k-sums Clustering: A Stochastic Optimization Approach

Wan-Lei Zhao*
Shi-Ying Lan
Run-Qing Chen
Xiamen University
China
wlzhao@xmu.edu.cn
sylan@stu.xmu.edu.cn
chenrq1010026261@stu.xmu.edu.cn

Chong-Wah Ngo
School of Computing and Information Systems, Singapore
Management University
Singapore
cwngo@smu.edu.sg

ABSTRACT

In this paper, we revisit the decades-old clustering method *k*-means. The egg-chicken loop in traditional *k*-means has been replaced by a pure stochastic optimization procedure. The optimization is undertaken from the perspective of each individual sample. Different from existing incremental *k*-means, an individual sample is tentatively joined into a new cluster to evaluate its distance to the corresponding new centroid, in which the contribution from this sample is accounted. The sample is moved to this new cluster concretely only after we find the reallocation makes the sample closer to the new centroid than it is to the current one. Compared with traditional *k*-means and other variants, this new procedure allows the clustering to converge faster to a better local minimum. This fundamental modification over the *k*-means loop leads to the redefinition of a family of *k*-means variants, such as hierarchical *k*-means, and Sequential *k*-means. As an extension, a new target function that minimizes the summation of pairwise distances within clusters is presented. Under l_2-norm, it could be solved under the same stochastic optimization procedure. The re-defined traditional *k*-means, hierarchical *k*-means, as well as Sequential *k*-means all show considerable performance improvement over their traditional counterparts under different settings and on various types of datasets.

CCS CONCEPTS

• **Theory of computation** → **Unsupervised learning and clustering**.

KEYWORDS

k-means, stochastic optimization, driven function

---

*Corresponding author.

1 INTRODUCTION

Clustering is a basic processing tool in many areas such as data mining [32], data compression [15], pattern recognition and computer vision. Since the first *k*-means methods [18, 19] was proposed in year *1982*, various clustering methods [13] have been proposed one after another in the last three decades. These methods range from classic density based methods such as mean shift [8], DB-SCAN [9], and recent clusterDP [24], to graph based methods such as spectral clustering [29] and Rank-Order [21], etc. Nevertheless, *k*-means [19] remains popular for its efficiency, versatility as well as simplicity. According to [30], it is recognized among the top ten most popular methods in data mining.

Given *n* data samples in *d*-dimensional space $\mathbb{R}^d$, and an integer *k*, the clustering task is modeled as a distortion minimization process in *k*-means. In one iteration, it assigns *n* samples to one of *k* sets where its corresponding centroid is the closest to the sample. The minimization target function is given as

$$\text{Min.} \sum_{q(x_i)=r} d(x_i, C_r), \tag{1}$$

where $x_i \in \mathbb{R}^d$ and C_r is the centroid of cluster *r*. In Eqn. 1, function $q(x_i)$ returns the closest centroid (among *k* centroids) for sample x_i. There are in general three major steps involved in *k*-means iterations. In the initial step, *k* samples are randomly selected as the initial centroids. In the assignment step, each sample is assigned to its closest centroid. In the centroid updating step, each centroid C_r is updated by taking the average over the assigned samples. The last two steps are repeated until there is no distortion variation (Eqn. 1) in the two consecutive iterations. This iteration process is widely known as the classic "egg-chicken" loop.

Although it is simple and effective, the major issues for this "egg-chicken" loop are in several aspects. Firstly, the target function is minimized in an implicit manner. The iteration in its nature minimizes the discrepancy between two consecutive iterations instead of Eqn. 1. Moreover, the update on the centroid is postponed to the moment when all the samples are assigned to their closest centroids. Given *t* and *t* + 1 are two consecutive iterations in *k*-means, the real target function that is minimized during the iteration is

$$\text{Min.} \sum_{q(x_i)=r} d(x_i, C_r^{(t)}). \tag{2}$$

After the assignment step, samples assigned to $C_r^{(t)}$ are averaged to produce $C_r^{(t+1)}$. Such kind of minimization is inefficient in the sense that the samples are compared to centroids produced from previous iteration t. No update happens when a sample is moved from other clusters to r. However, according to Eqn. 1, the centroids are expected to be updated as soon as the membership of one sample changes. Due to the delayed update, the samples are not allowed to compare with the centroids that reflect the real structure of clusters at each moment. For the above reasons, usually k-means converges slowly to a local minimum.

In the literature, several efforts have been devoted to enhancing the clustering quality. Particularly, the clustering quality is boosted by a careful seeding scheme [1, 6], for which the centroids are initialized based upon the data distribution. Recently, the k-means problem is approximated by a maximization procedure [31, 32]. Encouraging performance is achieved.

In this paper, the k-means clustering that is formulated in Eqn. 1 is addressed by an explicit stochastic minimization process. It turns out to be simpler as well as better over k-means and many of its variants. Under the same minimization framework, a family of k-means variants such as hierarchical k-means and Sequential k-means is redefined to achieve better performance. Moreover, a new target function that minimizes the summation of pairwise distances within each cluster is proposed. Based on the same stochastic optimization procedure, the target function is explicitly minimized with the same time complexity as the conventional k-means.

The remainder of this paper is organized as follows. The reviews on the representative k-means variants are presented in Section 2. In Section 3, the driven function derived from the k-means target function is presented. In addition, a new clustering target function and its driven function are proposed. The iteration procedures built upon these two driven functions are accordingly presented. The possible extensions, convergence and complexity analysis are presented in Section 4. The effectiveness of the proposed methods is studied in Section 5. Section 6 concludes the paper.

2 RELATED WORK

k-means has been widely adopted as a basic tool in data mining [32], various data preprocessing and pattern recognition [13] mainly due to its versatility and simplicity. Various improvement schemes are proposed during the last three decades to boost its performance in terms of either clustering quality or scalability.

A representative work in improving the clustering quality was proposed by S. Vassilvitskii et al. [1, 4]. In the method, the initial centroids are selected to be far from each other to reflect the underlying data distribution. It leads to higher clustering quality as well as faster convergence speed according to [1]. However, k-1 rounds of scanning over the whole data are necessary to find the initial centroids. The number of scanning rounds has been successfully reduced to a few [4] or even fewer [3]. However, all the above improvements focus on the initial assignment stage. The "egg-chicken" loop is still adopted. Therefore the aforementioned pitfalls that are caused by this loop remain unchanged.

In the literature, there are several efforts aiming to transform the "egg-chicken" loop into an optimization procedure [27, 28, 31, 32].

