Journal of Artificial Intelligence and Emerging Technologies
Volume 3, Issue 8, pp 41-52, August-2026
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2026.308005
Available Online at https://www.jaiet.com

Improved Privacy-Preserving Clustering Model for Distributed Data

[1]O. M. Omowaye, *[2]V. I. E. Anireh, *[3]N. D. Nwiabu

[1,2,3]Department of Computer Science, Rivers State University, Port-Harcourt, Nigeria
E-mail: [1]omowaye.oluwabukola@ust.edu.ng, [2]anireh.ike@ust.edu.ng, [3]nwiabu.nuka@ust.edu.ng

Abstract: Privacy-preserving distributed clustering lets organisations analyse data held in separate locations without centralising raw records. Homomorphic encryption can protect centroid exchange, but distributed k-means becomes slow and communication-intensive when every centroid feature is encrypted and transmitted separately. This article presents a CKKS-based distributed k-means method that represents each centroid as a packed ciphertext. Under a semi-honest coordinator assumption, participant records remain local, centroid updates are encrypted before transmission, and aggregation is performed on ciphertexts. The method was evaluated on Iris, Wine, Diabetes, and Credit Card Default datasets under simulated 3-participant and 5-participant settings. Three configurations were compared: centralised plaintext k-means, a per-dimension CKKS encrypted centroid baseline, and the packed CKKS method. Clustering quality was measured using silhouette score and Davies-Bouldin index, while efficiency was measured using encryption time, total execution time, and encrypted communication volume. Compared with the per-dimension encrypted baseline, the packed method reduced encryption time by 57.47% to 95.27% and encrypted communication volume by 75.00% to 95.65%. In the Credit Card Default 3-participant setting, total runtime fell from 24.08 seconds to 1.12 seconds, and encrypted communication fell from 450,504.18 KB to 19,587.03 KB. The silhouette-score difference relative to the encrypted baseline was 0.0000 in all reported encrypted comparisons, indicating that centroid packing reduced operational cost without adding clustering-quality loss.

Keywords: Privacy-preserving clustering; distributed clustering; CKKS homomorphic encryption; ciphertext packing; k-means clustering; encrypted aggregation.

I. INTRODUCTION

It Privacy-preserving distributed clustering is useful when organisations need to analyse data held in separate locations but cannot pool the underlying records. This situation is common in healthcare, finance, and public-sector analytics, where privacy rules, institutional boundaries, or bandwidth constraints can prevent direct data sharing [1], [2]. Distributed learning reduces raw-data transfer, but clustering still depends on repeated exchange of intermediate model information, especially centroids in k-means workflows [3], [4]. Without protection, centroid updates may reveal information about local data distributions.

Homomorphic encryption protects these exchanges by supporting computation on encrypted numerical values. The Cheon-Kim-Kim-Song (CKKS) scheme is relevant here because it supports approximate arithmetic on real-valued vectors, which matches centroid aggregation [5]. The cost is computational overhead. In distributed k-means, each iteration involves local assignment, local centroid computation, encrypted centroid exchange, and encrypted aggregation. If centroid features are encrypted and transmitted separately, the number of ciphertexts grows with feature dimensionality.

Recent CKKS optimisation studies have used segmentation, packing, and parallel processing to reduce encryption overhead in federated learning [6], [7], [8]. Most of this work targets supervised models or general encrypted parameter exchange, not centroid-based clustering. The gap is practical: distributed clustering needs an encrypted representation that fits the repeated vector structure of k-means centroids. Centroid vectors are well suited to ciphertext packing because multiple feature values can be encoded in one encrypted vector and processed through Single Instruction Multiple Data (SIMD)-style operations.

This article introduces a packed CKKS representation for encrypted centroid aggregation in distributed k-means clustering. Rather than encrypting each centroid feature separately, the method packs all feature values of a centroid into one CKKS ciphertext. This reduces encryption operations and ciphertext

© 2026 The Author(s). CC BY 4.0 License

Journal of Artificial Intelligence and Emerging Technologies
Volume 3, Issue 8, pp 41-52, August-2026
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2026.308005
Available Online at https://www.jaiet.com

payloads while keeping the clustering objective unchanged. Privacy is assessed through the implemented workflow: participant records remain local, local centroid updates are encrypted before transmission, and aggregation is performed on ciphertexts under a semi-honest coordinator assumption.

The method is implemented using Python, TenSEAL, scikit-learn, and NumPy, and evaluated on four numerical benchmark datasets: Iris, Wine, Diabetes, and Credit Card Default. Experiments use simulated 3-participant and 5-participant distributed settings. Clustering quality is assessed using silhouette score and Davies-Bouldin index, while computational and communication efficiency are measured using encryption time, total execution time, and encrypted communication volume. The central comparison is between the packed CKKS method and a per-dimension encrypted CKKS baseline, with centralised plaintext k-means included as a non-encrypted reference point.

II. RELATED WORK

Homomorphic encryption permits arithmetic on encrypted data and returns encrypted results that correspond to the same plaintext computation after decryption [9]. CKKS is a natural fit for this study because it handles approximate arithmetic on real-valued vectors [5]. Since k-means centroids are continuous-valued vectors, encrypted centroid aggregation can be expressed directly within the CKKS arithmetic model.

