A Parameter-free Affinity Based Clustering

Bhaskar Mukhoty, Ruchir Gupta Member IEEE

IIITDM, Jabalpur

\{p.bhaskar, rgupta\}@iiitdmj.ac.in

Abstract—Several methods have been proposed to estimate number of clusters in datasets; the basic approach has been to study an index over a range of cluster numbers, and select the number which gives an optimum or brings a sudden change in the parameter. In this paper we propose a simpler approach to find the number of clusters by directly detecting them. This has been done using a affinity histogram, where it is easy to identify a threshold, above which two points must come to the same cluster. For well separated clusters this would identify the clusters at once, for others it gives a good starting point to find larger clusters. The approached has worked well for several high dimensional synthetic and real datasets. Obvious implication of this is to bypass costly range search and report the clusters itself when supplied with the number of them.

Index Terms—Number of clusters, Affinity histogram, parameter-free clustering

I. INTRODUCTION

Cluster analysis is an unsupervised learning problem, where the objective is to suitably group n data points \( X = \{x_1, x_2, ..., x_n\}, \) \( x_i \in \mathbb{R}^d \), taken from a d-dimensional space. Traditionally this has been considered an important statistical problem with numerous applications in different fields including image analysis, bio-informatics and market research.

Generally, if the number of clusters \( k \) is given, finding the optimal set of clusters \( C = \{c_1, c_2, ..., c_k\} \) which minimizes the variance of the clusters or squared sum within, is an NP-hard problem\[1\].

\[
SSW(C) = \sum_{j=1}^{k} \sum_{x \in c_j} ||x - \bar{c}_j||^2
\]

Here, \( ||p|| \) denotes the magnitude of a vector \( p \), so that \( ||p - q|| \) is the euclidean distance between two points \( p, q \) and \( \bar{c}_j \) denotes the mean of \( x_s \in c_j \) or centroid of the cluster \( c_j \). Evidently SSW reduces as number of clusters increases, and ultimately goes to zero, with number of clusters equal to \( n \). Methods have been proposed to study such kind of functions for a point after which the curve flattens, to give optimal number of clusters[2].

Internal cluster evaluation indexes \[3, 4\] which evaluates quality of clustering depending upon some measures of intra cluster cohesion and inter cluster separation, are often used to predict suitable number of clusters, by searching the data with a range of cluster numbers and reporting the number which gives optimal value of the index. This followed by a clustering algorithm parameterized by number of clusters, gives the partition of the data.

Other parameterized algorithms supplied with some form information about the data, are also in use. As per example, DBSCAN[5] a density based clustering algorithm requires minimum number of points in a cluster as input, but having prerequisite domain knowledge to supply such parameters are not always easy, often additional work is needed to estimate such parameters.

In order to have a parameter free clustering algorithm, we have taken a different approach to identify the clusters, our method imitates the way human recognizes the groups in the data. Human when exposed to a representation of an intelligible dataset, at once recognizes the clusters present, because some data points appears so close, that they could hardly go to different clusters, such groups when counted gives the number of clusters. We never work in the redundant way to search through a possible set of clustering, to identify the optimum one.

Our algorithm works in a similar manner, it identifies closely grouped points, by calculating a affinity threshold. And it sequentially searches the data space for the points in their vicinity, forming the group incrementally; points which remains single in such a search are identified as outliers. The clusters detected by this process could be the clusters we are seeking if they are well separated, but they could also be merged to form new cluster if they are close enough.

We observe there could two kinds of data, one which would require further merging and the one which would not for the detected clusters are well separated. The data which could support merge, we estimate the number of clusters after merge for them from the size of the detected clusters. Merge is done in order of closeness. The nature of the data is decided by defining a compactness function which tells whether the data supports merge, unlike equation 1, this function will not always decrease with increase of number of clusters, indicating the nature of the data.

We have conducted experiments with standard datasets, and compared the performance of the algorithm with the existing algorithms, in terms of estimating number of clusters and quality of the obtained clusters, to claim a remarkable performance in convex datasets.

