Quantization-based clustering algorithm

Santiago del Rey Juárez

May 2022

Abstract

In research, a task that is usually done is to compare the methods/algorithms that one proposes to other similar/competing methods to show empirically the benefits of our approach. Sometimes, when looking into the state of the art for the most recent methods in our area, we find that the implementations of such methods are not available. In these cases, we must understand and implement the competing algorithms to be able to test and compare them with ours. The goal of this practical work is to simulate this scenario by implementing the Quantization-based clustering algorithm (QBCA) [8] and comparing it with other well-known clustering algorithms. In particular, we will focus on the K-Means [5] algorithm and some of its variants since all of them, including QBCA, use the distance between samples to determine the clusters.

1 Introduction

In the field of unsupervised learning, clustering techniques have been extensively researched due to their usefulness for knowledge discovery, data mining, and image segmentation, among other tasks. In particular, there has been a lot of research effort spent in finding how to optimize this problem since most of the proposed algorithms suffer a performance downgrade with very large quantities of data.

We can find multiple families of clustering techniques in the literature (e.g. density-based, hierarchical). In this paper, we will focus on the K-Means family which is, at the same time, inside the partitional algorithms. These clustering techniques try to minimize the distance of each example to the centroid of the cluster. QBCA is based on the same optimization problem and proposes an approach to reduce the number of distance computations to efficiently process large quantities of data without losing quality.

The rest of the paper is structured as follows. Section 2 gives an overview of the main steps of the QBCA algorithm. Section 3 explains how the experiments were conducted and their results. Section 4 concludes the paper and reports future work.

2 QBCA

In this section, the main steps of the QBCA algorithm are described. For the mathematical demonstrations and further details refer to the original paper [8].

QBCA aims to reduce the number of distance computations required in the clustering process. As indicated in Fig. 1, the algorithm first performs a quantization process to associate the data points with their corresponding histogram bins. Then, it performs the cluster center initialization stage (CCI) to find a set of "good" initial centers of the clusters. Then, it applies the cluster center assignment stage (CCA) to reduce the distance computation cost between the points and the cluster centers. Finally, the cluster centers are recomputed. These two last steps are repeated until the termination criteria is met. In Fig. 2 we can see the algorithm's pseudocode where the parameter δ is used to determine the termination condition.

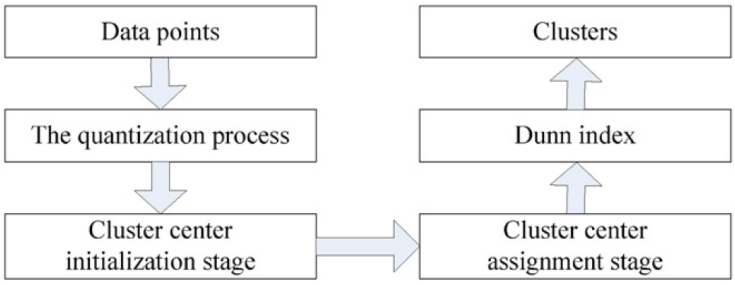

Figure 1: The framework of the quantization-based clustering algorithm. Source: [8]

Algorithm QBCA (Point set P, Seed number k, Threshold ϵ)

1. Perform quantization;
2. Perform CCI;
3. t=0;
4. $\delta = 2\,\epsilon$;
5. Repeat
6. Apply CCA;
7. Recompute the centers of the clusters as the new seeds according to the points assigned to them;
8. t = t +1;
9. Calculate δ;
10. Until $\delta < \epsilon$;

Figure 2: Quantization-based clustering algorithm. Source: [8]

2.1 Quantization

In the quantization step, QBCA maps all the data points to a m dimensional histogram B. To achieve this, it first computes the number of histogram bins per dimension ρ, and the size λ_l of the histogram bins in each dimension using the following equations:

$$\rho = \lfloor \log_m n \rfloor \tag{1}$$

$$\lambda_l = \frac{\overline{p}_l - \underline{p}_l}{\rho} \quad \forall l \in \{1, ..., m\} \tag{2}$$

where n is the number of data points, and m is the number of dimensions, $\overline{p}_l$ and $\underline{p}_l$ denote the maximum and minimum component of the l-th dimension, respectively.

