Lightweight Trustworthy Distributed Clustering

Hongyang Li*, Caesar Wu*, Mohammed Chadli†, Said Mammar†, Pascal Bouvry*
*University of Luxembourg
{hongyang.li, caesar.wu, pascal.bouvry}@uni.lu
†University Paris-Saclay
{said.mammar, mohammed.chadli}@univ-evry.fr

***Abstract*—Ensuring data trustworthiness within individual edge nodes while facilitating collaborative data processing poses a critical challenge in edge computing systems (ECS), particularly in resource-constrained scenarios such as autonomous systems sensor networks, industrial IoT, and smart cities. This paper presents a lightweight, fully distributed k-means clustering algorithm specifically adapted for edge environments, leveraging a distributed averaging approach with additive secret sharing, a secure multiparty computation technique, during the cluster center update phase to ensure the accuracy and trustworthiness of data across nodes.**

***Index Terms*—Trustworthiness, k-means Clustering, Edge Computing, Distributed Computing, Distributed Learning, Collaborative Data Processing**

I. Introduction

Edge computing, a paradigm emerging from distributed computing, emphasizes processing data at or near its source to minimize latency and reduce bandwidth consumption [1]–[3]. The rapid advancements in edge computing technologies, including algorithms for decentralized and efficient data processing, have significantly accelerated the deployment of distributed sensor networks. These networks, consisting of numerous spatially distributed sensor nodes, facilitate extensive data collection and enable real-time processing, particularly in dynamic and latency-sensitive environments [4], [5].

Two key properties of ECS are crucial in large-scale deployments. First, ECS often employ a distributed architecture [6], where data processing and communication are localized or involve collaboration among a subset of edge nodes, reducing reliance on centralized servers. This decentralized structure enables excellent scalability, as additional edge devices can integrate into the network with minimal dependencies, resembling the functionality of a randomly connected peer-to-peer (P2P) network [7]. Second, ECS frequently operate under strict resource constraints, including limited computational power, bandwidth, and energy availability, often due to the hardware's cost-efficiency requirements and latency-sensitive applications [8]. To address these challenges, lightweight and scalable distributed algorithms are essential to optimize resource utilization and ensure efficient operations in large-scale edge deployments.

Among existing clustering algorithms, the k-means algorithm [9]–[11] is particularly well-suited to resource-constrained environments due to its lightweight nature, balancing simplicity with computational efficiency. Unlike more sophisticated methods, k-means involves simple iterative updates with low computational complexity, which makes k-means an ideal lightweight clustering solution for edge computing systems, where maintaining low latency and resource efficiency is critical.

Specifically, the k-means algorithm is implemented through two primary steps: initially, the cluster assignment step, in which each data point is allocated to a cluster according to its distance from the cluster centers, and subsequently, the cluster center updating step, where the cluster centers are recalculated based on these new assignments.

However, the collaborative framework of distributed k-means clustering in sensor networks, while effective for certain tasks, may introduce significant vulnerabilities, especially in adversarial environments. When data from numerous sensor nodes is shared and aggregated, the potential for data leakage, tampering, and adversarial attacks grows, underscoring the need to protect sensitive information and maintain reliable system performance [12], [13]. For distributed k-means clustering, the cluster assignment step typically does not present trustworthy issues, as it is performed locally within each node. In contrast, the cluster center updating step in an ECS necessitates a data aggregation protocol, such as those found in average consensus algorithms [14], gossip algorithms [15], [16]. This step raises trustworthy concerns because these aggregation protocols require nodes to exchange information, which unavoidably exposes the sensitive data stored on each node. This interconnectedness heightens trust-related challenges, as a single node's malicious activity or compromised data can disrupt the entire network. Achieving trustworthiness demands ensuring data confidentiality, maintaining accuracy, and enhancing robustness against adversarial threats—key objectives that lie at the core of our research.

In this paper, we propose a lightweight, trustworthy distributed k-means clustering algorithm specifically designed for ECS, such as sensor networks. Our approach not only addresses the computational and communication constraints of these networks but also ensures trustworthiness and accuracy performance of system, enabling secure and cost-effective data processing in real-time environments.

II. Related Work

Recent studies have addressed various challenges in sensor networks, particularly in resource-constrained environments [17]–[22]. These works highlight the importance of collaborative tasks within ECS.

