International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:12, No:7, 2018

An Improved K-Means Algorithm for Gene Expression Data Clustering

Billel Kenidra, Mohamed Benmohammed

***Abstract*—**Data mining technique used in the field of clustering is a subject of active research and assists in biological pattern recognition and extraction of new knowledge from raw data. Clustering means the act of partitioning an unlabeled dataset into groups of similar objects. Each group, called a cluster, consists of objects that are similar between themselves and dissimilar to objects of other groups. Several clustering methods are based on partitional clustering. This category attempts to directly decompose the dataset into a set of disjoint clusters leading to an integer number of clusters that optimizes a given criterion function. The criterion function may emphasize a local or a global structure of the data, and its optimization is an iterative relocation procedure. The K-Means algorithm is one of the most widely used partitional clustering techniques. Since K-Means is extremely sensitive to the initial choice of centers and a poor choice of centers may lead to a local optimum that is quite inferior to the global optimum, we propose a strategy to initiate K-Means centers. The improved K-Means algorithm is compared with the original K-Means, and the results prove how the efficiency has been significantly improved.

***Keywords*—**Microarray data mining, biological pattern recognition, partitional clustering, k-means algorithm, centroid initialization.

I. Background

SINCE we focus on a strategy based on K-Means approach. In this section, several interrelated notions should be gradually presented before getting involved in such abstraction level. This section is dedicated to highlight the idea of gene expression data clustering in terms of microarray data mining with focus on partitional clustering approach.

A. Bioinformatics

With the growth of genomic datasets, it has become important to develop techniques being fast and accurate in order to quickly extract meaningful insight that a user can take advantage of. Computational tools have been involved in biology realm in order to tackle this challenge. The coupling of both disciplines is known under the name of bioinformatics.

The ultimate goal of bioinformatics is to better understand a living cell and how it functions at the molecular level. By analyzing raw molecular sequence and structural data, bioinformatics research can generate new insights and provide a global perspective of the cell [8]. High-speed genomic sequencing coupled with sophisticated computer technology will enable a physician in a clinic to rapidly sequence a patient's genome and easily detect potentially dangerous mutations and initiate early diagnosis and effective treatment of disease [1].

Billel Kenidra is with the National Superior Institute of Computer Science (ESI) Algiers, Algeria (e-mail: b_kenidra@esi.dz).

Mohamed Benmohammed is with the LIRE Laboratory, Constantine 2, Algeria (e-mail: mohamed.benmohammed@univ-constantine2.dz).

B. Gene Expression Data Clustering

In recent years, microarray gene expression studies have been actively pursued for extracting significant biological knowledge hidden under a large volume of gene expression profiles accumulated by microarray experiments. The idea of a microarray is to detect the presence and abundance of specific DNA molecules in biological samples of interests.

The expression level is represented as ratio of pixel intensities. It might represent an increase or reduction of the intensities. The analysis of microarray data, the gene expression profiles are typically employed in a p × n matrix form as:

$$\begin{pmatrix} X_{11} X_{12} & \dots & X_{1n} \\ X_{21} X_{22} & \dots & X_{2n} \\ & \vdots & \\ X_{p1} X_{p2} & \dots & X_{pn} \end{pmatrix}$$

where Xij denotes the expression intensity of the ith gene (i=1, …, p, p is the number of genes) in the j^{th} sample (j=1, …, n, n is the number of experiment conditions). After the above pre-processing steps, gene expression data can be represented by a real-valued expression matrix.

Gene expression matrix can be analyzed in two ways. For gene-based clustering, genes are treated as data objects, while samples are considered as features (dimensions). Conversely, for sample-based clustering, samples serve as data objects to be clustered, while genes play the role of features [2].

In cluster analysis, a group of objects is split up into a number of homogeneous subgroups (clusters) on the basis of a chosen measure of similarity, the objects that are close to each other are likely to be assigned to the same subgroup. The projection of these clusters on an interpretable conceptual-map leads to extract the required information, in order to take the right decision.

In bioinformatics, clustering techniques have been used to identify groups of genes with similar patterns of expression with the aim of helping to answer questions of how gene expression is affected by various diseases and which genes are responsible for specific diseases.

