Random Projection for k-means Clustering

Sami Sieranoja and Pasi Fränti(✉)

School of Computing, University of Eastern Finland, Joensuu, Finland
{sami.sieranoja,pasi.franti}@uef.fi

Abstract. We study how much the k-means can be improved if initialized by random projections. The first variant takes two random data points and projects the points to the axis defined by these two points. The second one uses furthest point heuristic for the second point. When repeated 100 times, cluster level errors of a single run of k-means reduces from CI = 4.5 to 0.8, on average. We also propose simple projective indicator that predicts when the projection-heuristic is expected to work well.

Keywords: Clustering · Random projection · K-means

1 Introduction

K-means groups *N* data points into *k* clusters by minimizing the sum of squared distances between the data points and their nearest cluster centers (*centroid*). It takes any initial solution as an input and then improves it iteratively. A random set of points is the most common choice for the initialization. K-means is very popular because of its simple implementation. It has also been used as a part of other clustering algorithms such as genetic algorithms [14, 25], random swap [16], spectral clustering [34] and density clustering [3].

The best property of k-means is its excellent fine-tuning capability. Given a rough location of initial cluster centers, it can optimize their locations locally. The main limitation is that k-means cannot optimize the locations globally. Problems appear especially when clusters are well separated, or when stable clusters block the movements of the centroids, see Fig. 1. A poor initialization can therefore lead to an inferior local minimum.

To compensate the problem, alternative initialization heuristics have been considered. These include *Maxmin* heuristic, *sorting* heuristic, *density* and *projection* heuristics. Comparative studies [8, 22, 27, 30] have found that no single technique would outperform the others in all cases. A clear state-of-the-art is missing.

Another way to improve k-means is to repeat it several times [11]. The idea is simply to restart k-means from different initial solutions, and then keeping the best result. This requires that the initialization technique include some randomness in the process to produce different initial solutions. We refer this as *repeated k-means* (RKM).

For the initialization, projection-based heuristics aim at finding one-dimensional projection that would allow the data to be divided to roughly equal size clusters [1, 31, 35]. Most obvious choices are *diagonal* and the *principal* axis. Diagonal axis works well

L. Rutkowski et al. (Eds.): ICAISC 2018, LNAI 10841, pp. 680–689, 2018.
https://doi.org/10.1007/978-3-319-91253-0_63

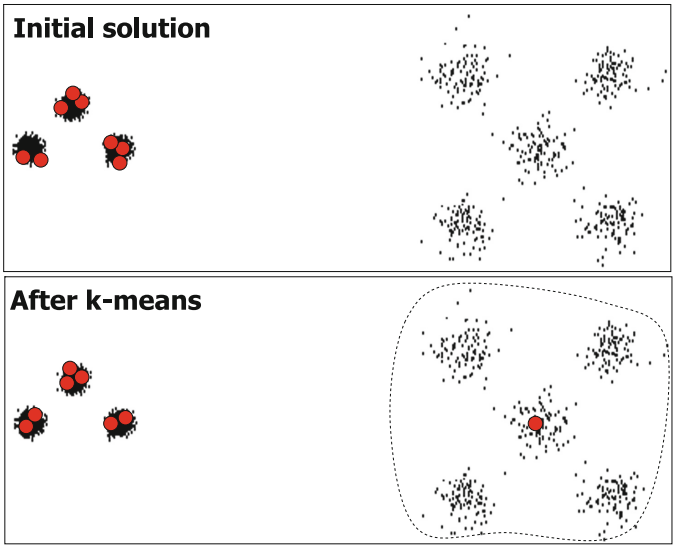

Fig. 1. Example of sub-optimal convergence of k-means.

with image data when the points are strongly correlated. Principal axis finds the direction with maximum variance. It has been widely used also in split-based algorithms that iteratively split one cluster at a time [4, 15, 23, 35].