Federated learning offers the basic pattern used here: local computation followed by global aggregation, without transferring raw data [10]. In distributed k-means, each participant keeps its records locally, assigns them to the nearest global centroids, computes local centroid updates, and sends those updates for aggregation. The privacy concern is that centroid updates may still reveal local distributional information. CKKS reduces this exposure by allowing the updates to be encrypted before transmission and aggregated while still encrypted.

Ciphertext packing is the efficiency mechanism used in this article. It places multiple values inside one encrypted representation, allowing SIMD-style processing [11]. For k-means, the repeated centroid-vector structure makes this especially useful: centroid features can be packed together instead of being encrypted one feature at a time.

Other privacy-preserving approaches include differential privacy and secure multi-party computation. Differential privacy adds calibrated noise to query results or model updates so that individual records have bounded influence on released outputs [12]. In clustering, however, centroids are aggregate statistics over clusters of varying size, so privacy noise must be calibrated carefully. Secure multi-party computation can also protect distributed computation through protocols such as garbled circuits, oblivious transfer, and secret sharing [13], but it usually requires more interaction among participants. Homomorphic encryption is therefore attractive for the present setting because it supports non-interactive encrypted centroid exchange and aggregation, although at higher computational cost.

Distributed data settings add practical constraints. Participants may hold disjoint data partitions that cannot be centralised because of regulation, institutional boundaries, or bandwidth limitations [10]. In k-means, this matters because each iteration requires centroid transmission from every participant. Homomorphic encryption overhead then accumulates with both feature dimensionality and iteration count, making efficiency optimisation necessary for deployment [6].

Federated clustering adapts conventional clustering algorithms to distributed environments where raw data cannot be centralised. K-means clustering, introduced by MacQueen [14], alternates between cluster assignment based on nearest centroid and centroid recomputation as cluster means. Federated k-means variants preserve this general structure but require local computation and global aggregation rather than direct access to all records. Garst and Reinders [15] examined federated k-means clustering and showed the importance of aggregation strategy, convergence, and cluster-quality preservation in distributed settings. Luo *et al.* [16] investigated privacy-preserving clustering federated learning for non-IID data and highlighted the effect of distributed data heterogeneity on clustering performance. These studies support the relevance of distributed clustering but also show the need for efficient aggregation mechanisms that can preserve privacy without making encrypted clustering impractical.

Evaluation must consider both cluster quality and operational overhead. Silhouette score measures how well samples fit their assigned cluster relative to the nearest alternative cluster, while the Davies-Bouldin index captures within-cluster scatter relative to between-cluster separation [17], [18]. For privacy-preserving clustering, these metrics are not enough on their own. Encryption time, aggregation time, and transmitted ciphertext volume also determine whether the method is practical.

© 2026 The Author(s). CC BY 4.0 License

Journal of Artificial Intelligence and Emerging Technologies
Volume 3, Issue 8, pp 41-52, August-2026
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2026.308005
Available Online at https://www.jaiet.com

Recent homomorphic encryption optimisation studies show that CKKS can be made more efficient through segmentation, packing, and parallel processing. Pan et al. [6] developed FedSHE, an adaptive segmented CKKS homomorphic encryption framework, and demonstrated 35-42% encryption overhead reduction through parameter vector segmentation. Zuo *et al.* [7] proposed Pack, a communication-efficient homomorphic encryption framework for federated learning, achieving 45-68% communication reduction through packed ciphertext representations. F. Chen *et al.* [8] explored high-throughput CKKS implementation using parallelism and reported large performance gains in cryptographic operations. These studies show the value of CKKS optimisation, but their emphasis is mainly on general federated learning, deep learning model parameters, or hardware acceleration rather than clustering-specific centroid representation.

The main gap is the handling of encrypted centroids in distributed clustering. Existing CKKS optimisation studies focus largely on supervised federated learning, general encrypted parameter exchange, or cryptographic architecture. Pan *et al.* [6] use adaptive segmentation for supervised federated learning, Kumar *et al.* [11] analyse configurable CKKS architectures, and Zuo *et al. [*7] address communication efficiency through packing for model parameters. These studies do not directly exploit the repeated centroid-vector structure of k-means. This article therefore evaluates centroid-level CKKS packing against a per-dimension encrypted baseline.

III. METHOD

The packed CKKS method takes local participant partitions as input and coordinates distributed k-means through encrypted centroid aggregation. Figure 1 shows the main stages: local data partitions, packed CKKS encryption, homomorphic aggregation, and final cluster assignment. The method focuses on how centroid information is represented and aggregated without exposing participant records.