After the computation of these two parameters, the quantization of the points occurs. The quantization function is defined as follows:

$$\begin{aligned} \mathbf{b}.id &= \sum_{l \in \{1,...,m\}} (\xi_l - 1)\, \rho^{m-l} + \xi_m \\ \xi_l &= \begin{cases} \left\lceil \frac{p_l - \underline{p}_l}{\lambda_l} \right\rceil & \text{if } p_l > \underline{p}_l \\ 1 & \text{if } p_l = \underline{p}_l \end{cases} \end{aligned} \tag{3}$$

where $\mathbf{b}.id$ is the index of the histogram bin $\mathbf{b}$, and p_l is the value of the data point $\mathbf{p}$ in the l-th dimension.

2.2 Cluster center initialization

The next step after quantizing the data point is to determine the initial centers (seeds) of the clusters. This CCI stage comes motivated by a property of Gaussian distributions that tells us that the closer we get to the center of the distribution, the higher the number of points in the region will be.

Assuming that the quantization step has been completed and we have all the data points assigned to their respective histogram bins and we know the number of points ($|\mathbf{b}_j|$) in each histogram bin $\mathbf{b}_j$ (where $1 \leq j \leq n_b$, n_b is the number of histogram bins). The CCI stage consists of the following steps:

1. Initialize an empty Max-heap H and a seed list L.
2. Process all non-zero bins.
3. While H is not empty pop the top of H and if the cardinality of the bin $\mathbf{b}$ is greater than that of its neighboring bins, $\mathbf{b}$ is added to L.
4. If $|L| < k$, where k is the desired number of clusters, add $k - |L|$ histogram bins with the largest cardinalities from the remaining histogram bins which are not yet in L.

5. Select the first k bins with the largest cardinalities in L and cluster all the points in each histogram bin as one cluster. The center of the cluster is viewed as a seed.

2.3 Cluster center assignment

The next step in QBCA is to obtain the seed candidate lists of the non-empty histogram bins and assign the bins' points to the closes seed. For this step, we first need to compute the optimal maximum distances and the minimum distances between the bins and the seeds. The optimal maximum distance $\overline{d}^{*}(\mathbf{s},\mathbf{b})$ is defined as follows:

$$\overline{d}^{*}(\mathbf{s},\mathbf{b}) = \min_{1\leq h\leq k} \overline{d}(\mathbf{s}_h, \mathbf{b}) \tag{4}$$

$$\overline{d}(s_h, \mathbf{b}_j) = \sqrt{\sum_{l=1}^{m} (s_{h,l} - \Upsilon_{j,l})^2} \tag{5}$$

$$\Upsilon_{j,l} = \begin{cases} \underline{b}_{j,l} & \text{if } s_{h,l} \geq \dfrac{\underline{b}_{j,l} + \overline{b}_{j,l}}{2} \\ \overline{b}_{j,l} & \text{otherwise} \end{cases} \tag{6}$$

where $\overline{d}(s_h, \mathbf{b}_j)$ is the maximum distance between seed $\mathbf{s}_h$ and a histogram bin $\mathbf{b}_j$ in the space $\mathbb{R}^m$.

