Federated Learning Strategies for Enhancing Privacy-Preserving Euclidean Distance Matrix Computation

Vikrant Shokeen
Department of Computer Science & Engineering, Maharaja Surajmal Institute of Technology, Delhi, India
shokeen18@gmail.com

Sandeep Kumar
Department of Computer Science & Engineering, Maharaja Surajmal Institute of Technology, Delhi, India
san.jaglan@gmail.com

Amit Sharma
Department of Computer Science, IMS Engineering College, Ghaziabad, UP, India
amit.faculty@gmail.com

Ahmad Taher Azar
College of Computer and Information Sciences, Prince Sultan University, Riyadh, Saudi Arabia | Automated Systems and Computing Lab (ASCL), Prince Sultan University, Riyadh, Saudi Arabia
aazar@psu.edu.sa (corresponding author)

Samah Almutlaq
College of Computer and Information Sciences, Prince Sultan University, Riyadh, Saudi Arabia | Automated Systems and Computing Lab (ASCL), Prince Sultan University, Riyadh, Saudi Arabia
salmutlaq@psu.edu.sa

Received: 7 March 2025 | Revised: 13 April 2025, 26 April 2025, and 1 May 2025 | Accepted: 4 May 2025

 DOI: https://doi.org/10.48084/etasr.10840

ABSTRACT

Numerous Machine Learning (ML) algorithms require pairwise Euclidean distance computations between all data points in horizontally partitioned datasets. To ensure data privacy, most existing solutions incorporate encryption or cryptographic techniques, allowing secure sharing and computation of data points. In this study, we propose two methodologies for generating Euclidean distance matrices: the Federated Euclidean Distance Matrix (FEDM) and the Predicted Euclidean Distance Matrix (PEDM), derived from the pairwise Euclidean distances of horizontally partitioned data to ensure privacy by design. The proposed approach has significant potential to transform the execution of ML algorithms that rely on Euclidean distance calculations and to eliminate the need for separate encryption methods, thereby potentially reducing communication and computation costs in a Federated Learning (FL) environment. The proposed methods achieve high accuracy and exhibit strong similarity to the actual Euclidean distance matrix. FL has gained prominence as a privacy-preserving ML solution that encapsulates data while appropriately sharing model parameters. To this end, we utilize artificial spike points for the creation of FEDM. We also elucidate the foundational workflows of the method and matrix construction and demonstrate their efficacy through comprehensive experimentation.

Keywords-Federated Learning (FL); Machine Learning (ML); Federated Euclidean Distance Matrix (FEDM); Predicted Euclidean Distance Matrix (PEDM); pairwise Euclidean distance

I. INTRODUCTION

To ensure data privacy, many privacy-preserving techniques have been introduced. Broadly, these techniques can be viewed as a trade-off between data accuracy and privacy, where a system relying on such a technique must sacrifice privacy for higher accuracy and vice versa. In traditional Machine Learning (ML) [1], data from different sites need to be sent and aggregated in a centralized location for model training, which is highly vulnerable to data breaches and leakage.

Federated Learning (FL) addresses these vulnerabilities by allowing personal data to remain on local sites, thereby reducing the risk of data breaches while enabling personalized solutions by preserving data privacy. FL trains ML algorithms using multiple local datasets to create a shared global model without exchanging the raw training data in a central location. This approach is particularly effective for large datasets, where the trade-off of adding noise to ensure privacy against accuracy can be sustainable.

Nevertheless, even FL and other privacy-preserving ML techniques face challenges. They often introduce significant communication, computational, and network overhead while sacrificing some accuracy, which can hinder the practical deployment of ML solutions across diverse sectors [2-10]. Nevertheless, all these above-mentioned privacy-preserving techniques create communication, computational and network overhead while losing a considerable amount of accuracy, which can be a major challenge for useful ML implementation. At the same time, ML techniques have demonstrated practical value across all aspects of life, making it essential to develop privacy-preserving solutions that maintain efficiency and accuracy.

II. BACKGROUND

FL, also referred to as collaborative learning, enables multiple participants to train a robust shared model without sharing raw data, addressing critical challenges such as privacy, accessibility, and heterogeneous datasets. FL revolutionizes ML by decentralizing model training across multiple devices without exposing individual device data. Additionally, FL frameworks, including horizontal, vertical, and federated transfer learning, cater to diverse data distribution scenarios, ensuring privacy and collaboration across different entities. This approach allows collaborative model training while preserving data privacy across different feature domains.

Unlike traditional methods that centralize data on a server, FL preserves privacy and mitigates communication overhead by allowing devices to independently train models with their respective data. In horizontal FL, entities share identical sets of features while training a centralized model without data sharing. Data partitioning by user or device IDs further enhances privacy and preserves a horizontal data partitioning structure. Research emphasizes on-device FL, optimizing communication costs, enhancing device data security, and addressing data distribution imbalances to improve overall model performance. In contrast, vertical FL involves dissimilar feature sets among entities, necessitating secure alignment of data through techniques such as private set interaction [11].

III. RELATED WORKS

Privacy-preserving techniques and dataset transformations are essential in both FL and non-FL environments to safeguard the privacy of data points. These approaches typically involve using actual or noise data points to calculate aggregated Euclidean distance matrices. However, many existing methods entail direct or indirect sharing of datasets among multiple owners, posing privacy risks. Authors in [12] proposed a method for calculating Euclidean distances in vertically partitioned data, highlighting the difficulty of integrating overall distances without the use of encryption techniques. In [13], the authors proposed using Fourier transforms to maintain the distance between points while ensuring data privacy. However, setting a distance difference threshold for non-uniform or unknown dataset distributions remains challenging. Although various algorithms exist for calculating Euclidean distance matrices for single datasets, they are not applicable to multiple datasets, and privacy concerns are often overlooked, which this study aims to address. Overall, existing studies focus on improving computational or memory efficiency for large-scale Euclidean distance matrix calculations.

IV. METHODS

The Federated Euclidean Distance Matrix (FEDM) is initially constructed by adding random points, referred to in this paper as spike points, to each participant's dataset. Then, pairwise distance matrices, called Local Spike Distance Matrices (LSDMs), are generated between the spike points and the data points at each participant's end. These LSDMs are concatenated locally and shared with the coordinator, where the pairwise Euclidean distances across the datasets of N participants are aggregated to form the final FEDM and the Predicted Euclidean Distance Matrix (PEDM). Both FEDM and PEDM are $N \times N$ matrices, where N is the total number of samples across all participants. Each LSDM is a $S \times N$ matrix, where S is the total number of spike points and N is the total number of data samples for that participant.

To improve the correlation of the FEDM with respect to the true Aggregated Distance Matrix (ADM), the PEDM is calculated using regression models such Linear, Huber, and Theil-Sen [14]. The regression coefficients and intercepts derived locally for each participant are shared with the coordinator, which then applies them to adjust the FEDM. Overall, both FEDM and PEDM are constructed at the coordinator's end, minimizing computational and network communication overhead for the participants.

A. Assumptions

The study assumes that both participants and the coordinator act as computational resources, meaning they have the computing capabilities required to perform tasks within the system. It accommodates the possibility of multiple participants (denoted by N) and a single coordinator. Additionally, the coordinator can also fulfill the role of a participant if required, ensuring flexibility in the system's operation. Each participant contributes data from a single dataset for every operation

involving distance matrices, implying a controlled sharing mechanism that maintains data privacy and integrity. All data points within the participant's datasets are treated as vectors and possess the same number of dimensions. This assumption ensures uniformity in data representation and simplifies the computational process. This study assumes that the number of spike points in the datasets provided by participants does not exceed the number of dimensions. This assumption helps maintain the scalability and feasibility of the distance matrix calculations, preventing potential computational issues.

B. Spike Points Generation Methods

Whether adjusting the number of spike points for compression or expansion, the underlying methods aim to optimize the dataset for subsequent analysis and interpretation. These methods interpolate additional spike points between existing data points, effectively densifying the dataset. Conversely, reducing the number of spike points can streamline the dataset but may increase the risk of oversimplification and information loss. For instance, in scenarios where the dataset exhibits a high degree of variability or complexity, generating a larger number of spike points may be warranted.