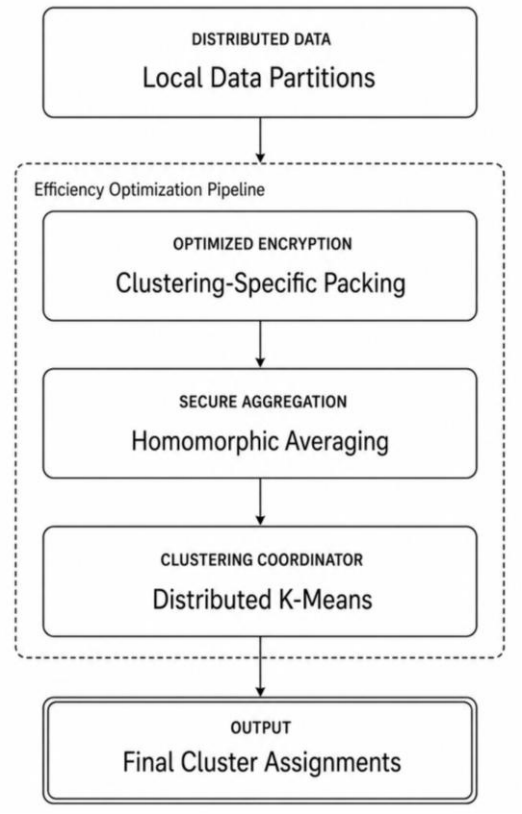

Figure 1: Packed CKKS Clustering Architecture

Each participant keeps a local data partition, assigns records to the current global centroids, and computes local centroid updates. Instead of encrypting each centroid dimension independently, the packed variant encodes the feature values of a centroid into

© 2026 The Author(s). CC BY 4.0 License

Journal of Artificial Intelligence and Emerging Technologies
Volume 3, Issue 8, pp 41-52, August-2026
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2026.308005
Available Online at https://www.jaiet.com

one CKKS vector. The coordinator then performs weighted aggregation on encrypted centroid updates and uses the decrypted aggregate centroids for the next iteration.

The contribution is the use of clustering-specific packing inside the encrypted aggregation step of iterative distributed k-means.

Workflow

Figure 2 provides the workflow for the packed CKKS distributed k-means process, showing how benchmark datasets move through preprocessing, participant simulation, centroid initialisation, local k-means computation, packed centroid encryption, homomorphic weighted aggregation, convergence checking, final cluster assignment, evaluation, and reporting.

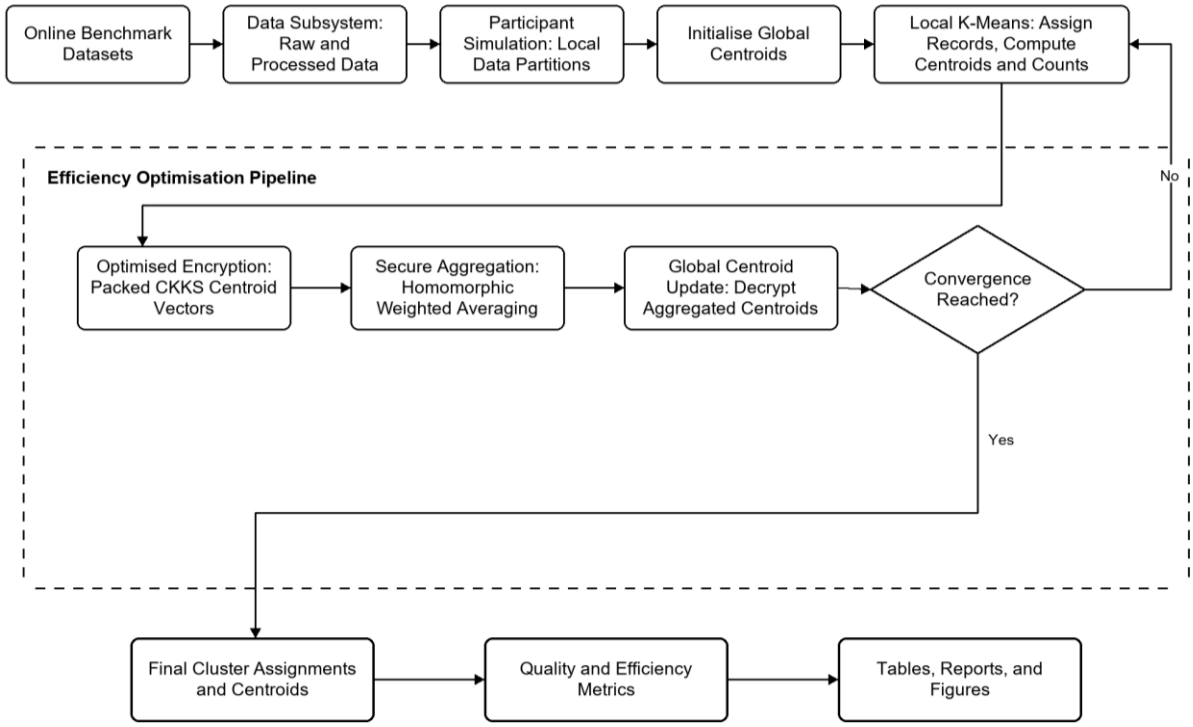

Figure 2: Workflow of the Packed CKKS Distributed K-Means Method

The workflow begins with benchmark datasets that are transformed into standardised numerical arrays. The processed arrays are partitioned into simulated participant datasets, after which global centroids are initialised and local records are assigned to their nearest centroids. Each participant computes local centroids and cluster counts, encrypts centroid vectors using the packed CKKS representation, and sends the encrypted updates for homomorphic weighted aggregation. The aggregated centroid is decrypted for the next clustering round, and the process repeats until convergence is reached or the maximum iteration limit is met. After termination, the method produces final cluster assignments and centroids, then computes the quality and efficiency metrics used for empirical evaluation.

