HeMI++: A Genetic Algorithm based Clustering Technique for Sensible Clusters

A. H. Beg
School of Computing and Mathematics
Charles Sturt University
Panorama Avenue, Bathurst 2795, Australia
E-mail: abeg@csu.edu.au

Md Zahidul Islam
School of Computing and Mathematics
Charles Sturt University
Panorama Avenue, Bathurst 2795, Australia
E-mail: zislam@csu.edu.au

Vladimir Estivill-Castro
Departament de Tecnologies de la Informació i les Comunicacions,
Universitat Pompeu Fabra, Roc Boronat, 138, 08018 Barcelona, Spain
E-mail:vladimir.estivill@upf.edu

***Abstract*—We propose a new clustering technique called HeMI++. It uses cleansing and cloning operations that help to produce sensible clusters. HeMI++ learns necessary properties of a reasonable clustering solution for a dataset from a high-quality initial population, without requiring any user input. It then disqualifies the chromosomes that do not satisfy the properties through its cleansing operation. In the cloning operation, HeMI++ replaces the chromosomes by high-quality chromosomes already found in the initial population. We compare HeMI++ with six (6) existing techniques on twenty (20) publicly available datasets regarding the Tree Index. Our experimental results indicate a clear superiority of HeMI++ over existing methods. We also apply HeMI++ on a brain dataset and demonstrate its ability to produce sensible clusters.**

Keywords—data mining, clustering, genetic algorithms, artificial intelligence (AI)

I. Introduction

A. Clustering

Clustering is a well-known and an important technique that aims to group the similar records in one cluster and dissimilar records in different clusters [1-6]. Clustering has enhanced data analysis in a wide range of areas such as business [11], machine learning [12] social network analysis [13] and medical imaging [14].

K-means [5] is among the top ten most used clustering techniques [15] for its simplicity and a low complexity of $O(dn)$ time where n is the size of the data set and d is the dimension. Although it is popular for its simplicity, it also has several well-known drawbacks. Fundamentally, K-means is statistically biased (even on data from a mixture of multivariate normal distributions with equal variance and just one dimension, K-means converges to incorrect means). Another issue commonly attributed to K-means is its requirement of a user-defined number (k) of clusters [6, 16]. In reality, it can be difficult for a user (data analyst) to estimate the suitable number of clusters in advance.

Because K-means is essentially a hill-climber/gradient descent search method on a least square's loss, K-means has a tendency to converge to poor local optima [10,16,17]. The random selection of the initial cluster centers in K-means is also considered to be a major drawback since it often results in a poor-quality final clustering solution [6,7]. Arthur and Vassilvitskii [6] proposed a clustering technique called K-means++ that shows a theoretical competitive ratio if initialisation is cleverly used. However, experimental evaluations show that this variant does not alleviate K-means problems in practice [7].

To overcome these limitations in recent years, many evolutionary algorithms such as stochastic search [18], simulated annealing [19] and genetic algorithms [2-4,7] for clustering have been proposed that achieve encouraging results.

Genetic algorithms (GA) use randomised search and optimisation techniques emulating the concepts of natural activity of genes, individual selection, and the evolutionary process [2-4,7]. Typically, in GA for clustering, a chromosome is an encoding of a clustering solution, and a gene within a chromosome corresponds to a vector that encodes the centre of a cluster.

However, there has some limitations on the existing GA-based clustering techniques. Typically, in the initial population, the number of genes of a chromosome are generated randomly. The genes are also selected randomly from a dataset instead of careful consideration [3,4]. Carefully genes selection increases the possibility of getting high-quality chromosome in the initial population (recall the theoretical result of K-means++). Having high-quality chromosomes in the initial population improves speed as it typically reduces the algorithm's number of iterations [2].

Rahman and Islam [2] presented a GA-based clustering technique known as GenClust that produces high-quality chromosomes in the initial population. However, users must provide a set of radius values for the chromosomes in the initial population. It can be difficult for a user to provide a suitable set of radius values.

Our previous technique, HeMI [7] produces high-quality chromosomes in the initial population without requiring any user-defined parameter and produces good-quality clustering results. We recently proposed a genetic algorithm-based clustering technique called GenClust++ [20] where we took a slightly similar approach for the initial population of HeMI. In the initial population, GenClust++ uses K-means [5] or K-means++ [16] multiple times with different values for the number *k* of clusters. That is, each value in {2, 3, …. 10} is used as the number *(k)* of clusters for K-means or K-means++. Thus, at least nine different clustering solutions progress in GenClust++ to a first ranking and culling phase. The initial population is probabilistically elected under a regime where chromosomes with higher cluster quality have greater chance to survive.