In [32], k-means is addressed as a maximization problem under *Cosine* distance. This maximization solution is extended to the whole l_2-space in [31]. While following Hartigan procedure [11], methods from [27, 28] perform the distortion minimization directly on the original k-means target function. There are two major differences in these methods from the other k-means variants. Firstly, a cluster and its corresponding centroid are updated as soon as the membership of one sample changes during the iteration. Secondly, the target function in each update step is monotonically optimized in a greedy manner. Another interesting discovery from [31] is that the improvement achieved from careful seeding [1] is minor in comparison to that from the modification of the iteration procedure. Nevertheless, the maximization model in [32] only works under *Cosine* distance. Although the methods in [27, 28, 31] are feasible in the whole l_2-space, the optimization converges in a slow pace as it has to guarantee a monotonic optimization in each update step. Specifically, a sample is not necessarily assigned to its closest centroid in one update [27, 28], which actually hinders the optimization process from reaching a better local optimum.

Although the time complexity of k-means is linear to the size of the dataset, it could become very slow as both k and dataset size n are large. The processing bottleneck comes from the operations of assigning samples to their closest centroids in every k-means iteration. As a result, many efforts have been made to speed-up the sample-to-centroid comparison. Solutions presented in [7, 22] reduce the comparisons with the support of indexing structures such as inverted file or KD-tree. However, the former is only effective for sparse vectors, while the latter performs poorly on dense high-dimensional vectors. The scalability issue of k-means is also addressed by subsampling strategy. In methods such as Mini-Batch [26] and [10], only a small portion of the whole dataset are sampled to update the cluster centroids. Such methods usually achieve high speed efficiency at the expense of low clustering quality.

Besides aforementioned k-means variants, there are still another two popular variants, namely hierarchical k-means [12] and Sequential k-means [20]. Hierarchical k-means conducts the clustering in a top-down hierarchical manner [12, 32, 33]. The clustering solution is obtained via a sequence of repeated partitions over intermediate clusters. When the fanout on each hierarchy is 2, it is called as "bisecting k-means" [32]. The advantages of such scheme are two folds. Firstly, it is able to produce a dendrogram view of the dataset. Moreover, the clustering time complexity of k-means is reduced from $O(t \cdot k \cdot n \cdot d)$ to $O(t \cdot \log(k) \cdot n \cdot d)$ [31], where t is the number of iterations. This is significant when n, d, and k are all very large. The dark side is that clustering performance could be poor as it breaks *Lloyd*'s condition [31]. Sequential k-means is also known as online k-means. It is designed for the case that samples come in sequentially. The clustering centroid is updated incrementally as a new sample joins in [20]. Given C_r is the closest centroid to sample x_i and n_r is the size of cluster r, the centroid is updated by

$$C_r = C_r + \frac{x_i - C_r}{n_r + 1}. \tag{3}$$

Different from the conventional k-means, it is supposed that there is only one single pass over the data, although it can be trivially repeated multiple times to reallocate samples until convergence.

Overall, although the various modifications are made over conventional k-means in the literature, most of the variants still build upon the "egg-chicken" loop. In this paper, the modification is undertaken on the "egg-chicken" loop itself. This leads to a fundamental change over k-means. It becomes simpler and considerably better while involving no additional computational costs. More importantly, this new iteration procedure can be easily implanted in various k-means variants to boost their performance.

3 K-SUMS CLUSTERING

As discussed in Section 1, the major issues that lie in the conventional k-means loop are that the centroids are not updated timely and the target function is not explicitly minimized. In the following, we are going to show it is possible to minimize Eqn. 1 directly by a stochastic optimization procedure. The optimization is driven by a function that minimizes Eqn. 1 greedily. This function is called as driven function $\mathcal{I}_m$. In addition, another target function that aims to minimize the summation of pairwise distances within each cluster is presented. Similarly, a driven function given as $\mathcal{I}_s$ is derived for this target function. We show that both minimization problems could be solved by the same stochastic optimization procedure.

To facilitate our discussions in this section and the later, several variables are introduced. The k clusters produced by a clustering method are given as $\{S_1, \cdots, S_r, \cdots, S_k\}$. Accordingly, the sizes of the clusters are given as $n_1, \cdots, n_r, \cdots, n_k$. The composite vector of one cluster is defined as $D_r = \sum_{x_i \in S_r} x_i$[1], which is nothing more than the summation of the samples in one cluster. The cluster centroid C_r is given as $C_r = \frac{D_r}{n_r}$.

In the following, we are going to first show the driven functions for two optimization problems. Based on the driven functions, the novel k-means iteration procedure is presented.

3.1 Driven Function $\mathcal{I}_m$

Given a sample x_i, it is currently located in cluster S_w, namely $x_i \in S_w$. According to Eqn. 1, its distance to the centroid of S_w is given as

$$d(x_i, C_w) = \| x_i - \frac{D_w}{n_w} \|^2 . \qquad (4)$$

This is also the distortion associated with sample x_i that contributes to Eqn. 1.

Let's now assume that the structure of cluster S_w has been changed in the previous iterations as some samples have been swapped in/out. For this reason, C_w may be no longer the closest centroid for x_i. Now we check whether there exists any other cluster S_v ($v \neq w$) that is more appropriate for x_i. The distance between x_i and C_v is measured supposing that x_i is already joined into cluster S_v. As a result, the distortion variation for x_i is given as Eqn. 5 for this possible movement.

$$\mathcal{I}_m(x_i, w, v) = d(x_i, C_w) - d(x_i, C_v),$$
$$\text{where } C_v = \frac{D_v + x_i}{n_v + 1}. \qquad (5)$$

Please be noted that Eqn. 5 is different from online k-means [20] in the sense that x_i is supposed to be a member of cluster S_v, rather than excluding x_i out of S_v in the distance evaluation. In

[1]Both x_i and D_r are column vectors by default.

above equation, as $\mathcal{I}_m(x_i, w, v) > 0$, assigning x_i to cluster S_v will decrease the distortion associated with x_i, which in turn leads to the possible decrease in the overall distortion for target function Eqn. 1. So the sample is moved from the current cluster to S_v as long as $\mathcal{I}_m(x_i, w, v)$ is positive and the maximum among all $k - 1$ tentative re-allocations. The movement of sample x_i from cluster S_w to S_v involves the update of membership for x_i as well as the update on C_v, n_v, C_w and n_w. Function $\mathcal{I}_m(x_i, w, v)$ is therefore called as driven function.

This driven function is essentially different from [27, 28], in which the distance between x_i and C_w is calculated assuming x_i has been removed out from S_w. This subtle difference leads to the very different interpretations about the effect. Function $\mathcal{I}_m(x_i, w, v)$ guarantees that x_i is placed to its closest centroid. While there is no guarantee that the movement of x_i leads to the decrease in Eqn. 1. The function in [27, 28] leads to the opposite effects. Namely, the movement of x_i leads to the lower of overall distortion in Eqn. 1, however x_i is not necessarily put into the cluster that is closest to it. In other words, $\mathcal{I}_m(x_i, w, v)$ allows the "individual interests" to be maximized in each movement, while function in [27, 28] guarantees the monotonic increase of "general interests" in each movement. As analyzed in Section 4.2, the former is less likely being trapped in a local optimum and therefore performs considerably better as is revealed in the experiments.

