Communication-efficient k-Means for Edge-based Machine Learning

Hanlin Lu, *Student Member, IEEE,* Ting He, *Senior Member, IEEE,* Shiqiang Wang, *Member, IEEE,* Changchang Liu, Mehrdad Mahdavi, Vijaykrishnan Narayanan, *Fellow, IEEE,* Kevin S. Chan, *Senior Member, IEEE,* and Stephen Pasteris

Abstract—We consider the problem of computing the k-means centers for a large high-dimensional dataset in the context of edge-based machine learning, where data sources offload machine learning computation to nearby edge servers. k-Means computation is fundamental to many data analytics, and the capability of computing provably accurate k-means centers by leveraging the computation power of the edge servers, at a low communication and computation cost to the data sources, will greatly improve the performance of these analytics. We propose to let the data sources send small summaries, generated by joint dimensionality reduction (DR), cardinality reduction (CR), and quantization (QT), to support approximate k-means computation at reduced complexity and communication cost. By analyzing the complexity, the communication cost, and the approximation error of k-means algorithms based on carefully designed composition of DR/CR/QT methods, we show that: (i) it is possible to compute near-optimal k-means centers at a near-linear complexity and a constant or logarithmic communication cost, (ii) the order of applying DR and CR significantly affects the complexity and the communication cost, and (iii) combining DR/CR methods with a properly configured quantizer can further reduce the communication cost without compromising the other performance metrics. Our theoretical analysis has been validated through experiments based on real datasets.

Index Terms—k-Means, dimensionality reduction, coreset, random projection, quantization, edge-based machine learning.

1 Introduction

Edge-based machine learning [2] is an emerging application scenario, where mobile/wireless devices collect data and transmit them (or their summaries) to nearby edge servers for processing. Compared to alternative approaches, e.g., transmitting locally learned model parameters as in federated learning [3], transmitting data summaries has the advantages that: (i) only one round of communications is required,[1] (ii) the transmitted data can potentially be used to compute other machine learning models [5], [6], and (iii) the edge server can solve the machine learning problem closer to the optimality than the data-collecting devices within the same time. In this work, we focus on k-means clustering under the framework of edge-based machine learning.

k-Means clustering is one of the most widely-used machine learning techniques. Algorithms for k-means are used in many areas of data science, e.g., for data compression, quantization, hashing; see the survey in [7] for more details. Recently, it was shown in [5], [6] that the centers of k-means can be used as a proxy of the original dataset in computing a broader set of machine learning models with sufficiently continuous cost functions. Thus, efficient and accurate computation of k-means can bring broad benefits to machine learning applications.

However, solving k-means is nontrivial. The problem is known to be NP-hard, even for two centers [8] or in the plane [9]. Due to its fundamental importance, how to speed up the k-means computation for large datasets has received significant attention. Most existing solutions can be classified into two approaches: *dimensionality reduction (DR)* methods that aim at generating a "thinner" dataset with a reduced number of attributes [10], and *cardinality reduction (CR)* methods that aim at generating a "smaller" dataset with a reduced number of data points (i.e., samples) [11]. However, these solutions assumed that the full dataset is available locally at the server that performs k-means computation, and hence ignored the communication cost.

To our knowledge, we are the first to explicitly analyze the communication cost in computing k-means over remote and possibly distributed data. The need of communications arises in the application scenario of edge-based machine learning, where edge devices collecting data wish to offload machine learning computation to nearby edge servers through wireless links. Given a large high-dimensional dataset, i.e., $n, d \gg 1$ (n: cardinality, d: dimension), residing at one or multiple data sources at the network edge, an obvious solution of solving k-means at the data sources and sending the centers to the server will incur a high computational complexity at the data sources that is not suitable for the limited computation power of edge devices, while another obvious solution of sending the raw data

- *H. Lu, T. He, M. Mahdavi, and V. Narayanan are with Pennsylvania State University, University Park, PA 16802, USA (email: {hzl263, tzh58, mzm616, vxn9}@psu.edu).*
- *S. Wang and C. Liu are with IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA (email: {wangshiq@us., Changchang.Liu33@}ibm.com).*
- *K. Chan is with Army Research Laboratory, Adelphi, MD 20783, USA (email: kevin.s.chan.civ@mail.mil).*
- *S. Pasteris is with University College London, London WC1E 6EA, UK (email: s.pasteris@cs.ucl.ac.uk).*

A preliminary version of this work was presented at ICDCS'20. [1].

1. In cases that the raw data are spread over multiple nodes, another round of communications is needed to decide the sizes of data summaries to collect from each node [4]. However, each node only sends one scalar in this round and hence the communication cost is negligible.

to the server and solving k-means there will incur a high communication cost that imposes too much stress on the wireless links. We seek to achieve a better tradeoff by letting the data sources send small data summaries generated by efficient data reduction methods and leaving the k-means computation to the server.

Besides DR and CR, quantization (QT) [12] can also reduce the communication cost by representing each data point with a smaller number of bits. While k-means itself has been used to design vector quantizers [13], we will show that simpler quantizers can be combined with DR/CR methods to compute approximate k-means at an even lower communication cost without negatively affecting the complexity or the quality of solution.

1.1 Summary of Contributions

We want to develop efficient k-means algorithms suitable for edge-based machine learning, by offloading as much computation as possible to edge servers at a low communication cost to data sources. Our contributions include:

1) If the data reside at a single data source, we show that (i) it is possible to solve k-means arbitrarily close to the optimal with constant communication cost and near-linear complexity at the data source by combining suitably selected DR/CR methods, (ii) the order of applying DR and CR methods will not affect the approximation error, but will lead to different tradeoffs between communication cost and computational complexity, and (iii) repeating DR both before and after CR can further improve the performance.

2) If the data are distributed over multiple data sources, we show that suitably combining DR/CR methods can solve k-means arbitrarily close to the optimal with near-linear complexity at the data sources and a total communication cost that is logarithmic in the data size.

3) We further extend our solution to include quantization. Using the rounding-based quantizer as an example, we demonstrate how to configure the quantizer to minimize the communication cost while guaranteeing a given approximation error.

4) Through experiments on real datasets, we verify that (i) joint DR and CR can drastically reduce the communication cost without incurring a high complexity at the data sources or significantly degrading the solution quality, (ii) the proposed joint DR-CR algorithms can achieve a solution quality similar to state-of-the-art algorithms while notably reducing the communication cost and the complexity, and (iii) combining DR/CR with quantization can further reduce the communication cost without compromising the other performance metrics.

Roadmap. Sections 2–3 review the background on DR/CR methods. Section 4 presents our results on joint DR/CR in the centralized setting, and Section 5 presents those in the distributed setting. Section 6 presents further improvement via joint DR, CR, and quantization. Section 7 evaluates our solutions on real datasets. Finally, Section 8 concludes the paper. Proofs are given in Appendix.

2 Related Work

Our work belongs to the studies on data reduction for approximate k-means. Existing solutions can be classified into the following categories:

Dimensionality reduction (DR): DR for k-means, initiated by [14], aims at speeding up k-means by reducing the number of features (i.e., the dimension). Two approaches have been proposed: 1) *feature selection* that selects a subset of the original features, and 2) *feature extraction* that constructs a smaller set of new features. For feature selection, the best known algorithms are from [15], including a random sampling algorithm that achieves a $(1+\epsilon)$-approximation using $O(k \log k/\epsilon^2)$ features, and a deterministic algorithm that achieves a $(1+\epsilon)$-approximation using $O(k/\epsilon^2)$ features. For feature extraction, there are two methods with guaranteed approximation, both based on linear projections. The first method is based on *principal component analysis (PCA)* via computing the *singular value decomposition (SVD)*, where exact SVD gives 2-approximation using k features [16] or $(1+\epsilon)$-approximation using $\lceil k/\epsilon \rceil$ features [15], and approximate SVD gives $(2+\epsilon)$-approximation using k features [17] or $(1+\epsilon)$-approximation using $\lceil k/\epsilon \rceil$ features [15]. The second method is based on *random projections* that preserve vector ℓ-2 norms with an arbitrarily high probability, whose existence is guaranteed by the *Johnson-Lindenstrauss (JL) lemma* [18]. The best known algorithm there is given by [10], which achieves a $(1+\epsilon)$-approximation using $O(log(k/\epsilon)/\epsilon^2)$ features.

Cardinality reduction (CR): CR for k-means, initiated by [19], aims at using a small weighted set of points in the same space, referred to as a *coreset*, to replace the original dataset. A coreset is called an ϵ-coreset (for k-means) if it can approximate the k-means cost of the original dataset for every candidate set of centers up to a factor of $1 \pm \epsilon$. Many coreset construction algorithms have been proposed for k-means. Early algorithms use geometric partitions to merge each group of nearby points into a single coreset point [19], [20], [21], which cause the cardinality of the coreset to be exponential in the dimension d. Later, [22] showed that sampling can be used to reduce the coreset cardinality to a polynomial in $k, \epsilon, \log n$, and d. Most state-of-the-art coreset construction algorithms are based on the *sensitivity sampling* framework that was first proposed in [23] and then formalized in [24]. To generate an ϵ-coreset, the solution in [24] needs a coreset cardinality of[2] $\tilde{O}(kd\epsilon^{-4})$, and its followup in [25] needs $\tilde{O}(k^2 d\epsilon^{-2})$. The best known solution[3] is the one in [11] (presented implicitly in the proof of Theorem 36), which showed that by reducing the intrinsic dimension of the dataset and adding a constant term to the coreset-based cost, the cardinality of an ϵ-coreset can be reduced to $\tilde{O}(k^3\epsilon^{-4})$.

Joint DR-CR: Among the above works, only [11], [27] considered joint DR and CR for k-means computation.

Algorithms in the distributed setting: In this setting, [4] proposed a distributed version of sensitivity sampling to construct an ϵ-coreset over a distributed dataset, and [27] further combined this algorithm with a distributed PCA algorithm from [11]. Besides these theoretical results, there are also system works on adapting centralized k-means algorithms for distributed settings, e.g., MapReduce

2. We use $\tilde{O}(x)$ to denote a value that is at most linear in x times a factor that is polylogarithmic in x.

3. There is another algorithm with a coreset cardinality independent of n and d (precisely, $k^{O(\epsilon^{-2})}$) in [26]. However, the algorithm has a high complexity and the coreset cardinality is larger than that in [11].

[28], sensor networks [29], and Peer-to-Peer networks [30]. However, these algorithms are only heuristics.

Limitations & improvements: While extensively studied, existing solutions mainly focused on reducing the computation time, leaving open the issue of communication cost. Moreover, we note that: (i) the state-of-the-art data reduction methods [11], [27] blindly assumed that DR should be applied before CR, leaving open whether it is possible to achieve better performance by reversing the order of DR/CR or applying them repeatedly, and (ii) most of the algorithms for the distributed setting are heuristics without guarantees on how well their solutions approximate the optimal solution. To fill this gap, we will perform a comprehensive analysis in terms of computational complexity, communication cost, and approximation error, while carefully designing the order of applying DR/CR. In addition to DR and CR, QT [12] is also an effective method for reducing the communication cost by lowering the data precision. While k-means itself has been used to design certain quantizers [13], the use of simpler quantizers for communication-efficient k-means computation has not been studied before. In this regard, we will show how to properly combine a simple rounding-based quantizer with DR/CR methods to further reduce the communication cost without compromising the other performance metrics.

