ISSN:2320-0790

An International Journal of Advanced Computer Technology

Effective generation of clusters for gene data

Neelambike S[1], Veeragangadhara Swamy T M[2], Netravati U M[3], Asha K[4]

[1] neelais@gmail.com,[2]Swamytm@gmail.com,[3]netswamytm@gmail.com,*asha_kathigi@yahoo.co.in*[4]

Abstract: In this paper we have analysed that which of the algorithms leads to efficient generation of cluster for gene data so that making use of the existing algorithms such as Hierarchical, partitional, fuzzy k-mean and ART1 NN and takes input from the NCBI and provide input to above algorithms and find the time taken to produced the result and also takes reference sequence from the NCBI and compare the clustering result with the reference sequence then decide how much of presentence the clustering result match with the reference sequence and with minimum amount of time after that rank all these algorithms in rank wise and depending upon this rank the developer pick the algorithm and develop their application.

Keywords: K-mean; DNA; reference sequence;FCM;

I. INTRODUCTION

Cluster analysis groups objects (observations, events) based on the information found in the data describing the objects or their relationships. The goal is that the objects in a group will be similar (or related) to one other and different from (or unrelated to) the objects in other groups. The greater the similarity (or homogeneity) within a group, and the greater the difference between groups, the "better" or more distinct the clustering. The definition of what constitutes a cluster is not well defined, and, in many applications clusters are not well separated from one another. Nonetheless, most cluster analysis seeks as a result, a crisp classification of the data into non-overlapping groups. Fuzzy clustering, described in an exception to this, and allows an object to partially belong to several groups.

Common Proximity Measures

Distance Measures: The most commonly used proximity measure, at least for ratio scales (scales with an absolute 0) is the Minkowski metric, which is a generalization of the normal distance between points in Euclidean space. It is defined as $p_{ij} = \left(\sum_{k=1}^{d} |xik - xjk|^r \right)^{1/r}$

where, *r* is a parameter, *d* is the dimensionality of the data object, and *xik* and *xjk* are, respectively, the *k*th components of the *i*th and *j*th objects, *xi* and *x*j. The following is a list of the common Minkowski distances for specific values of *r*

1) *r* = 1. City block (Manhattan, taxicab, L1 norm) distance. A common example of this is the Hamming distance, which is just the number of bits that are different between two binary vectors.

2) *r* = 2. Euclidean distance. The most common measure of the distance between two points.

3) *r* → ∞. "supremum" (Lmax norm, L∞ norm) distance. This is the maximum difference between any component of the vectors..

Basic Clustering Techniques

Many different clustering techniques that have been proposed over the years. These techniques can be described using the following criteria [1]:

- Hierarchical vs. partitional (nested and unnested).
- Divisive vs. agglomerative
- Incremental or non-incremental.

Hierarchical vs. partitional (nested and unnested). Hierarchical techniques produce a nested sequence of partitions, with a single, all inclusive cluster at the top 14 and singleton clusters of individual points at the bottom. Each intermediate level can be viewed as combining (splitting) two clusters from the next lower (next higher) level.

Divisive vs. agglomerative. Hierarchical clustering techniques proceed either from the top to the bottom or from the bottom to the top, i.e., a technique starts with one large cluster and splits it, or starts with clusters each containing a point, and then merges them.

Incremental or non-incremental. Some clustering techniques work with an item at a time and decide how to cluster it given the current set of points that have already been processed. Other techniques use information about all the points at once. Nonincremental clustering algorithms are far more common.[6].