2025 37th Chinese Control and Decision Conference (CCDC) | 979-8-3315-1056-5/25/$31.00 ©2025 IEEE | DOI: 10.1109/CCDC65474.2025.11090494

Most existing trustworthy k-means clustering approaches, such as [23]–[30], however, are not designed for fully distributed networks. Instead, they typically focus on data trustworthiness in server-centric settings [31], where multiple servers collect data from users and each server aims to protect its data from other servers. Techniques from secure multiparty computation, such as those in [32], are commonly applied in these settings, as they enable nodes to jointly compute a function while keeping their inputs secure. A comprehensive overview of these approaches can be found in [33].

Applying these algorithms directly to ECS, however, presents challenges, as they generally do not scale well with an increasing number of nodes and often assume only two-party settings. A more user-centric approach [31], which combines Paillier cryptosystems [34] and secure function evaluation (SFE) [32], [35] with a random gossip algorithm [15], offers better scalability and could theoretically be applied to ECS. This method securely computes cluster centers and assigns cluster labels in each k-means iteration, making it a promising candidate for distributed networks.

However, this approach is not practically applicable in sensor networks due to its high computational complexity and significant communication bandwidth requirements, which are particularly problematic in such resource-constrained environments. As shown in [33], the additive secret sharing scheme is a more promising alternative when compared to Yao's circuit evaluation [35] and Paillier cryptosystems [34], especially regarding computational efficiency. The key idea behind additive secret sharing is to break a secret into smaller shares, with the reconstruction relying on simple addition operations. This makes it more suitable for distributed sensor networks, where low-latency communication and minimal computational overhead are critical.

Therefore, we apply the additive secret sharing scheme in a fully distributed setting specifically designed for sensor networks. This method enables secure execution of k-means clustering while ensuring that the private data, such as the sensor measurements from individual nodes and their corresponding cluster labels, remain protected. This approach addresses the need for both trustworthiness and efficient communication in sensor networks.

III. Preliminaries and Problem Definition

A. Distributed k-Means Clustering over ECS

A ECS can be modeled as an undirected graph $G = (V, E)$, with $V = \{1, 2, \ldots, n\}$ representing the set of nodes and $E \subseteq V \times V$ denoting the set of edges. Each node $i \in V$ can communicate only with the nodes within its neighborhood, denoted by $d_i = \{j \mid (i, j) \in E\}$, which defines a fully distributed setting. Let x_i represent the observation data at node i, and let $X = [x_1, x_2, \ldots, x_n]$ denote the entire dataset across the network. The objective of k-means clustering is to partition X into k clusters $K = \{1, 2, \ldots, k\}$, with each cluster associated with a central point, c_j for $j \in K$. The two steps in the k-means algorithm at iteration t are :

1) **Cluster Assignment Step**

Each node i determines its closest cluster by calculating:

$$l_i(t) = \arg\min_{j \in K} \left\| x_i - c_j(t) \right\|^2, \tag{1}$$

where $\left\| x_i - c_j(t) \right\|^2$ represents the squared Euclidean distance between data point x_i and cluster center $c_j(t)$.

2) **Cluster Center Updating Step**

Each cluster center c_j is updated by averaging the data points assigned to it:

$$c_j(t+1) = \frac{1}{N_j} \sum_{x_i \in \mathcal{C}_{j(t)}} x_i, \tag{2}$$

where $\mathcal{C}_{j(t)} = \left\{ x_i \mid l_i(t) = j \right\}$ denotes the data points assigned to cluster j at iteration t, and $N_j = \left| \mathcal{C}_{j(t)} \right|$ is the number of points in that cluster.

In the fully distributed context, the cluster center updating step generally requires a distributed data aggregation protocol [14]–[16], [36], [37], enabling each node to iteratively exchange information only with its neighboring nodes to collectively compute the cluster centers. For simplicity, we assume each observation x_i is a scalar, though this approach can be extended to higher dimensions.

B. Trustworthy Concerns and Adversary Model

Trustworthy operation in ECS relies on clearly defining what qualifies as sensitive data. Each node, x_i, typically regards its observed data as private, even when collaborating on shared tasks like network localization. Participation in such tasks does not imply that a node consents to reveal its individual measurements. For example, while contributing to network localization, a node's specific location should remain undisclosed. Similarly, a node's assigned cluster label, l_i^f, must also be kept confidential, as it could inadvertently expose sensitive details. On the other hand, cluster centers are not treated as private because they represent aggregated information about the group rather than individual nodes.