However, unlike GenClust++, HeMI does not select the initial chromosomes probabilistically, instead, HeMI selects only the best chromosomes based on their fitness values. Moreover, HeMI++ generates 50% of the initial population from a *deterministic phase* and 50% from a *random phase*. In the *deterministic phase*, HeMI picks the initial population from the pool of chromosomes obtained through K-means using the value of k ranging from 2 to 10. In many data sets, the true number of clusters is more than 10. To handle such situations, in the *random phase* HeMI generates the initial population randomly with the value of k ranging from 2 to $\sqrt{n}$ (where n is the number of records in a dataset).

However, HeMI also has some limitations. Although it produces high-quality chromosomes in the initial population, it does not take full advantage of them since they are not used in any other genetic operations. Through our intelligent use of the high-quality initial population, the genetic operations could be made more effective. It also does not use any data-driven approach to learn the necessary properties of sensible clusters. Sometimes it produces non-sensible clustering solutions.

In our previous paper [8] we demonstrate that some recent clustering techniques such K-means [5], K-means ++ [6], AGCUK [3], GAGR [4], GenClust [2] and HeMI [7] produces non-sensible clusters. Sometimes, they obtain a huge number of clusters, and sometimes they derive only two clusters, where one cluster contains one record, and the other cluster contains all remaining records. Interestingly, these clustering solutions often attain high fitness values based on existing evaluation criteria. We in this paper propose a new clustering technique called HeMI++ that produces sensible clusters with high fitness values.

B. Main contributions of this study

In this paper we propose a new clustering technique called **HeMI++**. HeMI++ is inspired by our previous techniques called CSClust [8] and HeMI [7] however, significant innovations upon those methods are the following:

Technical contributions:

- The use of multiple streams:
  - CSClust did not use any multiple streams.
  - HeMI used multiple streams, but it did not learn the reasonable cluster property from a data set and did not apply such knowledge for producing reasonable clusters.
  - HeMI++ uses multiple streams; like HeMI and unlike CSClust. Unlike HeMI, it learns the reasonable clustering property from a data set and applies that to produce reasonable clustering solutions.
- Learning reasonable properties:
  - HeMI did not learn the reasonable clustering property and apply that.
  - CSClust learnt some reasonable property by applying DB-Index [26] on the initial population. This approach was problematic since the selection was biased by the inductive principle under of DB-Index.
  - HeMI++ learns properties of a reasonable clustering solution through a new approach (without using DB-Index), and then applies the knowledge in producing its clustering solutions (see Section 2.3.4).

Experimental contributions:

- Comparison: Techniques and Data sets
  - We compare HeMI++ with six other existing techniques on twenty (20) natural data sets.
  - CSClust was compared with five existing techniques on ten (10) data sets.
- Complexity analysis:
  - We also analyze the complexity of HeMI++ and compare it with six other existing techniques, whereas in CSClust we did not present any complexity analysis.

The rest of the paper is organized as follows: The proposed technique is described in Section II. In Section III, we discuss experimental results and Section IV provides the concluding remarks.

II. Our Technique

A. Main Components of HeMI++

We first mention the main steps of HeMI++ as follows and then explain each of them in detail.

```
BEGIN
    Step-1: Normalization
    DO: k=1to m /* m is the user defined number of streams */
        Step-2: Population Initialization
    END
    Step-3: Selection of Sensible Properties
    DO: j=1to G /* G is the user defined number of intervals*/
        DO: k=1to m /* m is the user defined number of streams */
            DO: t=1to I /* I=10; I is the number of iterations */
                Step-4: Noise-Based Selection
                Step-5: Crossover Operation
                Step-6: Twin Removal
                Step-7: Three Steps Mutation Operation
                Step-8: Health Improvement Operation
                Step-9: Cleansing Operation
                Step-10: Cloning Operation
                Step-11: The Elitist Operation
            END
        END
        Step-12: Neighbour Information Sharing
    END
    Step-13: Global Best Selection
END
```

a. Component 1: Normalization

HeMI++ first takes a data set D as input and then normalizes each attribute of the data set D in order to consider each attribute equally regardless of their domain size. This step is the same as HeMI [7].

b. Component 2: Multiple Stream

HeMI++ uses the multiple stream approach of HeMI in order to take the advantage of using a big population through multiple streams where each stream contains relatively small number of chromosomes. It can process each stream separately in parallel in order to reduce the time complexity. It generates the chromosomes for each stream separately through the population initialization. Various components such as noise-based selection, crossover and mutation are applied on each stream separately.

c. Component 3: Population Initialization

HeMI++ generates the initial population of $|P|$ chromosomes, $|P|/2$ from the *deterministic phase* and $|P|/2$ from the *random phase*. The value of $|P|$ in HeMI++ is set to 20. This component is same as HeMI [7].

d. Component 4: Selection of Sensible Properties

HeMI++ selects $|P|$ top chromosomes from the generated initial chromosomes based on their fitness (DB value). Davis Bouldin Index [26] is biased towards low number of clusters and low number of records in a cluster [8]. Therefore, HeMI++ finds the necessary properties of a sensible clustering solution. The properties of the sensible clustering solution are then used in each generation in order to ensure that chromosomes in a population do not contradict the properties.