II. SPECIFIC CLUSTERING TECHNOLOGY

K-means is one of the simplest unsupervised learning algorithm that partition feature vectors into k clusters so that the within group sum of squares is minimized. The procedure follows a simple way to classify a given data set and looks like that:

```
Step 1: start
Step 2: place randomly initial group centroids
Step 3: Assign each object to the group that has the closest centroid
Step 4: if the positions of the centroids didn't change go to
        the next step else go to step 2.
Step 5: End.
```

So, using this K-mean algorithm generate the clusters by using the centroids and distance. Which data point is very close to the centroids that data point belongs to that cluster.

fuzzy K-mean: This algorithm works by assigning membership to each data point corresponding to each cluster center on the basis of distance between the cluster center and the data point. More the data is near to the cluster center more is its members hip towards the particular cluster center. Clearly, summation of membership of each data point should be equal to one

Let $X = \{x_1, x_2, x_3 ..., x_n\}$ be the set of data points and $V = \{v_1, v_2, v_3 ..., v_c\}$ be the set of centers.

Step 1: Start

Step2 : Randomly select *'c'* cluster centers.

Step 3: Calculate the fuzzy membership $'\mu_{ij}'$ using below Equation (5.1)

$$\mu_{ij} = \frac{1}{\sum_{k=1}^{c} (d_{ij} / d_{ik})^{(2/m-1)}} \quad (5.1)$$

Step 4: Compute the fuzzy centers $'v_j'$ using below Equation (5.2)

$$v_j = (\sum_{i=1}^{n} (\mu_{ij})^m x_i) / (\sum_{i=1}^{n} (\mu_{ij})^m), \forall j = 1, 2, ...c \quad (5.2)$$

Step 5: Repeat step 2) and 3) until the minimum *'J'* value is achieved or $||U^{(k+1)} - U^{(k)}|| < \beta$.

'k' is the iteration step.
'β' is the termination criterion between [0, 1].
$'U = (\mu_{ij})_{n*c}'$ is the fuzzy membership matrix.
'J' is the objective function.

So, using the fuzzy C-means algorithm generate the cluster this is based on the fuzzy member ship matrix.

ART1 NN [Adaptive resonance theory1 neural network]

```
Step 1: Start
Step 2: Start with no cluster prototype vector
Step 3: Let I= next input vector
Step4: Find the closest cluster prototype vector and say this cluster vector T
Step 5: If T is too far from I or if there are no cluster prototype vector yet then create
        new cluster, with prototype vector equal to i. output to index of this cluster. Then
        go to step 1
Step6: otherwise (T is close to i) update T by moving it closer to I. output the T's index.
        Go to step 1.
```

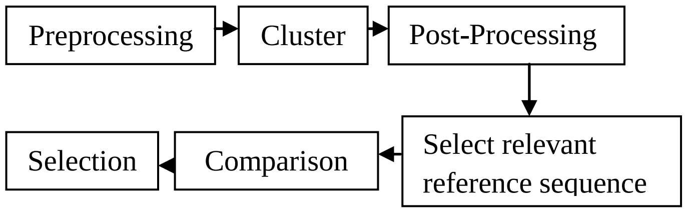
So, using the ART1 algorithm also generates the cluster based on the prototype vector initially prototype vector is null this will update depending upon the output index values.

Archicture Design:

Pre and Post Processing: Sometimes the quality of the clusters that are found can be improved by preprocessing the data. For example, when we use the squared error criteria, outliers can unduly influence the clusters that are found. Thus, it is common to try to find such values and eliminate them with a preprocessing step.

Another common approach uses post-processing steps to try to fix up the clusters that have been found. For example, small clusters are often eliminated since they frequently represent groups of outliers. Alternatively, two small clusters that are close together can be merged. Finally, large clusters can be split into smaller clusters.

While pre- and post-processing techniques can be effective, they are based on heuristics that may not always work. Furthermore, they often require that the user choose values for a number of parameters.



III. Experiment Result

In this paper executing cluster analysis algorithms and find out the time taken to produced the result as well as how much of reference sequence is match with resultant cluster.