1) Overview

Spike points serve as artificial data points, acting as centroids or uniformly distributed random samples, common across all participants' datasets. In a federated setting, where each participant has their own dataset (denoted as R for participant P_1 and $B_1 \dots B_N$ for participant P_2), the FEDM emerges as a meta-distance metric. FEDM is represented as $euclidean\_dist(euclidean\_dist(A_1 \dots A_N, S_1 \dots S_N)$, $euclidean\_dist(B_1 \dots B_N, S_1 \dots S_N))$, providing an approximation of the true Euclidean distance between datasets. The privacy of data points is ensured, as the total number of spike points is always less than the dimension of the data points. The PEDM utilizes derived coefficients and intercepts for each value of FEDM, serving as a distance matrix.

Several critical considerations govern the selection of spike locations, ensuring utility and replicability. These include the unpredictability of the underlying data distribution, the dataset's dimensionality (which limits the maximum number of spike points), and the trade-off between compression and expansion based on local data properties. Furthermore, participants can customize the number of spike points based on their computational resources and privacy needs, allowing flexible and consistent deployment across varied circumstances.

2) Centroid-Based Spike Points

In this method, the coordinator aggregates all centroids received from participants to form the final array of spike points. In this innovative approach for generating spike points, each participant's dataset's centroids are considered as the spike points. The process begins with participants employing clustering algorithms such as K-means or affinity propagation on their respective datasets to identify centroids. Once centroids are identified, each participant shares their centroids with the coordinator (refer to Algorithm 1).

```
Algorithm 1: Generate spike points based
on centroid(s) of each participants
dataset
START
Data: number_of_participants,
      number_of_cluster (optional)
Result: spike_point (Array)
Initialize spike_point [];
For each participant_index from 1 to
number_of_participants do
   Perform K-means clustering with k =
   number_of_cluster on participant_index's
   data
   Return the cluster centers to the
   coordinator
   Append the returned cluster centers to
   spike_point (Array)
Return spike_point (Array)
Output: spike_point (Array): Array
containing cluster centers
END
```

3) Uniformly-Based Spike Points

In this method, each participant performs Principal Component Analysis (PCA) on their dataset independently. Subsequently, random spike points are uniformly distributed across a half-open interval. This interval spans from the lowest value of all features across all samples to the highest value across all samples, serving as the local spike points. These spike points are then transformed back to their original feature magnitudes before being transmitted to the coordinator. Upon receiving spike points from all participants, the coordinator aggregates and broadcasts them to create the FEDM model (refer to Algorithms 2 and 3).

```
Algorithm 2: Generate spike points from
each participant using PCA covering
desired variance
START
Input: variance (desired variance for
     PCA), dataset (input dataset),
     no_of_spikes (number of spike points
     to generate), reduce (flag indicating
     whether to reduce dimensionality
     using PCA, default = 1), induce (flag
     indicating whether to induce variance
     or not, default = 1)
Initialize generated_spikes as an empty
array
Perform PCA on dataset with desired
variance:
     dataset_pca = perform_PCA(variance,
     dataset)
Generate spike points uniformly along each
principal component:
   Determine the minimum and maximum values
   along each principal component:
     min_values = min(dataset_pca)
     max_values = max(dataset_pca)
```

```
   Generate spike points uniformly within
   the range defined by min_values and
   max_values:
      generated_spikes =
      get_uniform_samples(low=min_values,
      high=max_values, size=(no_of_spikes,
      dimension_of_dataset_pca))
If reduce flag is set to 1, reduce the
dimensionality of generated_spikes using
inverse_transform() function:
     generated_spikes =
     inverse_transform(generated_spikes)
Return generated_spikes (Array)
Output: generated_spikes (Array): Array
containing spike points
END
```

```
Algorithm 3: Generate spike points based
on uniform random samples
START
Input: number_of_participants (the total
     number of participants),
     generated_spikes_of_each_participant
     (array containing the generated
     spikes of each participant)
Initialize generated_spikes as an empty
array
For each participant_index from 1 to
number_of_participants do:
   Perform Algorithm 2 on the dataset of
   the current participant
   Get the uniformly distributed random
   samples from the coordinator
   Append the random samples to the
   generated_spikes array
Return generated_spikes (Array)
Output: generated_spikes (Array): Array
containing all the generated spikes
END
```

4) Random Samples Spike Points with Centroid

In this method, participant generates consistent random spike positions according to Algorithm 2, alongside centroids determined by Algorithm 1. Subsequently, each participant merges all random samples and centroids as spike positions before transmitting them to the coordinator. Upon receiving submissions from all participants, the coordinator has the option to either return a predetermined number of points or all spike positions, as appropriate (refer to Algorithms 4 and 5).

```
Algorithm 4: Generate spike points from
each participant using PCA covering the
desired variance along with its
centroid(s)
START
Input: variance (desired variance to cover
     in PCA), dataset (original dataset),
     centroids (centroids of the
     clusters), no_of_spikes (number of
     spikes to generate), reduce (boolean
     indicating whether to perform
     dimensionality reduction), induce
     (boolean indicating whether to induce
     new spikes or not)
Initialize generated_spikes_with_centroids
as an empty array
Perform PCA on the dataset with the
desired variance:
     dataset_pca = perform_PCA(variance,
     dataset)
Generate uniformly distributed random
samples in the PCA space:
     generated_spikes =
     get_uniform_samples(low=min(dataset_p
     ca), high=max(dataset_pca),
     size=(no_of_spikes,
     dimension_of_dataset_pca))
Inverse transform the generated spikes to
the original dataset space:
     inverse_transform(generated_spikes)
Save the random samples in
generated_spikes_with_centroids
Concatenate the centroids with
generated_spikes_with_centroids
Return generated_spikes_with_centroids
(Array)
Output: generated_spikes_with_centroids
(Array): Array containing generated spikes
along with centroids
END
```

```
Algorithm 5: Generate spike points based
on uniform random samples using each
participant's dataset and its centroid(s)
START
Data: number_of_participants,
      generated_spikes_of_each_participant
Initialize
generated_spikes_for_all_participants as
an empty array
For each participant_index from 1 to
number_of_participants do:
   Retrieve the dataset and centroid(s) of
   the current participant
   Perform Algorithm 4 (algorithm for
   generating spike points based on dataset
   and centroids)
   Get uniformly distributed random samples
   to coordinator
   Append the generated random samples to
   generated_spikes_for_all_participants
   array
Return
generated_spikes_for_all_participants
(Array)
Output:
generated_spikes_for_all_participants
```

```
(Array): Array containing generated spike
points for all participants
END
```

C. Federated Euclidean Distance Matrix

One of the compiled distance matrices obtained through Euclidean distance calculations provided by various participants is the FEDM. It represents the aggregated Euclidean distances of pairwise distances between all data points and spike points from each participant. The steps involved in constructing the FEDM are outlined as follows:

1. Each participant generates spike points from their dataset(s) using any chosen methods; participants may opt for different methods.
2. Participants share the generated spike points along with their respective dataset sizes with the coordinator.
3. The coordinator consolidates all received spike points from participants and broadcasts them to all participants.
4. Each participant constructs a pairwise distance matrix, referred to as LSDM, between their data points and the spike points.
5. The coordinator concatenates all LSDMs from each participant.
6. The coordinator constructs the FEDM by computing the pairwise Euclidean distances between every point within the concatenated LSDMs shared by all participants.

D. Predicted Euclidean Distance Matrix

To evaluate the efficiency of the regression models used in PEDM creation, conventional measures such as the coefficient of determination (R^2), Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE) are computed between predicted and actual distance values. These evaluations are carried out locally by each participant during model training to ensure that the learnt regression parameters (M, C) appropriately describe the relationship between LSDM and Local Distance Matrix (LDM), which represents the true pairwise Euclidean distances within the participant's dataset. The choice of regression model has a considerable impact on the accuracy and robustness of the PEDM. While linear regression provides a simple and understandable mapping between LSDM and LDM, Huber regression enhances robustness to outliers, which is especially beneficial when participant datasets are noisy or contain abnormalities. Our tests show that using Huber regression improves PEDM stability and reduces reconstruction error, particularly in heterogeneous and imbalanced data scenarios.

V. DATASETS

Two types of datasets are used for experimental and evaluation purposes: artificial blobs of data points and a gene expression dataset. To handle circumstances with imbalanced participant datasets, our suggested approaches use class-weighted loss functions and data augmentation strategies during training. These strategies reduce bias toward majority classes and increase model robustness for minority participant groups. Furthermore, performance is evaluated using class-specific criteria to ensure a fair assessment across all classes.

A. Artificial Datasets

Two distinct categories were employed in artificial datasets: isotropic Gaussian blobs with non-uniformly distributed random samples [15] and uniformly distributed random data points [16]. The Gaussian blobs simulate structured clusters, allowing evaluation of algorithm performance on data with inherent grouping, whereas the uniformly distributed data provide a baseline representation of randomness, enabling assessment of robustness and scalability when handling unstructured or evenly distributed datasets.