C. Partitional Clustering

Many clustering methods have been proposed since 1950s. Among these algorithms, we consider partitional clustering ones, this kind of algorithms divide the dataset into a specified

number of clusters. These algorithms try to optimize a square error function. Thus, they are considered as NP-hard [3]. The criterion function may emphasize the local or global structure of the data, and its optimization is an iterative procedure [4].

Partitional clustering algorithms aim at discovering the partition that present in the dataset by optimizing a specific objective function and iteratively improving the quality of the partitions. These algorithms generally require a set of initial starting points. These points change at different iterations and they can be virtual points such as the centroid of the cluster.

K-Means is the most popular partitional clustering algorithm, because it is simple to implement, fast, and easily parallelized. It spends a vast majority of its time computing distances between each of the k cluster centers and the n data points. The measure of similarity can be the distance between the data points or some descriptive concept and can be chosen differently depending on the type of the dataset of interest and the purpose of clustering.

D. Formal Description

The aim of a clustering technique is to find a suitable partition of the input dataset so that some criteria are optimized. Formally, given a dataset $S=\{x_1,x_2, \ldots ,x_n\}$ consists of n objects to be grouped into k clusters. Solving this problem aims to find a partition $C^* = \{C_1^*,C_2^*,\ldots,C_k^*\}$ of S that optimizes a quality measure, where each Ci represents a cluster. The partition must satisfy the following conditions:

1. $\forall i \quad Ci \neq \emptyset$
2. $\forall i, j \quad Ci \cap Cj = \emptyset$
3. $\cup Ci = S$

The number of ways to group n objects in k clusters is $k^n/k!$. Consequently, the number of partitions that can be formed significantly increases, depending on both the size of S and the number k together. Hence, the problem of clustering is a combinatorial problem, because the number of partitions that can be obtained grows exponentially with the volume of dataset to be classified and the number of clusters.

The problem of clustering requires appropriate parameter selection and efficient search in complex and large spaces in order to attain optimal solutions.

E. Organization of This Paper

The rest of the paper is organized as follows. Section II is dedicated to related works. Precisely, we survey some of K-Means variations that continue to spring up in literature. In Section III, the proposed approach is revealed. Section IV is devoted to experimental study and comparison. Finally, conclusions resulted from this contribution will be drawn in Section V.

II. Related Works

A. K-Means Clustering

The K-Means clustering algorithmis one of the simplest and most efficient clustering algorithms proposed in the literature of data clustering. K-Means clustering is the most widely used partitional clustering algorithm. It starts by choosing K representative points as the initial centroids [6].

Each point is associated to the nearest centroid. Once the groups are formed, the centroids for each group are updated. These two steps will be iteratively repeated while the centroids change [5].

Algorithm 1 *K*-Means Clustering

1: Select K points as initial centroids.
2: **repeat**
3: Form K clusters by assigning each point to its closest centroid.
4: Recompute the centroid of each cluster.
5: **until** convergence criterion is met.

Algorithm 1 provides an overview of the basic K-Means algorithm. When it comes to calculate the nearest centroid, a wide range of proximity measurements can be used in the K-Means algorithm. For the K-Means algorithm, Euclidean distance measurement is the most popular choice. The choice can significantly affect the centroid assignment and the quality of the final solution.

Given a dataset $D=\{x_1,x_2, \ldots ,x_N\}$ consists of N points, let us denote the clustering obtained after applying K-Means clustering by $C = \{C_1,C_2, \ldots ,C_K\}$. The Sum of Squared Errors (SSE) is the objective function. The minimum SSE is, the better the clustering. SSE is defined in (1) where $O_m$is the centroid of cluster C_m.

$$SSE(C) = \sum_{m=1}^{K} \sum_{x_i \in C_m} \|x_i - O_m\|^2 \qquad (1)$$

$$O_m = \frac{\sum_{x_i \in C_m} x_i}{|C_m|} \qquad (2)$$

B. K-Means++ Clustering

The drawback of K-Means algorithm is the initial centroids selection, K-Means++ algorithm has come to tackle this problem.