An essential step in designing reliable algorithms is defining a clear adversarial model. In this work, we consider the honest-but-curious model, sometimes referred to as the semi-honest or passive model [38]. In this framework, nodes execute the algorithm as intended but may attempt to uncover private details about others. These curious nodes, while adhering to protocol, can share received data amongst themselves to infer sensitive information, such as observation data or cluster labels of the honest nodes.

C. Main Requirements

A trustworthy, distributed k-means clustering approach for ECS must meet the following two core requirements:

1) Clustering correctness: The clustering result produced by the trustworthy k-means algorithm should be identical to that of a traditional k-means algorithm without trust concerns.

2) Individual data protection: The sensitive data of each honest node—specifically, the observation data and final cluster label—must remain protected throughout the algorithm's execution under a passive adversary model.

IV. Building Block

In this paper, we opt for the additive secret sharing approach [39] due to its simplicity. Below, we introduce the additive secret sharing algorithm, which will later serve as a foundational component of our proposed method.

The purpose of the adopted algorithm is to securely compute the global average of each node's input, denoted as a_i, across the network while keeping individual inputs confidential. This algorithm ensures both accurate averaging and complete trustiness, provided that each honest node has at least one honest neighboring node.

The algorithm comprises three primary steps. First, each node obfuscates its input a_i by generating an additive secret share s_i^i, following the principles of additive secret sharing [38]. This step is critical for preserving privacy, as the obfuscation ensures that the distribution of s_i^i is statistically independent of the original input, providing perfect trustiness.

In the second step, since the obfuscated shares s_i^i no longer reveal sensitive information, they can be safely used in any distributed averaging protocol to compute the global average, denoted as $\bar{s}$, across all nodes $i \in V$.

Finally, the true average of the original inputs is reconstructed from the obfuscated average via:

$$s_{\mathrm{a}} = \frac{1}{n}\left(\bar{s} \times n \mod p\right),$$

where p is a sufficiently large prime that satisfies $p > \sum_{i\in V} a_i$. Algorithm 1 outlines the steps in detail, where p is chosen to exceed $\sum_{i\in V} a_i$, and F represents the set of integers modulo p.

V. Proposed Approach

The proposed approach includes two parts: cluster assignment and privacy-preserving cluster center updating. Firstly, each node i computes its cluster label $l_i(t)$ based on (1), this local computation will not violate the privacy as it does not require any information sharing. Secondly, a privacy-preserving cluster center updating algorithm is proposed. When updating the cluster centers in ECS, the information exchange required in the distributed data aggregation protocols will inevitably reveal the private data held by each node. Therefore, we apply the abovementioned privacy-preserving distributed averaging algorithm to protect the private data held by each node. Note that even we aim to protect the final cluster label l_i^f from revealing, the intermediate cluster label $l_i(t)$ should also be protected as secret since it is highly correlated with the final cluster label especially during the last few iterations. As a consequence, the goal is to protect both the observation data and intermediate cluster label in cluster center updating step. To do so, each node first constructs the extended observation data and index vector:

$$\begin{aligned}\mathbf{y}_i^{l_i}(t) &= \left[0, \ldots, x_i, \ldots, 0\right], \\ \mathbf{e}_i^{l_i}(t) &= \left[0, \ldots, 1, \ldots, 0\right],\end{aligned} \tag{3}$$

where $\mathbf{y}_i^{l_i}(t) \in \mathbb{R}^{d\times k}$ has the observation data x_i in the $l_i(t)$-th column and zeros in elsewhere. Similarly, the extended index vector $\mathbf{e}_i^{l_i}(t) \in \mathbb{R}^k$ has its $l_i(t)$-th entry as one and zeros elsewhere. Note that inserting zeros in $\mathbf{y}_i^{l_i}(t)$ and $\mathbf{e}_i^{l_i}(t)$ is to keep the sum of observation data and number of nodes in each cluster unchanged.