B. Gene Expression Datasets

In this context, PCA is conducted on the datasets to the first two or three principal components, condensing them while retaining maximum variance for visualization purposes. The first dataset comprises gene expression profiles from gastrectomy samples of 76 gastric cancer patients. The second dataset consists of gene expression profiles from 357 gastrectomy samples. Both datasets were analyzed using the HumanHT-12 v3.0 Expression BeadChip array (Illumina), denoted as GSE84426 and GSE84433, respectively. Total Ribonucleic Acid (RNA) was extracted from fresh-frozen gastrectomy specimens at Yonsei University Severance Hospital (South Korea) between 2000 and 2010 [17]. The gene expression profiles from both datasets are subsequently used to compute the FEDM and PEDM matrices.

VI. IMPLEMENTATION

To emulate a centralized federated setting during experimentation, each dataset underwent random shuffling and subsequent division into two distinct, non-overlapping subsets. The distribution of samples within each subset was deliberately uneven. Each subset represented a collection of distinct data entries from two separate participants yet sharing identical feature spaces. These participants exchanged intermediate results, namely LSDMs, to facilitate coordinator-based computation.

Throughout the simulation of the FL environment and the computation of FEDM and PEDM, network dependencies and communication overhead were not considered. In accordance with the fundamental principles outlined in [18], the FL environment for computing FEDM and PEDM involves two distinct roles: participants and coordinators, primarily defined based on their computational tasks. Participants generate local spike points using their chosen spike point generation method and transmit them to the coordinator. Each participant also conducts regression analysis using its LSDM values as independent variables (x) and its own data points as dependent variables (y).

It is assumed that there is a single coordinator in the FeatureCloud implementation. The system configurations used for the construction and evaluation of FEDM, PEDM, and ADM, as well as for comprehensive testing are shown in Table I. System 01 was utilized for development and preliminary testing, encompassing smaller artificially generated datasets in addition to the GSE84426 dataset. System 02 was primarily

utilized for extensive experimentation, particularly with larger datasets. Python was the chosen programming language, with version 3.12.2 used in System 01 and 3.11.8 in System 02. System 02, operating as an Ubuntu Linux server, used VIM (Vi Improved 8.1) as the preferred code editor. In contrast, System 01, primarily designated for coding and initial prototype development, utilized Visual Studio Code 1.87 as both the code editor and Integrated Development Environment (IDE).

TABLE I. SYSTEM CONFIGURATION FOR EXPERIMENTAL EVALUATION

Component	System 01	System 02
Processor	Intel® Core™ i5-4300U CPU @ 1.90 GHz	Intel® Xeon® CPU E5-2680 v3 @ 2.50 GHz
Architecture	x64-based	x64-based
RAM	8.00 GB (7.69 GB usable)	512 GB
Storage (HDD)	250 GB (approx.)	3.35 TB
Operating system	Windows 10 Home 20H2	Ubuntu 20.04.3 LTS

VII. RESULTS

A. Artificial Datasets

1) Non-Uniformly Distributed

Figure 1 illustrates the Pearson correlation between the ADM and FEDM across three distinct datasets, each varying in sample size and dimensionality, as the number of spike points increases. Similarly, Figure 2 portrays the correlation between ADM and PEDM. Figure 1 reveals that, regardless of the number of spike points used, FEDM constructed from participants' centroids exhibit almost 100% correlation with ADM, as indicated by the Pearson coefficient. As the number of spike points increases, the Pearson correlation between FEDM and ADM rises, regardless of whether the spike points are randomly generated or a combination of random points and centroids. In Figure 2, a similar trend is observed in the correlation of PEDM with ADM across all datasets compared to FEDM in Figure 1. Notably, PEDM, generated from FEDM constructed using spike points, exhibit almost identical correlation with ADM, with the Pearson correlation coefficient approaching 1.0.

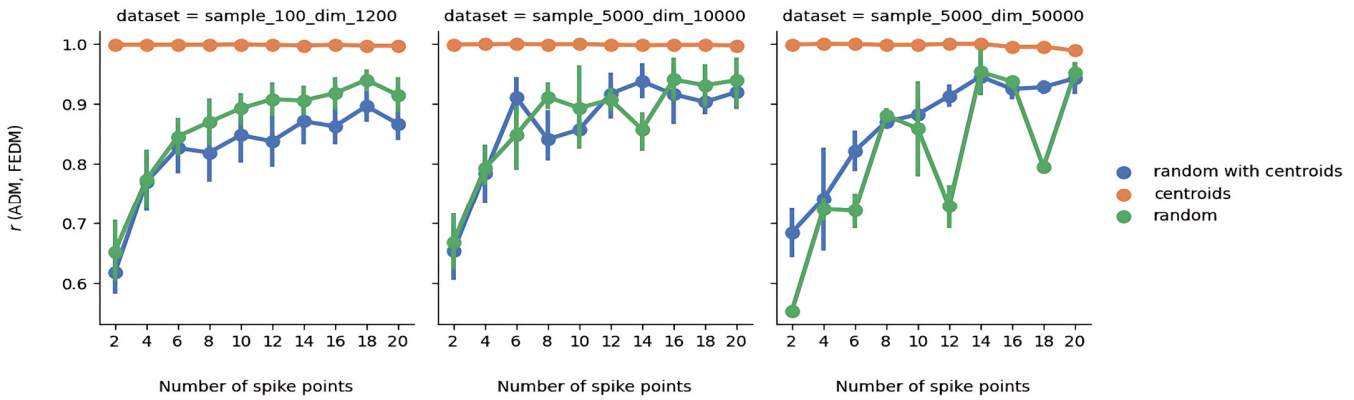

Fig. 1. Pearson coefficient of the FEDM with respect to ADM for datasets with non-uniform distribution.

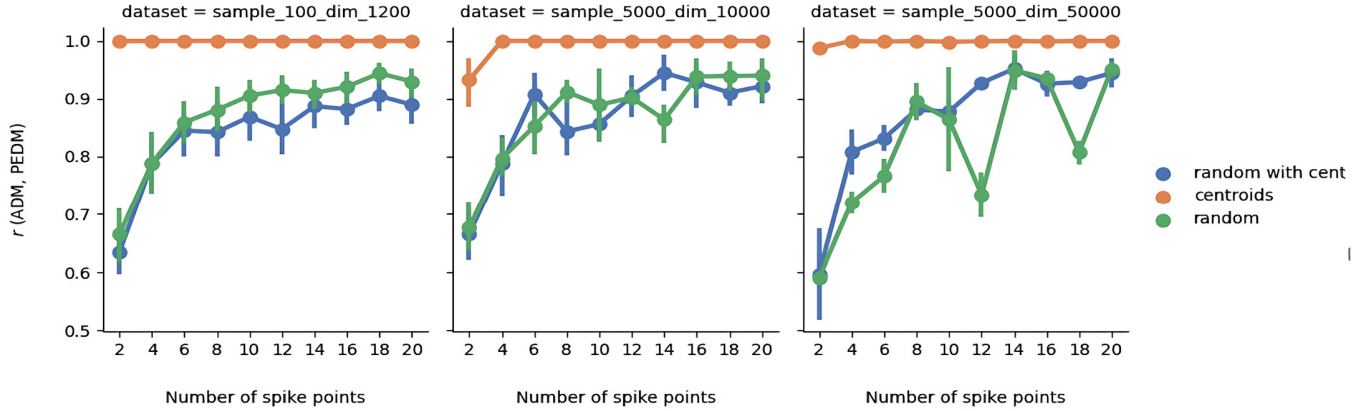

Fig. 2. Pearson coefficient of the PEDM with respect to ADM for datasets with non-uniform distribution.

Figure 3 illustrates the Spearman correlation between ADM and FEDM across three datasets, varying in sample size and dimensions, with an increasing number of spike points. Similarly, Figure 4 depicts the correlation between ADM and PEDM. In Figure 3, for the first two datasets, and with an increasing number of spike points, FEDM constructed from centroids as spike points generally demonstrates higher average Spearman correlation compared to FEDM created from random

points alone or a combination of random points and centroids. However, when the sample size reaches 5,000 and the number of dimensions increases to 50,000, the correlation trend becomes unstable, even with an increase in the number of spike points used to create FEDM with respect to ADM. In Figure 4, across all datasets, PEDM exhibit reasonably good reconstruction of distance matrices, indicated by highly positive Spearman rank correlation coefficient values.