3 Background and Formulation

We start with an overview of existing results on DR and CR for k-means, followed by our problem statement.

3.1 Notations & Definitions

Definitions: Consider a dataset $P \subset \mathbb{R}^d$ with cardinality n that resides at one or multiple data sources (i.e., data-collecting devices), where both $n \gg 1$ and $d \gg 1$. We want to find, with assistance of an edge server, the k points $X = \{x_i\}_{i=1}^k$ that minimize the following cost function[4]:

$$\text{cost}(P, X) := \sum_{p \in P} \min_{x_i \in X} \|p - x_i\|^2. \tag{1}$$

This is the *k-means clustering* problem, and the points in X are called *centers*. Equivalently, the k-means clustering problem can be considered as the problem of finding the partition $\mathcal{P} = \{P_1, \ldots, P_k\}$ of P into k clusters that minimizes the following cost function:

$$\text{cost}(\mathcal{P}) := \sum_{i=1}^{k} \min_{x_i \in \mathbb{R}^d} \sum_{p \in P_i} \|p - x_i\|^2. \tag{2}$$

Notations: We will use $\|x\|$ to denote the ℓ-2 norm if x is a vector, or the Frobenius norm if x is a matrix. We will use $A_P \in \mathbb{R}^{n \times d}$ to denote the matrix representation of a dataset $P \subset \mathbb{R}^d$, where each row corresponds to a data point. Let $\mu(P)$ denote the optimal 1-means center of P, which is well-known to be the sample mean, i.e., $\mu(P) = \frac{1}{|P|} \sum_{p \in P} p$. Let $\mathcal{P}_{P,X}$ denote the partition of dataset P induced by centers X, i.e., $\mathcal{P}_{P,X} = \{P_1, \ldots, P_{|X|}\}$ for $P_i := \{p \in P : \|p - x_i\| \leq \|p - x_j\|, \forall x_j \in X \setminus \{x_i\}\}$ (ties broken arbitrarily). Given scalars x, y, and ϵ ($\epsilon > 0$), we will use $x \approx_{1+\epsilon} y$ to denote $\frac{1}{1+\epsilon} x \leq y \leq (1+\epsilon)x$. In our analysis, we use $O(x)$ to denote a value that is at most linear in x, $\Omega(x)$ to denote a value that is at least linear in x, and $\tilde{O}(x)$ to denote a value that is at most linear in x times a factor that is polylogarithmic in x.

4. The norms in (1) and (2) refer to the ℓ-2 norm.

TABLE 1
Key Abbreviations and Notations

Notation	Explanation
DR	dimensionality reduction
CR	cardinality reduction
QT	quantization
PCA	principal component analysis
JL projection	a linear projection satisfying Theorem 3.1
FSS	the algorithm proposed in [11, Theorem 36]
BKLW	the distributed version of FSS proposed in [27, Algorithm 1]
P	original input dataset
n	cardinality of P
d	dimensionality of P
k	number of clustering centers
X	a set of clustering centers
$\mathcal{P}$	a partition of P
$\mu(P)$	the optimal 1-means center of P
$\mathcal{P}_{P,X}$	the partition of dataset P induced by centers X

Given a dimensionality reduction map $\pi : \mathbb{R}^d \to \mathbb{R}^{d'}$ ($d' < d$), we use $\pi(P) := \{\pi(p) : p \in P\}$ to denote the output dataset for an input dataset P, and $\pi(\mathcal{P}) := \{\pi(P_1), \ldots, \pi(P_k)\}$ to denote the partition of $\pi(P)$ corresponding to a partition $\mathcal{P} = \{P_1, \ldots, P_k\}$ of P. Moreover, given a partition $\mathcal{P}' = \pi(\mathcal{P})$, we use $\pi^{-1}(\mathcal{P}')$ to denote the corresponding partition of P, which puts $p, q \in P$ into the same cluster if and only if $\pi(p), \pi(q) \in P'$ belong to the same cluster under $\mathcal{P}'$. Finally, given $P' = \pi(P)$, we use $\pi^{-1}(P') := \{\pi^{-1}(p') : p' \in P'\}$ to denote a set of points in $\mathbb{R}^d$ that is mapped to P' by π. Note that there is no guarantee that $\pi^{-1}(P') = P$. However, suppose $\tilde{P}$ is the solution which satisfies $\pi(\tilde{P}) = P'$, then $\tilde{P}$ must exist (P is a feasible solution) and $\pi^{-1}(P')$ denotes an arbitrary solution. If π is a linear map, i.e., $\pi(P) := A_P \Pi$ for a matrix $\Pi \in \mathbb{R}^{d \times d'}$, then the *Moore-Penrose inverse* Π^+ [31] of Π gives a feasible solution $\pi^{-1}(P') := A_{P'} \Pi^+$.

The main notations and abbreviations used in the paper are summarized in Table 1.

3.2 Dimensionality Reduction for k-Means

Definition 3.1. *We say that a DR map $\pi : \mathbb{R}^d \to \mathbb{R}^{d'}$ ($d' < d$) is an* ϵ-projection *if it preserves the cost of any partition up to a factor of $1 + \epsilon$, i.e., cost$(\mathcal{P}) \approx_{1+\epsilon}$ cost$(\pi(\mathcal{P}))$ for every partition $\mathcal{P} = \{P_1, \ldots, P_k\}$ of a finite set $P \subset \mathbb{R}^d$.*

One commonly used method to construct ϵ-projection is random projection, where the cornerstone result is the JL Lemma:

Lemma 3.1 ([18]). *There exists a family of random linear maps $\pi : \mathbb{R}^d \to \mathbb{R}^{d'}$ with the following properties: for every $\epsilon, \delta \in (0, 1/2)$, there exists $d' = O(\frac{\log(1/\delta)}{\epsilon^2})$ such that for every $d \geq 1$ and all $x \in \mathbb{R}^d$, we have $\Pr\{\|\pi(x)\| \approx_{1+\epsilon} \|x\|\} \geq 1 - \delta$.*

Based on this lemma, the best known result achieved by random projection is the following:

Theorem 3.1 ([10]). *Consider any family of random linear maps $\pi : \mathbb{R}^d \to \mathbb{R}^{d'}$ that (i) satisfies Lemma 3.1, and (ii) is sub-Gaussian-tailed (i.e., the probability for the norm after mapping to be larger than the norm before mapping by a factor of at least $1 + t$ is bounded by $e^{-\Omega(d't^2)}$). Then for every $\epsilon, \delta \in (0, 1/4)$, there exists $d' = O(\frac{1}{\epsilon^2} \log \frac{k}{\epsilon\delta})$, such that π is an ϵ-projection with probability at least $1 - \delta$.*

There are many known methods to construct a random linear map that satisfies the conditions (i–ii) in Theorem 3.1, e.g., maps defined by matrices with i.i.d. *Gaussian* and sub-*Gaussian* entries [32], [33], [34]. We will refer to such a random projection as a *JL projection*.

Remark: Compared with PCA-based DR methods, JL projection has the advantage that the projection matrix is *data-oblivious*, and can hence be pre-generated and distributed, or generated independently by different nodes using a shared random number generation seed, both incurring negligible communication cost at runtime. As is shown later, this can lead to significant savings in the communication cost.

3.3 Cardinality Reduction for k-Means

CR methods, also known as *coreset construction algorithms*, aim at constructing a smaller weighted dataset (*coreset*) with a bounded approximation error as follows.

Definition 3.2 ([11]). *We say that a tuple (S, Δ, w), where $S \subset \mathbb{R}^d$, $w : S \to \mathbb{R}$, and $\Delta \in \mathbb{R}$, is an ϵ-coreset of $P \subset \mathbb{R}^d$ if it preserves the cost for every set of k centers up to a factor of $1 \pm \epsilon$, i.e.,*

$$(1 - \epsilon)cost(P, X) \leq cost(\mathbf{S}, X) \leq (1 + \epsilon)cost(P, X) \quad (3)$$

for any $X \subset \mathbb{R}^d$ with $|X| = k$, where

$$cost(\mathbf{S}, X) := \sum_{q \in S} w(q) \cdot \min_{x_i \in X} \|q - x_i\|^2 + \Delta \quad (4)$$

denotes the k-means cost for a coreset $\mathbf{S} := (S, \Delta, w)$ and a set of centers X.

We note that the above definition generalizes most of the existing definitions of ϵ-coreset, which typically ignore Δ.

The best known coreset construction algorithm for k-means was given in [11], which first reduces the intrinsic dimension of the dataset by PCA, and then applies sensitivity sampling to the dimension-reduced dataset to obtain an ϵ-coreset of the original dataset with a size that is constant in n and d.

Theorem 3.2 ([11]). *For any $\epsilon, \delta \in (0, 1)$, with probability at least $1 - \delta$, an ϵ-coreset (S, Δ, w) of size $|S| = O\left(\frac{k^3 \log^2 k}{\epsilon^4} \log(\frac{1}{\delta})\right)$ can be computed in time $O(\min(nd^2, n^2 d) + nk\epsilon^{-2}(d + k \log(1/\delta)))$.*

However, [11] only focused on minimizing the cardinality of coreset, ignoring the cost of transmitting the coreset. As is shown later (Section 5.3), its proposed algorithm can be severely suboptimal in the communication cost.

3.4 Problem Statement

The motivation of most existing DR/CR methods designed for k-means is to speed up k-means computation in a setting where the node holding the data is also the node computing k-means. In contrast, we want to develop efficient k-means algorithms in scenarios where the data generation and the k-means computation occur at different locations, such as in the case of edge-based learning. We will refer to the node(s) holding the original data as the *data source(s)*, and the node running k-means computation as the *server*.

We will evaluate each considered algorithm by the following performance metrics:

- *Approximation error:* We say that a set of k-means centers X is an α-approximation ($\alpha > 1$) for k-means clustering of P if $\text{cost}(P, X) \leq \alpha \cdot \text{cost}(P, X^*)$, where X^* is the optimal set of k-means centers for P.
- *Communication cost:* We say that an algorithm incurs a communication cost of y if a data source employing the algorithm needs to send y scalars to the server.
- *Complexity:* We say that an algorithm incurs a (time) complexity of z at the data source if a data source employing the algorithm needs to perform z elementary operations.

4 Joint DR and CR at a Single Data Source

We will first focus on the scenario where all the data are at a single data source (the *centralized setting*). We will show that: 1) using suitably selected DR/CR methods and a sufficiently powerful server, it is possible to solve k-means arbitrarily close to the optimal, while incurring a low communication cost and a low complexity at the data source; 2) the order of applying DR and CR does not affect the approximation error, but affects the complexity and the communication cost; 3) repeated application of DR/CR can lead to a better communication-computation tradeoff than applying DR and CR only once.

4.1 DR+CR

We first consider the approach of applying DR and then CR.

4.1.1 An Existing DR+CR Algorithm

The state-of-the-art joint DR and CR algorithm, referred to as *FSS* following the authors' last names, was implicitly presented in Theorem 36 in [11]. FSS first uses PCA to reduce the intrinsic dimension of the dataset and then applies sensitivity sampling. Theorem 3.2 gives the complexity of FSS, but the approximation error and the communication cost incurred when using FSS to generate a data summary for k-means were not given in [11]. Thus, we provide them (proved in Appendix A) to facilitate later comparison.