Abstract—Several methods have been proposed to estimate number of clusters in datasets; the basic approach has been to study an index over a range of cluster numbers, and select the number which gives an optimum or brings a sudden change in the parameter. In this paper we propose a simpler approach to find the number of clusters by directly detecting them. This has been done using a affection histogram, where it is easy to identify a threshold, above which two points must come to the same cluster. For well separated clusters this would identify the clusters at once, for others it gives a good starting point to find larger clusters. The approached has worked well for several high dimensional synthetic and real datasets. Obvious implication of this is to bypass costly range search and report the clusters itself when supplied with the number of them.

Index Terms—Number of clusters, Affinity histogram, parameter-free clustering

I. INTRODUCTION

Cluster analysis is an unsupervised learning problem, where the objective is to suitably group n data points \( X = \{x_1, x_2, ..., x_n\}, \) \( x_i \in \mathbb{R}^d \), taken from a d-dimensional space. Traditionally this has been considered an important statistical problem with numerous applications in different fields including image analysis, bio-informatics and market research.

Generally, if the number of clusters \( k \) is given, finding the optimal set of clusters \( C = \{c_1, c_2, ..., c_k\} \) which minimizes the variance of the clusters or squared sum within, is an NP-hard problem[1].

\[
SSW(C) = \sum_{j=1}^{k} \sum_{x \in c_j} ||x - \bar{c}_j||^2
\]

Here, \( ||p|| \) denotes the magnitude of a vector \( p \), so that \( ||p - q|| \) is the euclidean distance between two points \( p, q \) and \( \bar{c}_j \) denotes the mean of \( x_s \in c_j \) or centroid of the cluster \( c_j \). Evidently SSW reduces as number of clusters increases, and ultimately goes to zero, with number of clusters equal to \( n \). Methods have been proposed to study such kind of functions for a point after which the curve flattens, to give optimal number of clusters[2].

Internal cluster evaluation indexes [3, 4] which evaluates quality of clustering depending upon some measures of intra cluster cohesion and inter cluster separation, are often used to predict suitable number of clusters, by searching the data with a range of cluster numbers and reporting the number which gives optimal value of the index. This followed by a clustering algorithm parameterized by number of clusters, gives the partition of the data.

Other parameterized algorithms supplied with some form information about the data, are also in use. As per example, DBSCAN[5] a density based clustering algorithm requires minimum number of points in a cluster as input, but having prerequisite domain knowledge to supply such parameters are not always easy, often additional work is needed to estimate such parameters.

In order to have a parameter free clustering algorithm, we have taken a different approach to identify the clusters, our method imitates the way human recognizes the groups in the data. Human when exposed to a representation of an intelligible dataset, at once recognizes the clusters present, because some data points appears so close, that they could hardly go to different clusters, such groups when counted gives the number of clusters. We never work in the redundant way to search through a possible set of clustering, to identify the optimum one.

Our algorithm works in a similar manner, it identifies closely grouped points, by calculating a affinity threshold. And it sequentially searches the data space for the points in their vicinity, forming the group incrementally; points which remains single in such a search are identified as outliers. The clusters detected by this process could be the clusters we are seeking if they are well separated, but they could also be merged to form new cluster if they are close enough.

We observe there could two kinds of data, one which would require further merging and the one which would not for the detected clusters are well separated. The data which could support merge, we estimate the number of clusters after merge for them from the size of the detected clusters. Merge is done in order of closeness. The nature of the data is decided by defining a compactness function which tells whether the data supports merge, unlike equation 1, this function will not always decrease with increase of number of clusters, indicating the nature of the data.

We have conducted experiments with standard datasets, and compared the performance of the algorithm with the existing algorithms, in terms of estimating number of clusters and quality of the obtained clusters, to claim a remarkable performance in convex datasets.
In the next section we discuss some existing algorithms, section 3 produces the algorithm with suitable examples, which is followed by experimental results in section 4, we conclude the paper in section 5.

II. RELATED WORK

Following are the four basic of approach taken to address cluster analysis problem.

A. Partition Method

This kind of methods partition the data into k Voronoi parts, where k is supposed to be known. Starting with some initial clusters these methods reallocates the centroids in order to reduce the SSW error. k-means and its variations\[6], [7] are example of such method.