Table II presents the average Pearson and Spearman correlation coefficient values of both PEDM and FEDM with respect to ADM, independent of the spike generation method. Except for the dataset featuring 5,000 samples and 50,000 dimensions, 6-14 spike points generally suffice to create FEDM and PEDM with very high positive Pearson correlation relative to ADM. However, the Spearman correlation coefficient values are lower than the corresponding Pearson values, regardless of the spike point generation methods, sample size, and dataset dimensions.

TABLE II. AVERAGE PEARSON AND SPEARMAN CORRELATION COEFFICIENTS OF FEDM AND PEDM FOR NON-UNIFORMLY DISTRIBUTED DATASETS

Samples	Dimension	Spike points	Pearson Correlation Coefficient		Spearman Correlation Coefficient	
			ADM w.r.t PEDM	ADM w.r.t FEDM	ADM w.r.t PEDM	ADM w.r.t FEDM
100	1,200	2	0.77	0.76	0.58	0.59
100	1,200	6	0.9	0.89	0.7	0.63
100	1,200	14	0.93	0.92	0.64	0.57
100	1,200	20	0.94	0.92	0.68	0.62
5,000	10,000	2	0.76	0.77	0.66	0.64
5,000	10,000	6	0.92	0.92	0.65	0.59
5,000	10,000	14	0.94	0.93	0.7	0.49
5,000	10,000	20	0.95	0.95	0.73	0.54
5,000	50,000	2	0.75	0.78	0.68	0.69
5,000	50,000	6	0.86	0.84	0.71	0.61
5,000	50,000	12	0.89	0.88	0.74	0.67
5,000	50,000	20	0.97	0.96	0.72	0.32

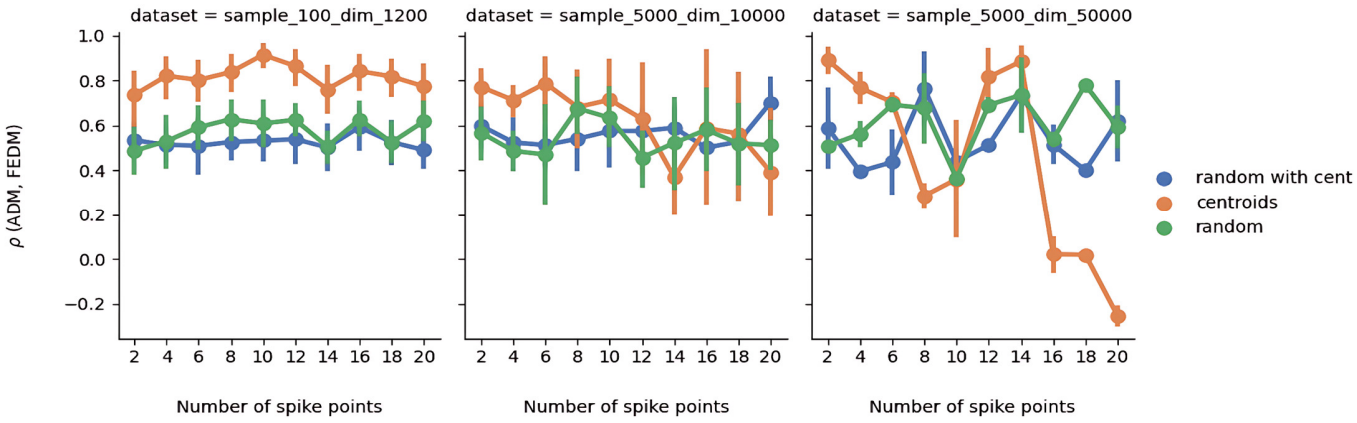

Fig. 3. Spearman coefficient of the FEDM with respect to ADM for datasets with non-uniform distribution.

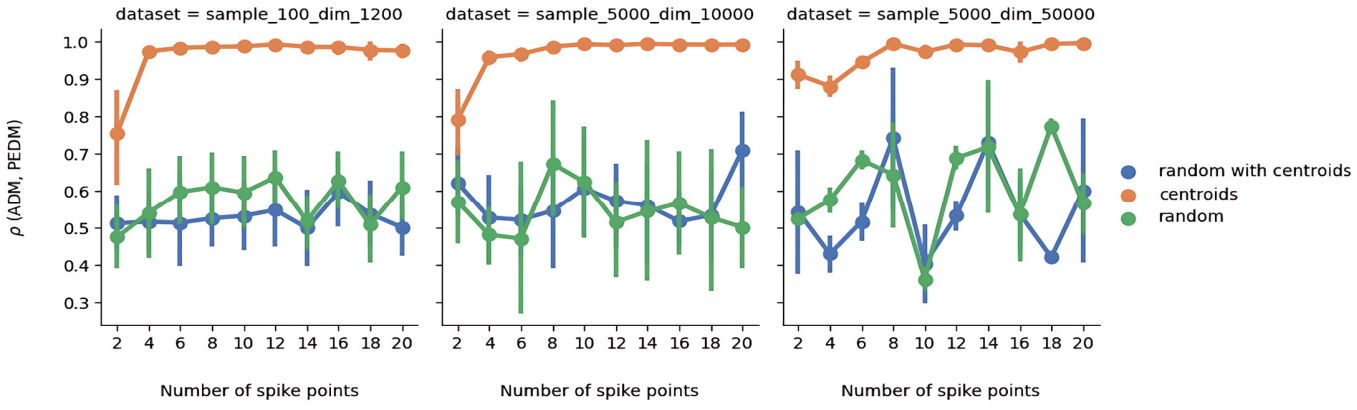

Fig. 4. Spearman coefficient of the FEDM with respect to ADM for datasets with non-uniform distribution.

2) Uniformly Distributed

Figure 5 presents the Pearson correlation between ADM and FEDM across four datasets varying in sample size and dimensions, with an increasing number of spike points. Figure 6 showcases the same correlation, but between ADM and PEDM. Due to the uniform distribution of random data points, it is evident from Figure 5 that random spike points outperform other spike point generation methods in constructing a more correlated distance matrix with respect to ADM. When the dimensionality of the data points is lower, FEDM exhibit less error in constructing a matrix close to ADM, and the Pearson correlation gradually increases with the number of spike points. Conversely, FEDM created via alternative spike point

generation methods exhibit significantly lower Pearson correlation coefficient values. In Figure 6, we observe the potential of PEDM and regression. After regression, PEDM achieves approximately 85–90% correlation with ADM, making it applicable for operational use cases related to Euclidean distance matrices in the FL environment. PEDM generated from FEDM exhibit similar Pearson correlation coefficient values regardless of the spike point generation method. Notably, both FEDM and PEDM created based on datasets with lower dimensionality demonstrate almost identical correlation with ADM.

Figure 7 illustrates the Spearman correlation between ADM and FEDM across four datasets, varying in sample size and dimensions, with an increasing number of spike points. Similarly, Figure 8 depicts the same correlation, but between ADM and PEDM. However, both Figures 7 and 8 indicate that Spearman correlation remains relatively consistent for both FEDM and PEDM with respect to spike point generation methods. Except for the dataset with samples of lower dimensionality (i.e., 100), all other datasets exhibit very low Spearman correlation coefficients (ranging from 0.0 to 0.3) due to significant differences in value and rank along each dimension of the data points, which further increases with the number of dimensions.

From Table III, we gain an overall understanding of how the number of spike points, regardless of the method used to generate them, impacts the construction of more correlated Euclidean distance matrices similar to ADM. With an increase in the number of spike points, both Pearson and Spearman correlations tend to rise to a certain extent.