Theorem 4.1. *Suppose that the data source reports the coreset $\mathbf{S} := (S, \Delta, w)$ computed by FSS [11] and the server computes the optimal k-means centers X of $\mathbf{S}$[5]. Then:*

1) *X is a $(1 + \epsilon)/(1 - \epsilon)$-approximation for k-means clustering of P with probability $\geq 1 - \delta$;*
2) *the communication cost is $O\left(kd/\epsilon^2\right)$,*

assuming $\min(n, d) \gg k, 1/\epsilon$, and $1/\delta$.

5. Given a coreset $\mathbf{S} = (S, \Delta, w)$, X can be computed by ignoring Δ and applying a weighted k-means algorithm to minimize $\sum_{q \in S} w(q) \cdot \min_{x_i \in X} \|q - x_i\|^2$, or by converting $\mathbf{S}$ into an unweighted dataset by duplicating each $q \in S$ for $w(q)$ times (on the average) and applying an unweighted k-means algorithm.

Algorithm 1: Communication-efficient k-Means under DR+CR

input : Original dataset P, number of centers k, JL projection π_1, FSS-based CR method π_2
output: Centers for k-means clustering of P

1 **data source:**
2 $P' \leftarrow \pi_1(P)$;
3 $(S', \Delta, w) \leftarrow \pi_2(P')$;
4 report (S', Δ, w) to the server;
5 **server:**
6 $X' \leftarrow \text{kmeans}(S', w, k)$;
7 $X \leftarrow \pi_1^{-1}(X')$;
8 return X;

Algorithm 2: Communication-efficient k-Means under CR+DR

input : Original dataset P, number of centers k, JL projection π_1, FSS-based CR method π_2
output: Centers for k-means clustering of P

1 **data source:**
2 $(S, \Delta, w) \leftarrow \pi_2(P)$;
3 $S' \leftarrow \pi_1(S)$;
4 report (S', Δ, w) to the server;
5 **server:**
6 $X' \leftarrow \text{kmeans}(S', w, k)$;
7 $X \leftarrow \pi_1^{-1}(X')$;
8 return X;

4.1.2 Communication-efficient DR+CR

Now the question is: can we further reduce the communication cost without hurting the approximation error and the complexity?

Our key observation is that the linear communication cost in d for FSS is due to the transmission of a basis of the projected subspace. In contrast, JL projections are *data-oblivious*. Thus, we can circumvent the cost of transmitting the projected subspace by employing a JL projection as the DR method, as the projected subspace can be predetermined. The following is directly implied by the JL Lemma (Lemma 3.1); see the proof in Appendix A.

Lemma 4.1. *Let $\pi : \mathbb{R}^d \to \mathbb{R}^{d'}$ be a JL projection. Then there exists $d' = O(\epsilon^{-2}\log(nk/\delta))$ such that for any $P \subset \mathbb{R}^d$ with $|P| = n$ and $X, X^* \subset \mathbb{R}^d$ with $|X| = |X^*| = k$, the following holds with probability at least $1-\delta$:*

$$cost(P, X) \approx_{(1+\epsilon)^2} cost(\pi(P), \pi(X)), \tag{5}$$

$$cost(P, X^*) \approx_{(1+\epsilon)^2} cost(\pi(P), \pi(X^*)). \tag{6}$$

Using a JL projection for DR and FSS for CR, we propose Algorithm 1, where the data source computes and reports a coreset in a low-dimensional space by first applying JL projection and then applying FSS (lines 2–4). Based on the dimension-reduced coreset (S', Δ, w), the server solves the k-means problem (line 6) and then converts the centers back to the original space (line 7). Here, kmeans(S', w, k) denotes a (centralized) k-means algorithm that returns the k-means centers for the data points in S' with weights w, and π_1^{-1} denotes an inverse of the JL projection π_1. We note that the inverse of π_1 is generally not unique as π_1 is noninvertible, but our analysis holds for any inverse (e.g., Moore-Penrose inverse). The following theorem quantifies the performance of Algorithm 1 (proved in Appendix A).

Theorem 4.2. *For any $\epsilon, \delta \in (0,1)$, if in Algorithm 1, π_1 satisfies Lemma 4.1, π_2 generates an ϵ-coreset with probability at least $1-\delta$, and kmeans(S', w, k) returns the optimal k-means centers of the dataset S' with weights w, then*

1) *the output X is a $(1+\epsilon)^5/(1-\epsilon)$-approximation for k-means clustering of P with probability at least $(1-\delta)^2$,*
2) *the communication cost is $O\left(k\epsilon^{-4}\log n\right)$, and*
3) *the complexity at the data source is $\tilde{O}\left(nd\epsilon^{-2}\right)$,*

assuming $\min(n, d) \gg k, 1/\epsilon$, and $1/\delta$.

Remark: We only focus on the complexity at the data source as the server is usually much more powerful. Theorem 4.2 shows that Algorithm 1 can solve k-means arbitrarily close to the optimal with an arbitrarily high probability, while incurring a complexity at the data source that is roughly linear in the data size (i.e., nd) and a communication cost that is roughly logarithmic in the data cardinality n.

4.2 CR+DR

While Algorithm 1 can reduce the communication cost without incurring much computation at the data source, it remains unclear whether its order of applying DR and CR is optimal. To this end, we consider applying CR first.

We again choose JL projection as the DR method to avoid transmitting the projection matrix at runtime, and choose FSS as the CR method as it generates an ϵ-coreset with the minimum cardinality among the existing CR methods for k-means. The algorithm, shown in Algorithm 2, differs from Algorithm 1 in that the order of applying DR and CR is reversed. That is, the data source first applies FSS (line 2) and then applies JL projection (line 3) to compute a dimension-reduced coreset (S', Δ, w), based on which the server computes a set of k-means centers X in the same way as Algorithm 1.

We now analyze the performance of Algorithm 2, starting with a counterpart of Lemma 4.1 (proved in Appendix A).

Lemma 4.2. *Let $\pi : \mathbb{R}^d \to \mathbb{R}^{d'}$ be a JL projection. Then there exists $d' = O(\epsilon^{-2}\log(n'k/\delta))$ such that for any coreset $\mathbf{S} := (S, \Delta, w)$, where $S \subset \mathbb{R}^d$ with $|S| = n'$, $w : S \to \mathbb{R}$, and $\Delta \in \mathbb{R}$, and any $X, X^* \subset \mathbb{R}^d$ with $|X| = |X^*| = k$, the following holds with probability at least $1-\delta$:*

$$cost(\mathbf{S}, X) \approx_{(1+\epsilon)^2} cost((\pi(S), \Delta, w), \pi(X)), \tag{7}$$

$$cost(\mathbf{S}, X^*) \approx_{(1+\epsilon)^2} cost((\pi(S), \Delta, w), \pi(X^*)). \tag{8}$$

Below, we will show that Algorithm 2 achieves the same approximation error as Algorithm 1, but at different cost and complexity (proved in Appendix A).

Theorem 4.3. *For any $\epsilon, \delta \in (0,1)$, if in Algorithm 2, π_1 satisfies Lemma 4.2, π_2 generates an ϵ-coreset with probability at least $1-\delta$, and kmeans(S', w, k) returns the optimal k-means centers of the dataset S' with weights w, then*

1) *the output X is an $(1+\epsilon)^5/(1-\epsilon)$-approximation for k-means clustering of P with probability $\geq (1-\delta)^2$,*
2) *the communication cost is $\tilde{O}(k^3/\epsilon^6)$, and*
3) *the complexity at the data source is $O\left(nd \cdot \min(n, d)\right)$,*

assuming $\min(n, d) \gg k, 1/\epsilon$, and $1/\delta$.

Algorithm 3: Communication-efficient k-Means under DR+CR+DR

input : Original dataset P, number of centers k, JL projection $\pi_1^{(1)}$ for P, FSS-based CR method π_2, JL projection $\pi_1^{(2)}$ for the output of π_2
output: Centers for k-means clustering of P

1 **data source:**
2 $P' \leftarrow \pi_1^{(1)}(P)$;
3 $(S, \Delta, w) \leftarrow \pi_2(P')$;
4 $S' \leftarrow \pi_1^{(2)}(S)$;
5 report (S', Δ, w) to the server;
6 **server:**
7 $X' \leftarrow \text{kmeans}(S', w, k)$;
8 $X \leftarrow (\pi_1^{(2)} \circ \pi_1^{(1)})^{-1}(X')$;
9 return X;

4.3 Repeated DR/CR

Theorems 4.2 and 4.3 state that to achieve the same approximation error with the same probability, DR+CR (Algorithm 1) incurs a communication cost of $O(k\epsilon^{-4}\log n)$ and a complexity of $\tilde{O}(\epsilon^{-2}nd)$, while CR+DR (Algorithm 2) incurs a communication cost of $\tilde{O}(k^3\epsilon^{-6})$ and a complexity of $O(nd \cdot \min(n, d))$. This shows a *communication-computation tradeoff*: the approach of DR+CR incurs a linear complexity and a logarithmic communication cost, whereas the approach of CR+DR incurs a super-linear complexity (which is still less than quadratic) and a constant communication cost. One may wonder whether it is possible to combine the strengths of both of the algorithms. Below we give an affirmative answer by applying some of these repeatedly.

We know from Theorem 3.2 that applying FSS once already reduces the cardinality to a constant (in n and d), and hence there is no need to repeat FSS. The same theorem also implies that if we apply FSS first, we will incur a super-linear complexity, and hence we need to apply JL projection before FSS. Meanwhile, we see from Lemmas 4.1 and 4.2 that applying JL projection on a dataset of cardinality n' can reduce its dimension to $O(\epsilon^{-2}\log(n'k/\delta))$ while achieving a $(1 + O(\epsilon))$-approximation with high probability. Thus, we can further reduce the dimension by applying JL projection again after reducing the cardinality by FSS. The above reasoning suggests a three-step procedure: JL→FSS→JL, presented in Algorithm 3. The data source applies JL projection both before and after FSS (lines 2 and 4), where $\pi_1^{(1)}$ projects from $\mathbb{R}^d$ to $\mathbb{R}^{O(\log n/\epsilon^2)}$, and $\pi_1^{(2)}$ projects from $\mathbb{R}^{O(\log n/\epsilon^2)}$ to $\mathbb{R}^{O(\log |S|/\epsilon^2)}$. The server first computes the k-means centers in the twice-protected space, and then converts them back to the original space (line 8). Note that by convention, $\pi_1^{(2)} \circ \pi_1^{(1)}(X)$ means $\pi_1^{(2)}(\pi_1^{(1)}(X))$.

Below, we will show that this seemingly small change is able to combine the low communication cost of Algorithm 2 and the low complexity of Algorithm 1, at a small increase in the approximation error; see Appendix A for the proof.

Theorem 4.4. *For any $\epsilon, \delta \in (0, 1)$, if in Algorithm 3, $\pi_1^{(1)}$ satisfies Lemma 4.1, $\pi_1^{(2)}$ satisfies Lemma 4.2, π_2 generates an ϵ-coreset of its input dataset with probability at least $1 - \delta$, and kmeans(S', w, k) returns the optimal k-means centers of the dataset S' with weights w, then*

1) *the output X is a $(1 + \epsilon)^9/(1 - \epsilon)$-approximation for k-means clustering of P with probability $\geq (1 - \delta)^3$,*
2) *the communication cost is $\tilde{O}(k^3/\epsilon^6)$, and*
3) *the complexity at the data source is $\tilde{O}(nd/\epsilon^2)$,*

assuming $\min(n, d) \gg k, 1/\epsilon$, and $1/\delta$.