In general, projection heuristic transforms the clustering to an easier segmentation problem, which can be optimally solved for a given objective function using dynamic programming [32]. However, it is unlikely that the data points would nicely fit along a linear axis in the space. Even a more complex non-linear principal curve was also considered in [9] but the final result still depended on the fine-tuning by k-means. Open questions are when the projection-based technique is expected to work, and how to choose the best projection axis.

In this paper, we study two simple variants to be used with repeated k-means: *random* and *furthest point* projections. The first takes two random data points without any sanity checks. The key idea is the randomness; single selection may provide poor initialization but when repeating several times, the chances to find one good initialization increases significantly.

The second heuristic is slightly more deterministic but still random. We start by selecting a random point, which will be the first reference point. We then calculate its furthest point and select it as the second reference point. The projection axis is the line passing through the two reference points. We again rely on randomness, but now the choices are expected to be more sensible, potentially providing better results using fewer trials.

The projection technique works when the data has one-dimensional structure, see Fig. 2. Otherwise, it may not provide additional benefit compared to the simple random initialization. We therefore introduce a simple *projective indicator* that predicts how well the data can be represented by a linear projection. This indicator can also be used

to evaluate the goodness of the random projection before performing the actual clustering, which can speed-up the process.

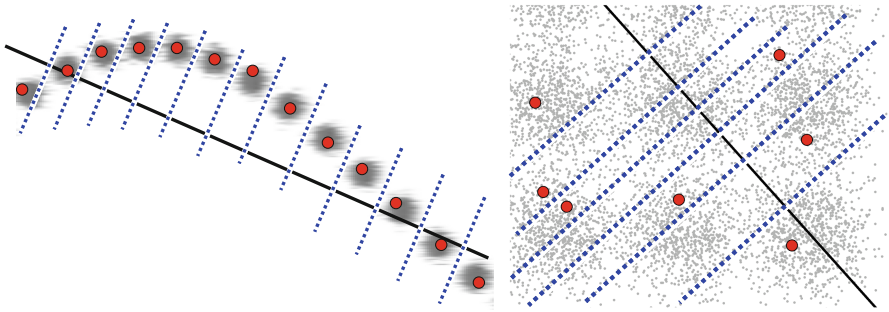

Fig. 2. The principle of the projection heuristic: when it works (left) and when not (right).

2 Projection-Based Initialization

The goal is to find a simple one-dimensional projection that can be used to create trivial equal size clusters along the axis, and yet, provide meaningful initial clustering. We merely seek to find a better initialization to k-means than the random heuristic, and generating it as simply as possible.

2.1 One-Dimensional Projections

The original idea was to project the data points on a single direction and then solving the initial partition along this direction [2]. Simplest way to do it is to sort the data according to the chosen axis, and then select every $(N/k)^{th}$ point. In [33], the points are sorted according to their distance to origin. If the attributes are non-negative, then this is essentially the same as projecting the data to the diagonal axis. Such projection is trivial to implement by calculating the average of the attribute values. The diagonal axis has also been used for speeding-up nearest neighbor searches in clustering [28].

In [1], the points are sorted according to the dimension with the largest variance. This adapts to the data slightly better than just using the diagonal. A more general approach would be to use *principal axis*, which is the direction along which the variance is maximal. It has also been used in divisive clustering algorithms [4, 15, 23, 35]. However, calculation of the principal axis takes $O(DN)$-$O(D^2N)$ depending on the variant [15]. To speed-up the process, only diagonal covariance matrix was computed in [31]. A more complex *principal curve* has also been used for clustering [9].

2.2 Random Projection in Higher Dimensions

Higher dimensional projections have also been considered. In [12], the data was projected in 2–dimensional subspace by selecting the highest variance attribute as the first dimension, and as second the one that has minimum absolute correlation with the

first. According to [10], Gaussian clusters can be projected into O(log k) dimensions while still preserving the approximate level of separation between the clusters. It was shown that random projection can change the shape of the clusters to be more spherical. The following methodology was also suggested: perform random projection into a lower dimensional space, perform clustering, and use the result for obtaining clustering in the high dimensional space. This is exactly what we do here: perform one-dimensional projection followed by k-means in the full space.