Encrypted Aggregation

The encrypted aggregation step is the main comparison point because it implements both the per-dimension encrypted representation and the packed representation. In the per-dimension encrypted baseline, each centroid feature is encrypted as a separate CKKS vector containing one value. Therefore, a centroid with d features requires d ciphertexts per cluster. In the packed CKKS method, all feature values of one centroid are encrypted together as one CKKS vector. Therefore, each centroid requires one ciphertext per cluster.

© 2026 The Author(s). CC BY 4.0 License

Journal of Artificial Intelligence and Emerging Technologies
Volume 3, Issue 8, pp 41-52, August-2026
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2026.308005
Available Online at https://www.jaiet.com

Following the weighted aggregation principle used in federated k-means clustering [15], both aggregation functions compute a weighted centroid update using participant cluster counts, as shown in Equation (1):

$$\mu_k^{new} = \sum_{i=1}^{N} \left(\frac{n_{ik}}{\sum_{i=1}^{N} n_{ik}} \right) \mu_{ik} \qquad (1)$$

Where mu_ik denotes participant i's local centroid for cluster k, n_ik denotes the number of local records assigned to cluster k, and N denotes the number of participants. In the encrypted configurations, the scalar weighting and summation are performed on CKKS ciphertexts before decryption of the aggregated centroid. Communication volume is measured by serialising each ciphertext and summing the byte length of the serialised payloads. The packed representation reduces communication volume because it serialises fewer ciphertexts for the same centroid update.

Privacy preservation is assessed analytically rather than as a percentage score. Under the semi-honest coordinator assumption, the workflow limits data exposure because raw participant records are not centralised, local computation is performed within each participant partition, local centroid updates are encrypted before transmission, and the coordinator performs weighted aggregation on CKKS ciphertexts rather than plaintext participant centroids.

Dataset and Experimental Setup

The experimental evaluation uses public benchmark datasets with different record counts and feature dimensionalities. Table 1 presents the four datasets used in the experiments.

Table 1: Benchmark Datasets for Experimental Evaluation

Dataset	Instances	Features	Classes	Source	Description
Iris	150	4	3	UCI	Flower species classification
Wine	178	13	3	UCI	Wine cultivar recognition
Diabetes	768	8	2	UCI/Kaggle	Diabetes prediction
Credit Card	1,000	23	2	UCI/Kaggle	Default prediction

The datasets vary by size, feature dimensionality, and class count. Iris provides a low-dimensional benchmark, Wine and Diabetes provide moderate-dimensional numerical clustering settings, and Credit Card Default provides the highest-dimensional evaluation case.

Before clustering, each dataset is downloaded from its configured source URL, converted to numeric format, imputed using feature medians where necessary, and standardised using Standard Scaler. The target column is retained for dataset description and validation only; the clustering algorithm operates on the feature matrix without using target labels.

The processed feature matrix is partitioned into 3 or 5 simulated participant subsets using a fixed random seed. Initial centroids are obtained with scikit-learn k-means using one initial iteration. Both encrypted configurations use the same CKKS context settings defined in the experiment configuration: polynomial modulus degree 8192, coefficient modulus bit sizes [60, 40, 40, 60], and global scale 2^40. The distributed clustering loop stops when the maximum centroid movement falls below the configured tolerance or when the maximum number of iterations is reached. The tolerance is 10^-4.

Packed CKKS Distributed K-Means Algorithm

The packed CKKS method follows the algorithm below. The algorithm is restricted to operations implemented in the source code.

© 2026 The Author(s). CC BY 4.0 License

Journal of Artificial Intelligence and Emerging Technologies
Volume 3, Issue 8, pp 41-52, August-2026
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2026.308005
Available Online at https://www.jaiet.com

Algorithm 1: Packed CKKS Distributed K-Means

INPUT:
X: standardised feature matrix
K: number of clusters
N: number of simulated participants
config: random seed, CKKS parameters, maximum iterations, tolerance

OUTPUT:
Final clustering results and evaluation summary

PROCEDURE:
1. Partition X into N participant subsets using the configured random seed.
2. Initialise global centroids with one scikit-learn k-means iteration.
3. Create the TenSEAL CKKS context from the configured CKKS parameters.
4. For each iteration until maximum iterations:
   4.1 For each participant:
      4.1.1 Assign local records to the nearest global centroid.
      4.1.2 Compute local centroids and cluster counts.
      4.1.3 Encrypt each local centroid as one packed CKKS vector.
      4.1.4 Add the serialised ciphertext size to communication volume.
   4.2 For each cluster:
      4.2.1 Weight each participant ciphertext by its cluster-count proportion.
      4.2.2 Add weighted ciphertexts homomorphically.
      4.2.3 Decrypt the aggregated ciphertext to obtain the updated centroid.
   4.3 Compute maximum centroid movement.
   4.4 Stop if the movement is below the configured tolerance.
5. Assign all records in X to the final centroids.
6. Compute silhouette score, Davies-Bouldin index, timing metrics, and communication metrics.
7. Return the final labels, centroids, quality metrics, timing values, and communication volume.