Remark: Theorem 4.4 implies that Algorithm 3 is essentially "optimal" in the sense that it achieves a $(1 + O(\epsilon))$-approximation with an arbitrarily high probability, at a near-linear complexity and a constant communication cost at the data source. Thus, no qualitative improvement will be achieved by applying further DR/CR methods.

5 Joint DR and CR across Multiple Data Sources

Consider the scenario where the dataset P is split across m data sources ($m \geq 2$). Let P_i denote the dataset at data source i and n_i be its cardinality. As shown below, the previous algorithms can be adapted to the distributed setting.

5.1 Distributed Version of FSS

It turns out that the state-of-the-art distributed DR and CR algorithm, proposed in [27, Algorithm 1], is exactly a distributed version of FSS, referred to as *BKLW* following the authors' last names. As in FSS, BKLW first uses PCA to reduce the intrinsic dimension of the dataset and then applies sensitivity sampling. However, it uses distributed algorithms to perform these steps.

For distributed PCA, BKLW applies an algorithm *disPCA* from [11] (formalized in Algorithm 1 in [35]), where:

1) each data source i ($i = 1, \ldots, m$) computes local SVD $A_{P_i} = U_i\Sigma_i V_i^T$, and sends $\Sigma_i^{(t_1)}$ and $V_i^{(t_1)}$ to the server ($\Sigma_i^{(t_1)}$ and $V_i^{(t_1)}$ contain the first t_1 columns of Σ_i and V_i, respectively);
2) the server constructs $Y^T = [Y_1^T, \ldots, Y_m^T]$, with $Y_i = \Sigma_i^{(t_1)}(V_i^{(t_1)})^T$, computes a global SVD $Y = U\Sigma V^T$;
3) the first t_2 columns of V are returned as an approximate solution to the PCA of $\bigcup_{i=1}^m P_i$.

For distributed sensitivity sampling, BKLW applies an algorithm *disSS* from [4] (Algorithm 1), where:

1) each data source i ($i = 1, \ldots, m$) computes a bicriteria approximation X_i for P_i and reports $\text{cost}(P_i, X_i)$;
2) the server allocates a global sample size s to each data source proportionally to its cost, i.e., $s_i = s \cdot \text{cost}(P_i, X_i)/\big(\sum_{j=1}^m \text{cost}(P_j, X_j)\big)$;
3) each data source i draws s_i i.i.d. samples S_i from P_i with probability proportional to $\text{cost}(\{p\}, X_i)$, and reports $S_i \cup X_i$ with their weights $w : S_i \cup X_i \to \mathbb{R}$, that are set to match the number of points per cluster;
4) the union of the reported sets $(\bigcup_{i=1}^m (S_i \cup X_i), 0, w)$ is returned as a coreset of $\bigcup_{i=1}^m P_i$.

BKLW first applies disPCA, followed by disSS with $s = O(\epsilon^{-4}(k^2/\epsilon^2 + \log(1/\delta)) + mk\log(mk/\delta))$ to the dimension-reduced dataset $\{A_{P_i}V^{(t_2)}(V^{(t_2)})^T\}_{i=1}^m$ to compute a coreset $(S, 0, w)$ at the server. Finally, the server computes the

optimal k-means centers X on $(S, 0, w)$ and returns it as an approximation to the optimal k-means centers of $\bigcup_{i=1}^m P_i$.

Although a theorem was given in [27] without proof on the performance of BKLW, the result is imprecise and incomplete. Here, we provide the complete analysis to facilitate later comparison. We will leverage the following results.

Theorem 5.1 ([35]). *For any $\epsilon \in (0, 1/3)$, let $t_1 = t_2 \geq k + \lceil 4k/\epsilon^2 \rceil - 1$ in disPCA and $\tilde{P}_i$ be the projected dataset at data source i (i.e., the set of rows of $A_{P_i} V^{(t_2)} (V^{(t_2)})^T$). Then there exists a constant $\Delta \geq 0$ such that for any set $X \subset \mathbb{R}^d$ with $|X| = k$,*

$$(1-\epsilon)cost(P, X) \leq cost(\tilde{P}, X)+\Delta \leq (1+\epsilon)cost(P, X), \quad (9)$$

where $P := \bigcup_{i=1}^m P_i$ and $\tilde{P} := \bigcup_{i=1}^m \tilde{P}_i$.

Theorem 5.2 ([4]). *For a distributed dataset $\{P_i\}_{i=1}^m$ with $P_i \subset \mathbb{R}^d$ and any $\epsilon, \delta \in (0, 1)$, with probability at least $1-\delta$, the output $(S, 0, w)$ of disSS is an ϵ-coreset of $\bigcup_{i=1}^m P_i$ of size*

$$|S| = O\left(\frac{1}{\epsilon^4}\Big(kd + \log(\frac{1}{\delta})\Big) + mk\log(\frac{mk}{\delta})\right). \quad (10)$$

Theorems 5.1 and 5.2 bound the performance of disPCA and disSS, respectively, based on which we have the following results for BKLW (see proof in Appendix A).

Theorem 5.3. *For any $\epsilon \in (0, 1/3)$ and $\delta \in (0, 1)$, suppose that in BKLW, disPCA satisfies Theorem 5.1 for the input dataset $\{P_i\}_{i=1}^m$ and disSS satisfies Theorem 5.2 for the input dataset $\{\tilde{P}_i\}_{i=1}^m$. Then*

1) *the output X is a $(1+\epsilon)^2/(1-\epsilon)^2$-approximation for k-means clustering of $\bigcup_{i=1}^m P_i$ with probability $\geq 1-\delta$,*
2) *the total communication cost over all the data sources is $O(mkd/\epsilon^2)$, and*
3) *the complexity at each data source is $O(nd \cdot \min(n, d))$,*

assuming $\min(n, d) \gg m, k, 1/\epsilon$, and $1/\delta$.

5.2 Enhancements

It is easy to see that each data source can apply JL projection independently at no additional communication cost. Following the ideas in Algorithms 1 and 2, we wonder: (i) Can we improve BKLW by combining it with JL projection? (ii) Is there an optimal order of applying BKLW and JL projection?

We first consider applying JL projection before invoking BKLW. For consistency with Algorithm 1, we only use the first two steps of BKLW, i.e., disPCA and disSS, that construct a coreset, which we refer to as a *BKLW-based CR method*. The algorithm, shown in Algorithm 4, is essentially the distributed counterpart of Algorithm 1. First, each source independently applies JL projection to its local dataset (line 2). Then, the sources cooperatively run BKLW, i.e., disPCA + disSS (line 3). Finally, the server uses the received dimension-reduced coreset to solve k-means and converts the centers back to the original space (lines 5–6).

We now analyze the performance of Algorithm 4, starting from a coreset-like property of the BKLW-based CR method π_2 (see proof in Appendix A).

Lemma 5.1. *Let $P := \bigcup_{i=1}^m P_i$ be the union of the input datasets for the BKLW-based CR method π_2 and $\mathbf{S} := (S, 0, w)$ be the resulting coreset reported to the server. For any $\epsilon \in (0, 1/3)$ and $\delta \in (0, 1)$, $\exists t_1 = t_2 = O(k/\epsilon^2)$, $s = O(\epsilon^{-4}(k^2/\epsilon^2 + \log(1/\delta)) + mk\log(mk/\delta))$, and $\Delta \geq 0$, such that with probability at least $1-\delta$, π_2 with parameters t_1, t_2, and s satisfies*

$$(1-\epsilon)^2 cost(P, X) \leq cost(\mathbf{S}, X)+\Delta \leq (1+\epsilon)^2 cost(P, X) \quad (11)$$

for any set X of k points in the same space as P.

Remark: Comparing Lemma 5.1 with Definition 3.2, we see that π_2 does not construct an $O(\epsilon)$-coreset of its input dataset. Nevertheless, its output can approximate the k-means cost of the input dataset up to a constant shift, which is sufficient for computing approximate k-means.

We have the following performance guarantee for Algorithm 4 (see proof in Appendix A).

Theorem 5.4. *For any $\epsilon \in (0, 1/3)$ and $\delta \in (0, 1)$, suppose that in Algorithm 4, π_1 satisfies Lemma 4.1, π_2 satisfies Lemma 5.1, and kmeans$(\bigcup_{i=1}^m S_i', w, k)$ returns the optimal k-means centers of the dataset $\bigcup_{i=1}^m S_i'$ with weights w. Then*

1) *the output X is a $(1+\epsilon)^6/(1-\epsilon)^2$-approximation for k-means clustering of $\bigcup_{i=1}^m P_i$ with probability $\geq (1-\delta)^2$,*
2) *the total communication cost over all the data sources is $O(mk\epsilon^{-4}\log n)$, and*
3) *the complexity at each data source is $\tilde{O}(nd\epsilon^{-4})$,*

assuming $\min(n, d) \gg m, k, 1/\epsilon$, and $1/\delta$.

Discussion: Comparing Theorems 5.4 and 5.3, we see that for $d \gg \log n$ (e.g., $d = \Omega(n)$), Algorithm 4 can significantly reduce the communication cost and the complexity at data sources, while achieving a similar $(1+O(\epsilon))$-approximation as BKLW. Note that although the possibility of applying another DR method before BKLW was mentioned in [27], no result was given there.

Meanwhile, although one could develop a distributed counterpart of Algorithm 2 that applies JL projection after BKLW, its performance will not be competitive. Specifically, using similar analysis, this approach incurs the same order of communication cost and complexity as BKLW. Meanwhile, the JL projection introduces additional error, causing its overall approximation error to be larger. It is thus unnecessary to consider this algorithm.

Furthermore, we note that repeated application of DR/CR is unnecessary in the distributed setting. This is because after one round of BKLW (with or without applying JL projection beforehand), we already reduce the cardinality to $O(\epsilon^{-4}(k^2/\epsilon^2 + \log(1/\delta)) + mk\log(mk/\delta))$

Algorithm 4: Communication-efficient Distributed k-Means under DR+CR

input : Distributed dataset $\{P_i\}_{i=1}^m$, number of centers k, JL projection π_1, BKLW-based CR method π_2
output: Centers for k-means clustering of P

1 **each data source** i **(**$i = 1, \ldots, m$**):**
2 | $P_i' \leftarrow \pi_1(P_i)$;
3 run π_2 on the distributed dataset $\{P_i'\}_{i=1}^m$, which results in each data source i reporting a local coreset $(S_i', 0, w)$ to the server;
4 **server:**
5 | $X' \leftarrow$ kmeans$(\bigcup_{i=1}^m S_i', w, k)$;
6 | $X \leftarrow \pi_1^{-1}(X')$;
7 | return X;

TABLE 2
Summary of Comparison

Algorithm	Communication cost	Computational complexity
FSS [11]	$O(kd/\epsilon_2^2)$	$O(nd \cdot \min(n,d))$
JL + FSS (Alg. 1)	$O(k \log n/\epsilon_1^4)$	$\tilde{O}(nd/\epsilon_1^2)$
FSS + JL (Alg. 2)	$\tilde{O}(k^3/\epsilon_1^6)$	$O(nd \cdot \min(n,d))$
JL + FSS + JL (Alg. 3)	$\tilde{O}(k^3/\epsilon_3^6)$	$\tilde{O}(nd/\epsilon_3^2)$
BKLW [27]	$O(mkd/\epsilon_4^2)$	$O(nd \cdot \min(n,d))$
JL + BKLW (Alg. 4)	$O(mk \log n/\epsilon_5^4)$	$\tilde{O}(nd/\epsilon_5^4)$

and the dimension to $O(k/\epsilon^2)$, both constant in the size (n, d) of the original dataset. Meanwhile, this round incurs a communication cost that scales with (n, d) as $O(\log n)$ (with JL projection) or $O(d)$ (without JL projection), and a complexity that scales as $\tilde{O}(nd)$ (with JL projection) or $O(nd \cdot \min(n, d))$ (without JL projection). Therefore, any possible reduction in the cost (or the complexity) achieved by further reducing the cardinality or dimension will be dominated by the cost (or the complexity) in the first round. Hence, repeated application of DR/CR will not improve the order of the communication cost or the complexity.