Algorithm 1 Trustworthy Distributed Average Consensus using Additive Secret Sharing

Step 1: Additive Randomization

1: Initialize each node $i \in V$ with input a_i.
2: **while** Node i has active neighbors $k \in d_i$ **do**
3: Generate a random value r_i^k uniformly from F.
4: Send r_i^k to neighboring node k.
5: **end while**
6: **for** each neighboring node $k \in d_i$ **do**
7: Compute obfuscated share $s_i^k = (r_i^k - r_k^i) \mod p$.
8: **end for**
9: Compute private share for node i:

$$s_i^i = \left(a_i - \sum_{k\in d_i} s_i^k\right) \mod p.$$

Step 2: Distributed Averaging

10: **while** Distributed averaging protocol is running **do**
11: Use s_i^i as input to compute the global obfuscated average $\bar{s}$ via protocols such as [14]–[16].
12: **end while**

Step 3: Average Consensus Computation

13: Each node computes the final average as:

$$s_{\mathrm{a}} = \frac{1}{n}\left(\bar{s} \times n \mod p\right).$$

After that, apply Algorithm 1 to securely compute the averages on the constructed $\mathbf{y}_i^{l_i}(t)$ and $\mathbf{e}_i^{l_i}(t)$ over the network, respectively. The computed averages would be normalized sum and node number in each cluster:

$$\begin{aligned}\bar{\mathbf{Y}}(t) &= \frac{1}{n}\sum_{i\in V} \mathbf{y}_i^{l_i}(t), \\ \bar{\mathbf{N}}(t) &= \frac{1}{n}\sum_{i\in V} \mathbf{e}_i^{l_i}(t).\end{aligned} \tag{4}$$

As a consequence, the cluster centers can be updated as:

$$\mathbf{c}_j^p(t+1) = \frac{\bar{\mathbf{Y}}_j(t)}{\bar{\mathbf{N}}_j(t)}, j \in K. \tag{5}$$

Details of the proposed approach are summarized in Algorithm 2.

VI. Analysis

In this section, we will analyze the proposed approach in terms of clustering accuracy and privacy.

Algorithm 2 Proposed approach

1: Randomly initialize the cluster centers as $c_j(0), j \in K$.
2: For iteration $t = 0, 1, 2, 3, ..., T$
Cluster assignment:
3: Each node i computes its cluster label $l_i(t)$ based on (1).
Privacy-preserving cluster center updating:
4: Each node i constructs its extended private $\mathbf{y}_i^{l_i}(t)$ and extended index vector $\mathbf{e}_i^{l_i}$ based on (3).
5: Each node i set $\mathbf{y}_i^{l_i}(t)$ and $\mathbf{e}_i^{l_i}(t)$ as the inputs (i.e., a_i) in Algorithm 1 to compute the average result $\bar{\mathbf{Y}}(t)$ and $\bar{\mathbf{N}}(t)$ over the network, respectively.
6: Each node update the cluster centers based on (5).
7: Repeat until Convergence (Clustering Center does not change anymore).
8: End

A. Clustering accuracy analysis

By inspecting (4) and (5), we can see that the cluster centers computed by the proposed approach are identical to the original one shown in (1):

$$c_j^p(t+1) = \frac{\bar{\mathbf{Y}}_j(t)}{\bar{\mathbf{N}}_j(t)} = \frac{1}{N_j} \sum_{x_i \in \mathcal{C}_{j(t)}} x_i = c_j(t+1), j \in K.$$

Hence, the proposed approach will output identical cluster centers to the conventional algorithms thereby having no tradeoff in clustering performance and privacy-preservation.

B. Privacy Analysis

As explained before, the cluster assignment step does not violate the privacy concern. Therefore, we will focus on the privacy analysis of the cluster center updating step.

Let V_c and V_h represent the sets of passively corrupted nodes and honest nodes, respectively. To simplify the analysis, we assume initially that the set of honest nodes V_h forms a connected subgraph. As shown in Algorithm 1, the input of any honest node $i \in V_h$ remains secure as long as $|N_i \cap V_h| \neq \emptyset$, recall N_i denotes the neighbors of node i. In this case, the only information disclosed is the aggregated sum of the inputs from all honest nodes. This ensures that no individual input from any honest node is directly exposed, thereby preserving privacy.

Since k-means clustering is an iterative process where the successive iterations are highly correlated. Therefore, we will first analyze the privacy in each iteration and then combine the information in all iterations.