Evaluation Metrics

After convergence or termination, the final centroids are used to assign labels to the complete dataset. The evaluation records silhouette score, Davies-Bouldin index, iteration count, convergence status, encryption time, aggregation time, total time, communication bytes, and communication kilobytes. Following the comparative efficiency evaluation approach used in CKKS optimisation studies [6], the comparison summary then computes percentage reduction in encryption time and communication volume using Equation (2):

$$Reduction = \frac{Baseline - Packed}{Baseline} \times 100 \qquad (2)$$

Clustering quality is assessed using silhouette score and Davies-Bouldin index. Computational efficiency is assessed using encryption, aggregation, and total timing measurements. Communication overhead is assessed through ciphertext payload size analysis. The central quality-preservation claim concerns comparison with the encrypted baseline, while the centralised plaintext baseline is used as a non-encrypted clustering quality reference.

© 2026 The Author(s). CC BY 4.0 License

Journal of Artificial Intelligence and Emerging Technologies
Volume 3, Issue 8, pp 41-52, August-2026
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2026.308005
Available Online at https://www.jaiet.com

IV. RESULTS & DISCUSSION

Efficiency and Communication Results

The experiments compare plaintext k-means, a per-dimension CKKS encrypted centroid baseline, and the packed CKKS method across the four benchmark datasets. The evaluation combines clustering quality, encryption time, total runtime, and encrypted communication volume. Privacy is not expressed as a percentage score; it is assessed from the implemented data-exposure path, where raw participant data stays local, centroid updates are encrypted before transmission, and aggregation is performed on ciphertexts.

Encryption time fell in every dataset and participant setting when centroid features were packed. The smallest reduction was 57.47% for Iris with 3 participants, and the largest was 95.27% for Credit Card Default with 3 participants.

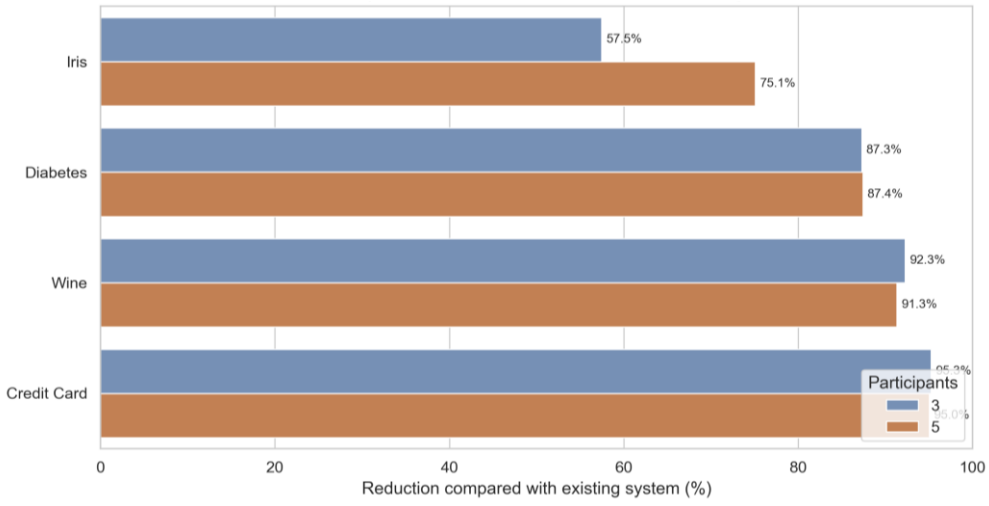

Figure 3: Encryption Time Reduction from Centroid Packing

The same pattern appears in communication volume. Because the packed method sends one ciphertext per centroid instead of one ciphertext per centroid feature, the reduction ranges from 75.00% for Iris to 95.65% for Credit Card Default. Higher-dimensional datasets produce more ciphertexts under the per-dimension baseline, so they benefit more from packing.

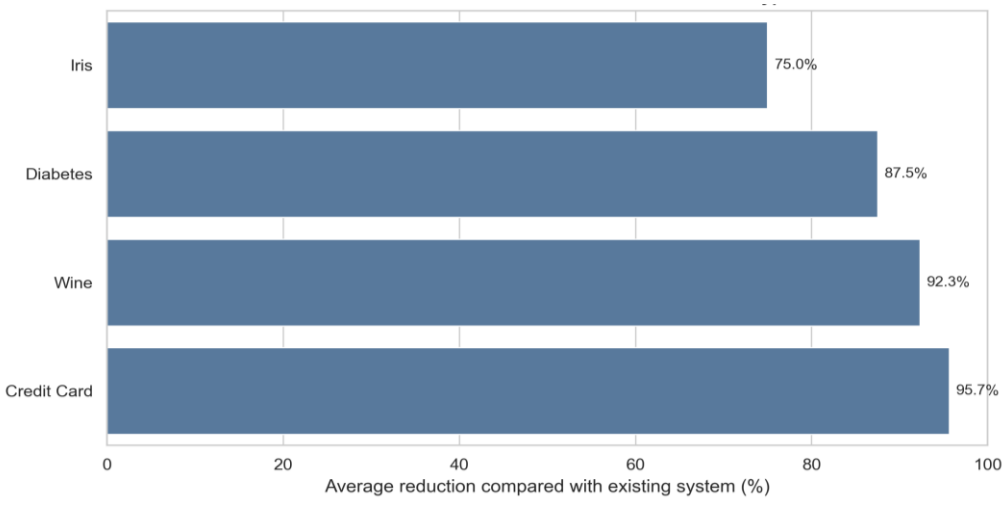

Figure 4: Average Communication Reduction from Packed Centroid Encryption

© 2026 The Author(s). CC BY 4.0 License

Journal of Artificial Intelligence and Emerging Technologies
Volume 3, Issue 8, pp 41-52, August-2026
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2026.308005
Available Online at https://www.jaiet.com

Table 2 summarises the main efficiency results. In all encrypted comparisons, packing reduced encryption time and communication volume without changing the silhouette score relative to the encrypted baseline.