The first centroid is selected at random, while the next centroid selected is the one being the farthest from the currently selected centroid. We repeat this procedure as long as the number of centroids is less than K. Otherwise, we use K-Means algorithm initiated with these centroids [9].

The authors of [6] propose a specific way of choosing centers for the K-Means algorithm. In particular, let $D(x)$ denote the shortest distance from a data point to the closest center we have already chosen. Then, they define the following algorithm, which they call *K-Means++*.

Algorithm 2 K-Means++Clustering

1: Take one center c_1, chosen uniformly at random from X.
2: Take a new center c_i, choosing $x \in X$ with probability $\frac{D(x)^2}{\sum_{x \in X} D(x)^2}$
3: Repeat step 2 until we have taken k centers altogether.
4: Proceed as with the standard *K-Means* algorithm.

We call the weighting used in step 2 simply "D^2 weighting".

III. The Proposed Approach

This section is devoted to describe the proposed approach. It is primarily based on the use of a strategy to initiate K-Means approach since K-Means is extremely sensitive to the

initial choice of centers, i.e. every different choice of K initial centers may lead to a different result, some results are poor and some are close or equal to the global optimum.

What distinguishes the proposed approach compared to K-Means approach resides in the quality of results. We adopted some techniques in order to skillfully detect K centers as a wise choice that is more likely to conduct to a best result (i.e. close or equal to global optimum).

A. Algorithmic Description

We can recapitulate the proposed idea through this algorithmic description. We have a set S of points to be grouped into K clusters, and every point is getting weighted to w = 0; let j = 0:

Step 1: for each point p ∈ S, for each point pp ∈ S and p ≠ pp, we verify if p is the closest point of pp. If this is the case, then p will get weighted to w = w + 1.

Step 2: sorting these weights in descending order in an array named weights-array.

Step 3: based on weights-array, we select only candidate points, i.e. only points having a weight greater than j.

Step 4: among these candidate points, we seek the longest distance between two points, and we use one of them as a reference point.

Step 5: calculating the distances between the reference point, and every candidate points.

Step 6: sorting these distances in ascending order in an indexed array named distances-array.

Step 7: calculating the difference between the distance i and the distance i+1 knowing that i varies, i = {0 ,..., the index of the penultimate distance in distances-array}.

Step 8: sorting these differences in descending order in an array named differences-array, by keeping the index of each value.

Step 9: subdividing the first K-1 values. Since each value equals distance i+1 minus distance i, we proceed to sort these values in ascending order based on their index i according to distances-array.

Step 10: we can detect a primary cluster based on index i according to distances-array in step 6, the distance of every non-clustered point of this cluster is less or equals than the distance having index = i.

Step 11: K-1 values in step 9 corresponding to K-1 primary clusters. The remaining non-clustered points form the K^{th} primary cluster.

Step 12: calculating centers of these detected clusters.

Step 13: integrating K-Means algorithm using these detected centers to cluster all points of S.

Step 14: computing the result quality, and storing it in an array named results-array.

Step 15: resetting j = j + 1, and we repeat step 3. If the number of candidate points is greater than K, then we repeat the process from step 4.

Step 16: choosing the best result from results-array.

B. Illustrative Example

An illustrative example might make the proposed approach clear and easier to understand what behind the idea.

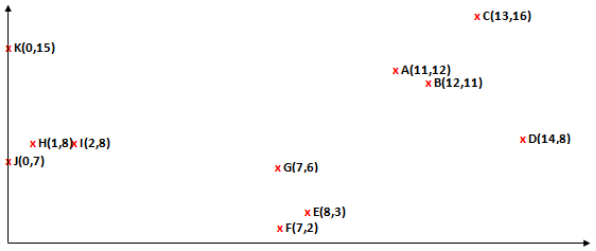

Fig. 1 Set of points to be grouped into 3 clusters

Given a set S={A(11,12), B(12,11), C(13,16), D(14,8), E(8,3), F(7,2), G(7,6), H(1,8), I(2,8), J(0,7), K(0,15)} of points to be grouped into three clusters, and every point is getting weighted to w = 0; let j = 0; and

Step 1: The number of clusters K = 3. weights-array should be:

TABLE I
STEP 2

Point	Closest of	Its weight		Point	Closest of	Its weight
A	B,C	2		H	I,J,K	3
B	A,D	2	➔	A	B,C	2
C	∅	0		B	A,D	2
D	∅	0	Step 2: weights-array in descending order	E	F,G	2
E	F,G	2		F	E	1
F	E	1		I	H	1
G	∅	0		C	∅	0
H	I,J,K	3	➔	D	∅	0
I	H	1		G	∅	0
J	∅	0		J	∅	0
K	∅	0		K	∅	0

Step 3: C,D,G,J,K considered as outliers, since its weights = 0. The other points are considered as candidate points.

