	6.481	7.149	10.538	13.713	17.726	19.508	24.474	32.635
NSGA-II	3.528	5.704	7.713	9.764	13.478	15.687	24.740	27.861	29.635	35.943

Table 8: Comparison of computation time (in minutes) metric for different algorithms

Approach	Metric	Net-1	Net-2	Net-3	Net-4	Net-5	Net-6	Net-7	Net-8	Net-9	Net-10
MOSSA	Accuracy	70.67	71.09	71.98	72.56	73.98	74.09	75.98	76.88	77.23	79.03
	#CR	13	24	45	98	189	12980	25567	51342	109786	203456
	δ	0.0181	0.0190	0.0211	0.0234	0.0242	0.0245	0.0250	0.0256	0.0266	0.0279
	CC	122	234	556	1234	2890	134561	290879	620987	1238976	2567890
MOACO	Accuracy	60.27	60.99	61.18	61.98	62.18	62.97	63.45	63.86	64.23	64.53
	#CR	17	34	70	135	299	17780	35597	71324	142386	283456
	δ	0.0201	0.0219	0.0221	0.0234	0.0244	0.0265	0.0263	0.0264	0.0268	0.0278
	CC	125	244	555	1299	2991	132321	291079	621197	1245716	2513566
MOCSA	Accuracy	63.24	63.77	63.98	64.46	65.08	65.99	66.23	66.88	67.23	68.07
	#CR	19	38	78	157	319	19780	38560	75123	153423	298336
	δ	0.0191	0.0210	0.0222	0.0233	0.0242	0.0245	0.0260	0.0265	0.0276	0.0288
	CC	131	274	586	1334	2909	14461	299423	634587	1300076	2589897
MO-Cuckoo	Accuracy	67.67	68.19	68.98	69.56	69.98	71.09	71.98	72.08	72.23	72.93
	#CR	20	39	77	156	323	20980	41557	81349	179786	333456
	δ	0.0192	0.0198	0.0201	0.0224	0.0248	0.0255	0.0259	0.0266	0.0276	0.0283
	CC	129	256	598	1301	2787	131123	272379	599787	1234767	2612334
MOPSO	Accuracy	68.14	68.79	70.98	71.05	71.98	72.09	72.45	72.88	73.23	74.13
	#CR	23	54	105	198	389	14980	28567	56349	129756	223456
	δ	0.0188	0.0197	0.0231	0.0244	0.0252	0.0255	0.0260	0.0276	0.0286	0.0289
	CC	133	274	656	1335	2995	144561	296879	633487	1294976	2617890
NSGA-II	Accuracy	64.22	64.59	64.98	65.56	65.98	66.09	66.78	67.88	67.23	68.65
	#CR	15	32	65	123	256	14985	29867	60342	119783	223355
	δ	0.0201	0.0213	0.0223	0.0238	0.0252	0.0258	0.0267	0.0276	0.0287	0.0293
	CC	127	267	596	1304	2888	131123	296754	630007	1234561	2607908

Table 9: Comparison of federated learning metrics for various algorithms

Topology based comparison of differentially private federated learning				
Federated learning in centralised topology				
Clients	δ	Accuracy(%)	CR	CC
Net- 1	0.0281	76.70	7	54
Net- 2	0.0290	77.90	12	112
Net- 3	0.0291	77.98	28	232
Net- 4	0.0304	78.56	55	456
Net- 5	0.0312	78.88	109	986
Net- 6	0.0325	79.03	6789	5189
Net- 7	0.0326	79.17	11908	9807
Net- 8	0.0333	80.18	20980	18765
Net- 9	0.0342	80.23	44567	34538
Net- 10	0.0352	80.99	85670	65743
Federated learning in decentralised topology				
Clients	δ	Accuracy (%)	CR	CC
Net- 1	0.0181	60.06	87	122
Net- 2	0.0190	60.29	243	234
Net- 3	0.0211	61.08	721	556
Net- 4	0.0234	61.56	1456	1234
Net- 5	0.0242	61.16	2786	2890
Net- 6	0.0245	61.09	79810	134561
Net- 7	0.0250	62.01	160879	290879
Net- 8	0.0256	62.88	334098	620987
Net- 9	0.0266	62.23	675432	1238976
Net- 10	0.0279	62.33	1348745	2567890
Federated learning with hybrid topology obtained through MOSSA				
Clients	δ	Accuracy(%)	CR	CC
Net- 1	0.0133	70.76	13	487
Net- 2	0.0145	71.09	24	986
Net- 3	0.0152	71.98	45	2310
Net- 4	0.0156	72.56	98	4321
Net- 5	0.0166	73.98	189	8176
Net- 6	0.0169	74.09	12980	487562
Net- 7	0.0173	75.98	25567	875634
Net- 8	0.0173	76.88	51342	1783421
Net- 9	0.0177	77.23	109786	3568654
Net- 10	0.0179	79.03	203456	7124567

Table 10: Comparison of federated learning metrics for centralised, decentralised and hybrid tolpogies

[2] Theodora S Brisimi, Ruidi Chen, Theofanie Mela, Alex Olshevsky, Ioannis Ch Paschalidis, and Wei Shi. Federated learning of predictive models from federated electronic health records. International Journal of Medical Informatics , 2018.

[3] Konečný, J., McMahan, H.B., Yu, F.X., Richtárik, P., Suresh, A.T. and Bacon, D., 2016. Federated learning: Strategies for improving communication efficiency.

[4] Sundermeyer, M., Schlüter, R. and Ney, H., 2012. LSTM neural networks for language modeling. In Thirteenth Annual Conference of the International Speech Communication Association.

[5] McMahan, H.B., Moore, E., Ramage, D. and Hampson, S., 2016. Communication-efficient learning of deep networks from decentralized data.

[6] H. Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. Federated learning of deep networks using model averaging, 2016

[7] McMahan, H.B., Ramage, D., Talwar, K. and Zhang, L., 2017. Learning differentially private language models without losing accuracy. .

[8] Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H.B., Patel, S., Ramage, D., Segal, A. and Seth, K., 2017, October. Practical Secure Aggregation for Privacy-Preserving Machine Learning. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (pp. 1175-1191).

[9] Abadi, Martin, et al. "Deep learning with differential privacy." Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016.

[10] Chaudhuri, K., Monteleoni, C. (2009). Privacy-preserving logistic regression. In Advances in neural information processing systems (pp. 289-296).

[11] Fredrikson, M., Jha, S. and Ristenpart, T., 2015, October. Model inversion attacks that exploit confidence information and basic countermeasures. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (pp. 1322-1333).

[12] Douceur, John R. "The sybil attack." In International workshop on peer-to-peer systems, pp. 251-260. Springer, Berlin, Heidelberg, 2002.

[13] Fung, Clement, Chris JM Yoon, and Ivan Beschastnikh. "Mitigating sybils in federated learning poisoning." arXiv preprint arXiv:1808.04866 (2018).

[14] Boyd, Stephen, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. "Randomized gossip algorithms." IEEE/ACM Transactions on Networking (TON) 14, no. SI (2006): 2508-2530.

[15] https://www.swirlds.com/downloads/SWIRLDS-TR-2016-01.pdf