1) Privacy analysis in each iteration: At each iteration t, denote $I(t)$ as the information set obtained by the corrupted node set V_c at current iteration. As the sum of honest nodes' inputs as it can be deduced from the mean of each cluster and the inputs of the corrupted nodes, we have

$$\mathcal{I}(t) = \{\mathbf{s}_y^H(t) = \sum_{i \in \mathcal{H}} \mathbf{y}_i^{l_i}(t), \mathbf{s}_e^H(t) = \sum_{i \in \mathcal{H}} \mathbf{e}_i^{l_i}(t)\}. \quad (6)$$

Let k subsets $\bigcup_j \Lambda_j(t) = V_h$ denotes the set of nodes in each cluster and there are $n_j(t)$ nodes in $\Lambda_j(t)$, the total number of honest nodes is $n_h = \sum_{j \in K} n_j(t)$. The information set is then $\mathcal{I}(t) = \{\mathbf{y}_j(t) = \sum_{i \in \Lambda_j(t)} \mathbf{x}_i | j \in K\}$. Define all the observation data x_i held by honest nodes i as random variable $X_i, i \in V_h$, and $\sum_{i \in \Lambda_j(t)} x_i$ in each cluster as random variable $Y_j(t)$. Assume the observation data x_i held by each node is independent with each other, therefore the random variable $Y_j(t)$ can be seen as the sum of a number of independent random variables as: $Y_j(t) = \sum_{i \in \Lambda_j(t)} X_i, j \in K$. Thus the information set $\mathcal{I}(t)$ contains the joint random variables $\{Y_1(t), Y_2(t), ..., Y_k(t)\}$, and they are also independent with each other. Without loss of generality, for any individual honest node $i \in V_h$ with differential entropy $h(X_i)$, our goal is to compute the information leakage regarding to the individual node i based on the information $\mathcal{I}(t)$ obtained by the corrupted nodes, which is the mutual information $I(X_i; Y_1(t), ..., Y_k(t))$. By chain rule

$$\begin{aligned} I(X_i; Y_1(t), ..., Y_k(t)) &= \sum_{j \in K} (I(X_i; Y_j(t) | Y_{j-1}(t), ..., Y_1(t))) \\ &= \sum_{j \in K} (I(X_i; Y_j(t))), \end{aligned} \quad (7)$$

note that there are at least $k-1$ zeros in $I(X_i; Y_j(t)), j \in K$ since X_i can only locate in one cluster. For example, if X_i is located in cluster 1, then all other mutual information $I(X_i; Y_j(t)), j \neq 1$ are zeros since X_i and $Y_j(t)$ are independent with each other. We then have

$$\begin{aligned} & I(X_i; Y_j(t)) \\ =& \int_{x_i} \int_{y_j(t)} p(x_i, y_j(t)) log \frac{p(x_i, y_j(t))}{p(x_i)p(y_j(t))} \\ =& \sum_{u=1}^{k} Pr\{x_i \in \Lambda_u(t)\} \int_{x_i} \int_{y_j(t)} p(x_i, y_j(t) | x_i \in \Lambda_u(t)) \\ & log \frac{p(x_i, y_j(t))}{p(x_i)p(y_j(t))} \\ =& Pr\{x_i \in \Lambda_j(t)\} I(X_i; Y_j(t) | x_i \in \Lambda_j(t))) \\ =& \frac{n_j(t)}{n_h} I(X_i; Y_j(t) | x_i \in \Lambda_j(t)) \\ =& \begin{cases} 0, n_j(t) > 1 \\ \frac{1}{n_h} h(X_i), n_j(t) = 1, \end{cases} \end{aligned} \quad (8)$$

where $p(x_i, y_j(t) | x_i \in \Lambda_u(t)) = p(x_i, y_u(t)) \sigma_{ju}$ and $\sigma_{ju} = \begin{cases} 0, j \neq u \\ 1, j = u, \end{cases}$ is the indicator function denoting if node i belongs to cluster j. $\frac{n_j(t)}{n_h} = Pr\{x_i \in \Lambda_j(t)\}$ is the probability when node i belongs to cluster j. We can see that mutual information is zero if there are more than one node in certain cluster as the perfect security is guaranteed. If there is only one honest node within certain cluster, the mutual information is $\frac{1}{n_h} h(X_i)$ since $I(X_i; Y_j(t) = X_i | x_i \in \Lambda_j(t)) = h(X_i)$.