This comes with the challenging problem to estimate the number of clusters. Internal validity indexes which evaluates a clustering depending upon cohesion and separation are frequently used for this purpose. These indices need to be evaluated the for all possible number of clusters usually 2 to \(\sqrt{n}\). Results are subject to change under different clustering algorithm used. Besides the time it takes to estimate the number, it is often left for user to determine which index to use. If the clustering algorithm used is not deterministic, as is the case for k-means, it may produce different clustering on different runs, so aggregation of results also becomes necessary.

B. Hierarchical Method

Here the clusters are formed either by agglomerating nearest points in a bottom up approach, or by partitioning the set in a divisive top down approach, until it gives the desired number of clusters or falsify some parametric criterion. The agglomeration or division typically depends upon the distance measure used to define separation between two clusters. There are several options available for this purpose. Single linkage defines this as the distance between two nearest points of the clusters, average linkage measures takes the mean of all pairs of distance, while Ward measures the increase in sum of squares when the clusters are merged.

The methods require estimation of parameter as before for stopping criterion, and the greedy search method often leads to a local optima in presence of outliers.

C. Distribution Method

In this method data points are assumed to come from a certain distribution, as per example in Gaussian Mixture Model, clusters are assumed to come from Gaussian distributions, and expectation maximization algorithm parameterized with number of clusters is used to estimate distributions parameters. Random outliers can severely degrade performance of such algorithm by giving misplaced cluster centers.

D. Density Method

Density based methods such as DBSCAN[5] method works by detecting points in a cluster using a radius length \(\epsilon\), and a parameter Minpts to represent minimum number of points withing the radius. This method although robust to outliers and works for arbitrary shapes, is susceptible to density variations. OPTICS[8] is an improvement which removes the \(\epsilon\) parameter. Finding suitable value for parameter in such algorithm requires domain knowledge.

III. METHOD

The algorithm works in two parts, first to identify those points which must come in the same cluster and then to merge the clusters further if necessary. The approach we have taken can be seen as a middle ground between partition based and hierarchical clustering.

A. Description of Algorithm

As a preprocessing each dimension is scaled separately by subtracting its mean and diving its standard deviation. This ensures all dimensions get equal weight in the distance function. We form a \(n \times n\) distance matrix D to store euclidean distance,

\[D(i, j) = ||x_i - x_j||\]

between all pair of points in normalized data matrix and pass it to affinity function taken as,

\[A(i, j) = \exp\left(-\frac{D(i, j)^2}{2\sigma_D}\right)\]  \hspace{1cm} (2)

\(\sigma_D\) being standard deviation of the elements of D, to get an affinity score. So as the pair of nodes which are closer will have a affinity near one, and nodes which are farther will have it near zero.

We then take a histogram H of all affinity scores, which divides the score in 10 bins of 0.1 range. If two points have affinity above the threshold identified from the histogram, the points would go to the same cluster. Figure 1 shows such histogram taken for d8c8N[9] data set.

The threshold is calculated by taking maximum of the difference between counts in consecutive bins, taken from high to low affinity; in the histogram of figure 1 this would come at 0.85, representing affinity of the ninth bin.

All points are then checked sequentially and an unclustered point is initialized as new cluster and all other points which have similarity above threshold are assigned to that cluster, in case the point being compared is already in a cluster and it is closer to a new forming cluster, then it is assigned to that cluster, this helps to form more cohesive clusters.

Result of the above process could be to have some single point clusters, we would identify them as outliers and separate them from future processing.

Now we reach to second part of the algorithm where the objective is to merge the clusters if necessary. We
define the cost of clustering as a slight modification of SSW:

\[ W(k) = \sum_{j=1}^{k} \frac{1}{n_j} \sum_{i \in S_j} (x_i - \overline{x})^2 \]  

For data with closely lying clusters \( W \) will decrease as clusters are merged, which signifies that data is favorable towards merge, for clusters which are well separated merging would increase it, signifying merge is not required.

Let \( [s_1, s_2, ..., s_p] \) be the size of the clusters in descending order. We observe, most data that supports merge are scale free, big size clusters being rare, and small sizes being frequent.

Figure 3 shows the distribution of detected clusters in descending order of size, for the thyroid dataset taken from UCI repository[5].

In order to estimate number of clusters for such dataset, the more skewed the distribution becomes, the less number of clusters become appropriate to represent the data.

