Neurocomputing 291 (2018) 195–206

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

k-means: A revisit

Wan-Lei Zhao[a,*], Cheng-Hao Deng[a], Chong-Wah Ngo[b]

[a] Fujian Key Laboratory of Sensing and Computing for Smart City, and the School of Information Science and Engineering, Xiamen University, Xiamen 361005, PR China

[b] Department of Computer Science, City University of Hong Kong, Hong Kong

ARTICLE INFO

Article history:
Received 3 December 2016
Revised 12 February 2018
Accepted 21 February 2018
Available online 28 February 2018

Communicated by Deng Cai

Keywords:
Clustering
k-means
Incremental optimization

ABSTRACT

Due to its simplicity and versatility, *k*-means remains popular since it was proposed three decades ago. The performance of *k*-means has been enhanced from different perspectives over the years. Unfortunately, a good trade-off between quality and efficiency is hardly reached. In this paper, a novel *k*-means variant is presented. Different from most of *k*-means variants, the clustering procedure is driven by an explicit objective function, which is feasible for the whole l_2-space. The classic egg-chicken loop in *k*-means has been simplified to a pure stochastic optimization procedure. The procedure of *k*-means becomes simpler and converges to a considerably better local optima. The effectiveness of this new variant has been studied extensively in different contexts, such as document clustering, nearest neighbor search and image clustering. Superior performance is observed across different scenarios.

1. Introduction

Clustering problems arise from variety of applications, such as documents/web pages clustering [1], pattern recognition, image linking [2], image segmentation [3], data compression via vector quantization [4] and nearest neighbor search [5–7]. In the last three decades, various clustering algorithms have been proposed. Among these algorithms, *k*-means [8] remains a popular choice for its simplicity, efficiency and moderate but stable performance across different problems. It was known as one of top ten most popular algorithms in data mining [9]. On one hand, *k*-means has been widely adopted in different applications. On the other hand, continuous efforts have been devoted to enhance the performance *k*-means as well.

Despite its popularity, it actually suffers from several latent issues. Although the time complexity is linear to data size, traditional *k*-means is still not sufficiently efficient to handle the web-scale data. In some specific scenarios, the running time of *k*-means could be even exponential in the worst case [10,11]. Moreover, *k*-means usually only converges to local optima. As a consequence, recent research has been working on either improving its clustering quality [12,13] or efficiency [2,13–19]. *k*-means has been also tailored to perform web-scale image clustering [2,20].

There are in general three steps involved in the clustering procedure. Namely, 1. initialize *k* cluster centroids; 2. assign each sample to its closest centroid; 3. recompute cluster centroids with assignments produced in Step 2 and go back to Step 2 until convergence. This is known as *Lloyd* iteration procedure [8]. The iteration repeats Step 2 and Step 3 until the centroids do not change between two consecutive rounds. Given $C_{1\ldots k} \in R^d$ are cluster centroids, $\{x_i \in R^d\}_{i=1\ldots n}$ are samples to be clustered, above procedure essentially minimizes the following objective function:

$$\min \sum_{q(x_i)=r} \| C_r - x_i \|^2. \quad (1)$$

In Eq. (1), function $q(\cdot)$ returns the closest centroid for sample x_i. Unfortunately, searching an optimal solution for the above objective function is NP-hard. In general *k*-means only converges to local minimum [21]. The reason that *k*-means maintains its popularity is mainly due to its linear complexity in terms of the number of samples to be clustered. The complexity is $O(t \cdot k \cdot n \cdot d)$, given t as the number of iterations to converge. Compared with other well-known clustering algorithms such as DBSCAN [22], Mean shift [23] and clusterDP [24], this complexity is considerably low. However, the efficiency of traditional *k*-means cannot cope with the massive growth of data in Internet. In particular, in the case that the size of data (n), the number of clusters (k) and the dimension (d) are all very large, *k*-means becomes unbearably slow. The existing efforts [16,18] in enhancing the scalability of *k*-means for web-scale tasks often come with price of lower clustering quality. On the other hand, *k*-means++ proposed in [12,17] focuses on enhancing the clustering quality by a careful design of the

* Corresponding author.
E-mail addresses: wlzhao@xmu.edu.cn (W.-L. Zhao), chenghaodeng@stu.xmu.edu.cn (C.-H. Deng), cscwngo@gapps.cityu.edu.hk (C.-W. Ngo).

https://doi.org/10.1016/j.neucom.2018.02.072
0925-2312/

initialization procedure. However, k-means slows down as a few rounds of scanning over the dataset is still necessary in the initialization.

In this paper, a novel variant of k-means is proposed, which aims to make a better trade-off between clustering quality and efficiency. Inspired by the work in [1], a novel objective function is derived from Eq. (1). With the development of this objective function, the traditional k-means iteration procedure has been revised to a simpler form, in which the costly initial assignment becomes unnecessary. In addition, driven by the objective function, sample is moved from one cluster to another cluster when we find this movement leads to higher objective function score, which is known as incremental clustering [1,25]. These modifications lead to several advantages.

- k-means clustering without initial assignment results in better quality as well as higher speed efficiency.
- k-means iteration driven by an explicit objective function converges to considerably lower clustering distortion in faster pace.
- Different from traditional k-means, it is not necessary to assign a sample to its closest centroid in each iteration, which also leads to higher speed.