Step 4: The longest distance is between H and B. We can take B as a reference point.

Step 5: distances-array should be:

TABLE II
STEP 6

Point	Its distance to B		Point	Its distance to B	Index
A	1.41	➔	B	0	0
B	0	Step 6: Distances -array in Ascending order	A	1.41	1
E	8.94		E	8.94	2
F	10.30		F	10.3	3
H	11.40	➔	I	10.44	4
I	10.44		H	11.4	5

Step 7: differences-array should be:

TABLE III
STEP 8

differences	Index	→	differences	Index
1.41 - 0 = 1.41	0	Step 8: Differences -array in Descending order →	7.53	1
8.94 - 1.41 = 7.53	1		1.41	0
10.3 - 8.94 = 1.36	2		1.36	2
10.44 - 10.3 = 0.14	3		0.96	4
11.4 - 10.44 = 0.96	4		0.14	3

Step 9: subdividing the first K-1 values.

TABLE IV
STEP 9

differences	Index	→	differences	Index
7.53	1		1. 41	0
1. 41	0		7.53	1

Step 10: The first primary cluster should contain:

TABLE V
STEP 10-A

Point	Its distance to B	Index
B	0	0

The second primary cluster should contain:

TABLE VI
STEP 10-B

Point	Its distance to B	Index
A	1.41	1

Step 11: The third primary cluster should contain the remaining points of distances-array:

TABLE VII
STEP 11

Point	Its distance to B	Index
E	8.94	3
F	10.3	4
I	10.44	5
H	11.4	6

Step 12: Center of first primary cluster = Center of B(12,11) = (12,11)

Center of second primary cluster = Center of A(11,12) = (11,12)

Center of third primary cluster = Center of E(8,3) F(7,2) I(2,8) and H(1,8) = (4.5,5.25)

Step 13: By integrating K-Means algorithm using these detected centers to cluster all points of S. the partition will be: Cluster1={A,C}, Cluster2={B,D}, Cluster3={E,F,G,H,I,J,K}. consequently, it is a bad result.

Step 14: computing the result quality, and storing it in results-array.

Step 15: based on weights-array in step 2, the candidate points will be:

TABLE VIII
STEP 15

Point	Closest to	Its weight
H	I,J,K	3
A	B,C	2
B	A,D	2
E	F,G	2

F,G,I,C,D,J,K are considered as outliers, since its weights > j / j has been incremented to 1.

Since the number of candidate points is greater than K / K=3, then we repeat the process from step 4.

Step 5: distances-array should be:

TABLE IX
STEP 6-SECOND ITERATION

Point	Its distance to B	→	Point	Its distance to B	Index
A	1.41	Step 6: Ascending order →	B	0	0
B	0		A	1.41	1
E	8.94		E	8.94	2
H	11.40		H	11.4	3

Step 7: differences-array should be:

TABLE X
STEP 8-SECOND ITERATION

differences	Index	→	differences	Index
1.41 - 0 = 1.41	0	Step 8: Descending order →	7.53	1
8.94 - 1.41 = 7.53	1		2.46	2
11.4 - 8.94 = 2.46	2		1.41	0

Step 9: subdividing the first K-1 values.