TABLE III. AVERAGE PEARSON AND SPEARMAN CORRELATION COEFFICIENTS OF FEDM AND PEDM FOR UNIFORMLY DISTRIBUTED DATASETS

Samples	Dimension	Spike points	Pearson Correlation Coefficient		Spearman Correlation Coefficient	
			ADM w.r.t PEDM	ADM w.r.t FEDM	ADM w.r.t PEDM	ADM w.r.t FEDM
1,200	1,000	10	0.83	0.07	0.05	0.05
1,200	1,000	40	0.82	0.1	0.07	0.08
1,200	1,000	70	0.82	0.13	0.1	0.1
1,200	1,000	100	0.81	0.13	0.11	0.11
5,000	100	10	0.28	0.15	0.14	0.14
5,000	100	40	0.36	0.25	0.27	0.27
5,000	100	70	0.38	0.3	0.3	0.31
5,000	100	100	0.41	0.33	0.35	0.35
5,000	10,000	10	0.92	0.03	0.02	0.02
5,000	10,000	40	0.91	0.05	0.03	0.03
5,000	10,000	70	0.91	0.05	0.03	0.04
5,000	10,000	100	0.89	0.05	0.04	0.15
5,000	20,000	10	0.96	0.03	0.01	0.01
5,000	20,000	40	0.95	0.03	0.01	0.01
5,000	20,000	70	0.95	0.04	0.02	0.03
5,000	20,000	100	0.93	0.03	0.02	0.02

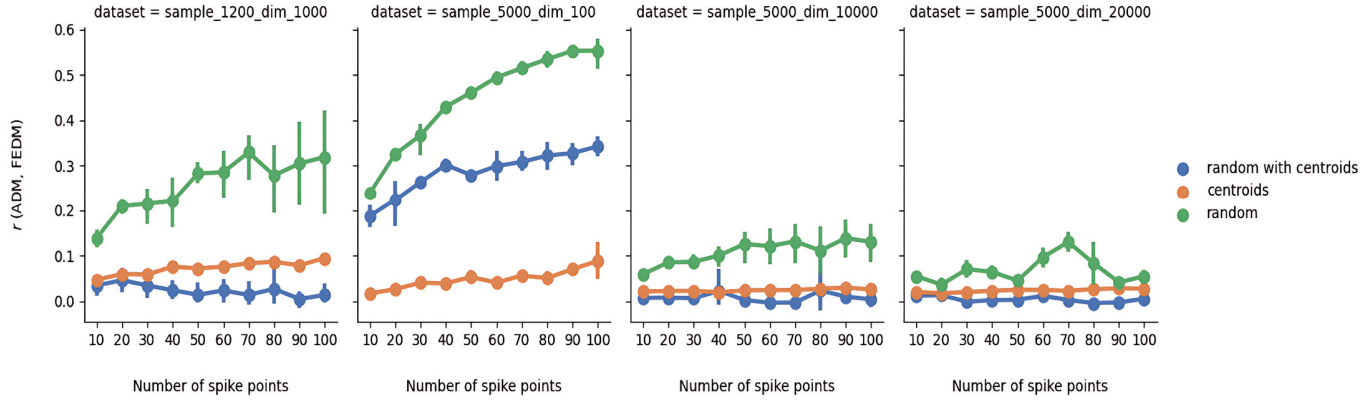

Fig. 5. Pearson coefficient of the FEDM with respect to ADM for datasets with uniform distribution.

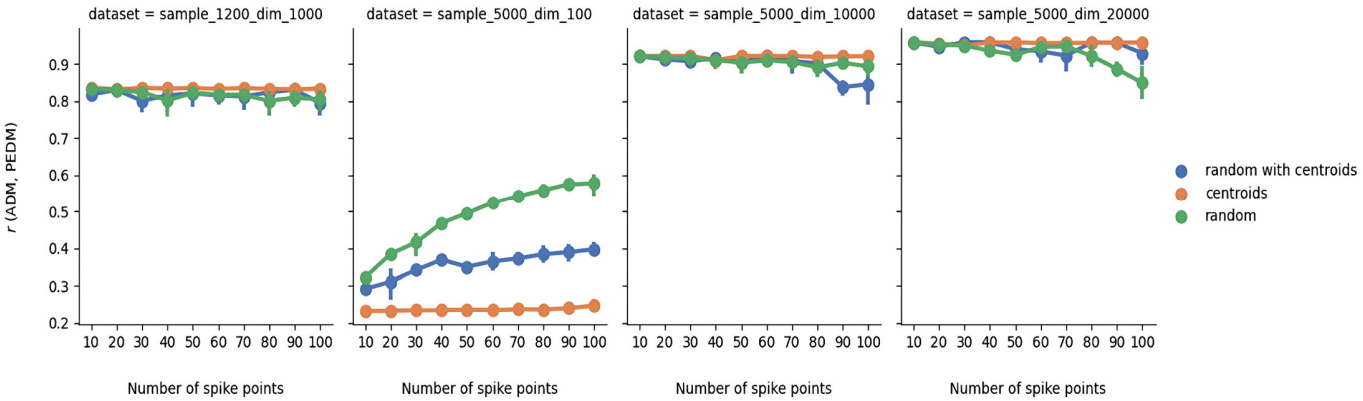

Fig. 6. Pearson coefficient of the PEDM with respect to ADM for datasets with uniform distribution.

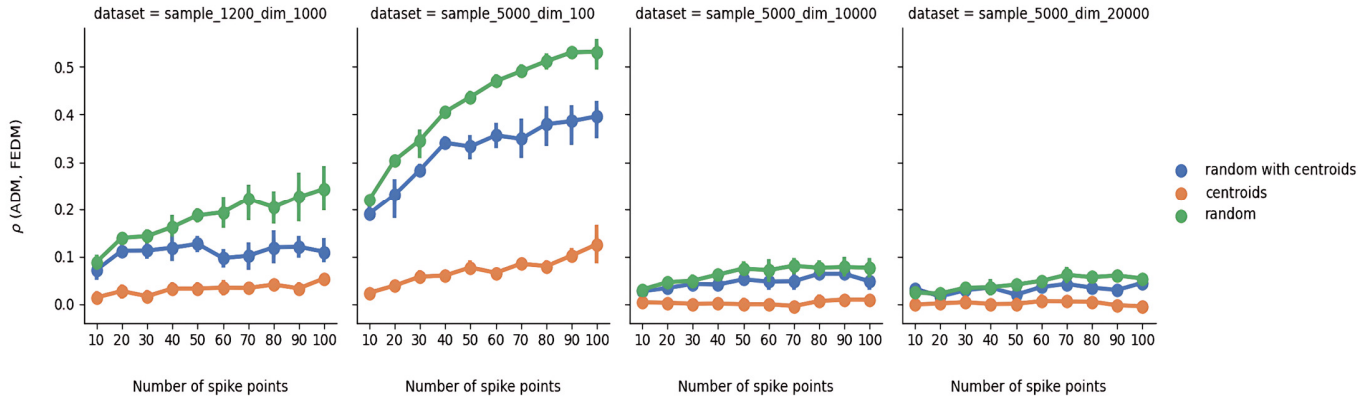

Fig. 7. Spearman coefficient of the FEDM with respect to ADM for datasets with uniform distribution.

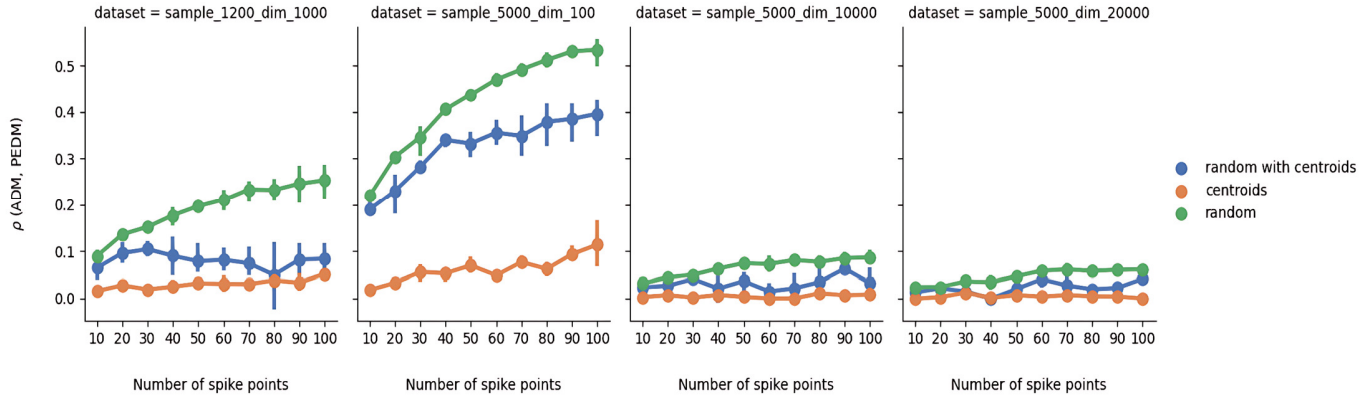

Fig. 8. Spearman coefficient of the PEDM with respect to ADM for datasets with uniform distribution.