We capture this notion by taking minimum \( k \), such that sum of the differences from the bigger clusters weighted by cluster sizes exceeds that of from the smaller clusters.

\[ \min_k \left( \sum_{i=1}^{k-1} (s_i - s_k) s_i > \sum_{j=k+1}^{p} (s_k - s_j) s_j \right) \tag{4} \]

In the figure 3, the number of clusters satisfying the above criteria is at \( k=2 \), giving ground truth value of the dataset.

We then form a \( p \times p \) matrix to get distance between all clusters using centroid distance that could be replaced by any other distance measure, and merge two clusters which are closest in each step, \( p - k \) such merge would produce the required \( k \) clusters.

Finally we compare \( W(k) \) with \( W(p) \) to ensure that merger was necessary. The merge is discarded in favor of initially detected clusters, if found otherwise.

B. Algorithm

Algorithm: Find_Clustres

Input: \( X_{n,d} \) (n data points in d dimensions)

Output: \( C_{n,1} \) (vector of integers denoting class of each point)

1. Normalize columns of \( X \); \( X^{(i)} \) denotes \( j^{th} \) column \( X \).
   for \( j = 1 \) to \( d \) do
      \( \mu_j = \text{mean}(X^{(i)}); \quad \sigma_j = \text{SD}(X^{(i)}) \)
      \( X^{(i)} := \frac{X^{(i)} - \mu_j}{\sigma_j} \)
   end for

2. Form distance matrix \( D_{n \times n} \); \( X_i \) denotes \( i^{th} \) row of \( X \):
   for \( i = 1 \) to \( n \) do
      for \( j = 1 \) to \( n \) do
         \( D(i, j) = ||X_i - X_j|| \)
      end for
   end for

3. Get Affinity matrix \( A_{n \times n} \):
   for \( i = 1 \) to \( n \) do
      for \( j = 1 \) to \( n \) do
         \( A(i, j) = \exp\left(-\frac{D(i, j)^2}{2 \sigma_D^2}\right) \quad | \sigma_D \text{ being SD of } D| \)
      end for
   end for

4. Take histogram \( H_{10 \times 1} \) of affinity values in 10 bins:
   for \( i = 1 \) to \( n \) do
      for \( j = 1 \) to \( n \) do
         \( H([A(i, j) \times 10]) = H([A(i, j) \times 10]) + 1 \)
      end for
   end for

5. Find affinity threshold:
   \( k = \max(H(i + 1) - H(i)), i \in [1..9] \)
   \( \text{thresh} = 0.1(k - 1) + 0.05 \)

6. Initialize cluster size and centroid vector
   \( C_{n \times 1} = 0; \quad S_{n \times 1} = 0; \quad Cnt_{n \times d} = 0 \)
   for \( i = 1 : n \) do
      if \( C(i) == 0 \) then
         \( k = k + 1 \)
      \( C(i) = k; \quad S(k) = 1; \quad Cnt(k) = X_i \)
end if
for $j = 1 : n$ do
  if $C(j) == 0$ then
    if $A(cnt(k), X_i) > \text{threshold}$ then
      addPoint($C(cnt(k), j)$); $C(j) = k$
    end if
  else if $D(Cnt(k), j) < D(Cnt(C(j)), j)$ then
    removePoint($Cnt(j)$)
    addPoint($Cnt(k), j$); $C(j) = k$
  end if
end for
end for

Remove outliers:
for $i = 1 : n$ do
  if $S(C(i)) == 1$ then
    print $X_i$ as outlier.
  end if
end for

Let $p$ be remaining number of clusters after removing outliers.

Find $k$ which satisfies equation (4).

$C' = C$
for $i = 1 : p - k$ do
  Merge two clusters that are closest; Update $C$.
end for
if $W(k) > W(p)$ then
  Discard merge.
  print number of clusters $p$
  return $C'$
else
  print number of clusters $k$
  return $C$
end if
end Find_clusters

addPoint($Cnt(k), j$)
\[ Cnt(k) = \frac{\sum_{i} Cnt(k) + X_i}{S(k) + 1} \]
end

removePoint($Cnt(j)$)
\[ k' = C(j) \]
\[ Cnt(k') = \frac{\sum_{i} Cnt(k') - X_i}{S(k') - 1} \]
end

1) Complexity: Distance calculation, affinity matrix formation and detection of initial clusters by threshold, requires to check all pair of points. As addition or removal of a point from a cluster can be performed in constant time, as performed in addPoint and removePoint functions, the worst case complexity of the algorithm becomes $O(n^2)$, with dimension of the data taken as constant.