5.3 Summary of Comparison

We are now ready to compare the performances of all the proposed algorithms and their best existing counterparts in both centralized (i.e., single data source) and distributed (i.e., multiple data sources) settings.

To ensure the same approximation error for all the algorithms, we set the error parameter 'ϵ' to ϵ_1 for Algorithms 1 and 2, ϵ_2 for FSS, ϵ_3 for Algorithm 3, ϵ_4 for BKLW, and ϵ_5 for Algorithm 4, where for any $\epsilon \in (0, 1)$, ϵ_1 satisfies $(1+\epsilon_1)^5/(1-\epsilon_1) = 1+\epsilon$, ϵ_2 satisfies $(1+\epsilon_2)/(1-\epsilon_2) = 1+\epsilon$, ϵ_3 satisfies $(1+\epsilon_3)^9/(1-\epsilon_3) = 1+\epsilon$, ϵ_4 satisfies $(1+\epsilon_4)^2/(1-\epsilon_4)^2 = 1+\epsilon$, and ϵ_5 satisfies $(1+\epsilon_5)^6/(1-\epsilon_5)^2 = 1+\epsilon$.

The comparison, summarized in Table 2, is in terms of the communication cost and the complexity at the data source(s) for achieving a $(1+\epsilon)$-approximation for k-means clustering of an input dataset of cardinality n and dimension d, where the first four rows are for the centralized setting and the last two rows are for the distributed setting. Our focus is on the scaling with n and d, which are assumed to dominate the other parameters (i.e., k, m, $1/\epsilon_i$). Clearly, for high-dimensional datasets satisfying $d \gg \log n$, the best proposed algorithms (Algorithm 3 and Algorithm 4) significantly outperforms the best existing algorithms (FSS and BKLW) in both centralized and distributed settings.

6 Extension to Joint DR, CR, and QT

Besides cardinality and dimensionality, the volume of a dataset also depends on its *precision*, defined as the number of bits used to represent each attribute in the dataset. While DR and CR methods can be used to reduce the dimensionality and the cardinality, quantization techniques [12] can be used to reduce the precision and hence further reduce the communication cost. While the optimal efficient quantization in support of k-means is worth a separate study, our focus here is on properly combining a given quantizer with the proposed DR/CR methods. To this end, we will use a simple rounding-based quantizer as a concrete example.

6.1 Rounding-based Quantization

Given a scalar $x \in \mathbb{R}$, we denote the b_0-bit binary floating number representation of x by

$$x = (-1)^{\text{sign}(x)} \times 2^{e_x} \times (a(0) + a(1) \times 2^{-1} + \ldots + a(b_0 - 1 - m_e) \times 2^{-(b_0-1-m_e)}), \quad (12)$$

where $\text{sign}(x) = 0$ if $x \geq 0$ and $\text{sign}(x) = 1$ if $x < 0$, m_e is the number of exponent bits, e_x is the m_e-bit exponent of x, and $a(\cdot) \in \{0, 1\}$ are the significant bits ($a(0) \equiv 1$). The rounding-based quantizer Γ with s significant bits is

$$\Gamma(x) := (-1)^{\text{sign}(x)} \times 2^{e_x} \times (a(0) + a(1) \times 2^{-1} + \ldots + a(s) \times 2^{-s} + a'(s) \times 2^{-s}), \quad (13)$$

where $a'(s)$ is the result of rounding the remaining bits (0: rounding down; 1: rounding up).

For simplicity of notation, we also use $p' := \Gamma(p)$ to denote the element-wise rounding-based quantization of a data point $p = (p_i)_{i=1}^d \in \mathbb{R}^d$. Defining the maximum quantization error as $\Delta_{QT} := \max_{p \in P} \|p - p'\|$, we know that by the definition of rounding-based quantizer, the quantization error in each element satisfies $|p_i - p_i'| \leq 2^{e_{p_i} - s} \leq |p_i| 2^{-s}$ since $|p_i| \geq 2^{e_{p_i}}$. Therefore, the maximum quantization error is bounded as

$$\Delta_{QT} = \max_{p \in P} \sqrt{\sum_{i=1}^d (p_i - p_i')^2} \leq \max_{p \in P} \sqrt{\sum_{i=1}^d 2^{-2s} p_i^2} = 2^{-s} \max_{p \in P} \|p\|. \quad (14)$$

6.2 Approximation Error Analysis

We now analyze the performance after adding quantization to the proposed communication-efficient k-means algorithms. As DR and CR can generate data points of arbitrary values that may not be representable with a given number of significant bits, we add quantization after all the DR/CR steps. That is, we assume that right before a data source reports its dimension-reduced coreset (S, Δ, w) to the server, it will apply the rounding-based quantizer Γ and report (S_{QT}, Δ, w) instead[6], where $S_{QT} := \{\Gamma(p) : p \in S\}$. Obviously, the quantization further reduces the communication cost. It also incurs a computational complexity that is linear in the size of S, which is sub-linear in the size of the original

6. Here we only apply quantization to the coreset points in S as their transfer dominates the communication cost, but our approach can be extended to other cases.

dataset (i.e., nd) and thus subsumed by the complexity of DR/CR (as shown in Table 2). The only performance metric it can negatively impact is the approximation error, which is analyzed in the following theorem (see proof in Appendix A).

Theorem 6.1. *Let X denote the optimal k-means centers computed by the server based on the received coreset, X^* denote the optimal k-means centers based on the original dataset, and Δ_D denote the diameter of the input space.*

1) *In Algorithm 1, suppose that π_1 satisfies Lemma 4.1 with ϵ_1, π_2 generates an ϵ_2-coreset with probability $\geq 1-\delta$, and π_{QT} is a quantizer with maximum error Δ_{QT}. If we update Line 4 to: $S'_{QT} \leftarrow \pi_{QT}(S')$ and report (S'_{QT}, Δ, w) to the server, then the approximation error will be*

$$cost(P,X) \leq \frac{(1+\epsilon_1)^4(1+\epsilon_2)}{(1-\epsilon_2)} cost(P,X^*) + \frac{(1+\epsilon_1)^2}{(1-\epsilon_2)} 4n\Delta_D\Delta_{QT} \quad (15)$$

with probability at least $(1-\delta)^2$.

2) *In Algorithm 2, suppose that π_1 satisfies Lemma 4.2 with ϵ_1, π_2 generates an ϵ_2-coreset with probability $\geq 1-\delta$, and π_{QT} is a quantizer with maximum error Δ_{QT}. If we update Line 4 to: $S'_{QT} \leftarrow \pi_{QT}(S')$ and report (S'_{QT}, Δ, w) to the server, then the approximation error will be*

$$cost(P,X) \leq \frac{(1+\epsilon_1)^4(1+\epsilon_2)}{(1-\epsilon_2)} cost(P,X^*) + \frac{(1+\epsilon_1)^2}{(1-\epsilon_2)} 4n\Delta_D\Delta_{QT} \quad (16)$$

with probability at least $(1-\delta)^2$.

3) *In Algorithm 3, suppose that $\pi_1^{(1)}$ satisfies Lemma 4.1 with $\epsilon_1^{(1)}$, $\pi_1^{(2)}$ satisfies Lemma 4.2 with $\epsilon_1^{(2)}$, π_2 generates an ϵ_2-coreset with probability $\geq 1-\delta$, and π_{QT} is a quantizer with maximum error Δ_{QT}. If we update Line 5 to: $S'_{QT} \leftarrow \pi_{QT}(S')$ and report (S'_{QT}, Δ, w) to the server, then the approximation error will be*

$$cost(P,X) \leq \frac{(1+\epsilon_1^{(1)})^4(1+\epsilon_2)(1+\epsilon_1^{(2)})^4}{(1-\epsilon_2)} cost(P,X^*) + \frac{(1+\epsilon_1^{(1)})^2(1+\epsilon_1^{(2)})^2}{(1-\epsilon_2)} 4n\Delta_D\Delta_{QT} \quad (17)$$

with probability at least $(1-\delta)^3$.

4) *In Algorithm 4, suppose that π_1 satisfies Lemma 4.1, π_2 satisfies Lemma 5.1, and π_{QT} is a quantizer with maximum error Δ_{QT}. If we update Line 3 to: run π_2 on the distributed dataset $\{P'_i\}_{i=1}^m$ to compute a local coreset $(S'_i, 0, w)$ at each data source i and report $(\pi_{QT}(S'_i), 0, w)$ to the server, then the approximation error will be*

$$cost(P,X) \leq \frac{(1+\epsilon_1)^4(1+\epsilon_2)^2}{(1-\epsilon_2)^2} cost(P,X^*) + \frac{(1+\epsilon_1)^2}{(1-\epsilon_2)^2} 4n\Delta_D\Delta_{QT} \quad (18)$$

with probability at least $(1-\delta)^2$.

6.3 Configuration of Joint DR, CR, and QT

Based on the analysis of the approximation error under given DR, CR, and QT (quantization) methods, we aim to answer the following question: how can we configure the DR, CR, and QT methods such that we can minimize the communication cost while keeping the approximation error within a given bound? We will present a detailed solution for the four-step procedure JL+FSS+JL+QT, as the solutions for the other procedures are similar.