Similarly, the minimum distance $\underline{d}(s_h, \mathbf{b}_j)$ between seed $\mathbf{s}_h$ and a histogram bin $\mathbf{b}_j$ in the space $\mathbb{R}^m$ is

$$\underline{d}(s_h, \mathbf{b}_j) = \sqrt{\sum_{l=1}^{m} (s_{h,l} - b_{j,l})^2} \tag{7}$$

$$b_{j,l} = \begin{cases} \underline{b}_{j,l} & \text{if } s_{j,l} < \underline{b}_{j,l} \\ \overline{b}_{j,l} & \text{if } s_{j,l} > \overline{b}_{j,l} \\ s_{h,l} & \text{otherwise} \end{cases} \tag{8}$$

With these definitions we can make use of the **lemma 1** from the original paper, which states that if $\underline{d}(\mathbf{s}', \mathbf{b}) \leq \overline{d}^{*}(\mathbf{s}, \mathbf{b})$ then $\mathbf{s}'$ is not the closest seed of a point $\mathbf{p}$ which belongs to the histogram bin $\mathbf{b}$. Thus, all seeds that fulfill this condition for a concrete histogram been will be considered as seed candidates for it.

The CCA stage considers the non-empty histogram bins $\mathbf{b}_j$ one by one. Thus, with the previous concepts in mind, the CCA stage can be described with the following steps:

1. Calculate the maximum distance $\overline{d}$ between the bin b_j and the seeds.

2. Sort the maximum distances in ascending order and select the first value as $\overline{d}^*(s, b)$.

3. For each seed, if it fulfills lemma 1 add it to the seed candidate list of b_j.

4. For each point p in the histogram bin b_j compute the distance between p and the bin's candidate seeds and assign p to the seed candidate which is closest to it.

2.4 Centroid recomputation

After assigning all the points to the seeds, QBCA recomputes the centers of the clusters and uses these centers as the new seeds, as in the standard K-Means, and calculates the gap δ. The algorithm terminates when $\delta < \varepsilon$, where ε is a chosen threshold.

3 Experiment

3.1 Experimental methodology and data sets

This paper tries to evaluate the performance of the QBCA algorithm by trying to replicate some of the experiments conducted in the original paper. First, as in the original work, two synthetic datasets (10000 points) are generated from a Gaussian distribution. Specifically, the datasets "Gaussian (5)" and "Gaussian (25)" are synthesized with 5 and 25 Gaussian clusters, with corresponding standard deviations of 0.06 and 0.02.

Then, QBCA is applied to two real data sets (the iris data set and the wine data set) in the UCI machine learning repository [3].

Finally, two randomly chosen images from the internet have been chosen to perform image segmentation. As in the original paper, we represent each pixel of the image in the CIE L*a*b color space, and we also apply a symmetric Gaussian low pass filter of size 5×5 with $\sigma = 0.5$.

In the following experiments, we apply QBCA, K-Means, K-Means++ ,[1] and Mini-batch K-Means [6]. Specifically, we use the implementation from the scikit-learn[1] Python package.

All four algorithms have been initialized with the same parameters. In addition, the K-Means and its variants have the `n_init` parameter set to 1 to run the main algorithms only once. In this way, we expect to make the comparisons fairer.

To assess the performance of QBCA we used three types of metrics. To evaluate its computational efficiency we use the number of distance computations per execution. To evaluate the internal quality of the clusters we use the Dunn index, described in the original paper, and the Calinski-Harabasz index (CH) [2]. Finally, to evaluate the external quality of the clusters we use the Adjusted Mutual Information (AMI) [7] and the Fowlkes-Mallows Score (FM) [4].

[1] `https://scikit-learn.org/stable/modules/clustering.html#k-means`

All the reported metrics are obtained by averaging the results of 10 repetitions, except for the image segmentation that is executed only once. The qualitative results are given with the best result among the 10 repetitions.

3.2 Comparison on the synthetic data sets

In this first experiment, we evaluate the performance of the algorithm in two synthetic data sets. The value of k is set to 5 and 25 for the Gaussian (5) and Gaussian (25), respectively. The termination threshold ε of all the algorithms is set to 0.0001. The maximum number of iterations for all the algorithms is set to 30. With this setting, we try to replicate the original experiment as much as possible.