To simplify the computation, the distance between x_i and C_w is given as

$$d(x_i, C_w) = \frac{\| n_w \cdot x_i - D_w \|^2}{n_w^2}. \qquad (6)$$

Accordingly, the distance between x_i and C_v is given as

$$d(x_i, C_v) = \frac{\| n_v \cdot x_i - D_v \|^2}{(n_v + 1)^2}. \qquad (7)$$

In some scenarios, we may use *Cosine* distance instead of l_2-norm to measure the distance between samples and the distance between samples and the centroids. One would have the following equations to measure the distance between sample x_i and centroid C_w and C_v respectively.

$$cos(x_i, C_w) = \frac{x_i' \cdot D_w}{\sqrt{x_i' \cdot x_i} \cdot \sqrt{D_w' \cdot D_w}} \qquad (8)$$

$$cos(x_i, C_v) = \frac{x_i' \cdot D_v + x_i' \cdot x_i}{\sqrt{x_i' \cdot x_i} \cdot \sqrt{D_v' \cdot D_v + 2x_i' \cdot D_v + x_i' \cdot x_i}} \qquad (9)$$

Since the l_2-norm of x_i could be pre-computed, the terms we should consider in Eqn. 8 and Eqn. 9 are the inner-products between x_i and the composite vectors, and the l_2-norms of composite vectors D_w and D_v.

It is clear to see that C_w and C_v are not involved in any case of the distance computation. Only D_rs and n_rs are required. The composite vectors D_rs are nothing more than k summations of samples within k clusters. To this end, the "means" are replaced by "sums". For this reason, our new clustering method is called as **k-sums** from now on. Please be noted that it is possible to formulate the driven function (Eqn. 5) in terms of centroids. However, the computing cost of updating centroids turns out to be much higher

than updating only the composite vectors as the update operation is frequently undertaken in the iteration.

3.2 Driven Function $\mathcal{I}_s$

In some scenarios, defining the centroid for a clustering problem would be hard or even impossible. For instance, the sample vectors could not be averaged when the values in each data dimension/property are discrete. A good case is the gender of a person. This is where the clustering method such as PAM [16] comes, in which cluster modes instead of centroids are defined. Moreover, the criterion of being a cluster may change. Instead of minimizing summations of distances to a mode/centroid, we may need to minimize the intra-distances within each cluster. This leads to a new target function. Namely, the target function is simply defined as

$$\text{Min.} \sum_{r=1}^{k} \sum_{i,j \in S_r \& i < j} d(x_i, x_j). \tag{10}$$

Notice that this minimization target function is different from $\mathcal{I}_2$ proposed in [32], because it aims to minimize the weighted intra-distances within each cluster. In [32], the average pairwise distance within each cluster is weighted by the size of a cluster. To seek for the optimal solution for Eqn. 10, intuitively one has to try out all the possible combinations of the samples in one cluster. This is unfortunately NP-hard as PAM [16]. As a consequence, we only seek for a local minimal solution to this problem. In particular, in l_2-space, this target function can be addressed with a greedy procedure in a very efficient fashion.

Given that $x_i \in S_w$ and the distance between samples is measured by l_2-norm, the overall distance between sample x_i and cluster S_w is defined as

$$d(x_i, S_w) = \sum_{x_j \in S_w} \| x_i - x_j \|^2, \tag{11}$$

which is the summation of distances between sample x_i and each sample in S_w. Eqn. 11 can be further simplified as

$$d(x_i, S_w) = n_w \cdot x_i' \cdot x_i - 2 \cdot x_i' \cdot D_w + \sum_{x_j \in S_w} x_j' \cdot x_j, \tag{12}$$

where D_w is the composite vector of cluster S_w. Eqn. 12 can be efficiently calculated given the l_2-norms of each sample can be pre-computed and kept in a look-up table. The second term is the inner-product between sample x_i and the composite vector, which is comparable to calculating the distance between sample x_i and a centroid in the conventional k-means model. Given sample vectors are l_2-normalized, Eqn. 12 is further simplified as

$$d(x_i, S_w) = 2 \cdot n_w - 2 \cdot x_i' \cdot D_w. \tag{13}$$

Eqn. 13 could be used as *Cosine* distance when we want to adopt *Cosine* to measure the distances between vectors. They are interchangable as the vectors are l_2-normalized.

Now let's consider the similar driven strategy that we derive for target function Eqn. 1. Given sample $x_i \in S_w$, we consider whether it could be better if we put x_i into S_v. The distance between x_i and

---

Algorithm 1: k-sums driven by $\mathcal{I}_m$

Data: Input: $X_{d \times n}$, k
Result: Output: $S_1, \cdots, S_r, \cdots, S_k$
1 Lables$[1, \cdots, n] \leftarrow 0$;
2 Assign each $x_i \in X$ with a random cluster label;
3 Calculate $D_1, \cdots, D_r, \cdots, D_k$ and $n_1, \cdots, n_r, \cdots, n_k$;
4 **while** *not convergent* **do**
5   **for** *each $x_i \in X$ (in random order)* **do**
6         $w \leftarrow$ Lables[i];
7   Seek S_v that $\mathcal{I}_m(x_i, w, v)$ reaches the maximum;
8   **if** $\mathcal{I}_m(x_i, w, v) > 0$ **then**
9   Lables$[i] \leftarrow v$;
10             $D_w \leftarrow D_w - x_i$; $n_w \leftarrow n_w - 1$;
11             $D_v \leftarrow D_v + x_i$; $n_v \leftarrow n_v + 1$;
12   **end**
13   **end**
14 **end**

S_v is given as

$$\begin{aligned} d(x_i, S_v) &= (n_v + 1) \cdot x_i' \cdot x_i - 2 \cdot x_i' \cdot (D_v + x_i) \\ &\quad + \sum_{x_j \in S_v \& j \neq i} x_j' \cdot x_j + x_i' \cdot x_i \\ &= n_v \cdot x_i' \cdot x_i - 2 \cdot x_i' \cdot D_v + \sum_{x_j \in S_v \& j \neq i} x_j' \cdot x_j. \end{aligned} \tag{14}$$

Comparing distance $d(x_i, S_v)$ to $d(x_i, S_w)$, it is easy to judge whether such movement is "profitable" for x_i. Namely, we work out the driven function to minimize target function Eqn. 10 as

$$\mathcal{I}_s(x_i, w, v) = d(x_i, S_w) - d(x_i, S_v). \tag{15}$$

As shown in Eqn. 12 and Eqn. 14, it is unnecessary to maintain C_r. Similar as driven function $\mathcal{I}_m$, one only needs to maintain D_rs and n_rs during the optimization for computational efficiency.

In the minimization step, we check Eqn. 15 with all $k - 1$ clusters, and move x_i to the cluster where $\mathcal{I}_s(x_i, w, v)$ is positive and the maximum. Notice that each such kind of movement will lead to a steady decrease in the target function (Eqn. 10). While it is not guaranteed that Eqn. 1 steadily decreases when driven by $\mathcal{I}_m$.