TABLE XI
STEP 9-SECOND ITERATION

differences	Index
7.53	1
2.46	2

Step 10: The first primary cluster should contain

TABLE XII
STEP 10-A-SECOND ITERATION

Point	Its distance to B	Index
B	0	0
A	1.41	1

The second primary cluster should contain:

TABLE XIII
STEP 10-B-SECOND ITERATION

Point	Its distance to B	Index
E	8.94	2

Step 11: The third primary cluster should contain the remaining points of distances-array:

TABLE XIV
STEP 11-SECOND ITERATION

Point	Its distance to B	Index
H	11.4	3

Step 12: Center of first primary cluster = Center of A(11,12) and B(12,11) = (11.5,11.5)
- Center of second primary cluster = Center of E(8,3) = (8,3)
- Center of third primary cluster = Center of H(1,8) = (1,8)

Step 13: By integrating K-Means algorithm using these detected centers to cluster all points of S. the partition will be: Cluster1={A,B,C,D}, Cluster2={E,F,G}, Cluster3={H,I,J,K}. It is a good result.

Step 14: computing the result quality, and storing it in results-array.

Step 15: condition not verified, so we stop running.

Step 16: choosing the best result from results-array.

C. Outline of the Proposed Approach

The purpose of step 1, step 2, and step 3 is to detect the densest regions in the dataset. Every region should be represented by a candidate point at least, we can know if this region is more or less dense according to the weight of its candidate point, the greater the weight, the densest the region is. Conversely, every outlier point will have a weight equal to zero.

But, it is not enough, we need to detect the longest distance between two candidate points, and we use one of them as a reference point in order to reveal the density and the remoteness among these candidate points, and that is what steps from step 4 to step 8 are all about.

Steps from step 9 to step 12 are to discover primary clusters and their centers. Step 13 aims to integrate K-Means, based on these detected centers.

By using this strategy, we ensure that the initial choice of centers is reasonable, since these centers are remote from each other and they are in the densest regions. Consequently, the drawback of K-Means has been fixed up by adopting this strategy.

IV. Experiments and Results

In this section, we present different genomic dataset as well as the evaluation measures used in our experiments, then we display the obtained results and theirs comparison, finally a discussion is conducted to evaluate the correctness and the efficiency of the proposed approach.

TABLE XV
Genomic Dataset Used in Evaluation

Dataset Name	Tissue	Total Samples	Classes Number	Samples per Class	Total Genes
alizadeh-v2	Blood	62	3	42, 9, 11	2093
alizadeh-v3	Blood	62	4	21, 21, 9, 11	2093
armstrong-v1	Blood	72	2	24, 48	1081
armstrong-v2	Blood	72	3	24, 20, 28	2194
bredel	Brain	50	3	31, 14, 5	1739
chen	Liver	179	2	104, 75	85
chowdary	Breast, Colon	104	2	62, 42	182
dyrskjot	Bladder	40	3	9, 20, 11	1203
garber	Lung	66	4	17, 40, 4, 5	4553
golub-v1	Bone Marrow	72	2	47, 25	1877
golub-v2	Bone Marrow	72	3	38, 9, 25	1877
khan	Multi-Tissue	83	4	29, 11, 18, 25	1069
laiho	Colon	37	2	8, 29	2202
Nutt-v3	Brain	22	2	7, 15	1152
pomeroy-v1	Brain	34	2	25, 9	857
pomeroy-v2	Brain	42	5	10, 10, 10, 4, 8	1379
shipp-v1	Blood	77	2	58, 19	798
singh	Prostate	102	2	50, 52	339
west	Breast	49	2	25, 24	1198
yeoh-v1	Bone Marrow	248	2	43, 205	2526

A. Dataset Used in Experiments

To assess the performance of the proposed approach, an experimental study was conducted using 20 different publicly available gene expression datasets, having the proprieties shown in Table XV.

As we mentioned above, gene expression matrix can be analyzed in two ways, either by treating genes as data objects, while samples are considered as features (dimensions), or conversely, treating samples as data objects to be clustered, while genes play the role of features.

In our empirical studies, we adopted the second strategy, i.e. the clustering of samples (experiment conditions). The significance of this clustering assists in diagnosis of the disease condition, and it discloses the effect of certain treatment on genes.

B. Evaluation Measures Used in Experiments

One of fundamental challenges of clustering is how to evaluate results, without auxiliary information. A common approach for evaluation of clustering results is to use validity indexes. Clustering validity approaches can use: external evaluation measure like F-measure & internal evaluation measure like Davies-Bouldin index.