Random projections have also been used in clustering ensemble [13] where individual clustering result is aggregated into a combined similarity matrix. Final clustering is then obtained by agglomerative clustering using the similarity matrix instead of the original data. This was shown to provide better and more robust clustering than PCA + EM (k-means variant for Gaussian clusters).

The diversity of the individual solution was also found to be an important element of the algorithm in [13]. We had similar observation in [18], where having some level of randomness in the algorithm was essential in swap-based clustering. Both these results suggest that random projections might be suitable for k-means initialization better than some fixed deterministic projection.

In [7], random projections were motivated by the need for speed in clustering of data streams. Quasilinear time complexity, O(N·log$N$), was set as a requirement. The speed was also the main motivation in [5] where the problem was considered as feature extraction, which aims at selecting such (linear) combination of the original features that the clustering quality would not be compromised.

In [6], k-means is applied several times, each time increasing the dimensionality after the convergence of k-means. In other words, least accurate version of the data is clustered first, and at each of the following steps, the accuracy is gradually increased. This approach is analogous to cooling in simulated annealing.

2.3 Random Projection

Although the above-mentioned solutions are also based on random projections, they do it in higher-dimensions. Their main goal was not to loose the clustering accuracy. However, we merely seek a simple initialization to k-means. We perform the random projection in one-dimensional space for two particular reasons:

- Simplicity
- Speed-up benefits

The random projection works as follows:

1. Select random data point r_1
2. Select random data point r_2

Unlike higher dimensional projections, one-dimensional projection induces unique sorting of the data. This could also be used to speed-up the nearest centroid searches without any further loss in the accuracy [28]. Significant reduction of processing time was reported when used jointly with the *centroid activity detection* [24]: speed-up factor

of 27:1 (Bridge dataset) and 62:1 (Miss America dataset); both 16-dimensional image data. Although speed-up is not our goal, it is worth to taken into account.

2.4 Furthest Point Projection

Our second heuristic is based on the classical furthest point heuristic where the next centroid is always selected as the one that is furthest to all other centroids [21]. However, instead of selecting every centroid as the next furthest, we only need to find two reference points. To involve some level of randomness in the process, we select the first reference point randomly:

1. Select random data point r_1
2. Find the furthest point r_2 of r_1

The projection axis is then defined by the reference points r_1 and r_2. The process is illustrated in Fig. 3. Since the first point is chosen randomly, the heuristic may create different solution in different runs, which is the purpose.

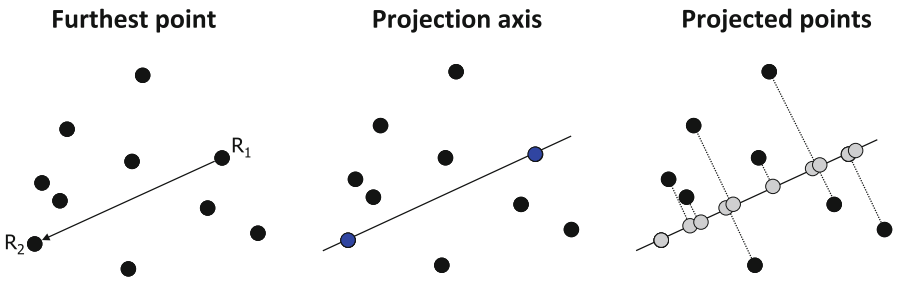

Fig. 3. Sample data and the projections.