2) Privacy analysis across all the iterations: Here, we will now analyze whether a clever combination of information across different iterations will increase the information

leakage. The accumulated information $\mathcal{I}$ obtained by the corrupted nodes across all the iterations is given by $\mathcal{I} = \{\mathcal{I}(t)|t = 1, 2, ..., T\}$. We can derive some extra information by comparing the information obtained in any two iterations $m, k \in 1, 2, ..., T$: $\mathcal{I}_{mk} = \{\mathbf{s}_y^H(m) - \mathbf{s}_y^H(k), \mathbf{s}_e^H(m) - \mathbf{s}_e^H(k)\}$. Assuming the worst case, namely that only one node moves from one cluster to another within two different iterations, then the private value of this node can be computed. Repeat this process, we can see that the accumulated information in the most malicious case would contain a collection of all observation data held by each honest node, i.e., $\mathcal{I} = \{X_i|i \in V_h\}$. The probability of inferring any specified X_i held by node i based on $\mathcal{I}$ would be $\frac{1}{n_h}$. Similar as (7) and (8) , the maximum mutual information can thus be computed as

$$I(X_i; X_1, ..., X_i, ...) = \frac{1}{n_h} h(X_i), \tag{9}$$

Combined with the analysis within each iteration, we can conclude that the maximum information leakage about the observation data x_i throughout the whole algorithm execution is $\frac{1}{n_h} h(X_i)$. Note that if the honest node set V_h is not connected, then it can be divided into several connected subsets $\bigcup_m V_{h_m} = V_h$, the privacy analysis is exactly the same as above by considering each connected subset separately.

Privacy preservation in this framework is ensured under the condition of an honest majority, as is similarly assumed in [40]. This assumption means that more than half of the nodes in the network remain honest and do not collude to compromise privacy. Consider an honest node i within a connected subset V_{h_m} consisting of M nodes. Based on this setting, the following observations can be made regarding privacy:

1) Final cluster label l_i^f: The privacy of the final cluster label is effectively preserved under the honest majority assumption. Specifically, l_i^f can only be revealed if all M honest nodes within the subset V_{h_m} end up in the same cluster. However, this scenario is highly improbable under the honest majority condition. For instance, assuming each node is equally likely to be assigned to any of the k clusters, the probability of all M nodes being in the same cluster is $(\frac{1}{k})^M$. As M increases, this probability diminishes exponentially, making it exceedingly unlikely in practical settings.
2) Private data x_i: The privacy of the observation data x_i is similarly safeguarded. The information leakage associated with x_i is upper-bounded by $\frac{1}{M} h(X_i)$. This upper bound indicates that as the number of honest nodes M in the connected subset increases, the leakage of information about x_i decreases. Under the honest majority condition, M is typically large, ensuring that the actual information leakage is minimal. This relationship highlights the importance of maintaining a robust honest majority, as it significantly mitigates privacy risks.

TABLE I
COMPARISON OF THE PROPOSED AND USER-CENTRIC APPROACHES

	Proposed	**User-centric approach** [31]
Adversary Model	Passive	Passive
Graph Topology	Fully distributed	Fully distributed
Involved Function	Linear	Exponential
Signal Bit Length	$8 - 16$	$512/1024$
Trusted Third Party	No	Yes
Security Model	Perfect	Computational
Distributed Algorithm	Arbitrary [14]–[16], [36], [37]	Gossip [15]
Privacy Concerns	$x_i, l_i(t)$	$x_i, l_i(t), c_j(t)$

VII. COMPARISONS

In this section, we compare the proposed approach with the existing user-centric approach in terms of several parameters shown in Table I.

We can see that both approaches consider the same distributed setting adversary model. The user-centric approach makes use of encryption techniques like Paillier cryptosystems and secure function evaluation adopted in [31], which usually require expensive exponential functions. As a consequence, the required computational complexity and communication bandwidth are much higher than for the proposed approach. For example, if a signal sample usually take 8-16 bits but its encrypted counterparts are usually 1024 or more bits long [41]. In addition, an extra trusted third party is required in the user-centric approach, something that is usually not practical in applications like ECS. Moreover, the achieved perfect (i.e., information-theoretic) security model in the proposed approach is stronger than the computational security model, where the former can guarantee the security against the adversary equipped with unlimited computational power while the later one can not (see [38, Section 1.3.1] for further details). As shown in the distributed averaging step of Algorithm 1, the proposed approach is quite general and can adopt all distributed averaging algorithms for the cluster center updating step while only random gossip algorithm is used in [31]. Concerning privacy, the cluster centers are not considered private information in the proposed approach as they only reflect group property owned by all the individuals within each cluster. The proposed approach has an assumption of honest majority, while this assumption is not required in [31].