B. Gene Expression Datasets

1) GSE84426

Figure 9 illustrates the Pearson correlation between ADM and FEDM, as well as between ADM and PEDM, for the GSE84426 dataset [19], with an increasing number of spike points. Figure 10 presents the Spearman correlation instead. From Figure 9, it is evident that spike points generated through centroid-based and random methods lead to higher Pearson correlation between ADM and FEDM as the number of spike points increases. However, the inclusion of both random points and centroids results in an FEDM with lower Pearson correlation for every given number of spike points. This occurs because many random points tend to appear close to centroids, reducing the information gained by LSDMs and consequently leading to less accurate FEDM. However, with PEDM, which corrects FEDM using the slope and intercept obtained from regression performed between ADM and FEDM, we observe consistent Pearson correlation regardless of the spike point generation method used for FEDM. Specifically, when FEDM is created with centroid-based spike points, PEDM exhibit better Pearson correlation compared to random points or centroids, especially when the number of spike points is equal to or greater than 28. This is because randomly generated points may reduce information gain due to their random placement. In Figure 10, it is apparent that, for every number of spike points, the FEDM created by both random points and centroids, along with the corresponding PEDM, exhibit significantly lower Spearman correlation with respect to ADM. Despite the similar correlation between FEDM and PEDM resulting from random spike points and centroids for different numbers of spike points, the centroid-based method generally yields slightly better correlation. This is because centroid-based spike points consider average distances between data points, whereas random spike point generation may lead to multiple points in close proximity, reducing information gain due to random placement.

Figure 11 demonstrates that the relationship between FEDM, PEDM, and ADM is more linear, as the correlation of both matrices decreases monotonically with respect to ADM. Moreover, PEDM exhibit higher linear correlation than FEDM, even though FEDM can achieve similar Pearson or Spearman correlation with ADM.

2) GSE84433

Figure 12 depicts the Pearson correlation between ADM and FEDM, as well as between ADM and PEDM, for the GSE84433 dataset [20], with a varying number of spike points. Figure 13 presents the Spearman correlation instead. Observing Figure 12, it becomes evident that for each number of spike points, FEDM generated by all spike generation methods exhibit progressively increasing Pearson correlation coefficients. However, FEDM derived from uniformly distributed random spike points demonstrates higher linear correlation with ADM for each number of spike points due to the uniform scattering of a high number of data points.

Notably, the highest Pearson correlation is observed in PEDM produced by FEDM resulting from a combination of centroids and random points. Given a significant number of data points falling within the range of -10,000 and 0, the difference in coefficient values between PEDM originated from FEDM created solely from random points is negligible.

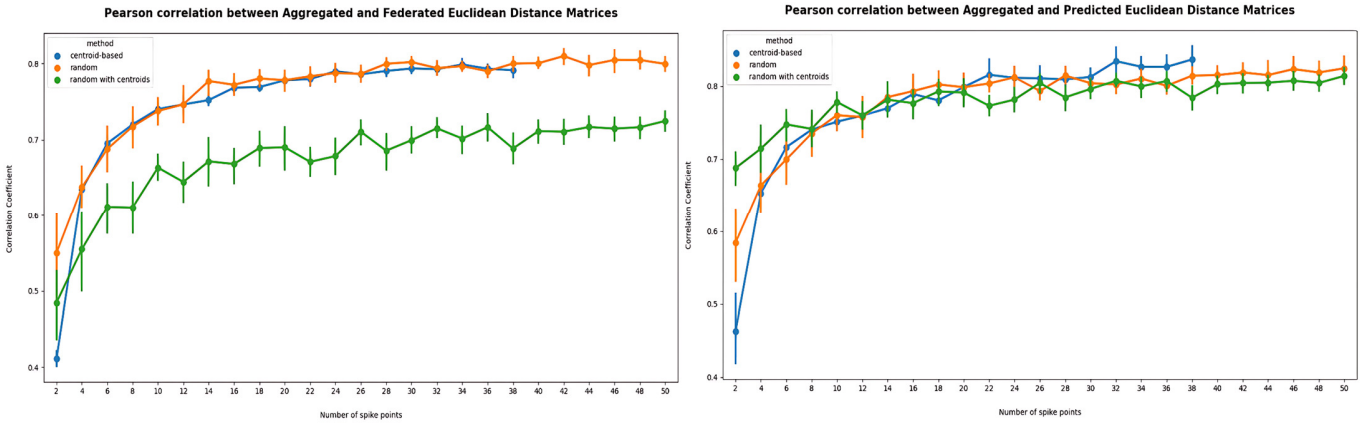

Fig. 9. Pearson correlation coefficient between FEDM and PEDM with respect to ADM for the GSE84426 dataset.

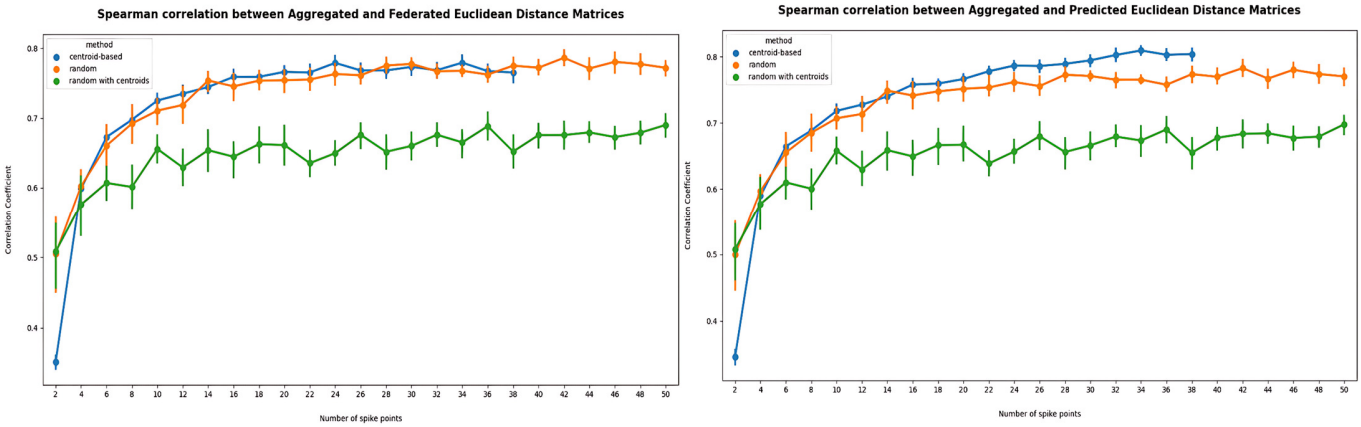

Fig. 10. Spearman correlation coefficient between FEDM and PEDM with respect to ADM for the GSE84426 dataset.

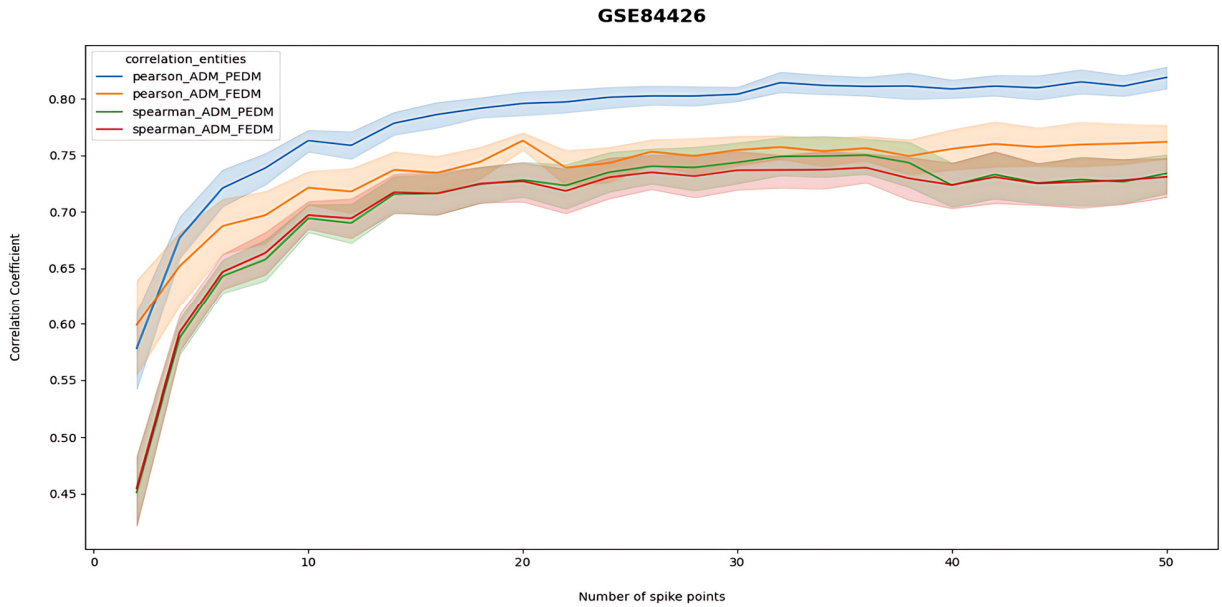

Fig. 11. Pearson and Spearman correlation coefficients of FEDM and PEDM with respect to ADM for the GSE84426 dataset (up to 50 spike points).