3.3 Stochastic Optimization Procedure

With two driven functions $\mathcal{I}_m$ and $\mathcal{I}_s$ derived in the above sections, it becomes natural to work out the clustering iteration. Since the optimization procedures for $\mathcal{I}_m$ and $\mathcal{I}_s$ are similar, let's take $\mathcal{I}_m$ as an example. In one step of the iteration, sample x_i is randomly selected, then it is checked with $k-1$ clusters to seek for the maximal $\mathcal{I}_m$. A sample reallocation is undertaken as long as $\mathcal{I}_m$ reaches the maximum and is positive. The details of the clustering method k-sums are presented in Alg. 1, which is in general similar as [28, 31] yet driven by different function.

As shown in Alg. 1, following the practice in [31], no initial centroid selection or initial sample-to-centroid assignment is involved in k-sums. Each sample is assigned with a random cluster label. With these random labels, it is possible to calculate D_rs and n_rs (Alg. 1, *Line 3*). At the beginning, the samples from different clusters

are mixed up with each other at the initial stages [31]. However, the boundaries between clusters become clearer after only a few iterations. In each iteration, samples are evaluated in random order with $\mathcal{I}_m$. The movement happens when it is the most appropriate (Alg. 1, *Lines 8-12*). In the iteration procedure, D_rs instead of C_rs are maintained and updated. Since this procedure is driven by $\mathcal{I}_m$, it is given as k-sums-$\mathcal{I}_m$.

Different from optimization procedure proposed in [31], k-sums aims to minimize the original target function of k-means instead of its approximation. The conventional k-means clustering is transformed into a pure stochastic minimization process with the target function unchanged. Additionally, our minimization procedure is also essentially different from methods in [27, 28], for which the clustering distortion drops monotonically after each movement. In our method, when moving x_i from S_w to S_v, it is the most "profitable" act for "individual" x_i, however this might not be true for other members in S_v and S_w. As a result, there will be a few bumps in the trend of distortion, while it still shows a general decreasing trend. In contrast, methods in [27, 28, 31] seek for the movement that leads to the decrease of overall distortion in each step. As revealed in the later experiments, the optimization driven by seeking for the better of "individual interests" instead of "general interests" converges to a better optimum in most of the cases. When $\mathcal{I}_m$ is replaced by $\mathcal{I}_s$ in Alg. 1, it becomes the clustering method driven by $\mathcal{I}_s$, which is given as k-sums-$\mathcal{I}_s$.

Fig. 1 shows the function value variations after each step (Alg. 1, *Lines 6-12*) on four consecutive iterations driven by $\mathcal{I}_m$ and $\mathcal{I}_s$ respectively on a SIFT image feature dataset [14]. According to our observation, the distortions from k-sums-$\mathcal{I}_m$ decrease steadily as a general trend. However, the function value of Eqn. 1 may increase in some steps in one round of iteration. This is visible in the zoom-in view of one iteration curve (Fig. 1(b)). This is mainly because k-sums-$\mathcal{I}_m$ is driven by "individual interests" instead of "general interests". Only the distortion associated with a sample is decreased in one movement when driven by $\mathcal{I}_m$. The movement may lead to the increase of Eqn. 1 temporarily. However, this invokes other samples (from all clusters) to seek for a better reallocation in the following steps. As a result, the distortion still decreases steadily. The bumps are not observed with k-sums-$\mathcal{I}_s$ in Fig. 1(b) since one movement driven by $\mathcal{I}_s$ leads to the steady decrease in both individual distance to a cluster and the overall intra-cluster distances of Eqn. 10. The function curve of k-sums-$\mathcal{I}_s$ that is measured by Eqn. 1 is shown in Fig. 1(d). The curve shows a general trend of steady decrease. This does indicate two target functions are correlated to some extent. However, they are essentially different given the fact that the decreasing pace in Fig. 1(d) is considerably slower than that of Fig. 1(a).

4 EXTENSIONS AND DISCUSSIONS OVER K-SUMS

4.1 k-means Variants Driven by Optimization

As presented in Section 3, our modification on k-means is simple but fundamental. Theoretically speaking, many k-means variants that are built upon the "egg-chicken" loop could be optimized following the framework of k-sums. In this section, the modification on two

popular k-means variants is presented. We first consider bisecting k-means. Typically, it produces k clusters by repeatedly bisecting the intermediate clusters into two [32]. On each bisecting step, k-means is called. As a result, when k-sums is adopted in the bisecting step, it becomes bisecting k-sums. Moreover, k-sums driven by either $\mathcal{I}_s$ or $\mathcal{I}_m$ is feasible. Alg. 2 shows the details of the bisecting k-sums.

Algorithm 2: bisecting k-sums

Data: Input: matrix $X_{d \times n}$, k
Result: Output: $S_1, \cdots, S_r, \cdots, S_k$

1 $S_1 \leftarrow 1 \cdots n$;
2 Push S_1 into a priority queue Q;
3 $i \leftarrow 1$;
4 **while** $i < k$ **do**
5 $\quad$ Pop cluster S_t from Q;
6 $\quad$ Call Alg. 1 to cluster S_t into $\{S_t^*, S_{i+1}\}$;
7 $\quad$ Push S_t^*, S_{i+1} into queue Q;
8 $\quad$ $i \leftarrow i + 1$;
9 **end**

As shown in Alg. 2, Alg. 1 is called to partition a cluster S_t into two in each step. There could be several ways to decide which cluster S_t to be partitioned. Following the practice in [32], cluster with the largest size is selected each time from queue Q in our implementation[2].

The second k-means variant we consider to redefine is Sequential k-means, which scans the data only one round and runs online. Given Eqn. 1 is adopted as the target function for online k-means, the update function is revised as
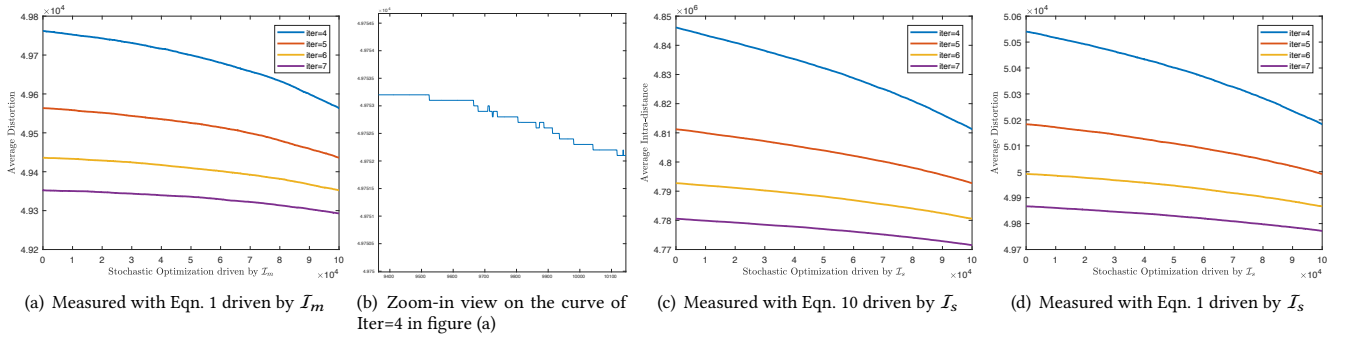
$$D_r = D_r + x_i, \tag{16}$$