2.5 Repeated K-means

Both heuristics are used with the k-means algorithm as follows. We repeat the algorithm $R = 100$ times, and select the best final result. The pseudo code of this overall algorithm is shown in Fig. 4. For a single projection axis, we predict its goodness as follows:

$$\Delta R = \sum_{a,b} (dist(a,b))/N^2, \forall a,b \tag{1}$$

$$\Delta L = \sum_{x} dist\left(x, p_x\right)/N \tag{2}$$

$$\text{Projective} = \frac{\Delta R}{\Delta L} \tag{3}$$

```
Repeated K-means(x):
f_best = ∞;
FOR r=1 to 100 DO
    y ← Projection(x);
    y ← Sort(y);
    FOR i=1 to k DO
        mid = (int) i*(N/k) + 0.5;
        c[i] = y[mid];
    f ← Evaluate(x,c);
    IF f < f_best THEN  f ← f_best

RandomProjection(x)
r1 ← RandomNumber(1,N);
r2 ← RandomNumber(1,N);
FOR i=1 to N DO
    p[i] ← Project(x[i], r1, r2);

FurthestPointProjection(x):
r1 ← Random(1,N);
r2 ← FindFurthest(X, x[r0]);
FOR i=1 to N DO
    p[i] ← Project(x[i], r1, r2);
```

Fig. 4. Pseudo code of the considered projection-based techniques.

The divider ΔR is the average pairwise distance of all data points in the original space, and the nominator ΔL is the average distance of the points (x) to the projection axis (p_x).

3 Experiments

We compare the two projection-based heuristics against the two most common heuristics: *random* initialization, *Maxmin* heuristic [21]. We use the datasets shown in Table 1. These include several artificial datasets, and real image datasets.

In case of artificial data with known ground truth, the success of the clustering is evaluated by *Centroid Index* (CI) [17], which counts how many cluster centroids are wrongly located. Value CI = 0 indicates correct clustering. Reference results are also given for single run of k-means, and when k-means is re-started 100 times (repeated). In case of image data, we calculate sum of squared errors (Fig. 5).

Table 1. Datasets used in the experiments.

Data set	Ref.	Type of data	Points (N)	Clusters (k)	Dimension (d)
Bridge	[16]	Image blocks	4096	256	16
House	[16]	*RGB* image	34112	256	3
Miss America	[16]	Residual blocks	6480	256	16
Europe		Differential values	169673	256	2
Birch, Bich2	[36]	Artificial	100000	100	2
$S_1 - S_4$	[19]	Artificial	5000	15	2
$A_1 - A_3$	[26]	Artificial	3000-7500	20-50	2
Unbalance	[29]	Artificial	6500	8	2
DIM-32	[20]	Artificial	1024	16	32