A. Initial patterns: The bellow fig1 illustrates the initial centroid how it looks here fetching the Ruspini DNA sequence

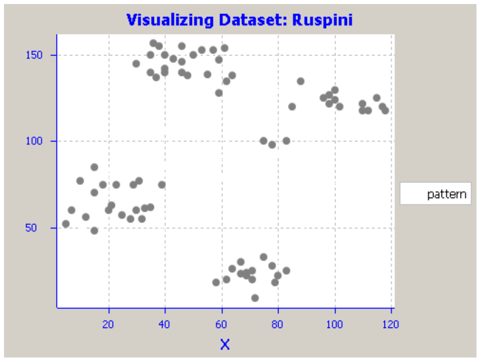


Fig 1: original initial patterns

B. *Hard partition algorithms:* The below fig2 showbench mark result for the ruspini sequence.

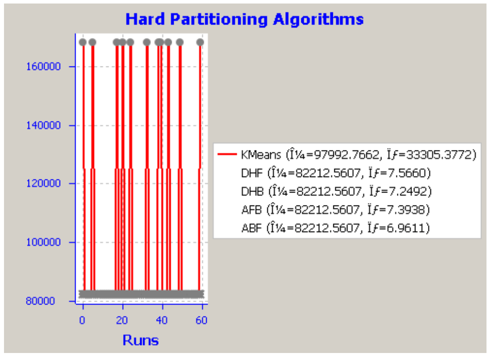


Fig 2: Hard partition algorithms

C. Hierarchical Clustering: The below fig3 show the Dendogram result for the Ruspini sequence

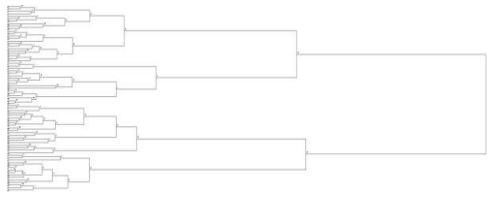


Fig 3: Dendogram representation of Hierarchical Clustering

D. Comparision of clustering algorithms: The below fig4 show the comparison of above algorithms and from this chart conclude that all Partitional algorithms (DBH,DFB,AFB,ABF,K-means) takes minimum amount of time to produces cluster and it match only 45% of reference sequence and Fuzzy and ART1 NN algorithm also takes very small amount of time to produce cluster but it match only 15 and 20% of reference sequence and hierarchical clustering it takes more time but it match the 90% of reference sequence

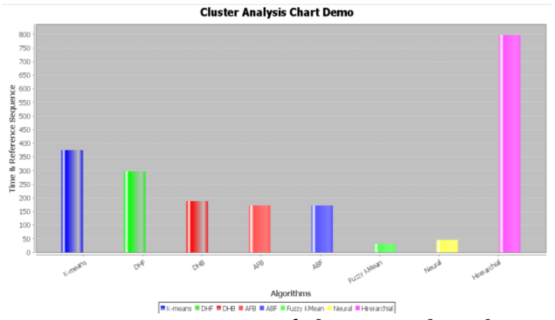


Fig4: Comparison of clustering algorithms

IV. References

[1] Y. Shi and M. Mizumoto, An improvement of neuro-fuzzy learning algorithm for tuning fuzzy rules, Fuzzy Sets and Systems, vol.118, no.2, pp.339-350, 2012..

[2] L. O. Hall and I. B. Ozyurt, Clustering with a genetically optimized approach, IEEE Trans, vol.7, no.3, pp.103-112, 2010.

[3] J. Li, X. Gao and L. Jiao, A new feature weighted fuzzy clustering algorithm, Acta Electronic Sinica, vol.34, no.1, pp.89-92, 2009.

[4] Y. Lu and X. Fan, Fuzzy weighting distance and its rationality discussing, Journal of Northern Transportation University, Beijing, 2007.

[5] Dembele, D. and Kastner, P., Fuzzy C-means method for clustering microarray data, *Bioinformatics*, 19:973–980, 2002.

[6] Kupiec, M., Ayers, B., Esposito, R.E., and Mitchell, A.P., The molecular and cellular biology of the yeast *Saccaromyces*, *Cold Spring Harbor*, 889–1036, 2001.

[7] http://www.raid-symposium.org

[8] http://www.ncbi.nlm.nih.gov

A. chittur, "Model Generation for an Intruction Detection System Using Genetic Algoritms",(accessed in Jan 2005)