given that $d(x_i, C_r) = \frac{\|n_r \cdot x_i - D_r\|^2}{(n_r+1)^2}$ is the minimum among k clusters. The similar way applies to the case when target function Eqn. 10 is adopted. This revised online clustering method is given as Sequential k-sums. Different from conventional Sequential k-means, the distance between sample x_i and C_r is calculated assuming that x_i is already joined in S_r. The codes of our implementation about k-sums and its variants are available at GitHub[3].

4.2 Complexity, Convergence and Optimality Analysis

It is apparent to see the time complexity of Alg. 1 is on the same par as conventional k-means. Compared to k-means, k-sums actually saves up the cost of initial sample-to-centroid assignment, which is equivalent to one round of iteration. In contrast, the time complexity of Hartigan procedure in [27, 28] is much higher than it is supposed to be as the optimization is defined on cluster centroids. Unlike conventional k-means "egg-chicken" loop, the centroid update is a frequent operation in all incremental optimization based methods, namely approaches from [27, 28] and k-sums. To its worse case, the centroids will be updated n times in one round. k-sums is computationally more efficient in the sense that it operates on the

---

[2]In practice, one may choose to split the most loose one.
[3]https://github.com/cc-cyber/k-sums.

(a) Measured with Eqn. 1 driven by $\mathcal{I}_m$

(b) Zoom-in view on the curve of Iter=4 in figure (a)

(c) Measured with Eqn. 10 driven by $\mathcal{I}_s$

(d) Measured with Eqn. 1 driven by $\mathcal{I}_s$

Figure 1: The target function variation curves produced on *100*K SIFT data by Alg. 1 on four consecutive iterations (4~7). Alg. 1 is driven by $\mathcal{I}_m$ (figure (a)) and $\mathcal{I}_s$ (figure (c)) respectively. The function value measured by Eqn. 1 when Alg. 1 is driven by $\mathcal{I}_s$ is shown in figure (d). All function values are normalized by the size of dataset.

composite vectors, on which only addition/subtraction operations are involved.

Since Alg. 1 could be driven by either $\mathcal{I}_m$ or $\mathcal{I}_s$, the convergence analysis on Alg. 1 is divided into two cases. Let's first consider the case as it is driven by $\mathcal{I}_m$. Essentially the iteration is driven by the motivation that x_i seeks for the better allocation such that $d(x_i, C_v) < d(x_i, C_w)$, where $x_i \in S_w$ and is tentatively put into S_v. Since $d(x_i, C_v) \geq 0$, there will be a moment for any sample x_i ($d(x_i, C_v) = 0$ to its best) that no movement could take place. At this moment, Alg. 1 converges.

When Alg. 1 is driven by $\mathcal{I}_s$, it is clear that target function Eqn.10 decreases monotonically after each movement. Given function value Eqn.10 after each movement is $\mathcal{F}^{(t)}$, following inequation series holds.

$$\mathcal{F}^{(1)} > \mathcal{F}^{(2)} > \cdots > \mathcal{F}^{(t)} > \cdots \geq \mathcal{F}^o, \qquad (17)$$

where $\mathcal{F}^o$ is the function value as we reach the optimal solution. As a result, the monotonically decreasing function is lower-bounded by $\mathcal{F}^o$. Apparently, it converges.

k-sums optimization driven either by $\mathcal{I}_m$ or $\mathcal{I}_s$ is greedy. Each optimization step is triggered by the decrease in the distance from an individual sample to its closest centroid (with $\mathcal{I}_m$) or cluster (with $\mathcal{I}_s$). Particularly for k-sums-$\mathcal{I}_m$, this is the essential difference as well as the advantage of our method over methods built upon Hartigan procedure [27, 28] and k-means#. The minimization in k-sums-$\mathcal{I}_m$ is driven by the "individual interests" of each sample instead of the "general interests" that is regulated by the Hartigan procedure [27, 28] . The latter imposes implicitly much tighter constraint over the movement of one sample. In these methods, one has to consider the impact to other members from two involved clusters, namely S_w and S_v. The "consensus" has to be reached among members from two clusters before sample x_i is allowed to move from one to another. In contrast, in k-sums-$\mathcal{I}_m$ sample x_i is free to move as long as the new centroid is closer to it than the previous is. It is no need to care about whether this movement is "beneficial" to the other members from cluster S_w or S_v. Due to the tight constraint, the existing methods [27, 28, 31] tend to be trapped in a local minimum easier than k-sums-$\mathcal{I}_m$.

It is possible that other samples in the two involved clusters become further from their centroids after the movement. However, they are therefore invoked to move to other closer clusters under the same rule. As a result, the seemingly "selfish" act allows each sample to finally find its closest centroid. Target function Eqn. 1 is a simple linear summation over distances of each individual to its assigned centroid. The lower of each individual distance leads to the lower overall function value.

Similar as k-means, there is no significant change in the structure of the clusters after a few iterations for k-sums. Although it turns out to be better than k-means and many of its variants, it only reaches a local minimum as k-means, k-means++ as well as k-means#.

5 EXPERIMENTS

In this section, the effectiveness of proposed clustering method, namely k-sums is studied in comparison to k-means and its representative variants. They include k-means++ [1], LVQ [17], the method based on Hartigan procedure (given as "Hartigan" in the following) [28], k-means# [31], incremental k-means (IKM) [32], Sequential k-means [20], Mini-Batch [26] and bisecting k-means [32]. For Sequential k-means and our redefined Sequential k-sums, there is only one single pass over the whole dataset.
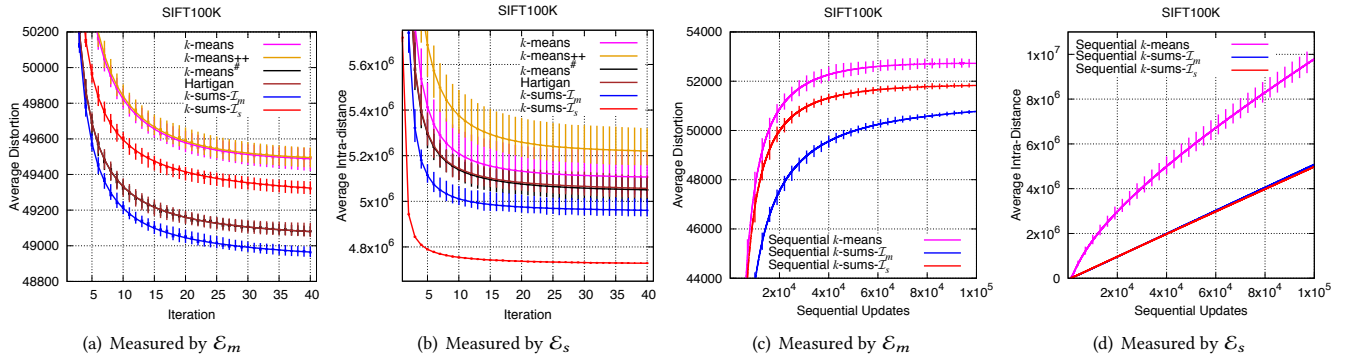
Following the practice in [2], the average distortion (or mean squared error [14]) is adopted to evaluate the clustering quality. It is nothing more than the function value of Eqn. 1 that is averaged by the size of dataset. The lower the distortion is, the better the clustering quality is.