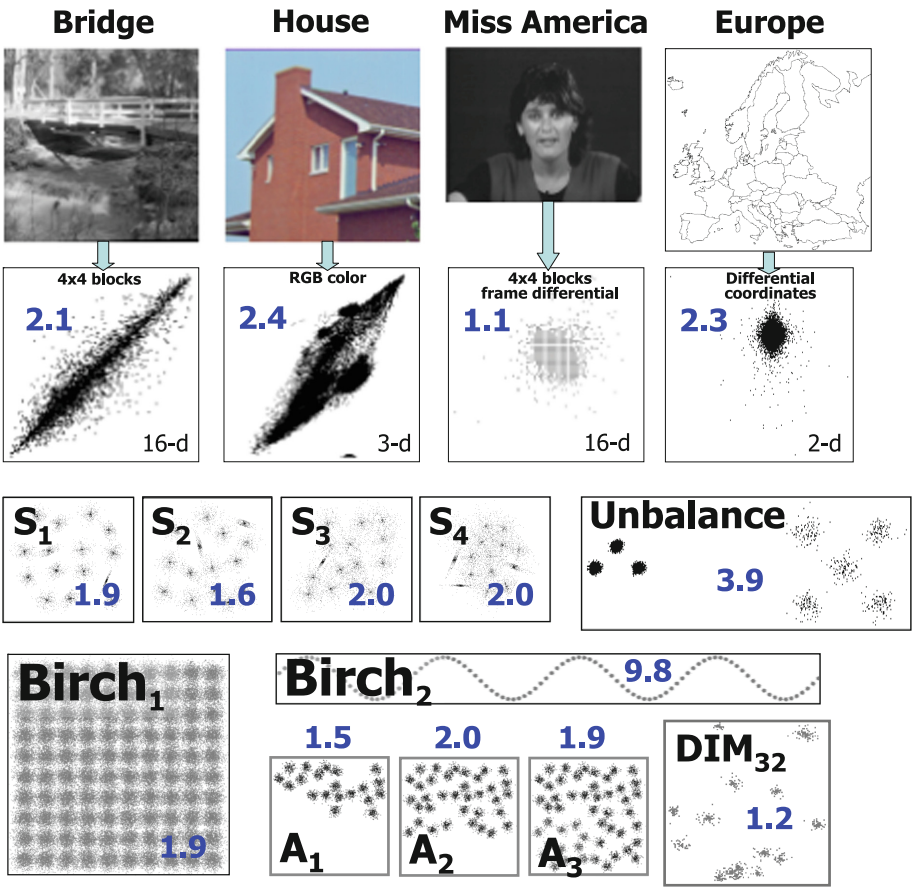

Fig. 5. Datasets and their calculated projective values according to (3).

The results are summarized in Table 2 where the correct clustering results are emphasized by green color. As expected, single run of k-means rarely finds the correct clustering regardless of the initialization. On average, CI = 4.5 centroids are wrongly located with random heuristic, and about CI = 2 with the other heuristics.

Table 2. Performance comparison of the various k-means initialization heuristics: random, furthest point, and the two projection-based: random (Proj-Rand) and the furthest point (Proj-FP).

Artificial data (Single run):

Method	S1	S2	S3	S4	A1	A2	A3	Unb	B1	B2	D32	Av.
Random	1.9	1.4	1.3	0.9	2.5	4.6	6.6	3.9	6.7	16.6	3.6	**4.5**
Furthest point	0.7	1.0	0.7	1.0	1.0	2.6	2.9	0.9	5.5	7.3	0.0	**2.1**
Proj-Rand	1.2	0.9	0.8	0.6	1.2	3.3	5.2	4.0	5.3	0.2	0.9	**2.2**
Proj-FP	1.2	0.9	0.8	0.6	1.1	3.3	5.0	4.0	5.4	0.0	1.0	**2.1**

Artificial data (100 repeats):

Method	S1	S2	S3	S4	A1	A2	A3	Unb	B1	B2	D32	Av.
Random	0.0	0.0	0.0	0.0	0.3	1.8	2.9	2.9	2.8	11	1.1	**2.0**
Furthest point	0.0	0.0	0.0	0.0	0.0	0.5	0.6	0.0	2.8	3.9	0.0	**0.7**
Proj-Rand	0.0	0.0	0.0	0.0	0.0	0.9	2.0	3.9	1.9	0.0	0.0	**0.8**
Proj-FP	0.0	0.0	0.0	0.0	0.0	0.8	1.9	4.0	1.8	0.0	0.0	**0.8**

Image data (100 repeats):

Method	Bridge	House	Miss America	Europe
Random	176.71	**6.43**	5.83	3.66
Furthest point	**172.09**	6.93	**5.68**	**3.56**
Proj-Rand	176.66	6.44	5.83	3.65
Proj-FP	176.83	6.46	5.83	3.65

However, when the k-means is repeated 100 times, the results are significantly better. The projection-based heuristics solve now 7 of the 11 artificial datasets, having CI = 0.8 centroids incorrectly located, on average. We also observe that the different heuristics work better with different datasets. In specific, Maxmin heuristic works best when the clusters have different densities and the projection-based better with data having high projective values (Birch2).