6.3.1 Problem Formulation

Let $\mathcal{Y}_0$ denote a desired bound on the approximation error and $1-\delta_0$ the desired confidence level, i.e., $\text{cost}(P,X) \leq \mathcal{Y}_0 \text{cost}(P,X^*)$ with probability $\geq 1-\delta_0$, where X is the computed k-means solution and X^* the optimal solution. By Theorem 6.1, the QT step introduces an additive error. We now convert it into a multiplicative error to enforce the bound $\mathcal{Y}_0$. To this end, suppose that we are given a lower bound $\mathcal{E}$ on the optimal k-means cost $\text{cost}(P,X^*)$. For example, by [36], we can estimate $\mathcal{E}$ by selecting $O(k)$ points from P according to a certain probability distribution, repeating this process for $\log(1/\delta)$ times, and outputting the set X of selected points with the minimum $\text{cost}(P,X)$. This result is proven to be at most 20-time worse than the optimal solution, i.e., $\mathcal{E} := \text{cost}(P,X)/20 \leq \text{cost}(P,X^*)$, with probability at least $1-\delta$. Define $\epsilon_{QT} := \frac{4n\Delta_D\Delta_{QT}}{\mathcal{E}}$. Then based on (17), we have:

$$\begin{aligned}
&\text{cost}(P,X) \quad (19)\\
&\leq \frac{(1+\epsilon_1^{(1)})^4(1+\epsilon_2)(1+\epsilon_1^{(2)})^4}{(1-\epsilon_2)} \text{cost}(P,X^*) + \frac{(1+\epsilon_1^{(1)})^2(1+\epsilon_1^{(2)})^2}{(1-\epsilon_2)} 4n\Delta_D\Delta_{QT}\\
&\leq \frac{(1+\epsilon_1^{(1)})^4(1+\epsilon_2)(1+\epsilon_1^{(2)})^4}{(1-\epsilon_2)} \text{cost}(P,X^*) + \frac{(1+\epsilon_1^{(1)})^2(1+\epsilon_1^{(2)})^2}{(1-\epsilon_2)} \frac{4n\Delta_D\Delta_{QT}}{\mathcal{E}} \text{cost}(P,X^*)\\
&= \frac{(1+\epsilon_1^{(1)})^2(1+\epsilon_1^{(2)})^2}{(1-\epsilon_2)} \times ((1+\epsilon_1^{(1)})^2(1+\epsilon_2)(1+\epsilon_1^{(2)})^2 + \epsilon_{QT}) \text{cost}(P,X^*). \quad (20)
\end{aligned}$$

Let $f(\epsilon_1^{(1)}, \epsilon_2, \epsilon_1^{(2)}, \epsilon_{QT})$ denote the communication cost as a function of the configuration parameters $\epsilon_1^{(1)}$, ϵ_2, $\epsilon_1^{(2)}$, and ϵ_{QT}. Our goal is to find the optimal configuration that minimizes the communication cost while satisfying the given bound on the approximation error:

$$\min_{\epsilon_1^{(1)}, \epsilon_2, \epsilon_1^{(2)}, \epsilon_{QT}} \quad \mathcal{X} := f(\epsilon_1^{(1)}, \epsilon_2, \epsilon_1^{(2)}, \epsilon_{QT}) \quad (21a)$$

$$\text{s.t.} \quad \mathcal{Y} := \frac{(1+\epsilon_1^{(1)})^2(1+\epsilon_1^{(2)})^2}{(1-\epsilon_2)} \times ((1+\epsilon_1^{(1)})^2(1+\epsilon_2)(1+\epsilon_1^{(2)})^2 + \epsilon_{QT}) \leq \mathcal{Y}_0, \quad (21b)$$

where $\mathcal{X}$ denotes the communication cost and $\mathcal{Y}$ denotes (an upper bound on) the approximation error. The parameter δ is set to $1-(1-\delta_0)^{1/3}$ such that the desired confidence level is satisfied.

6.3.2 Analysis

The communication cost $\mathcal{X}$ is dominated by the transfer of the dimension-reduced, quantized coreset S'_{QT}. Let its cardinality, dimensionality, and precision be n', d', and b'. By [11], the cardinality of an ϵ_2-coreset generated by FSS is $n' = O(\frac{k^3 \log^2(k) \log(1/\delta)}{\epsilon_2^4})$. To satisfy Lemma 4.1 with $\epsilon_1^{(2)}$, the dimensionality needs to satisfy $d' = O(\frac{\log(n'k/\delta)}{(\epsilon_1^{(2)})^2})$. By the analysis of quantization error in Section 6.1, $b' = O(\log(\frac{n\sqrt{d}}{\epsilon_{QT}}))$. Denoting the constant factors in these big-O terms by C_1, C_2, and C_3, we have

$$\mathcal{X} \approx n' \cdot d' \cdot b' \tag{22}$$

$$= C_1 \frac{k^3 \log^2(k) \log(1/\delta)}{\epsilon_2^4} \cdot C_2 \frac{\log(n'k/\delta)}{(\epsilon_1^{(2)})^2} \cdot C_3 \log\left(\frac{n\sqrt{d}}{\epsilon_{QT}}\right) \tag{23}$$

$$= \tilde{C_1} \cdot \frac{\log(\tilde{C_2}/\epsilon_2^4)}{\epsilon_2^4 (\epsilon_1^{(2)})^2} \cdot \log\left(\frac{\tilde{C_3}}{\frac{\mathcal{Y}_0(1-\epsilon_2)}{(1+\epsilon_1^{(1)})^2(1+\epsilon_1^{(2)})^2} - (1+\epsilon_1^{(1)})^2(1+\epsilon_2)(1+\epsilon_1^{(2)})^2}\right), \tag{24}$$

where $\tilde{C_1} = k^3 \log^2(k) \log(\frac{1}{\delta}) C_1 C_2 C_3$, $\tilde{C_2} = \frac{k^4 \log^2(k) \log(\frac{1}{\delta})}{\delta}$, and $\tilde{C_3} = n\sqrt{d}$. Assuming $k \geq 2$, by plugging the constant factors from [23], [37], [38] into Theorem 36 from [11], we can use $C_1 = 54912(1+\log_2(3))(1+\log_2(26/3))/225$. By JL projection, $d' \leq \lceil 8 * \log(\frac{4n'k}{\delta})/\epsilon^2 \rceil$, and therefore C_2 could be 24. Assuming $n > 8$ and $\mathcal{E} \geq 1$, C_3 could be 2. Equation (24) is obtained by replacing ϵ_{QT} by its maximum value derived from (21b).

While solving (21) in the general case is nontrivial, we note that for the rounding-based quantizer defined in Section 6.1, ϵ_{QT} only has a finite number of possible values, corresponding to the number of significant bits $s = 1, \ldots, b_0 - 1 - m_e$. Thus, under the simplifying constraint of $\epsilon_1^{(1)} = \epsilon_2 = \epsilon_1^{(2)} =: \epsilon$, we can enumerate each possible value of ϵ_{QT}, compute the maximum ϵ under this ϵ_{QT} from (21b), and plug it into (24) to evaluate $\mathcal{X}$. We can then select the configuration that yields the minimum $\mathcal{X}$.

7 Performance Evaluation

We now use experiments on real datasets to validate our analysis about the proposed joint DR, CR and QT algorithms in comparison with the state of the art in both the single-source and the multiple-source cases.

7.1 Datasets, Metrics, and Test Environment

We use two datasets: (1) MNIST training dataset [39], a handwritten digits dataset which has $60,000$ images in 784-dimensional space; (2) NeurIPS Conference Papers 1987-2015 dataset [40], a word counts dataset of the NeurIPS conference papers published from 1987 to 2015, which has $11,463$ instances (words) with $5,812$ attributes (papers). Both of these two datasets are normalized to $[-1, 1]$ with zero mean. In the case of multiple data sources, we randomly partition each dataset among 10 data sources.

We measure the performance by (i) the approximation error, measured by the normalized k-means cost $\text{cost}(P, X)/\text{cost}(P, X^*)$, where X is the set of centers returned by the evaluated algorithm and X^* is the set of centers computed from P, (ii) the normalized communication cost, measured by the ratio between the number of bits transmitted by the data source(s) and the size of P, and (iii) the complexity at the data source(s), measured by the running time of the evaluated DR/CR algorithm. We set $k = 2$ in all the experiments. Because of the randomness of the algorithms, we repeat each test for 10 Monte Carlo runs[7].

We run an edge-based machine learning system in a simulated environment and consider both cases of single and multiple data sources. All the experiments are conducted on a Windows machine with Intel i7-8700 CPU and 48GB DDR4 memory. We note that our simulated results closely resemble those in an actual distributed system, as the performance metrics we measure are either independent of the test environment (approximation error and communication cost) or only dependent through a scaling factor (running time). Although the absolute value of the running time depends on the processor speed at the data source, different processor speeds will only cause different scaling factors, and thus the running times obtained in our experiments can still be used to compare the complexities between algorithms.

7.2 Results for Joint DR and CR

7.2.1 Evaluated Algorithms

In the case of a single data source, we evaluate the following algorithms:

- "FSS": the benchmark algorithm introduced in [11],
- "JL+FSS": Algorithm 1, where we use JL projection before applying FSS,
- "FSS+JL": Algorithm 2, where we use JL projection after applying FSS, and
- "JL+FSS+JL": Algorithm 3, where we apply JL projection both before and after FSS.

In the case of multiple data sources, we evaluate the following algorithms:

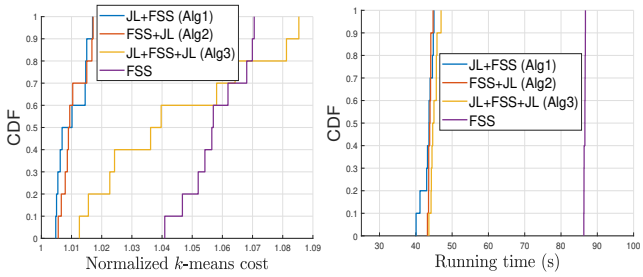
- "BKLW": the benchmark algorithm from [27], and
- "JL+BKLW": Algorithm 4, where we apply JL projection before BKLW.

In both cases, we have tuned the parameters of both the benchmark and proposed algorithms to make all the algorithms achieve a similar empirical approximation error. As a baseline, we also include the naive method of "no reduction (NR)", i.e., transmitting the raw data.

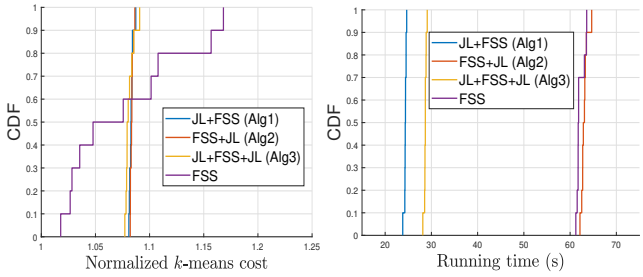
7.2.2 Results

In the case of a single data source, the data source computes and reports a data summary using the evaluated DR/CR algorithms, based on which a server computes k-means centers. The results are given in Figure 1 and Table 3. Note that by definition, the baseline (NR) has a normalized k-means cost of 1, a normalized communication cost of 1,

7. The number of Monte Carlo runs is limited by the running time of the experiment, which takes around 30 hours to complete one Monte Carlo run for all the algorithms and all the settings in Section 7.3.