$$\mathcal{E}_m = \frac{\sum_{q(x_i)=r} \| C_r - x_i \|^2}{n} \qquad (18)$$

Similarly, $\mathcal{E}_s$ is introduced to evaluate to what extent target function Eqn. 10 is minimized.

$$\mathcal{E}_s = \frac{\sum_{r=1}^{k} \sum_{i,j \in S_r \& i < j} \| x_i - x_j \|^2}{n} \qquad (19)$$

Twenty-one datasets are used in the evaluation. The brief information about these datasets is summarized in Tab. 1. In the

(a) Measured by $\mathcal{E}_m$	(b) Measured by $\mathcal{E}_s$	(c) Measured by $\mathcal{E}_m$	(d) Measured by $\mathcal{E}_s$

Figure 2: The significance test for k-sums-$\mathcal{I}_m$ and k-sums-$\mathcal{I}_s$. The function values measured by $\mathcal{E}_m$ (figure (a)) and $\mathcal{E}_s$ (figure (b)) are calculated after each iteration. *128* runs are carried out for each method on SIFT100K. The candle chart is plotted based on $\mathcal{E}_m$ and $\mathcal{E}_s$ of 128 runs from each iteration. Notice that all the k-means variants minimize Eqn. 1 except for k-sums-$\mathcal{I}_s$. The function values measured by $\mathcal{E}_m$ and $\mathcal{E}_s$ that are produced by Sequential k-means and Sequential k-sums are shown in figure (c) and (d) respectively.

Table 1: Overview of Datasets

Datasets	Scale	Dim.
SIFT100K [14]	1×10^4	128
SIFT1M [14]	1×10^6	128
GloVe1M [23]	1.1×10^6	100
MSD [25]	0.99×10^6	60
SUSY [5]	5×10^6	19
UMD [32]	$[878 \sim 9, 558]$	$[2, 880 \sim 36, 306]$

first experiment, dataset SIFT100K [14] is adopted to perform significance test to confirm that the improvement achieved by our approach is not by random. In the second experiment, k-sums is tested on four large-scale datasets. The types of data range from image local features (SIFT1M) [14], vectorized text word features (GloVe1M) [23], to audio features (MSD) [25] and event descriptions (SUSY) [5]. In the last experiment, *15* document datasets (UMD) [32] are adopted. The documents are represented with TF/IDF model and are l_2-normalized. On this document clustering task, the performance is evaluated by entropy [32].

$$Entropy = \sum_{r=1}^{k} \frac{n_r}{n} \frac{1}{\log c} * \sum_{i=1}^{c} \frac{n_r^i}{n_r} * \log \frac{n_r^i}{n_r}, \qquad (20)$$

where c is the number of classes in the ground-truth, and n_r^i is the size of intersection between class i and cluster S_r. The entropy obtained from *15* document datasets are averaged for each method.

5.1 Significance Test

The initialization on k-means clustering is based on either random seeding or random label assignment. Moreover, the optimization is a stochastic procedure for the methods such as IKM, k-means$^\#$, Hartigan, and k-sums. For these two reasons, the clustering results from k-means and its variants vary from one run to another. The first experiment investigates the general performance trends of

k-sums-$\mathcal{I}_m$ and k-sums-$\mathcal{I}_s$ and the variations across different runs. The experiment is conducted on SIFT100K. For each considered method, *128* runs are undertaken. The cluster number k is fixed to *1,024*. $\mathcal{E}_m$ and $\mathcal{E}_s$ are calculated after one iteration.

The candle charts for four methods from $\mathcal{E}_m$ and $\mathcal{E}_s$ are shown in Fig. 2(a) and Fig. 2(b) respectively. The trend curves produced by Sequential k-means and Sequential k-sums with respect to $\mathcal{E}_m$ and $\mathcal{E}_s$ are shown in Fig. 2(c) and Fig. 2(d). As shown from the figure, k-sums-$\mathcal{I}_m$ and k-sums-$\mathcal{I}_s$ achieve the lowest function score with respect to their target functions after *3* iterations. The performance gap between our methods and the rest is much more significant than the possible variations between different runs. As k-sums-$\mathcal{I}_s$ is the only method that aims to minimize target function Eqn. 10, a wide performance gap is observed in Fig. 2(b). The performance from Hartigan nearly overlaps with that of k-means$^\#$. Although k-means$^\#$ addresses k-means clustering as a maximization problem, it behaves similarly as Hartigan [28] as both of them incrementally optimize the k-means target function in a monotonic manner. The performance gap between k-means and k-means++ is nearly invisible from Fig. 2(a). This indicates the improvement from seeding scheme is limited. In terms of online k-means, all the curves given by $\mathcal{E}_m$ and $\mathcal{E}_s$ rise up as more and more samples join in. This is because the overall function values of $\mathcal{E}_m$ and $\mathcal{E}_s$ increase as more samples are incorporated in the equation. As shown in Fig. 2(c) and Fig. 2(d), Sequential k-sums show the lowest function value in each iteration with respect to the corresponding target function. Moreover, they demonstrate a much narrower variation range than that of Sequential k-means.