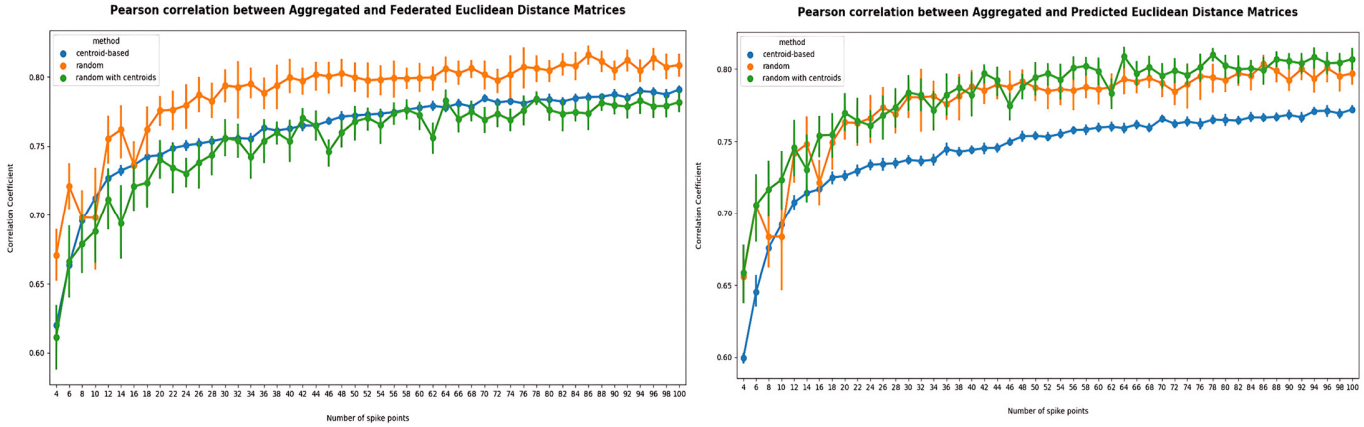

Fig. 12. Pearson coefficient between FEDM and PEDM with respect to ADM for the GSE84433 dataset.

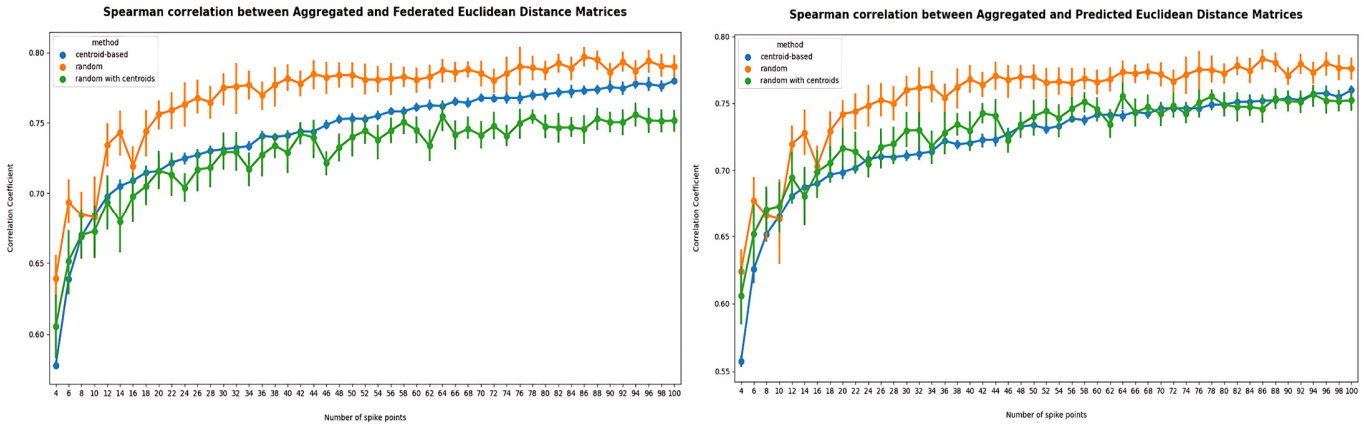

Fig. 13. Spearman coefficient between FEDM and PEDM with respect to ADM for the GSE84433 dataset.

However, in Figure 13, we observe that FEDM created with pairwise distances between random spike points and data points exhibit higher and better monotonic correlation with ADM, whereas FEDM generated by the other two spike generation methods have lower correlation. Although centroid based FEDM show slightly better correlation with ADM, FEDM created by spike points that are either centroids or random points with centroids have very close correlation coefficient values with ADM, respectively. We also observe an increase in Spearman correlation regardless of the method used for spike point generation as the number of spike points increases. The same pattern and results are observed for PEDM, calculated after performing regression by each participant and applying the respective coefficients and intercepts for each FEDM value.

In Figure 14, it is notable that the average Pearson correlation of FEDM and PEDM with respect to ADM increases with the number of spike points, with FEDM exhibiting slightly better correlation coefficient values. This suggests that the gap in values, along with differences in dataset dimensionality, is smaller in FEDM, and a high number of spike points can generate distance matrices very close to ADM. Despite the decreased effectiveness of Euclidean distance with increasing dimensionality, it performs well as many datasets exhibit similarities with a non-central t-distribution, for which the Euclidean distance function can be highly effective for higher dimensions (10,000 and above).

While the primary goal of this study is to preserve privacy in distance computations, the impact of the proposed FEDM and PEDM techniques on the overall accuracy of ML models in federated contexts was also taken into consideration. This demonstrates that the proposed approach strikes a balance between data privacy and model quality, making it suitable for actual FL applications where pairwise similarity or distance is important in model updates, grouping, or personalization.

VIII. DISCUSSION

The FEDM and PEDM for this type of dataset could exhibit very high positive correlation with respect to ADM regardless of the number of spike points if and only if the spike points are generated from centroids. For a dataset that has a skewed distribution and a high number of outliers, spike points generated using centroids and random points together create a better-correlated FEDM than spike points consisting of just centroids. When the FEDM is created from an array of spike points where some are centroids and others are random points, it tends to have low to moderate positive correlation with ADM, considering both Pearson and Spearman correlation. For a dataset that portrays more visible clusters or have multiple high-density regions formed by the values of the data points,

centroids used as spike points will provide more strongly correlated FEDM and PEDM. To maintain the mathematical validity and structural integrity of the generated distance matrices, our method ensures that the number of spike points never exceeds the data's dimensionality.

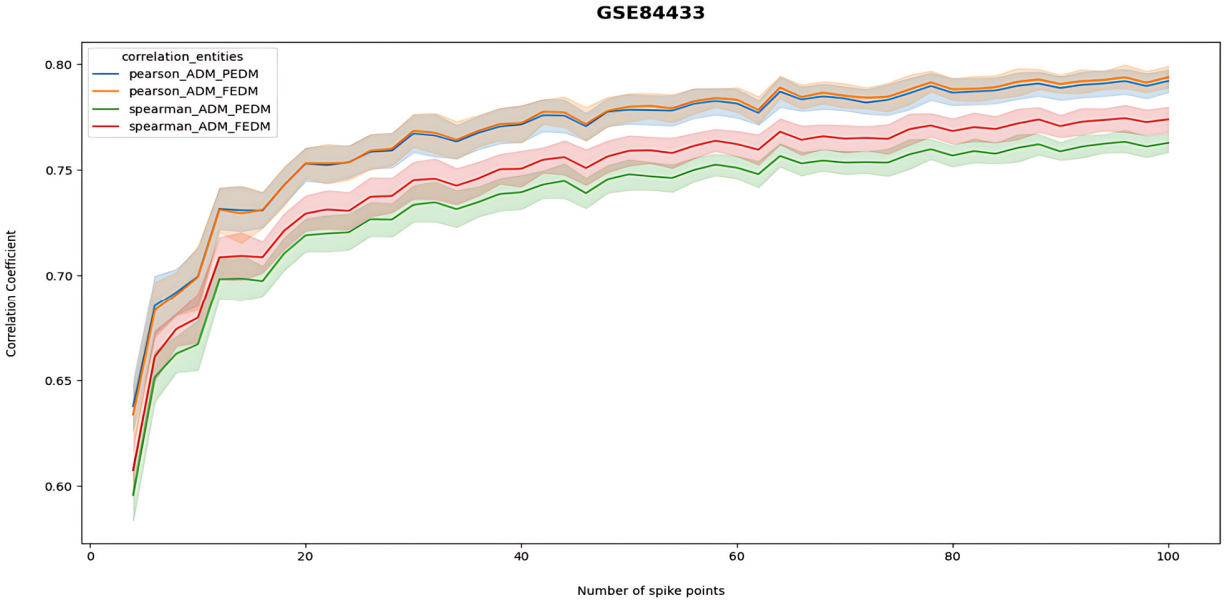

Fig. 14. Pearson and Spearman coefficients of FEDM and PEDM with respect to ADM for the GSE84433 dataset (up to 50 spike points).