None of the image sets have high projective values, and therefore, the projection heuristic works basically as the random heuristic. There is also no significant difference between the two projection-techniques. The furthest point projection works slightly better but when repeated, the minor difference disappear. Figure 6 demonstrates what kinds of projections are generated with the A2 dataset.

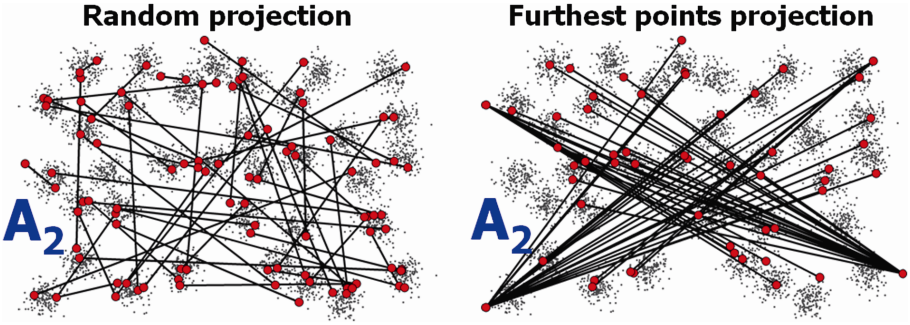

Fig. 6. Examples of the random projections axes (left), and the furthest point projection axes (right).

4 Conclusions

Both the random projection and furthest points heuristics outperform random initialization, and provide competitive performance to the Maxmin heuristic. The performance of the heuristics can be predicted by the projective indicator, which helps to recognize when the data is suitable to be projected. This could also be useful in split-based algorithms where the data is hierarchically split into smaller clusters.

References

1. Al-Daoud, M.B., Roberts, S.A.: New methods for the initialisation of clusters. Pattern Recogn. Lett. **17**(5), 451–455 (1996)
2. Anderberg, M.R.: Cluster Analysis for Applications. Academic Press, NewYork (1973)
3. Bai, L., Cheng, X., Liang, J., Shen, H., Guo, Y.: Fast density clustering strategies based on the k-means algorithm. Pattern Recogn. **71**, 375–386 (2017)
4. Boley, D.: Principal direction divisive partitioning. Data Min. Knowl. Disc. **2**(4), 325–344 (1998)
5. Boutsidis, C., Zouzias, A., Mahoney, M.W., Drineas, P.: Randomized dimensionality reduction for k-means clustering. IEEE Trans. Inf. Theory **61**(2), 1045–1062 (2015)
6. Cardoso, A., Wichert, A.: Iterative random projections for high-dimensional data clustering. Pattern Recogn. Lett. **33**, 1749–1755 (2012)
7. Carraher, L.A., Wilsey, P.A., Moitra, A., Dey, S.: Random projection clustering on streaming data. In: IEEE International Conference on Data Mining Workshops, pp. 708–715 (2016)
8. Celebi, M.E., Kingravi, H.A., Vela, P.A.: A comparative study of efficient initialization methods for the k-means clustering algorithm. Expert Syst. Appl. **40**, 200–210 (2013)
9. Cleju, I., Fränti, P., Wu, X.: Clustering based on principal curve. In: Kalviainen, H., Parkkinen, J., Kaarna, A. (eds.) SCIA 2005. LNCS, vol. 3540, pp. 872–881. Springer, Heidelberg (2005). https://doi.org/10.1007/11499145_88
10. Dasgupta, S.: Experiments with random projection. In: Uncertainty in Artificial Intelligence, pp. 143–151 (2000)
11. Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis. Wiley, New York (1973)
12. Erisoglu, M., Calis, N., Sakallioglu, S.: A new algorithm for initial cluster centers in k-means algorithm. Pattern Recogn. Lett. **32**(14), 1701–1705 (2011)