5.2 Quality Evaluation on Various Data Types

In the second experiment, four large-scale datasets of various data types are adopted in the evaluation. They are SIFT1M, GloVe1M, MSD and SUSY. The general trends of $\mathcal{E}_m$ and $\mathcal{E}_s$ from k-means$^\#$, Hartigan, k-means++, k-sums-$\mathcal{I}_m$ and k-sums-$\mathcal{I}_s$ are studied on these datasets. Since k-means++ usually shows better clustering quality than k-means and many other variants, it is treated as the

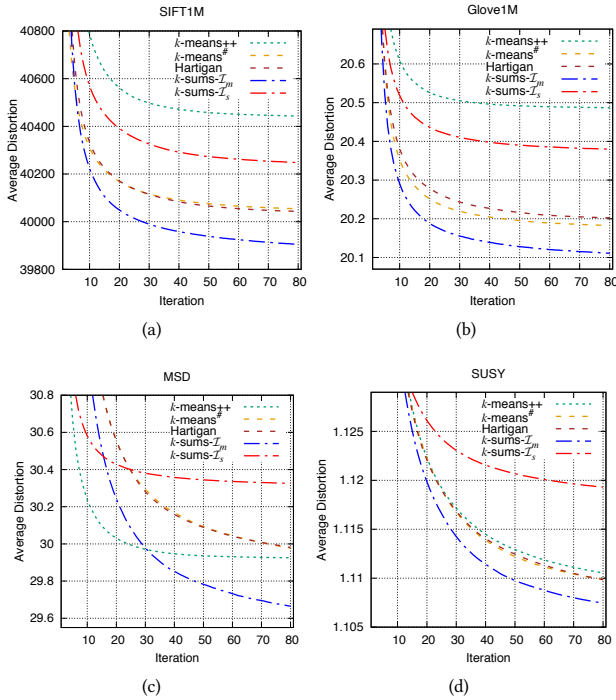**Figure 3: The general trend of function values measured by $\mathcal{E}_m$.**

Figure 4: The general trend of function values measured by $\mathcal{E}_s$.

comparison baseline. k is fixed to *10,000* for all the methods on each dataset. According to the previous experiment, the performance gap between the methods is more significant than the possible variations between different runs. It is therefore valid to only show the distortion curve of one run. The curves from $\mathcal{E}_m$ and $\mathcal{E}_s$ are shown in Fig. 3 and Fig. 4 respectively.

As shown in the figures, k-sums remains the best method with respect to two evaluation criterion, which is consistent with the previous observations. The function values from k-sums decrease at a much faster pace than the other three methods. Moreover, the performance gap gets wider as the number of iterations grows for k-sums-$\mathcal{I}_m$. This basically indicates that k-sums less likely gets trapped in a local minimum when driven by $\mathcal{I}_m$. In contrast, the rankings of cluster quality from k-means++ and k-means$^{\#}$ vary across different datasets. Generally the performance becomes saturated within *30* iterations for both of them. Similar as the previous observation, the performance trend from k-means$^{\#}$ and Hartigan remains similar. k-sums-$\mathcal{I}_s$ shows poorer performance than the others in Fig. 3. However, it converges quickly to a much better local optimum than the rest when measured by $\mathcal{E}_s$ (shown in Fig. 4). It is the only method that is designed to minimize target function Eqn. 10. This indicates two target functions considered in the paper are correlated yet still essentially different.

5.3 Document Clustering

In the third experiment, the performance of our methods is studied on the classic document clustering task. Fifteen datasets from UMD
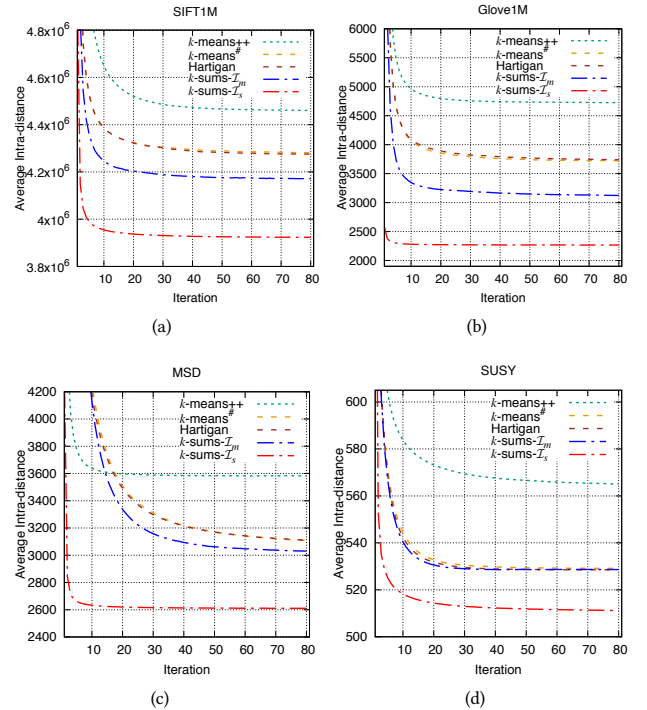
are adopted. In the experiments, k-means and the other five variants are considered. *Cosine* distance is adopted for all the methods. For the methods such as k-means, k-means++, k-means$^{\#}$ and k-sums, they could be undertaken in a bisecting manner, namely in the way of Alg. 2. As a result, the performance under the bisecting strategy for these methods is also reported. For each method, k is set to *5*, *10*, *15* and *20* on each dataset. Following the practice in [32], the clustering result of one method is selected from *10* runs with the lowest $\mathcal{E}_m$ or $\mathcal{E}_s$ for k-sums-$\mathcal{I}_s$. The average entropies of each method with both the direct k-way and the bisecting clustering are reported on Tab. 2(a) and Tab. 2(b) respectively.

As shown on the tables, k-sums driven by $\mathcal{I}_m$ and $\mathcal{I}_s$ outperform other methods considerably on the direct k-way case. On the bisecting case, k-sums-$\mathcal{I}_m$ still shows the best results, while k-sums-$\mathcal{I}_s$ shows similar performance as k-means$^{\#}$. k-sums-$\mathcal{I}_s$ shows relatively poor performance because it converges quickly and therefore is unable to reach a better local optimum in the bisecting case. IKM is the only method that shows close performance with k-sums. Unfortunately, it only works under *Cosine* distance [31, 32]. k-means$^{\#}$ and Hartigan perform similarly as they essentially optimize the target function in the similar manner. As explained in Section 4.2, both of them tend to be trapped in a local optimum easier than k-sums due to the tight constraint over the sample reallocation.

6 CONCLUSION

In this paper, the simple "egg-chicken" loop in k-means has been modified to an even simpler stochastic optimization procedure.

Table 2: Clustering performance on UMD *15* document datasets

(a) Clustering performance by direct *k*-way

	k=5	*k*=10	*k*=15	*k*=20
k-means [18]	0.539	0.443	0.402	0.387
k-means++ [1]	0.550	0.441	0.403	0.389
Mini-Batch [26]	0.585	0.488	0.469	0.475
LVQ [17]	0.800	0.761	0.681	0.674
k-means$^{\#}$ [31]	0.552	0.442	0.388	0.368
Hartigan [28]	0.451	0.358	0.331	**0.307**
IKM [32]	0.465	0.401	0.366	0.358
k-sums-$\mathcal{I}_m$	0.452	0.362	0.330	0.312
k-sums-$\mathcal{I}_s$	**0.445**	**0.357**	**0.325**	0.308

(b) Clustering performance by bisecting

	k=5	*k*=10	*k*=15	*k*=20
k-means [18]	0.532	0.438	0.410	0.373
k-means++ [1]	0.507	0.422	0.400	0.379
k-means$^{\#}$ [31]	0.514	0.388	0.353	0.329
IKM [32]	0.465	0.390	0.353	0.330
k-sums-$\mathcal{I}_m$	**0.449**	**0.367**	**0.335**	**0.311**
k-sums-$\mathcal{I}_s$	0.494	0.408	0.359	0.345

Different from *k*-means and many of its variants, the distortion minimization is driven by seeking for the better reallocation of each individual sample. The clusters are updated as soon as the reallocation of one sample leads to the lower distortion that is associated with the sample. A family of *k*-means variants are redefined under this optimization framework and show considerably better clustering quality. Moreover, another target function is proposed to handle the case that cluster centroid/mode cannot be defined. It is then solved under the same optimization procedure. To generalize this new clustering model to the generic metric space is our future research direction.