A. Limitations

The purpose of using both centroid-based and random spike points is to balance representativeness, robustness, and privacy across varied data distributions. Centroid-based spike points, produced using clustering methods such as K-means, are useful when participant data contain well-defined clusters or Gaussian-like blobs because they capture core tendencies within dense portions of the feature space. To address this, we use randomly generated spike points to provide broader and more uniform coverage of the feature space, improving the generalizability of the LSDMs and lowering the risk of privacy leakage or overfitting. When a participant has a dataset with many outliers and performs K-means or other clustering algorithms to create spike points based on centroids or a combination of random points and centroids, there is a chance that an outlier itself or very similar data points may be considered as a centroid and returned as a spike point for inclusion.

IX. CONCLUSION

This study proposed two types of Euclidean distance matrices, derived from spike points and regression: the Federated Euclidean Distance Matrix (FEDM) and the Predicted Euclidean Distance Matrix (PEDM), which utilize the pairwise distances between data points and spike points locally for each participant. The proposed approach successfully constructs distance matrices exhibiting approximately 80% to 99% similarity with the Aggregated Distance Matrix (ADM), which contains the pairwise distances of all data points from all participants, through the application of regression and the use of spike points generated via various methodologies. In summary, the objective was to approximate and compute the aggregated Euclidean distance matrix for data points from one or multiple datasets in a Federated Learning (FL) system without sharing the actual data values, when the samples are horizontally partitioned and distributed across different participants.

ACKNOWLEDGMENTS

This paper is based on a research grant funded by the Research, Development, and Innovation Authority (RDIA), Kingdom of Saudi Arabia, with grant number 13382-PSU-2023-PSNU-R-3-1-EI. The authors would like to acknowledge the support of Prince Sultan University, Riyadh, Saudi Arabia, for covering the article processing charges of this publication. This research is supported by the Automated Systems and Computing Lab (ASCL), Prince Sultan University, Riyadh, Saudi Arabia. The authors would like to thank the principals, teachers, and students of special schools in the districts of Ernakulam, Palakkad, Thrissur, and Kannur, India, for their invaluable support in facilitating data collection.

REFERENCES

[1] N. Bouacida and P. Mohapatra, "Vulnerabilities in Federated Learning," *IEEE Access*, vol. 9, pp. 63229–63249, 2021, https://doi.org/10.1109/ACCESS.2021.3075203.

[2] U. Hameed, M. U. Rehman, A. Rehman, R. Damaševičius, A. Sattar, and T. Saba, "A deep learning approach for liver cancer detection in CT scans," *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, vol. 11, no. 7, Jan. 2024, Art. no. 2280558, https://doi.org/10.1080/21681163.2023.2280558.

[3] H. Inbarani H., A. T. Azar, and J. G, "Leukemia Image Segmentation Using a Hybrid Histogram-Based Soft Covering Rough K-Means Clustering Algorithm," *Electronics*, vol. 9, no. 1, Jan. 2020, Art. no. 188, https://doi.org/10.3390/electronics9010188.

[4] S. Al-Otaibi, M. Mujahid, A. R. Khan, H. Nobanee, J. Alyami, and T. Saba, "Dual Attention Convolutional AutoEncoder for Diagnosis of Alzheimer's Disorder in Patients Using Neuroimaging and MRI Features," *IEEE Access*, vol. 12, pp. 58722–58739, 2024, https://doi.org/10.1109/ACCESS.2024.3390186.

[5] C. Kaushal, K. Kaushal, and A. Singla, "Firefly optimization-based segmentation technique to analyse medical images of breast cancer," *International Journal of Computer Mathematics*, vol. 98, no. 7, pp. 1293–1308, Jul. 2021, https://doi.org/10.1080/00207160.2020.1817411.

[6] S. R. Waheed, N. M. Suaib, M. S. M. Rahim, A. R. Khan, S. A. Bahaj, and T. Saba, "Synergistic Integration of Transfer Learning and Deep Learning for Enhanced Object Detection in Digital Images," *IEEE Access*, vol. 12, pp. 13525–13536, 2024, https://doi.org/10.1109/ACCESS.2024.3354706.

[7] A. Koubaa, A. Ammar, M. Alahdab, A. Kanhouch, and A. T. Azar, "DeepBrain: Experimental Evaluation of Cloud-Based Computation Offloading and Edge Computing in the Internet-of-Drones for Deep Learning Applications," *Sensors*, vol. 20, no. 18, Sep. 2020, Art. no. 5240, https://doi.org/10.3390/s20185240.

[8] A. Jha, M. Dave, and S. Madan, "Performance Evaluation of Binary and Multi-Class Dataset using Ensemble Classifiers," *International Journal of Engineering Research & Technology*, vol. 11, no. 3, pp. 425–430, Apr. 2022, https://doi.org/10.17577/IJERTV11IS030164.

[9] K. Hicham, S. Laghmati, B. Cherradi, S. Hamida, and A. Tmiri, "Enhancing Colorectal Polyps Detection using Transfer Learning on DICOM Metadata," *Engineering, Technology & Applied Science Research*, vol. 15, no. 1, pp. 19417–19423, Feb. 2025, https://doi.org/10.48084/etasr.9024.

[10] V. Kulkarni *et al.*, "Air Quality Decentralized Forecasting: Integrating IoT and Federated Learning for Enhanced Urban Environmental Monitoring," *Engineering, Technology & Applied Science Research*, vol. 14, no. 4, pp. 16077–16082, Aug. 2024, https://doi.org/10.48084/etasr.7869.

[11] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated Machine Learning: Concept and Applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, Jan. 2019, Art. no. 12, https://doi.org/10.1145/3298981.

[12] W.-J. Lee, R. P. W. Duin, A. Ibba, and M. Loog, "An experimental study on combining Euclidean distances," in *2010 2nd International Workshop on Cognitive Information Processing*, Elba, Italy, 2010, pp. 304–309, https://doi.org/10.1109/CIP.2010.5604238.

[13] S. Mukherjee, Z. Chen, and A. Gangopadhyay, "A privacy-preserving technique for Euclidean distance-based mining algorithms using Fourier-related transforms," *The VLDB Journal*, vol. 15, no. 4, pp. 293–315, Nov. 2006, https://doi.org/10.1007/s00778-006-0010-5.

[14] P. Kairouz *et al.*, "Advances and Open Problems in Federated Learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, Jun. 2021, https://doi.org/10.1561/2200000083.

[15] "sklearn.datasets.make_blobs." scikit-learn. [Online]. Available: https://scikit-learn/stable/modules/generated/sklearn.datasets.make_blobs.html.

[16] "numpy.random.random — NumPy v1.15 Manual." SciPy. https://docs.scipy.org/doc/numpy-1.15.0/reference/generated/numpy.random.random.html.

[17] S.-J. Yoon *et al.*, "Deconvolution of diffuse gastric cancer and the suppression of CD34 on the BALB/c nude mice model," *BMC Cancer*, vol. 20, no. 1, Apr. 2020, Art. no. 314, https://doi.org/10.1186/s12885-020-06814-4.

[18] J. Matschinske *et al.*, "The FeatureCloud AI Store for Federated Learning in Biomedicine and Beyond." arXiv, May 12, 2021, https://doi.org/10.48550/arXiv.2105.05734.

[19] "Molecular subtypes in gastric cancer. [I]." NCBI, 2018. [Online]. Available: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE84426.

[20] "Molecular subtypes in gastric cancer. [II]." NCBI, 2018. [Online]. Available: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE84433.