13. Fern, X.Z., Brodley, C.E.: Random projection for high dimensional data clustering: a cluster ensemble approach. In: International Conference on Machine Learning (ICMC), Washington, DC (2003)
14. Fränti, P.: Genetic algorithm with deterministic crossover for vector quantization. Pattern Recogn. Lett. **21**(1), 61–68 (2000)
15. Fränti, P., Kaukoranta, T., Nevalainen, O.: On the splitting method for VQ codebook generation. Opt. Eng. **36**(11), 3043–3051 (1997)
16. Fränti, P.: Efficiency of random swap clustering. J. Big Data **5**(13), 1–29 (2018)
17. Fränti, P., Rezaei, M., Zhao, Q.: Centroid index: cluster level similarity measure. Pattern Recogn. **47**(9), 3034–3045 (2014)
18. Fränti, P., Tuononen, M., Virmajoki, O.: Deterministic and randomized local search algorithms for clustering. In: IEEE International Conference on Multimedia and Expo, Hannover, Germany, pp. 837–840, June 2008
19. Fränti, P., Virmajoki, O.: Iterative shrinking method for clustering problems. Pattern Recogn. **39**(5), 761–765 (2006)
20. Fränti, P., Virmajoki, O., Hautamäki, V.: Fast agglomerative clustering using a k-nearest neighbor graph. IEEE Trans. Pattern Anal. Mach. Intell. **28**(11), 1875–1881 (2006)
21. González, R., Tou, J.: Pattern Recognition Principles. Addison-Wesley, Boston (1974)
22. He, J., Lan, M., Tan, C.-L., Sung, S.-Y., Low, H.-B.: Initialization of cluster refinement algorithms: a review and comparative study. In: IEEE International Joint Conference on Neural Networks (2004)
23. Huang, C.-M., Harris, R.W.: A comparison of several vector quantization codebook generation approaches. IEEE Trans. Image Process. **2**(1), 108–112 (1993)
24. Kaukoranta, T., Fränti, P., Nevalainen, O.: A fast exact GLA based on code vector activity detection. IEEE Trans. Image Process. **9**(8), 1337–1342 (2000)
25. Krishna, K., Murty, M.N.: Genetic k-means algorithm. IEEE Trans. Syst. Man Cybern. Part B **29**(3), 433–439 (1999)
26. Kärkkäinen, I., Fränti, P.: Dynamic local search algorithm for the clustering problem. Research Report A-2002-6 (2002)
27. Peña, J.M., Lozano, J.A., Larrañaga, P.: An empirical comparison of four initialization methods for the k-means algorithm. Pattern Recogn. Lett. **20**(10), 1027–1040 (1999)
28. Ra, S.-W., Kim, J.-K.: A fast mean-distance-ordered partial codebook search algorithm for image vector quantization. IEEE Trans. Circ. Syst. **40**, 576–579 (1993)
29. Rezaei, M., Fränti, P.: Set-matching methods for external cluster validity. IEEE Trans. Knowl. Data Eng. **28**(8), 2173–2186 (2016)
30. Steinley, D., Brusco, M.J.: Initializing k-means batch clustering: a critical evaluation of several techniques. J. Classif. **24**, 99–121 (2007)
31. Su, T., Dy, J.G.: In search of deterministic methods for initializing k-means and Gaussian mixture clustering. Intell. Data Anal. **11**(4), 319–338 (2007)
32. Wu, X.: Optimal quantization by matrix searching. J. Algorithms **12**(4), 663–673 (1991)
33. Wu, X., Zhang, K.: A better tree-structured vector quantizer. In: IEEE Data Compression Conference, Snowbird, UT, pp. 392–401 (1991)
34. Yan, D., Huang, L., Jordan, M.I.: Fast approximate spectral clustering. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 907–916, 2009
35. Yedla, M., Pathakota, S.R., Srinivasa, T.M.: Enhancing k-means clustering algorithm with improved initial center. Int. J. Comput. Sci. Inf. Technol. **1**(2), 121–125 (2010)
36. Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: a new data clustering algorithm and its applications. Data Min. Knowl. Disc. **1**(2), 141–182 (1997)