1) External Evaluation Measure

It is a comparison between the obtained result and the expected result (benchmark as a reference of comparison see Table XV). One of the most widely used external evaluation measure is *F-measure*.

- **F-measure**

If we have a reference partition P of the dataset (benchmark), which is probably derived from previously known domain knowledge, we can simply evaluate the cluster result C by comparing the similarity between P and C through some *statistic* such as *F-measure*. The *F-measure* values are within the interval [0,1] and larger values indicate higher clustering quality. *F-measure* equals 1, that means C is identical to P, and it is an optimal solution.

F-measure combines the precision and recalls concepts from information retrieval. We then calculate the recall and precision of that cluster for each class as:

$$R(i,j) = \frac{n_{ij}}{n_i} \text{ and } Precision(i,j) = \frac{n_{ij}}{n_j}$$

where n_{ij} is the number of objects of class i that are in cluster j, n_j is the number of objects in cluster j, and n_i is the number of objects in class i. The *F-measure* of cluster j and class i is given by the following equation:

$$F(i,j) = \frac{2 \,.\, Recall(i,j) \,.\, Precision(i,j)}{Precision(i,j) + Recall(i,j)}$$

2) Internal Evaluation Measure

Other approaches measure the quality of generated clusters from the concept of "homogeneity and separation". They are also defined to measure to what degree the data objects are similar inside one cluster, while dissimilar between different clusters, Davis-*Bouldin index* is one of the most commonly used.

- **Davies-Bouldin index (DB)**

This index aims to identify sets of clusters that are compact and well separated. The *Davies-Bouldin index* is defined as:

$$DB = \frac{1}{C}\sum_{i=1}^{C} Max_{i \neq j}\left\{\frac{d(X_i)+d(X_j)}{d(C_i,C_j)}\right\}$$

where C denotes the number of clusters, i, j are cluster labels, then $d(X_i)$ and $d(X_j)$ are all samples in clusters i and j to their respective cluster centroids, $d(C_i, C_j)$ is the distance between these centroids. Smaller value of DB indicates a better clustering solution [7].

C. Empirical Results and Comparison

Since all these algorithms are stochastic, we performed multiple runs over all 20 benchmarks, and each value is the average of 50 runs.

Table XVI shows the results of F-measure, obtained by each algorithm. F-measure is a quantitative comparison between obtained clusters and benchmarks clusters; F-measure = 1 means optimal solution. Consequently, the clustering of better quality is the one that maximizes F-measure (in bold). From this table, we can conclude that the proposed algorithm is the best in terms of F-measure.

Table XVII presents the results of the Davies-Bouldin index obtained by each algorithm. This measure is lower as the obtained clusters are compact and far from each other. Consequently, the clustering of better quality is the one that minimizes the Davies-Bouldin index (in bold). Based on this table, we can conclude that the proposed algorithm is ranked first in terms of Davies-Bouldin index.

TABLE XVI
RESULTS OF F-MEASURE

Benchmark	Improved K-Means	K-Means	K-Means++
alizadeh-v2	**1**	0.8201	0.8518
alizadeh-v3	**0.7599**	0.6771	0.6432
armstrong-v1	0.6667	0.7215	**0.7329**
armstrong-v2	0.7105	0.7567	**0.8264**
bredel	**0.7996**	0.7024	0.6590
chen	0.6446	**0.8096**	0.6895
chowdary	0.6697	0.6697	**0.6764**
dyrskjot	**0.7990**	0.7367	0.5376
garber	**0.6126**	0.5784	0.5753
golub-v1	0.7240	**0.8460**	0.8357
golub-v2	**0.8859**	0.7789	0.8069
khan	**0.6926**	0.6230	0.6031
laiho	0.6771	**0.7511**	0.7289
Nutt-v3	**1**	0.7854	0.7045
pomeroy-v1	0.6644	**0.6750**	0.6572
pomeroy-v2	**0.7358**	0.6273	0.5859
shipp-v1	**0.7341**	0.6891	0.7134
singh	0.6286	0.6286	0.6286
west	0.6571	**0.7437**	0.6607
yeoh-v1	**0.9836**	0.8991	0.8143