Table 2: Efficiency and Communication Reduction Summary

Dataset	Participants	Encryption Time Reduction (%)	Communication Reduction (%)	Clustering Quality Outcome
Iris	3	57.47	75.00	No measurable silhouette-score difference
Iris	5	75.07	75.00	No measurable silhouette-score difference
Wine	3	92.27	92.31	No measurable silhouette-score difference
Wine	5	91.33	92.31	No measurable silhouette-score difference
Diabetes	3	87.29	87.50	No measurable silhouette-score difference
Diabetes	5	87.42	87.50	No measurable silhouette-score difference
Credit Card Default	3	95.27	95.65	No measurable silhouette-score difference
Credit Card Default	5	95.02	95.65	No measurable silhouette-score difference

The feature-count pattern is consistent with the design of the packed representation. Credit Card Default, with 23 features, achieved the strongest communication reduction, while Iris, with 4 features, achieved the lowest reduction.

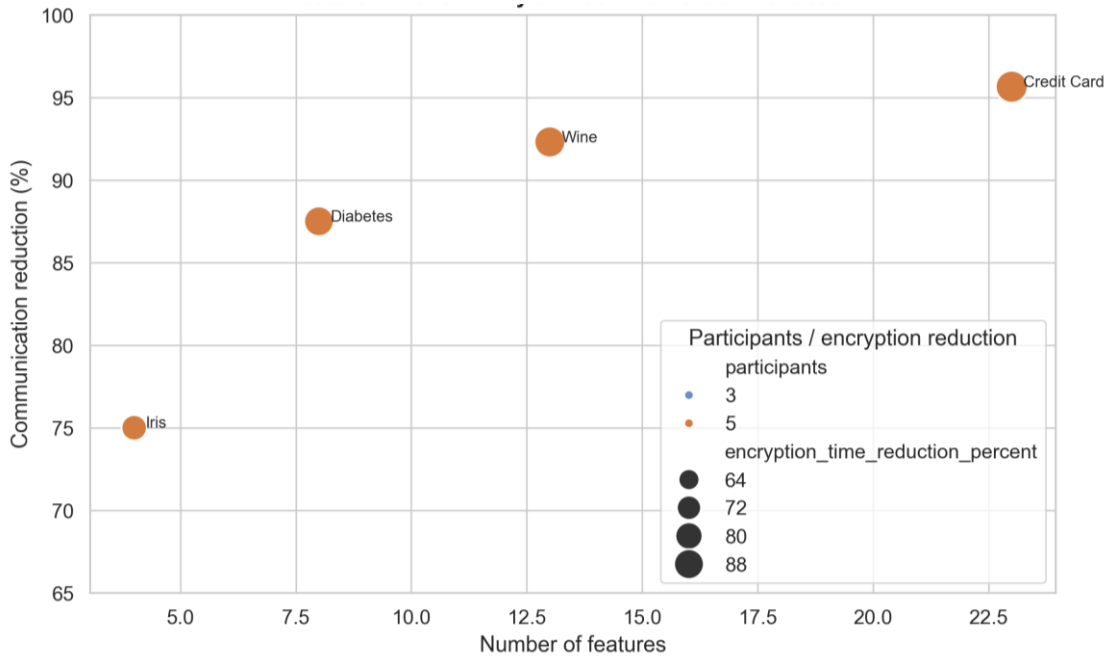

Figure 5: Feature Dimensionality and Communication Reduction

© 2026 The Author(s). CC BY 4.0 License

Journal of Artificial Intelligence and Emerging Technologies
Volume 3, Issue 8, pp 41-52, August-2026
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2026.308005
Available Online at https://www.jaiet.com

Clustering Quality Results

Clustering quality was evaluated using silhouette score and Davies-Bouldin index. A higher silhouette score indicates stronger separation and cohesion, while a lower Davies-Bouldin index indicates better-separated clusters. Table 3 reports the plaintext baseline and the 3-participant encrypted results. The 5-participant encrypted runs produced the same silhouette scores as the corresponding 3-participant runs because the optimisation changes the encryption representation, not the clustering objective.