The results of this experiment are quite promising. If we look at Tabs. 1 and 2, we can see that QBCA achieves the best results in the Gaussian(5), together with K-Means++ and Mini-batch K-Means, which appear highlighted in bold. Although the results in Gaussian(25) are not as good, it still outperforms the standard K-Means and achieves very good results on the external quality of the clusters.

Table 1: Cluster quality metrics for the Gaussian(5) data set.

	External		Internal	
	AMI	FM	CH	Dunn Index
qbca	**1.0000**	**1.0000**	**27966597.2907**	**2.4173**
kmeans	0.8611	0.8022	5791614.5463	0.4837
kmeans_plusplus	**1.0000**	**1.0000**	**27966597.2907**	**2.4173**
minibatch_kmeans	**1.0000**	**1.0000**	**27966597.2907**	**2.4173**

Table 2: Cluster quality metrics for the Gaussian(25) data set.

	External		Internal	
	AMI	FM	CH	Dunn Index
qbca	0.9760	0.9139	889282.0314	0.0002
kmeans	0.9230	0.7651	39717.5530	0.0001
kmeans_plusplus	**1.0000**	**1.0000**	**37606397.0128**	**2.6709**
minibatch_kmeans	**1.0000**	**1.0000**	**37606397.0128**	**2.6709**

In the efficiency aspect is where we can see how QBCA greatly outperforms the rest of the algorithms in this experiment as shown in Fig. 3.

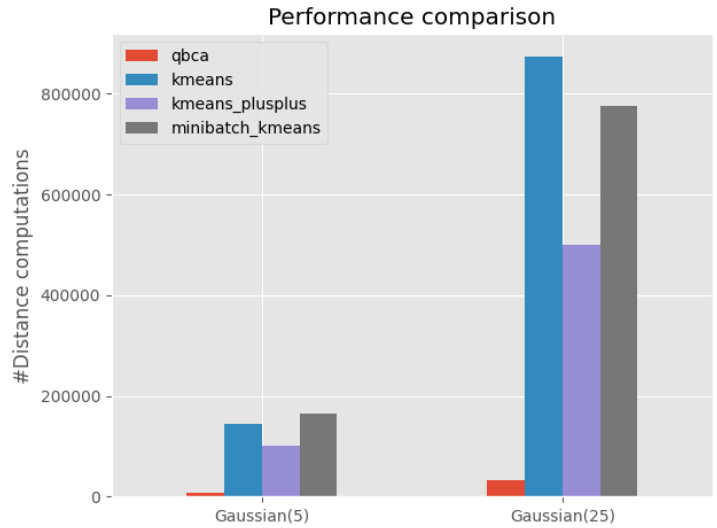

Figure 3: Efficiency comparison in the synthetic data sets.

3.3 Comparison on the real data sets

In this second experiment, we evaluate the performance of the algorithm in two real data sets. The value of k is set to 3 for the two data sets based on the number of different classes in our ground-truth. The termination threshold ε and the maximum number of iterations are set to the same values as in the previous experiment (i.e. 0.0001 and 30 respectively).

When tested in real data, QBCA seems to perform considerably well in terms of cluster quality compared to the other algorithms. In fact, we can see how it obtains the best Dunn index for the Iris data set (see Tab. 3) and the best AMI and CH scores for the Wine data set (see Tab. 4).

Table 3: Cluster quality metrics for the Iris data set.

	External		Internal	
	AMI	FM	CH	Dunn Index
qbca	0.7387	0.8112	561.5937	**0.1094**
kmeans	0.6917	0.7660	616.1957	0.0680
kmeans_plusplus	0.7387	0.8112	**693.7084**	0.0783
minibatch_kmeans	**0.7500**	**0.8242**	679.3648	0.0587

Table 4: Cluster quality metrics for the Wine data set.