TABLE XVII
RESULTS OF DAVIS-BOULDIN INDEX

Benchmark	Improved K-Means	K-Means	K-Means++
alizadeh-v2	**1.6366**	2.2306	1.7122
alizadeh-v3	**1.6032**	2.3249	1.7575
armstrong-v1	**1.9288**	1.9636	1.9683
armstrong-v2	2.0268	**1.8721**	2.1469
bredel	2.1958	2.1241	**1.8838**
chen	**1.2488**	2.4253	1.4717
chowdary	**0.8818**	0.8819	0.8889
dyrskjot	1.8901	1.7330	**1.4242**
garber	1.7822	2.6147	**1.7252**
golub-v1	**1.8092**	1.8891	1.8786
golub-v2	**1.7489**	1.9410	1.9531
khan	1.7493	1.9387	**1.4035**
laiho	1.7147	1.8561	**1.6953**
Nutt-v3	1.7910	**1.5726**	1.6410
pomeroy-v1	**1.4546**	1.6875	1.8476
pomeroy-v2	**1.4930**	1.7259	1.6233
shipp-v1	**0.3683**	1.4970	1.2275
singh	0.8425	0.8425	0.8425
west	**0.8815**	2.0582	1.7026
yeoh-v1	3.0918	2.5013	**2.3834**

D. Discussion and Evaluation

We notice that yeoh-v1 benchmark outperforms all

algorithms in terms of f-measure (the higher the value the better the result is), but it does not in terms of Davis-Bouldin. The interpretation of this phenomenon amounts to conclude that even the resulting clusters of an algorithm are more similar than other algorithm' ones to benchmark clusters, these resulting clusters do not guarantee the best ratio of compactness and remoteness.

In the end, these experimental results have clearly revealed the difference before and after the improvement of K-Means algorithm. This algorithm could be used in big data (millions of objects), and it is more likely to provide very good results, on one hand, since the problem of the local optimum had been fixed up, on the other hand, the improved K-Means is simple to implement, fast, and easily parallelized.

V. Conclusion

The K-Means algorithm is one of the most widely used clustering techniques, many researchers have developed several variations of the K-Means clustering during the last four decades. This paper has surveyed some of these variations and has presented an improvement of K-Means algorithm.

Due to the inefficiency of the K-Means strategy in terms of selecting the initial centroids, an improved K-Means algorithm has been proposed in this paper in order to fix up this problem, it consists of predicting a reasonable choice of the initial centers whose result is close or equal to the global optimum.

Because the objective is to optimize the grouping, comparative studies have been achieved between some state of art approaches and the proposed one. The obtained results in terms of internal and external validity indexes show the efficiency of the improvement.

References

[1] Xiong J, Essential Bioinformatics. Texas A&M University, 2006.
[2] Miyoung Shin and Jaeyoung Kim, Microarray Data Mining for Biological Pathway Analysis, ISBN 978-3-902613-53-0, 2009, pp.438.
[3] Prasad G.V.S.N.R.V, Venkata K, and Vijaya K. Automatic Clustering Approaches Based On Initial Seed Points. International Journal on Computer Science and Engineering (IJCSE), ISSN: 0975-3397 Vol. 3 No. 12 December 2011.
[4] Journal of Intelligent Information Systems, Kluwer Academic Publishers. Manufactured in The Netherlands, 2001, pp.107–145.
[5] S. Lloyd. Least squares quantization in pcm. IEEE Transactions on Information Theory, 28(2):129{137, 1982.
[6] D. Arthur and S. Vassilvitskii. K-means++: The advantages of careful seeding. In Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, 2007, pp.1027–1035.
[7] E. Rendon, I. Abundez, A. Arizmendi, Internal versus External cluster validation indexes. International Journal of Computers and Communications 5(1), 2011, pp.27–34.
[8] K. Alemu, The Role and Application of Bioinformatics in Plant Disease Management. Advances in Life Science and Technology, ISSN 2225-062X, Vol.28, 2015.
[9] K. Reddy, B Vinzamuri, A Survey of Partitional and Hierarchical Clustering Algorithms, Data Clustering: Algorithms and Applications, 2007.