IV. Results

A. Dataset

Table 1 shows real world and synthetic datasets used for experiments, they are introduced by respective researchers mentioned in the reference column. Dimension of the datasets varies from 2 to 1024. The last five datasets included have density varying clusters, also they have 10 percent noise/outliers added w.r.t original number of points. By the nature of the centroid distance function used, the algorithm would not work for shape based datasets, thus all synthetic data sets taken, are convex.

B. Evaluation

Fig 3. shows in first two dimensions the d8c8N data set clustered by the algorithm, where clusters are joined by virtual line to their centroid, while the outliers are single.

The evaluation has been performed in following two respects.

1) Estimation of cluster number: Correctness of cluster number estimated by the algorithm are compared other available methods. For the comparison we have taken nine indexes which are best performing[15, 16] among varied datasets. Published implementation[17] available for the indexes are used with k-means[6] as clustering algorithm. Cluster number which is the most reported in hundred test runs are taken as reported value, in contrast the proposed algorithm requires single run, as the output does not change if the order of the input data points remains the same. Ground truth datasets available online[13, 9] for which number of clusters are already known, are used for comparison, exact match of reported value with the known number is considered as success.

Table 2 shows the performance in this respect. It can be seen that the algorithm drastically outperforms the
other methods, with an accuracy of 78 percent, while the best among the algorithms taken are Silhouette and Davis-Bouldin.

2) Quality of clusters: Second, to compare the quality of clusters obtained from the proposed algorithm with that of existing algorithms. Most of the existing algorithms requires some parameter, in order to compare we have taken algorithms for which number of cluster could be supplied as parameter. Data for which correct number of clusters could be supplied, the ground truth number of clusters are compared with other clustering algorithms by supplying them the ground truth number of clusters as parameter. We made comparison with k-means++ [7] from partitioning method, single, average and ward from hierarchical method, and Gaussian mixture model with EM algorithm.

External evaluation indexes compare partition obtained from a clustering with actual partitions given in ground truth data, in order to evaluate quality of clustering. Adjusted Rand Index [24], Jaccard index and FI measure are standard such indices used for this purpose, all of them increase to represent better a match, the maximum value being 1.

Table 3 shows the quality of the clusters, in the noised
data the algorithm outperforms others, while in other data the quality is comparable. The relative performance of the algorithms are same with respect to all three external indices.

V. Conclusion

We have proposed a parameter free clustering algorithm, which shows promising results on convex datasets. The algorithm shows robustness to outliers, density variations and high dimensional input data. It could be noted that the algorithm can used dedicatedly to predict number of clusters, or if the number of clusters are known the algorithm can also be used to approximate optimal clusters, where prediction functionality would be turned off and cluster detected by affinity threshold would be merged to give required number of clusters.

References

[1] M. Mahajan, P. Nimbhorkar, and K. Varadarajan, “The planar k-means problem is np-hard,” in Proceedings of the 3rd International Workshop on Algorithms and Computation, ser. WALCOM ’09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 274–285.

[2] R. Tibshirani, W. Guenther, and T. Hastie, “Estimating the number of clusters in a data set via the gap statistic,” Journal of the Royal Statistical Society Series B, 2001.

[3] P. Rousseuw, “Silhouettes: A graphical aid to the interpretation and validation of cluster analysis,” J. Comput. Appl. Math., vol. 20, no. 1, pp. 53–65, Nov. 1987. [Online]. Available: http://dx.doi.org/10.1016/S0377-0427(87)80012-5

[4] I. Ciszews and J. Harabasz, “A dendrite method for cluster analysis,” Communications in Statistics, vol. 3, no. 1, pp. 1–27, Jan. 1974. [Online]. Available: http://dx.doi.org/10.1080/03610927408827101

[5] M. Ester, H. Peter Kriegel, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise.,” AAAI Press, 1996, pp. 226–231.