	External		Internal	
	AMI	FM	CH	Dunn Index
qbca	**0.4227**	0.5835	**561.8157**	0.0163
kmeans	0.4196	0.5869	536.5407	0.0142
kmeans_plusplus	0.4169	**0.5887**	556.2101	0.0167
minibatch_kmeans	0.4210	0.5885	553.6274	**0.0173**

To better appreciate the clusterings and their quality, we show the results of the different algorithms in Figs. 4 and 5. In these, we observe how the four algorithms reach very similar results.

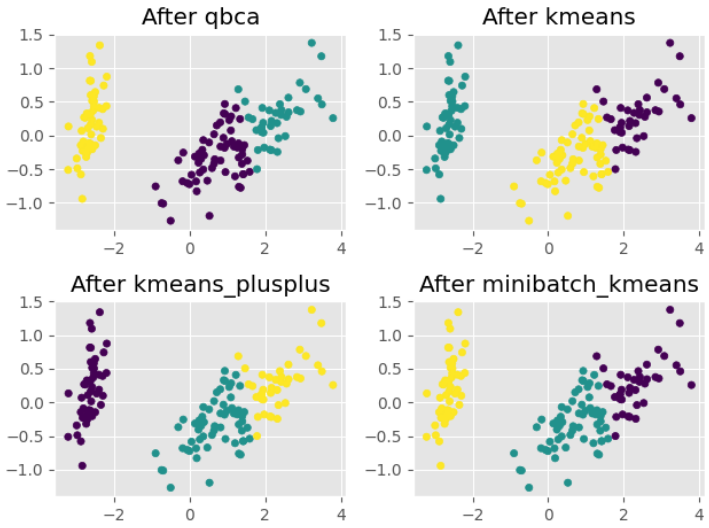

Figure 4: Clustering of the Iris data set by the different algorithms.

Contrary to our first experiment, we observed a downgrade in the efficiency performance of QBCA in the real data sets. As shown in Fig. 6, QBCA is the second-worst algorithm in terms of the number of distance computations. A possible reason for this huge downgrade is that QBCA is not suited for small and medium-sized data sets (i.e. < 5000 samples) and simpler algorithms can perform better for this reduced number of samples.

3.4 Comparison on the image segmentation task

The last experiment we conduct is the segmentation of two images. As in the original paper, we have set k to 5, the threshold ε to 0.01, and the maximum

Figure 5: Clustering of the Wine data set by the different algorithms.

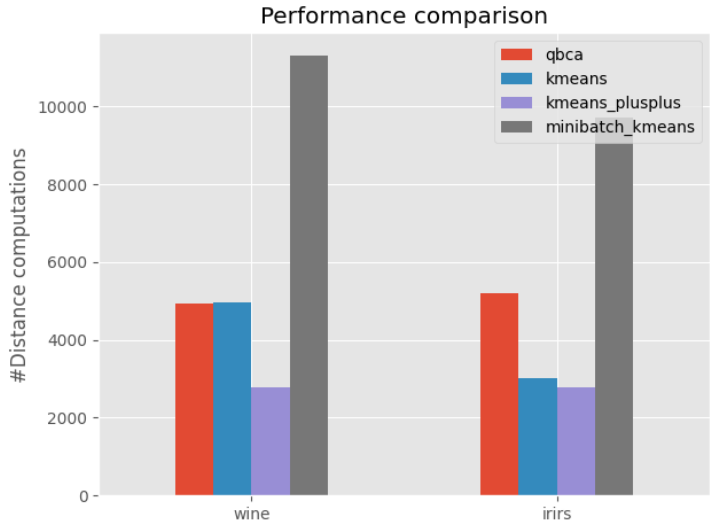

Figure 6: Efficiency comparison in the Iris and Wine data sets.

Image(**a**)
Size 150×225
Pixels 33750

Image(**b**)
Size 300×548
Pixels 164400

Figure 7: Selected images for the image segmentation experiment.

source (a): `https://negativespace.co/butterfly-close-up-monarch-flower/`

source (b): `https://www.flickr.com/photos/pinkcowphotography/16407544970`

number of iterations to 50. The original images can be seen in Fig. 7.