In addition, when clustering undertaken in hierarchical bisecting fashion, the proposed method achieves the highest scalability among all top-down hierarchical clustering methods. Extensive experiments are conducted to contrast the performance of proposed method with k-means and its variants including tasks document clustering [1], nearest neighbor search (NNS) with product quantization [4] and image clustering.

The remainder of this paper is organized as follows. The reviews about representative works on improving the performance of traditional k-means are presented in Section 2. In Section 3, the clustering objective functions are derived based on Eq. (1). Based on the objective function, Section 4 presents the clustering method. Extensive experiment studies over proposed clustering method are presented in Section 5. Section 6 concludes the paper.

2. Related works

Clustering is a process of partitioning a set of samples into a number of groups without any supervised training. Due to its versatility in different contexts, it has been studied in the last three decades [26]. As the introduction of Web 2.0, millions of data in Internet has been generated on a daily basis. Clustering becomes one of the basic tools to process such big volume of data. As a consequence, traditional clustering methods have been shed with new light. People are searching for clustering methods that are scalable [16–18,27] to web-scale data. In general, boosting the performance of traditional k-means becomes the major trend due to its simplicity and relative higher efficiency over other clustering methods.

In general, there are two major ways to enhance the performance of k-means. For the first kind, the aim is to improve the clustering quality. One of the important work comes from Bahmani et al. [12,17]. The motivation is based on the observation that k-means converges to a better local optima if the initial cluster centroids are carefully selected. According to [12], k-means iteration also converges faster due to the careful selection on the initial cluster centroids. However, in order to adapt the initial centroids to the data distribution, k rounds of scanning over the data are necessary. Although the number of scanning rounds has been reduced to a few in [17], the extra computational cost is still inevitable.

In each k-means iteration, the processing bottleneck is the operation of assigning each sample to its closest centroid. The iteration becomes unbearably slow when both the size and the dimension of the data are very large. Considering that this is a nearest neighbor search problem, Kanungo et al. [14] proposed to index dataset in a KD Tree [28] to speed-up the sample-to-centroid nearest neighbor search. However, this is only feasible when the dimension of data is in few tens. Similar scheme has been adopted by Pelleg and Moore [29]. Unfortunately, due to the curse of dimensionality, this method becomes ineffective when the dimension of data grows to a few hundreds. A recent work [18] takes similar way to speed-up the nearest neighbor search by indexing dataset with inverted file structure. During the iteration, each centroid is queried against all the indexed data. Thanks to the efficiency of inverted file structure, one to two orders of magnitude speed-up is observed. However, inverted file indexing structure is only effective for sparse vectors.

Alternatively, the scalability issue of k-means is addressed by subsampling over the dataset during k-means iteration. Namely, methods in [16,30] only pick a small portion of the whole dataset to update the cluster centroids each time. For the sake of speed efficiency, the number of iterations is empirically set to small value. It is therefore possible that the clustering terminates without a single pass over the whole dataset, which leads to higher speed but also higher clustering distortion. Even though, when coping with high dimensional data in big size, the speed-up achieved by these methods is still limited.

Apart from above methods, there is another easy way to reduce the number of comparisons between the samples and centroids, namely performing clustering in a top-down hierarchical manner [1,31,32]. Specifically, the clustering solution is obtained via a sequence of repeated bisections. The clustering complexity of k-means is reduced from $O(t \cdot k \cdot n \cdot d)$ to $O(t \cdot log(k) \cdot n \cdot d)$. This is particularly significant when n, d and k are all very large. In addition to that, another interesting idea from [1,32] is that cluster centroids are updated incrementally [1,25]. Moreover, the update process is explicitly driven by an objective function (called as criterion function in [1,32]). Unfortunately, objective functions proposed in [1,31,32] are based on the assumption that input data are in unit length. The clustering method is solely based on *Cosine* distance, which makes the clustering results unpredictable when dealing with data in the general l_2-space.

In this paper, a new objective function is derived directly from Eq. (1), which makes it suitable for the whole l_2-space. In other word, objective function proposed in [1] is the special case of our proposed form. Based on the proposed objective function, conventional egg-chicken k-means iteration is revised to a simpler form. On one hand, when applying the revised iteration procedure in direct k-way clustering, k-means is able to reach to considerably lower clustering distortion within only a few rounds. On the other hand, as the iteration procedure is undertaken in top-down hierarchical clustering manner (specifically bisecting), it shows faster speed while maintaining relatively lower clustering distortion in comparison to traditional k-means and most of its variants.

3. Clustering objective functions

In this section, the clustering objective functions upon which our k-means method is built are presented. Basically, two objective functions that aim to optimize the clustering results from different aspects are derived. Furthermore, we also show that these two objective functions can be reduced to a single form.

3.1. Preliminaries

In order to facilitate the discussions that are followed, several variables are defined. Throughout the paper, the size of input data is given as n, while the number of clusters to be produced is given as k. The partition formed by a clustering method is represented as $\{S_1, \ldots, S_r \ldots, S_k\}$. Accordingly, the sizes of clusters are given as $n_1, \ldots, n_r, \ldots, n_k$. The composite vector of a cluster is defined as