Table 3: Clustering Quality Results

Dataset	Configuration	Participants	Silhouette Score	Davies-Bouldin Index
Iris	Plaintext k-means baseline	Not applicable	0.4599	0.8336
Iris	Per-dimension CKKS baseline	3	0.4799	0.7894
Iris	Packed CKKS method	3	0.4799	0.7894
Wine	Plaintext k-means baseline	Not applicable	0.2849	1.3892
Wine	Per-dimension CKKS baseline	3	0.2849	1.3892
Wine	Packed CKKS method	3	0.2849	1.3892
Diabetes	Plaintext k-means baseline	Not applicable	0.1960	2.0118
Diabetes	Per-dimension CKKS baseline	3	0.1646	2.0994
Diabetes	Packed CKKS method	3	0.1646	2.0994
Credit Card Default	Plaintext k-means baseline	Not applicable	0.2811	1.7319
Credit Card Default	Per-dimension CKKS baseline	3	0.1690	1.8985
Credit Card Default	Packed CKKS method	3	0.1690	1.8985

Table 3 shows that the packed CKKS method produced the same clustering-quality values as the per-dimension encrypted baseline. For Iris and Wine, the encrypted distributed results are equal to or slightly higher than the plaintext baseline. For Diabetes and Credit Card Default, they are lower than the plaintext baseline, indicating some loss relative to centralised k-means in the simulated distributed setting. The packing step itself does not add further clustering-quality loss.

© 2026 The Author(s). CC BY 4.0 License

Journal of Artificial Intelligence and Emerging Technologies
Volume 3, Issue 8, pp 41-52, August-2026
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2026.308005
Available Online at https://www.jaiet.com

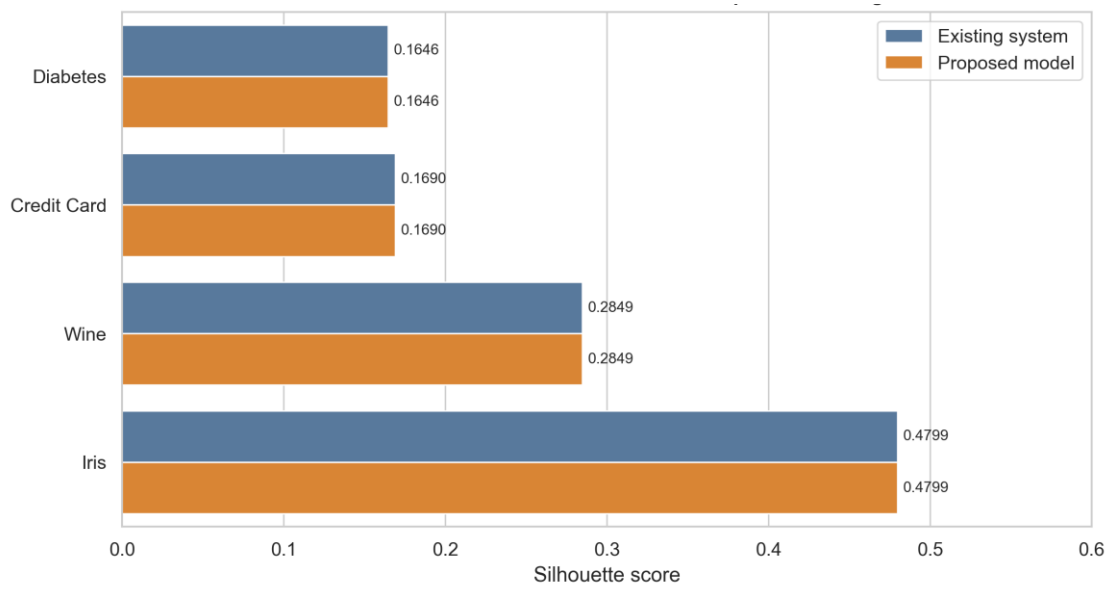

Figure 6: Silhouette Score Preservation in 3-Participant Clustering

Overall, the packed CKKS method gives the strongest encrypted cost-benefit balance. Compared with the per-dimension CKKS baseline, it reduced total execution time by 56.88% to 95.34% and encrypted communication volume by 75.00% to 95.65%, while preserving the same silhouette score and Davies-Bouldin index. The largest gain occurred on Credit Card Default with 3 participants: total execution time dropped from 24.08 seconds to 1.12 seconds, and encrypted communication dropped from 450,504.18 KB to 19,587.03 KB. Plaintext k-means remained cheaper, as expected, but it does not provide the encrypted distributed workflow required for privacy-preserving clustering.

Discussion

The results make clear that encrypted clustering cost depends heavily on how centroid updates are represented before encryption. In the per-dimension baseline, the number of ciphertexts grows with feature dimensionality. In the packed method, the centroid vector becomes the encryption unit, which explains why higher-dimensional datasets such as Credit Card Default and Wine show larger communication reductions.

The efficiency gains did not come from changing the clustering objective. Across the reported encrypted comparisons, the packed method produced the same silhouette score and Davies-Bouldin index as the per-dimension encrypted baseline. Packing therefore acts mainly on operational overhead: it reduces encryption and transmission cost while preserving the encrypted baseline output.

These matters in privacy-sensitive settings such as healthcare and finance, where institutions may need to collaborate without centralising raw data. Plaintext k-means is still computationally cheaper, but it does not meet the privacy-preserving distributed requirement. Under the stated semi-honest coordinator assumption, packed CKKS is a more practical encrypted alternative than per-dimension centroid encryption.

The cost results should be read as operational reductions, not direct monetary savings. The experiments did not use priced cloud instances, paid bandwidth plans, or specialised cryptographic accelerators. Real multi-site deployments, larger datasets, measured network latency, and fuller adversarial security analysis remain necessary next steps.