By looking at Tabs. 5 and 6 we can see how QBCA is the best algorithm, in general, when assessing the internal cluster quality. However, it suffers again in terms of computation efficiency as shown in Fig. 8. In this case, it is, with difference, the worst among the four algorithms. The results of the image segmentation for both images are shown in Figs. 9 and 10.

Table 5: Internal cluster quality metrics for the Image(1).

	CH	Dunn Index
qbca	32726.1676	**0.0089**
kmeans	**39424.2561**	0.0072
kmeans_plusplus	39355.7857	0.0072
minibatch_kmeans	39373.8792	0.0072

Table 6: Internal cluster quality metrics for the Image(2).

	CH	Dunn Index
qbca	**218335.3756**	**0.0082**
kmeans	199795.4988	0.0078
kmeans_plusplus	203798.4533	0.0079
minibatch_kmeans	208695.3298	0.0080

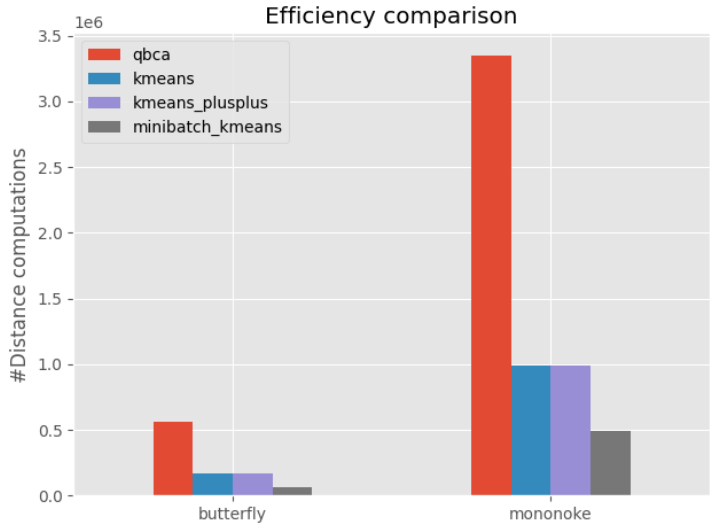

Figure 8: Efficiency comparison in the image segmentation experiment.

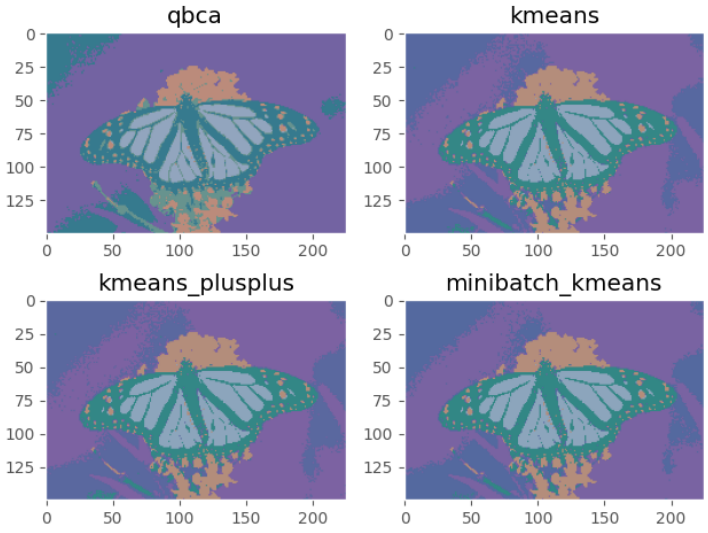

Figure 9: Clustering of the Image(1) image by the different algorithms.