ACKNOWLEDGMENT

REFERENCES

[1] David Arthur and Sergei Vassilvitskii. 2007. k-means++: The Advantages of Careful Seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms.* 1027–1035.
[2] Yannis Avrithis, Yannis Kalantidis, Evangelos Anagnostopoulos, and Ioannis Z. Emiris. 2015. Web-scale image clustering revisited. In *ICCV.* 1502–1510.
[3] Olivier Bachem, Mario Lucic, Hamed Hassani, and Andreas Krause. 2016. Fast and Provably Good Seedings for k-Means. In *Advances in Neural Information Processing Systems 29.* 55–63.
[4] Bahman Bahmani, Benjamin Moseley, Andrea Vattani, Ravi Kumar, and Sergei Vassilvitskii. 2012. Scalable k-means++. *In Proceedings of the VLDB Endowment* 5, 7 (2012), 622–633.
[5] Pierre Baldi, Przemysław Sadowski, and D. Whiteson. 2014. Searching for exotic particles in high-energy physics with deep learning. *Nature communications* 5 (Jul. 2014), 4308.
[6] Andrei Broder, Lluis Garcia-Pueyo, Vanja Josifovski, Sergei Vassilvitskii, and Srihari Venkatesan. 2014. Scalable K-Means by ranked retrieval. In *Proceedings of the 7th ACM international conference on Web search and data mining.* 233–242.
[7] Andrei Broder, Lluis Garcia-Pueyo, Vanja Josifovski, Sergei Vassilvitskii, and Srihari Venkatesan. 2014. Scalable K-Means by Ranked Retrieval. In *Proceedings of the 7th ACM international conference on Web search and data mining.* 233–242.
[8] Yizong Cheng. 1995. Mean shift, mode seeking, and clustering. *Trans. PAMI* 17, 8 (Aug. 1995), 790–799.
[9] Martin Ester, Hans peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In IEEE *Transactions on Knowledge and Data Engineering.* 226–231.
[10] A. Goswami, Ruoming Jin, and G. Agrawal. 2004. Fast and exact out-of-core k-means clustering. In *Fourth IEEE International Conference on Data Mining.* 83–90.
[11] John A. Hartigan. 1975. *Clustering Algorithms (Probability & Mathematical Statistics).* John Wiley & Sons Inc.
[12] Anil K. Jain and Richard C. Dubes. 1988. *Algorithms for Clustering Data.*
[13] A. K. Jain, M. N. Murty, and P. J. Flynn. 1999. Data Clustering: A Review. *Comput. Surveys* 31, 3 (Sep. 1999), 264–323.
[14] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. 2011. Product quantization for nearest neighbor search. *Trans. PAMI* 33, 1 (Jan. 2011), 117–128.
[15] Hervé Jégou, Florent Perronnin, Matthijs Douze, Jorge Sánchez, Patrick Pérez, and Cordelia Schmid. 2012. Aggregating local descriptors into compact codes. *Trans. PAMI* 34, 9 (Sep. 2012), 1704–1716.
[16] Leonard Kaufman and Peter J. Rousseeuw. 1987. Clustering by Means of Medoids. *Reports of the Faculty of Mathematics and Informatics* 87 (1987), 405–416.
[17] T. Kohonen, M. R. Schroeder, and T. S. Huang (Eds.). 2001. *Self-Organizing Maps* (3rd ed.). Springer-Verlag New York, Inc., Secaucus, NJ, USA.
[18] Stuart P. Lloyd. 1982. Least Squares Quantization in PCM. *IEEE Trans. Information Theory* 28 (Mar. 1982), 129–137.
[19] MacQueen, James, et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability,* Vol. 1. 281–297.
[20] J. MacQueen. 1967. Some Methods for classification and Analysis of Multivariate Observations. In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability.* 281–297.
[21] C. Otto, D. Wang, and A. Jain. 2017. Clustering Millions of Faces by Identity. *Trans. PAMI* (Mar. 2017), 1–14.
[22] Dan Pelleg and Andrew Moore. 1999. Accelerating Exact K-means Algorithms with Geometric Reasoning. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* 277–281.
[23] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP).* 1532–1543. http://www.aclweb.org/anthology/D14-1162
[24] Alex Rodriguez and Alessandro Laio. 2014. Clustering by fast search and find of density peaks. *Science* 344, 6191 (2014), 1492–1496.
[25] Alexander Schindler, Rudolf Mayer, and Andreas Rauber. 2012. Facilitating comprehensive benchmarking experiments on the million song dataset. In *In Proceedings of the 13th International Society for Music Information Retrieval Conference.* 469–474.
[26] D. Sculley. 2010. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web.* 1177–1178.
[27] Noam Slonim, Ehud Aharoni, and Koby Crammer. 2013. Hartigan's K-means versus Lloyd's K-means: is it time for a change?. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence.* 1677–1684.
[28] Matus Telgarsky and Andrea Vattani. 2010. Hartigan's Method: k-means Clustering without Voronoi. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics.* 820–827.
[29] Ulrike von Luxburg. 2007. A tutorial on spectral clustering. *Statistics and Computin* 17, 4 (Aug. 2007), 395–416.
[30] Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J. McLachlan, Angus Ng, Bing Liu, Philip S. Yu, Zhi-Hua Zhou, Michael Steinbach, David J. Hand, and Dan Steinberg. 2007. Top 10 Algorithms in Data Mining. *Knowledge and Information System* 14, 1 (Dec. 2007), 1–37.
[31] Wan-Lei Zhao, Cheng-Hao Deng, and Chong-Wah Ngo. 2018. k-means: a revisit. *Neurocomputing* 291 (2018), 195–206.
[32] Ying Zhao and Geoge Karypis. 2004. Empirical and Theoretical Comparisons of Selected Criterion Functions for Document Clustering. *Machine Learning* 55 (Jun. 2004), 311–331.
[33] Ying Zhao and George Karypis. 2005. Hierarchical Clustering Algorithms for Document Datasets. *Data Mining and Knowledge Discovery* 10, 2 (Mar. 2005), 141–168.