VIII. CONCLUSIONS AND FUTURE WORK

In this paper, we presented a lightweight, trustworthy, and fully distributed k-means clustering algorithm tailored for ECS, such as sensor networks. By incorporating the additive secret sharing scheme during the cluster center update process, the proposed algorithm not only achieves identical clustering results to the original k-means but also significantly reduces computational complexity and communication bandwidth. These characteristics make it ideal for the resource-constrained and latency-sensitive environments of ECS.

The method's accuracy and Trustworthiness remain valid as long as the honest majority assumption holds, which is generally reasonable in the context of ECS. However, to enhance the algorithm's robustness in less cooperative or

more adversarial scenarios, future work will explore strategies to relax or eliminate the reliance on the honest majority assumption.

In summary, the proposed lightweight approach effectively addresses the performance and trustworthiness requirements of edge computing systems. We also plan to validate the proposed approach on real-world edge computing datasets and platforms to further demonstrate its practical applicability and scalability.

REFERENCES

[1] W. Z. Khan, E. Ahmed, S. Hakak, I. Yaqoob, and A. Ahmed, "Edge computing: A survey," *Future Generation Computer Systems*, vol. 97, pp. 219–235, 2019.

[2] Q. Li, J. S. Gundersen, M. Lopuhaä-Zwakenberg, and R. Heusdens, "Adaptive differentially quantized subspace perturbation (adqsp): A unified framework for privacy-preserving distributed average consensus," *IEEE Transactions on Information Forensics and Security*, 2023.

[3] H. Li, C. Wu, M. Chadli, S. Mammar, and P. Bouvry, "Trustworthiness of stochastic gradient descent in distributed learning," *arXiv preprint arXiv:2410.21491*, 2024.

[4] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.

[5] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE communications surveys & tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[6] D. Mosk-Aoyama, T. Roughgarden, and D. Shah, "Fully distributed algorithms for convex optimization problems," *SIAM Journal on Optimization*, vol. 20, no. 6, pp. 3260–3279, 2010.

[7] R. Azimi, H. Sajedi, and M. Ghayekhloo, "A distributed data clustering algorithm in p2p networks," *Applied Soft Computing*, vol. 51, pp. 147–167, 2017.

[8] C. Jiang, T. Fan, H. Gao, W. Shi, L. Liu, C. Cérin, and J. Wan, "Energy aware edge computing: A survey," *Computer Communications*, vol. 151, pp. 556–580, 2020.

[9] S. Pei, Y. Sun, F. Nie, X. Jiang, and Z. Zheng, "Adaptive graph k-means," *Pattern Recognition*, vol. 161, p. 111226, 2025.

[10] J. Huang, Q. Feng, Z. Huang, J. Xu, and J. Wang, "Near-linear time approximation algorithms for k-means with outliers," in *Forty-first International Conference on Machine Learning*, 2024.

[11] G. Hamerly and C. Elkan, "Learning the k in k-means," *Advances in neural information processing systems*, vol. 16, 2003.

[12] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[13] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *ACM Sigmod Record*, vol. 29, no. 2. ACM, 2000, pp. 439–450.

[14] B. Johansson and M. Johansson, "Faster linear iterations for distributed averaging," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 2861–2866, 2008.

[15] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE transactions on information theory*, vol. 52, no. 6, pp. 2508–2530, 2006.

[16] A. G. Dimakis, S. Kar, J. M. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1847–1864, 2010.

[17] O. S. Egwuche, A. Singh, A. E. Ezugwu, J. Greeff, M. O. Olusanya, and L. Abualigah, "Machine learning for coverage optimization in wireless sensor networks: a comprehensive review," *Annals of Operations Research*, pp. 1–67, 2023.

[18] K. R. Rao, B. N. K. Reddy, and A. S. Kumar, "Using advanced distributed energy efficient clustering increasing the network lifetime in wireless sensor networks," *Soft Computing*, vol. 27, no. 20, pp. 15 269–15 280, 2023.

[19] D. K. Bangotra, Y. Singh, A. Selwal, N. Kumar, and P. K. Singh, "A trust based secure intelligent opportunistic routing protocol for wireless sensor networks," *Wireless Personal Communications*, vol. 127, no. 2, pp. 1045–1066, 2022.