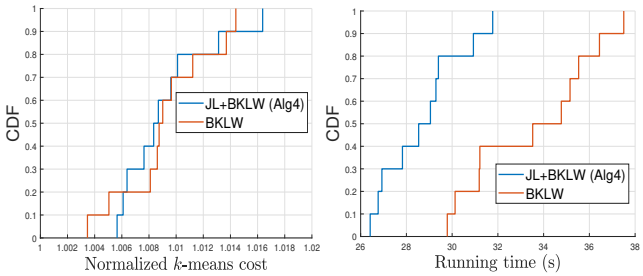


(a) MNIST

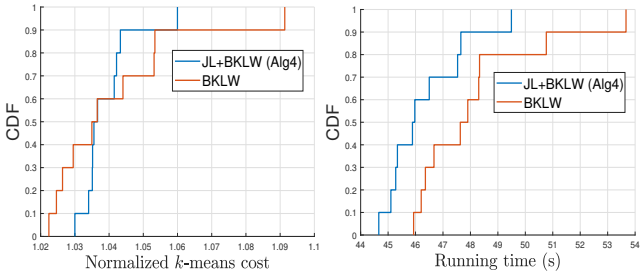


(b) NeurIPS

Fig. 1. Single-source case: normalized k-means cost and running time



(a) MNIST



(b) NeurIPS

Fig. 2. Multiple-source case: normalized k-means cost and running time

and no computation at the data source. We observe the following: (i) Compared to the naive method of transmitting the raw data (NR), the proposed algorithms can dramatically reduce the communication cost (by $> 99\%$) with a moderate increase in the k-means cost ($< 10\%$). (ii) Compared to the benchmark (FSS), the proposed algorithms can achieve a similar or smaller k-means cost while significantly reducing the communication cost and/or complexity, which is thanks to the proper application of JL projection and consistent with our theoretical analysis in Table 2. (iii) Between the approaches of DR+CR (JL+FSS) and CR+DR (FSS+JL), we see that the DR+CR approach yields a better performance for the NeurIPS dataset, where JL+FSS has a substantially shorter running time than FSS+JL but similar k-means cost and communication cost. This is because $\log n \ll \min(n, d)$ for this dataset, allowing JL+FSS to significantly reduce the complexity compared with FSS+JL without blowing up the communication cost according to our analysis in Table 2. (iv) For a sufficiently high-dimensional dataset such as NeurIPS, JL+FSS+JL can further improve the communication-computation tradeoff while achieving a similar k-means cost, which is again consistent with our analysis in Table 2.

In the case of multiple data sources, $m = 10$ data sources cooperatively compute and report a data summary using the evaluated distributed DR/CR algorithms, based on which a server computes k-means centers for the union of the m local datasets. The results are shown in Figure 2 and Table 4. We see that the proposed algorithm (JL+BKLW) achieves a k-means cost comparable to the benchmark (BKLW), while incurring a lower complexity and a lower communication cost. This improvement is again thanks to the suitable application of JL projection, which is efficient

TABLE 3
Single-source Case: Normalized Communication Cost

Dataset	NR	FSS	JL+FSS	FSS+JL	JL+FSS+JL
MNIST	1	8.95e-3	5.82e-3	5.82e-3	5.97e-3
NeurIPS	1	5.87e-3	3.60e-3	3.59e-3	2.84e-3

TABLE 4
Multiple-source Case: Normalized Communication Cost

Dataset	NR	BKLW	JL+BKLW
MNIST	1	1.97e-2	1.69e-2
NeurIPS	1	1.28e-2	1.05e-2

in both computational complexity and communication cost; the observations are consistent with the analysis in Table 2. Recall that applying JL projection after BKLW will not reduce the communication cost or the complexity as explained after Theorem 5.4.

7.3 Results for Joint DR, CR, and QT

We assume double precision for the original dataset before applying QT. For tractability, in the experiments we set all the ϵ values except ϵ_{QT} to be equal when solving (21).

7.3.1 *Evaluated Algorithms*

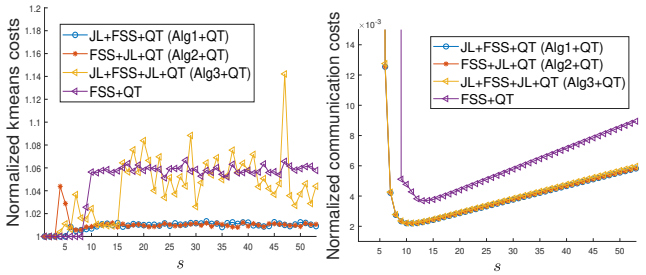
In the case of a single data source, we evaluate the following:

- "FSS+QT": the quantization-added version of FSS [11],
- "JL+FSS+QT": the quantization-added version of Algorithm 1,
- "FSS+JL+QT": the quantization-added version of Algorithm 2, and
- "JL+FSS+JL+QT": the quantization-added version of Algorithm 3.

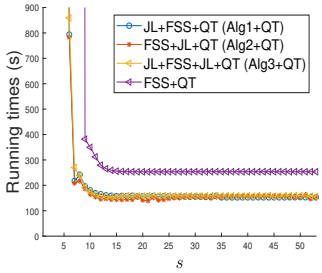
In the case of multiple data sources, we evaluate the following:

- "BKLW+QT": the quantization-added version of BKLW [27], and
- "JL+BKLW+QT": the quantization-added version of Algorithm 4.

For each algorithm, we construct a data summary under each configuration that corresponds to a possible number of significant bits s, and then solve k-means based on

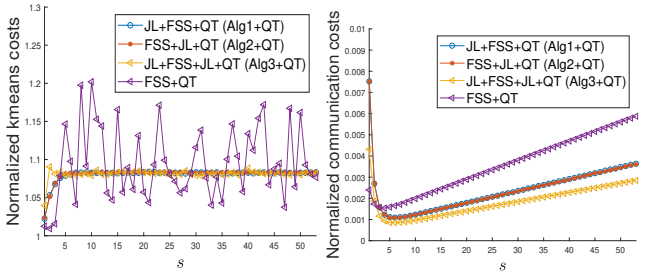


(a) Normalized k-means cost (b) Normalized communication cost

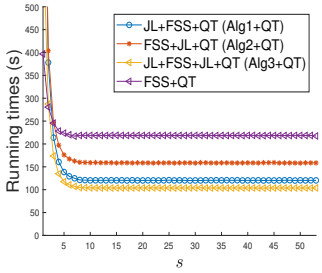


(c) Running time (s)

Fig. 3. Single-source case with quantization: MNIST

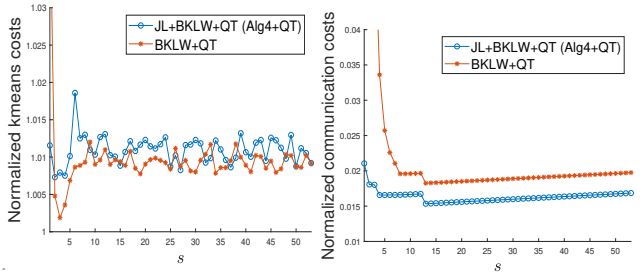


(a) Normalized k-means cost (b) Normalized communication cost

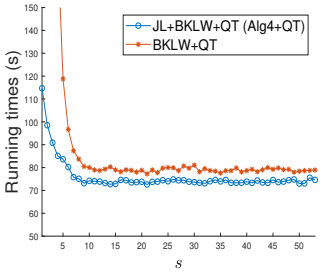


(c) Running time (s)

Fig. 4. Single-source case with quantization: NeurIPS

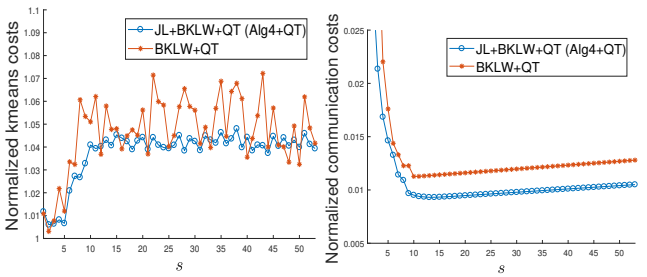


(a) Normalized k-means cost (b) Normalized communication cost

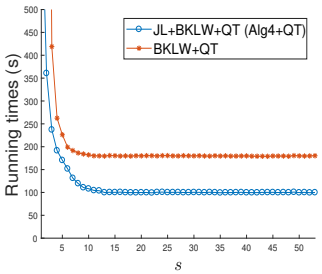


(c) Running time (s)

Fig. 5. Multiple-source case with quantization: MNIST



(a) Normalized k-means cost (b) Normalized communication cost



(c) Running time (s)

Fig. 6. Multiple-source case with quantization: NeurIPS

the data summary. Since the IEEE Standard 754 floating number representation [41] consists of 53 significant bits, we enumerate $s = 1, \ldots, 53$.

7.3.2 Results

The results for the single data source scenario are given in Figures 3–4. We have the following key observations: (i) Compared with the methods without quantization (the right-most points under $s = 53$), adding suitably configured quantization can further reduce the communication cost by $2/3$ without increasing the k-means cost or the running time. This is because the cluster structure for k-means clustering has certain robustness to minor shifts of data points (caused by rounding off a few least significant bits). (ii) However, it is nontrivial to find the optimal configuration to achieve a comparable k-means cost and running time with the least communication cost, as very small and very large values of s both lead to suboptimal performance. Intuitively, setting s too large will fail to take advantage of the communication cost saving due to quantization, and setting it too small will cause too much quantization error and leave no room of error for DR/CR. (iii) When the dimensionality is not too high (e.g., MNIST), a three-step procedure such as JL+FSS+QT or FSS+JL+QT suffices; for a high-dimensional dataset such as NeurIPS, the four-step procedure JL+FSS+JL+QT can further reduce the communication cost and the running time while achieving a comparable k-means cost, which is consistent with the predicted advantage of JL+FSS+JL in the regime of $n, d \gg 1$ as shown in Table 2.

The results for the multiple data source scenario are given in Figures 5–6. We have similar observations as in the single-source case: (i) Compared with no quantization (the right-most points under $s = 53$), adding suitably configured quantization can further reduce the communication cost by 10% without increasing the k-means cost or the running time. (ii) Choosing a proper configuration (by selecting the optimal number of significant bits to retain in quantization) is nontrivial. (iii) Compared with BKLW+QT, JL+BKLW+QT can reduce both the communication cost and the running time while achieving a similar k-means cost, which is consistent with the comparison between BKLW and JL+BKLW (Figure 2 and Table 4) as well as the theoretical prediction in Table 2. This result again demonstrates the benefit of properly combining existing DR/CR methods with JL projection.

7.4 Summary of Observations

Our experimental results imply the following observations:

- Solving k-means based on data summaries generated by DR/CR methods can provide a reasonably good solution at a drastically reduced communication cost without incurring a high complexity at data sources.
- Compared with state-of-the-art algorithms, suitable combination of DR and CR can effectively reduce the communication cost and the complexity while providing a k-means solution of a similar quality.
- Augmenting DR and CR with suitably configured quantization can further reduce the communication cost without adversely affecting the other metrics.

8 Conclusion

In this paper, we considered the problem of using data reduction methods to efficiently compute the k-means centers for a large high-dimensional dataset located at remote data source(s), with focus on DR and CR. Through a comprehensive analysis of the approximation error, the communication cost, and the complexity of various combinations of state-of-the-art DR/CR methods, we proved that it is possible to achieve a near-optimal approximation of k-means at a near-linear complexity at the data source(s) and a very low (constant or logarithmic) communication cost. In the process, we developed algorithms based on carefully designed combinations of existing DR/CR methods that outperformed two state-of-the-art algorithms in the scenarios of a single data source and multiple data sources, respectively. We also demonstrated how to combine DR/CR methods with quantizers to further reduce the communication cost without compromising the other performance metrics. Our findings were validated through experiments on real datasets.

Acknowledgments

This research was partly sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

[1] H. Lu, T. He, S. Wang, C. Liu, M. Mahdavi, V. Narayanan, K. S. Chan, and S. Pasteris, "Communication-efficient k-means for edge-based machine learning," in *ICDCS*, November 2020.