V. CONCLUSION

This article presented a packed CKKS method for privacy-preserving distributed k-means clustering. Participant records remain local, local centroid updates are encrypted before transmission, and weighted aggregation is performed on ciphertexts under a semi-honest coordinator assumption. Encoding each centroid as one packed encrypted vector, instead of encrypting each centroid feature separately, reduced the number of ciphertexts and encryption operations required during centroid exchange.

© 2026 The Author(s). CC BY 4.0 License

Journal of Artificial Intelligence and Emerging Technologies
Volume 3, Issue 8, pp 41-52, August-2026
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2026.308005
Available Online at https://www.jaiet.com

Experiments on Iris, Wine, Diabetes, and Credit Card Default datasets showed encryption-time reductions of 57.47% to 95.27% and encrypted communication-volume reductions of 75.00% to 95.65% compared with the per-dimension encrypted baseline, with no measurable silhouette-score loss in the reported encrypted comparisons. The results suggest that privacy-preserving distributed clustering becomes more practical when the encryption representation matches the structure of clustering operations. Future work should evaluate the method in real distributed deployments, test larger datasets and additional clustering algorithms, and include deeper security analysis, comparison with alternative privacy-preserving approaches, and automated CKKS parameter selection.

REFERENCES

[1] Chen, H., Chiang, R. H. L., & Storey, V. C. (2012). Business intelligence and analytics: From big data to big impact. *MIS Quarterly: Management Information Systems*, 36(4), 1165-1188. https://doi.org/10.2307/41703503

[2] Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., & Khan, S. U. (2015). The rise of "big data" on cloud computing: Review and open research issues. *Information Systems*, 47, 98-115.

[3] Li, Q., Wen, Z., Wu, Z., Hu, S., Wang, N., Li, Y., Liu, X., & He, B. (2024). A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering*, 36(2), 559-578.

[4] Zhang, C., Xie, Y., Bai, H., Yu, B., Li, W., & Gao, Y. (2021). A survey on federated learning. *Knowledge-Based Systems*, 216, 106775.

[5] Cheon, J. H., Kim, A., Kim, M., & Song, Y. (2017). Homomorphic encryption for arithmetic of approximate numbers. *Advances in Cryptology - ASIACRYPT* 2017, 409-437.

[6] Pan, Y., Chao, Z., He, W., & Xue, M. (2024). FedSHE: Privacy preserving and efficient federated learning with adaptive segmented CKKS homomorphic encryption. *Cybersecurity*, 7(1), 40.

[7] Zuo, Z., Su, N., Li, B., & Zhang, T. (2024). Pack: Towards communication-efficient homomorphic encryption in federated learning. *Proceedings of the 2024 ACM Symposium on Cloud Computing*, 470-486. https://doi.org/10.1145/3698038.3698557

[8] Chen, F., Dong, J., Hu, X., Dong, Z., & Dai, W. (2024). HI-CKKS: Is high-throughput neglected? Reimagining CKKS efficiency with parallelism. Cryptology ePrint Archive, Paper 2024/1976. https://eprint.iacr.org/2024/1976

[9] Gentry, C. (2009). Fully homomorphic encryption using ideal lattices. *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, 169-178.

[10] Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Eichner, H., El Rouayheb, S., Evans, D., Gardner, J., Garrett, Z., Gascon, A., Ghazi, B., Gibbons, P. B., *et al*. (2021). Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 17(1-2), 1-210.

[11] Kumar, S., Gupta, R., & Singh, A. (2023). Configurable encryption and decryption architectures for CKKS-based homomorphic encryption. *Sensors*, 23(17), 7389.

[12] Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., & Zhang, L. (2016). Deep learning with differential privacy. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 308-318.

[13] Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., & Seth, K. (2017). Practical secure aggregation for privacy-preserving machine learning. *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 1175-1191.

[14] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In L. M. Le Cam & J. Neyman (Eds.), *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability* (Vol. 1, pp. 281-297). *University of California Press.*

[15] Garst, S., & Reinders, M. (2024). Federated K-means clustering. *In Proceedings of the International Conference on Pattern Recognition (ICPR 2024), Lecture Notes in Computer Science* (Vol. 15302). *Springer*.

[16] Luo, G., Chen, N., He, J., Jin, B., Zhang, Z., & Li, Y. (2024). Privacy-preserving clustering federated learning for non-IID data. *Future Generation Computer Systems*, 161, 395-410.

[17] Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20, 53-65.

[18] Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning: Data mining, inference, and prediction (2nd ed.). *Springer*.

© 2026 The Author(s). CC BY 4.0 License

Journal of Artificial Intelligence and Emerging Technologies
Volume 3, Issue 8, pp 41-52, August-2026
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2026.308005
Available Online at https://www.jaiet.com

Citation of this Article:

O. M. Omowaye, V. I. E. Anireh, & N. D. Nwiabu. (2026). Improved Privacy-Preserving Clustering Model for Distributed Data. *Journal of Artificial Intelligence and Emerging Technologies (JAIET).* 3(8), 41-52. Article DOI: https://doi.org/10.47001/JAIET/2026.308005

***** End of the Article *****

© 2026 The Author(s). CC BY 4.0 License