[20] C. Sureshkumar and S. Sabena, "Fuzzy-based secure authentication and clustering algorithm for improving the energy efficiency in wireless sensor networks," *Wireless Personal Communications*, vol. 112, no. 3, pp. 1517–1536, 2020.

[21] C. Lv, Y. Ren, X. Li, P. Wang, Z. Du, G. Ma, and H. Chi, "Unmanned aerial vehicle-assisted sparse sensing in wireless sensor networks," *IEEE Wireless Communications Letters*, vol. 12, no. 6, pp. 977–981, 2023.

[22] H. Feng, J. Wang, Z. Fang, J. Qian, and K.-C. Chen, "Age of information in uav aided wireless sensor networks relying on blockchain," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 9, pp. 12 430–12 435, 2023.

[23] S. Samet, A. Miri, and L. Orozco-Barbosa, "Privacy preserving k-means clustering in multi-party environment." in *SECRYPT*, 2007, pp. 381–385.

[24] C. Su, F. Bao, J. Zhou, T. Takagi, and K. Sakurai, "Privacy-preserving two-party k-means clustering via secure approximation," in *21st International Conference on Advanced Information Networking and Applications Workshops (AINAW'07)*, vol. 1. IEEE, 2007, pp. 385–391.

[25] P. Bunn and R. Ostrovsky, "Secure two-party k-means clustering," in *Proceedings of the 14th ACM conference on Computer and communications security*. ACM, 2007, pp. 486–497.

[26] M. C. Doganay, T. B. Pedersen, Y. Saygin, E. Savaş, and A. Levi, "Distributed privacy preserving k-means clustering with additive secret sharing," in *Proceedings of the 2008 international workshop on Privacy and anonymity in information society*. ACM, 2008, pp. 3–11.

[27] K. Xing, C. Hu, J. Yu, X. Cheng, and F. Zhang, "Mutual privacy preserving *k*-means clustering in social participatory sensing," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 2066–2076, 2017.

[28] Y. Fan, J. Bai, X. Lei, W. Lin, Q. Hu, G. Wu, J. Guo, and G. Tan, "Ppmck: Privacy-preserving multi-party computing for k-means clustering," *Journal of Parallel and Distributed Computing*, vol. 154, pp. 54–63, 2021.

[29] D. Zhao, X. Hu, S. Xiong, J. Tian, J. Xiang, J. Zhou, and H. Li, "K-means clustering and knn classification based on negative databases," *Applied soft computing*, vol. 110, p. 107732, 2021.

[30] Q. Li and L. Luo, "On the privacy of federated clustering: A cryptographic view," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 4865–4869.

[31] J. Sakuma and S. Kobayashi, "Large-scale k-means clustering with user-centric privacy-preservation," *Knowledge and Information Systems*, vol. 25, no. 2, pp. 253–279, 2010.

[32] O. Goldreich, *Foundations of cryptography: volume 2, basic applications*. Cambridge university press, 2009.

[33] F. Meskine and S. N. Bahloul, "Privacy preserving k-means clustering: a survey research." *Int. Arab J. Inf. Technol.*, vol. 9, no. 2, pp. 194–200, 2012.

[34] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 1999, pp. 223–238.

[35] A. C.-C. Yao, "How to generate and exchange secrets," in *Foundations of Computer Science, 1986., 27th Annual Symposium on*. IEEE, 1986, pp. 162–167.

[36] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[37] G. Zhang and R. Heusdens, "Distributed optimization using the primal-dual method of multipliers," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 4, no. 1, pp. 173–187, 2018.

[38] R. Cramer, I. B. Damgrd, and J. B. Nielsen, *Secure Multiparty Computation and Secret Sharing*, 1st ed. New York, NY, USA: Cambridge University Press, 2015.

[39] N. Gupta, J. Katz, N. Chopra, "Privacy in distributed average consensus," *IFAC-PapersOnLine, vol. 50, no. 1, pp. 9515-9520*, 2017.

[40] X. Lin, C. Clifton, and M. Zhu, "Privacy-preserving clustering with distributed em mixture modeling," *Knowledge and information systems*, vol. 8, no. 1, pp. 68–81, 2005.

[41] R. L. Lagendijk, Z. Erkin, and M. Barni, "Encrypted signal processing for privacy protection: Conveying the utility of homomorphic encryption and multiparty computation," *IEEE Signal Processing Magazine*, vol. 30, no. 1, pp. 82–105, 2013.