[6] S. Lloyd, “Least squares quantization in pcm,” IEEE Trans. Inf. Theor., vol. 28, no. 2, pp. 129–137, Sep. 2006. [Online]. Available: http://dx.doi.org/10.1109/TIT.1982.1056489

[7] T. Calinski and J. Harabasz, “A dendrite method for cluster analysis,” Communications in Statistics, vol. 3, no. 1, pp. 1–27, Jan. 1974. [Online]. Available: http://dx.doi.org/10.1016/S0377-0427(87)80012-5

[8] D. Arthur and S. Vassilvitskis, “K-means++: The advantages of careful seeding,” in Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, ser. SODA ’07. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035. [Online]. Available: http://dl.acm.org/citation.cfm?dir=1283383.1283494

[9] D. Arthur and S. Vassilvitskis, “K-means++: The advantages of careful seeding,” in Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, ser. SODA ’07. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035. [Online]. Available: http://dl.acm.org/citation.cfm?dir=1283383.1283494

[10] D. A. Ratkowsky and G. N. Lance, “A criterion for determining the number of groups in a data set using sum-of-squares clustering,” Biometrics, vol. 39, no. 1, pp. 129–137, 1978. [Online]. Available: http://www.jstor.org/stable/25294713

[11] P. Franti and O. Virmajoki, “Iterative shrinking method for clustering problems,” Pattern Recogn., vol. 39, no. 5, pp. 761–775, May 2006. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2005.09.012

[12] C. J. Veenman, M. J. T. Renders, and E. Backer, “A maximum variance cluster algorithm,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 24, no. 9, pp. 1273–1280, Sep. 2002. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2002.1033218

[13] J. C. Dunn, “A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters,” Journal of Cybernetics, vol. 3, no. 3, pp. 32–57, 1973. [Online]. Available: http://dx.doi.org/10.1080/0196972720854046

[14] D. L. Davies and D. W. Bouldin, “A cluster separation measure,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 1, no. 2, pp. 224–227, Feb. 1979. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.1979.4766909

[15] P. Franti, O. Virmajoki, and V. Hautamaki, “Fast agglomerative clustering using a k-nearest neighbor graph,” Journal of Statistical Software, vol. 67, no. 1, pp. 137–159, 2002. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2006.227

[16] O. Arbelaitz, I. Gurrutxaga, J. Muguerza, J. M. Pérez, and I. Perona, “An extensive comparative study of cluster validity indices,” Pattern Recogn., vol. 46, no. 1, pp. 243–256, Jan. 2013. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2012.07.021

[17] M. Ankerst, M. M. Breunig, H. Peter Kriegel, and J. Sander, “Optics: Ordering points to identify the clustering structure.” ACM Press, 1999, pp. 49–60.

[18] M. Ankerst, M. M. Breunig, H. Peter Kriegel, and J. Sander, “Optics: Ordering points to identify the clustering structure.” ACM Press, 1999, pp. 49–60.

[19] J. C. Dunn, “A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters,” Journal of Cybernetics, vol. 3, no. 3, pp. 32–57, 1973. [Online]. Available: http://dx.doi.org/10.1080/0196972720854046

[20] W. J. Krzanowski and Y. T. Lai, “A criterion for determining the number of separating compact well-separated clusters,” Journal of the Royal Statistical Society, vol. 39, no. 1, pp. 129–137, 1978. [Online]. Available: http://www.jstor.org/stable/25294713

[21] D. L. Davies and D. W. Bouldin, “A cluster separation measure,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 1, no. 2, pp. 224–227, Feb. 1979. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.1979.4766909

[22] A. J. Scott and M. J. Symons, “Clustering methods based on likelihood ratio criteria,” Biometrics, vol. 27, no. 2, pp. 387–397, 1971. [Online]. Available: http://www.jstor.org/stable/25294713

[23] J. R. Hubert and J. R. Levin, A General Statistical Framework for Assessing Categorical Clustering in Free Recall. Distributed by ERIC Clearinghouse [Washington, D.C.], 1975. [Online]. Available: http://www.eric.ed.gov/contentdelivery/servlet/ERICServlet?accno=ED116162
[24] L. Hubert and P. Arabie, “Comparing Partitions,” *Journal of Classification*, 1985.