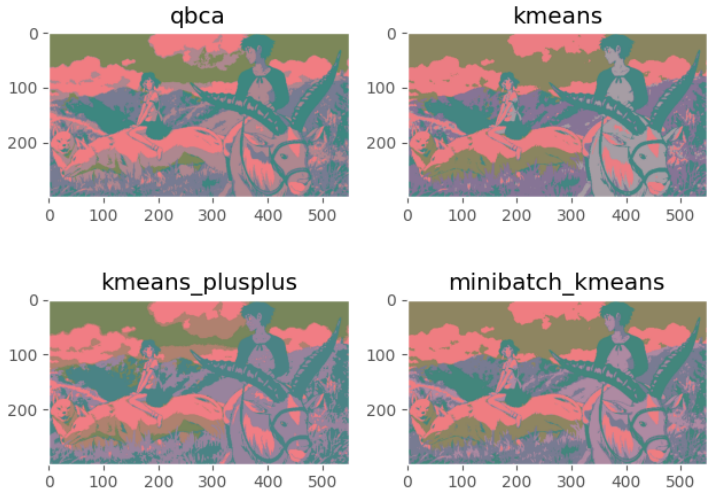

Figure 10: Clustering of the Image(2) by the different algorithms.

4 Conclusions

In this work, we have implemented the QBCA algorithm based on its original paper and compared it with other similar clustering algorithms. Looking at the results of the different experiments, we can conclude that QBCA obtains very good results in terms of cluster quality, outperforming in some cases well-known algorithms such as K-Means. Nevertheless, the main goal of QBCA was to propose an algorithm that improved the computing efficiency for large datasets. In this regard, we have seen how in our experiments QBCA usually was one of the worst algorithms, which is in disagreement with the stated in the original paper. However, we can not say that the results obtained in this work can be followed blindly since several factors might have biased the results. In fact, one should keep in mind that the implemented QBCA could have been implemented more efficiently by more experienced researchers. In the same regard, the algorithms used from the scikit-learn package are implemented to be extremely efficient. This might have caused extreme differences in the efficiency in some of the experiments. Another important factor is that the algorithms used for the comparison are not the same that the ones in the original paper.

In future work, it would be interesting to conduct the same experiments with the algorithms used in the original paper to see if we get similar results. Additionally, we could implement the two improvements proposed in the paper to see in what measure they improve the results obtained.

References

[1] David Arthur and Sergei Vassilvitskii. *k-means++: The advantages of careful seeding*. Tech. rep. Stanford, 2006.

[2] Tadeusz Caliński and Jerzy Harabasz. "A dendrite method for cluster analysis". In: *Communications in Statistics-theory and Methods* 3.1 (1974), pp. 1–27.

[3] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository*. 2017. URL: http://archive.ics.uci.edu/ml.

[4] Edward B Fowlkes and Colin L Mallows. "A method for comparing two hierarchical clusterings". In: *Journal of the American statistical association* 78.383 (1983), pp. 553–569.

[5] J. Macqueen. "Some methods for classification and analysis of multivariate observations". In: *In 5-th Berkeley Symposium on Mathematical Statistics and Probability*. 1967, pp. 281–297.

[6] David Sculley. "Web-scale k-means clustering". In: *Proceedings of the 19th international conference on World wide web*. 2010, pp. 1177–1178.

[7] Nguyen Xuan Vinh, Julien Epps, and James Bailey. "Information Theoretic Measures for Clusterings Comparison: Is a Correction for Chance Necessary?" In: *Proceedings of the 26th Annual International Conference on Machine Learning*. ICML '09. Montreal, Quebec, Canada: Association for Computing Machinery, 2009, pp. 1073–1080. ISBN: 9781605585161. DOI: 10.1145/1553374.1553511. URL: https://doi.org/10.1145/1553374.1553511.

[8] Zhiwen Yu and Hau-San Wong. "Quantization-based clustering algorithm". In: *Pattern Recognition* 43.8 (2010), pp. 2698–2711. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2010.02.020. URL: https://www.sciencedirect.com/science/article/pii/S0031320310000981.