[2] J. Park, S. Samarakoon, M. Bennis, and M. Debbah, "Wireless network intelligence at the edge," *Proceedings of the IEEE*, vol. 107, no. 11, pp. 2204–2239, 2019.

[3] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[4] M. F. Balcan, S. Ehrlich, and Y. Liang, "Distributed k-means and k-median clustering on general topologies," in *NIPS*, December 2013.

[5] H. Lu, M.-J. Li, T. He, S. Wang, V. Narayanan, and K. S. Chan, "Robust coreset construction for distributed machine learning," in *IEEE Globecom*, December 2019.

[6] H. Lu, M. Li, T. He, S. Wang, V. Narayanan, and K. S. Chan, "Robust coreset construction for distributed machine learning," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 10, pp. 2400–2417, 2020.

[7] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.

[8] D. Aloise, A. Deshpande, P. Hansen, and P. Popat, "NP-hardness of Euclidean sum-of-squares clustering," *Machine Learning*, vol. 75, no. 2, pp. 245–248, 2009.

[9] M. Mahajan, P. Nimbhorkar, and K. R. Varadarajan, "The planar k-means problem is NP-hard," *Theoretical Computer Science*, vol. 442, no. 13, pp. 13–21, 2012.

[10] K. Makarychev, Y. Makarychev, and I. Razenshteyn, "Performance of Johnson-Lindenstrauss transform for k-means and k-medians clustering," in *STOC*, June 2019.

[11] D. Feldman, M. Schmidt, and C. Sohler, "Turning big data into tiny data: Constant-size coresets for k-means, PCA, and projective clustering," 2018. [Online]. Available: https://arxiv.org/abs/1807.04518

[12] K. Sayood, *Introduction to Data Compression, Fourth Edition*, 4th ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2012.

[13] A. Gersho and R. M. Gray, *Vector quantization and signal compression*. Springer Science & Business Media, 2012, vol. 159.

[14] C. Boutsidis, A. Zouzias, and P. Drineas, "Random projections for k-means clustering," in *NIPS*, December 2010.

[15] M. B. Cohen, S. Elder, C. Musco, C. Musco, and M. Persu, "Dimensionality reduction for k-means clustering and low rank approaximation," in *STOC*, June 2015.

[16] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay, "Clustering large graphs via the singular value decomposition," *Machine Learning*, vol. 56, no. 1-3, pp. 9–33, 2004.

[17] C. Boutsidis, A. Zouzias, M. W. Mahoney, and P. Drineas, "Randomized dimensioinality reduction for k-means clustering," *IEEE Trans. IT*, vol. 61, no. 2, pp. 1045–1062, February 2015.

[18] W. Johnson and J. Lindenstrauss, "Extensions of Lipschitz mappings into a Hilbert space," in *Conference in Modern Analysis and Probability*, 1982.

[19] S. Har-Peled and S. Mazumdar, "On coresets for k-means and k-median clustering," in *STOC*, 2004.

[20] G. Frahling and C. Sohler, "Coresets in dynamic geometric data streams," in *STOC*, 2005.

[21] S. Har-Peled and A. Kushal, "Smaller coresets for k-median and k-means clustering," *Discrete & Computational Geometry*, vol. 37, no. 1, pp. 3–19, 2007.

[22] K. Chen, "On coresets for k-median and k-means clustering in metric and Euclidean spaces and their applications," *SIAM Journal on Computing*, vol. 39, no. 3, pp. 923–947, 2009.

[23] M. Langberg and L. J. Schulman, "Universal ϵ approximators for integrals," in *SODA*, 2010.

[24] D. Feldman and M. Langberg, "A unified framework for approximating and clustering data," in *STOC*, June 2011.

[25] V. Braverman, D. Feldman, and H. Lang, "New frameworks for offline and streaming coreset constructions," *CoRR*, vol. abs/1612.00889, 2016.

[26] A. Barger and D. Feldman, "k-means for streaming and distributed big sparse data," in *SDM*, 2016.

[27] M. F. Balcan, V. Kanchanapally, Y. Liang, and D. Woodruff, "Improved distributed principal component analysis," in *NIPS*, December 2014.

[28] Y. Mao, Z. Xu, P. Ping, and L. Wang, "An optimal distributed k-means clustering algorithm based on CloudStack," in *International Conference on Frontier of Computer Science and Technology*, August 2015.

[29] M. C. Naldi and R. J. G. B. Campello, "Distributed k-means clustering with low transmission cost," in *Brazilian Conference on Intelligent Systems*, October 2013.

[30] C. R. Giannella, H. Kargupta, and S. Datta, "Approximate distributed k-means clustering over a peer-to-peer network," *IEEE Trans. KDE*, vol. 21, pp. 1372–1388, October 2009.

[31] A. Ben-Israel and T. N. E. Greville, *Generalized Inverses: Theory and Applications*. Springer, 2003.

[32] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *ACM STOC*, 1998.

[33] D. Achlioptas, "Database-friendly random projections: Johnson-Lindenstrauss with binary coins," *Journal of Computer and System Sciences*, vol. 66, no. 4, pp. 671–687, 2003.

[34] B. Klartag and S. Mendelson, "Empirical processes and random projections," *Journal of Functional Analysis*, vol. 225, no. 1, pp. 229–245, August 2005.

[35] M. F. Balcan, V. Kanchanapally, Y. Liang, and D. Woodruff, "Improved distributed principal component analysis," 2014. [Online]. Available: https://arxiv.org/abs/1408.5823

[36] A. Aggarwal, A. Deshpande, and R. Kannan, "Adaptive sampling for k-means clustering," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Springer, 2009, pp. 15–28.

[37] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth, "Learnability and the vapnik-chervonenkis dimension," *Journal of the ACM (JACM)*, vol. 36, no. 4, pp. 929–965, 1989.

[38] Y. Li, P. M. Long, and A. Srinivasan, "Improved bounds on the sample complexity of learning," *Journal of Computer and System Sciences*, vol. 62, no. 3, pp. 516–527, 2001.

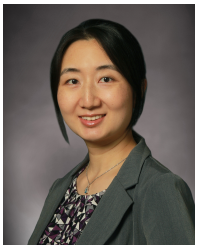
[39] Y. LeCun, C. Cortes, and C. Burges, "The MNIST database of handwritten digits," http://yann.lecun.com/exdb/mnist/, 1998.

[40] V. Perrone, P. A. Jenkins, D. Spano, and Y. W. Teh, "Poisson random fields for dynamic feature models," *arXiv preprint arXiv:1611.07460*, 2016.

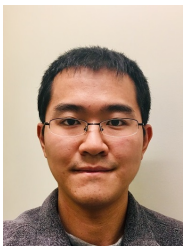
[41] IEEE, "754-2019 - ieee standard for floating-point arithmetic," 2019. [Online]. Available: https://ieeexplore.ieee.org/servlet/opac?punumber=8766227

[42] A. Aggarwal, A. Deshpande, and R. Kannan, "Adaptive sampling for k-means clustering," in *APPROX*, August 2009.

Hanlin Lu (S'19) received the Ph.D. degree in Computer Science and Engineering from Pennsylvania State University in 2021. He is currently a Research Scientist at ByteDance, Mountain View, CA, USA. His research interests include coreset construction and distributed machine learning training.



Ting He (SM'13) is an Associate Professor in the School of Electrical Engineering and Computer Science at Pennsylvania State University, University Park, PA. Her work is in the broad areas of computer networking, network modeling and optimization, and machine learning. Dr. He is a senior member of IEEE, an Associate Editor for IEEE Transactions on Communications (2017-2020) and IEEE/ACM Transactions on Networking (2017-2021), a TPC Co-Chair of IEEE ICCCN (2022), and an Area TPC Chair of IEEE INFOCOM (2021). She received multiple Outstanding Contributor Awards from IBM, multiple awards for Military Impact, Commercial Prosperity, and Collaboratively Complete Publications from ITA, and multiple paper awards from ICDCS, SIGMETRICS, ICASSP, and IEEE Communications Society.



Shiqiang Wang (S'13–M'15) received his Ph.D. from the Department of Electrical and Electronic Engineering, Imperial College London, United Kingdom, in 2015. Before that, he received his master's and bachelor's degrees at Northeastern University, China, in 2011 and 2009, respectively. He has been a Research Staff Member at IBM T. J. Watson Research Center, NY, USA since 2016. In the fall of 2012, he was at NEC Laboratories Europe, Heidelberg, Germany. His current research focuses on the interdisciplinary areas in distributed computing, machine learning, networking, optimization, and signal processing. Dr. Wang served as a technical program committee (TPC) member of several international conferences, including ICML, NeurIPS, ICDCS, AISTATS, IJCAI, IFIP Networking, IEEE GLOBECOM, IEEE ICC, and as an associate editor of the IEEE Transactions on Mobile Computing. He received the IEEE Communications Society Leonard G. Abraham Prize in 2021, IBM Outstanding Technical Achievement Award (OTAA) in 2019 and 2021, multiple Invention Achievement Awards from IBM since 2016, Best Paper Finalist of the IEEE International Conference on Image Processing (ICIP) 2019, and Best Student Paper Award of the Network and Information Sciences International Technology Alliance (NIS-ITA) in 2015.

Changchang Liu received the Ph.D. degree in electrical engineering from Princeton University. She is currently a Research Staff Member with the Department of Distributed AI, IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA. Her current research interests include federated learning, big data privacy, and security.

Mehrdad Mahdavi is an Assistant Professor of the Computer Science Department at the Penn State. He received the Ph.D. degree in Computer Science from Michigan State University in 2014. Before joining PSU in 2018, he was a Research Assistant Professor at Toyota Technological Institute, at University of Chicago. His research interests lie at the interface of machine learning and optimization with a focus on developing theoretically principled and practically efficient algorithms for learning from massive datasets and complex domains. He has won the Mark Fulk Best Student Paper award at Conference on Learning Theory (COLT) in 2012.

Vijaykrishnan Narayanan (F'11) is the Robert Noll Chair Professor of Computer Science and Engineering and Electrical Engineering at the Pennsylvania State University. His research interests are in Embedded System Design, Computer Architecture and Power-Aware Systems. He is a fellow of National Academy of Inventors, IEEE, and ACM.

Kevin S. Chan (S'02–M'09–SM'18) received the B.S. degree in electrical and computer engineering and engineering and public policy from Carnegie Mellon University, Pittsburgh, PA, USA and the M.S. and Ph.D. degrees in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, GA, USA. He is currently an Electronics Engineer with the Computational and Information Sciences Directorate, U.S. Army Combat Capabilities Development Command, Army Research Laboratory, Adelphi, MD, USA. He is actively involved in research on network science, distributed analytics, and cybersecurity. He received the 2021 IEEE Communications Society Leonard G. Abraham Prize and multiple best paper awards. He is the Co-Editor of the IEEE Communications Magazine—Military Communications and Networks Series.

Stephen Pasteris gained a BA+MA in Mathematics from Kings College of the University of Cambridge. After completing his BA he then went on to gain a PhD in Computer Science from University College London: his thesis focusing on the development of efficient algorithms for machine learning on networked data. Stephen is now a Research Associate at University College London where he primarily researches online machine learning.