In the initial population, HeMI++ produces 9×4=360 chromosomes as it has 4 streams. Each stream generates 90 chromosomes. HeMI++ finds the minimum number of records in a cluster for each of the 360 chromosomes. It then sorts these number in descending order and calculates the median of these numbers. The median value is then used as a property (of a sensible clustering solution) relating to the minimum number of records in a cluster. HeMI++ similarly finds the minimum and maximum number of clusters in a clustering solution, based on the 360 chromosomes. These values are then used in the cleansing operation in order to identify a sensible clustering solution.

Note that CSClust [8] uses a similar component. However, there are some significant differences, First, CSClust does not use multiple streams and hence it finds the properties based on 90 chromosomes of its single stream. Second, in order to find the minimum number of records in a cluster CSClust identifies the best 20 chromosomes according to their DB Index values and then picks the minimum number of records in a cluster out of all clusters in these 20 chromosomes. Since the best 20 chromosomes are selected based on their DB Index values, CSClust also suffers from the drawbacks of DB Index in identifying the properties of a sensible clustering solution.

e. Component 5: Noise Based Selection

The chromosomes of two generations are compared in order to select the chromosomes for the consequent genetic operations. HeMI also uses this component.

f. Component 6: Crossover Operation

All chromosomes in a population participate in crossover pair by pair. The best chromosome (available in the current population) is selected as the 1st chromosome of the pair and the 2nd chromosome of the pair is selected probabilistically using the roulette wheel technique [6, 9]. The probability of a chromosome P_j is computed as $T_j = (f_j / \sum_{i=1}^{|P|} f_i)$. Here, f_j is the fitness of the chromosome P_j and $|P|$ is the size of the current population

There are many approaches for crossover such as single-point, multi-point, arithmetic and path-based crossover [2, 4, 6, 20]. However, many existing techniques [2, 6, 7] use a single-point crossover, and it is commonly used in genetic algorithms. Moreover, Peng et al. [21] suggest that a single-point crossover performs better than a multi-point crossover. Therefore, HeMI++ uses a single-point crossover where each chromosome of a pair is randomly divided into two segments, where some genes of the chromosome fall in one segment and other genes fall in the other segment. A segment of the first chromosome is then swapped with a segment of the second chromosome. Thus, two offspring chromosomes are generated from two-parent chromosomes.

g. Component 7: Twin Removal

We use the twin removal approach [2] to change/remove the identical genes. If a chromosome has two identical genes and, if the length of the chromosome is more than two, then HeMI++ deletes one of the two identical genes. If the number of genes of a chromosome is two and both genes are identical, HeMI++ then randomly changes an attribute value of a gene to ensure that there are no identical genes.

h. Component 8: Three Steps Mutation Operation

The mutation operation of the proposed technique changes each chromosome using three operations: division, absorption and a random change This component is same as HeMI [7].

i. Component 9: Health Improvement Operation

This component aims to continuously improve the health of chromosomes within a population in order to ensure the presence of healthy (high-quality) chromosomes in each generation. This component is same as HeMI [7].

j. Component 10: Cleansing Operation

The aim of this component is to identify the chromosomes in a population with sensible and non-sensible solutions. This component is same as CSClust [8].

k. Component 11: Cloning Operation

The cloning operation replaces the sick chromosomes found in the cleansing operation. This component is same as CSClust [8].

l. Component 12: The Elitist Operation

The elitist operation carries the best chromosome throughout the generations and pass it to the next generation in order to improve the quality of the population [2-4, 6, 20]. This component is same as HeMI [7].

m. Component 13: Neighbor Information Sharing

HeMI++ uses the neighbor information sharing approach of HeMI [7] in order to share/exchange the best chromosome among neighboring streams at a regular interval such as at every 10th iteration. This component is same as HeMI [7].

n. Component 14: Global Best Selection

HeMI++ uses this component in order to find the global best chromosome among multiple streams. This component is the same as HeMI [7].

B. The HeMI++ Algorithm

We now present the HeMI++ algorithm, whose main steps is shown in Section 2 (A). HeMI++ first takes a data set *D* as an input and normalizes all attributes separately. It then takes the user defined number of multiple streams as like HeMI. The default number of multiple streams in HeMI++ is set to 4.

HeMI++ then produces initial chromosomes for each stream separately through the Population Initialization. It then applies the Selection of Sensible Properties same as CSClust in order to find the necessary properties of a sensible clustering solution. HeMI++ applies the noise-based selection operation same as HeMI.

HeMI++ then sequentially applies the Crossover, Twin Removal, Mutation and Health Improvement operation. HeMI++ applies the cleansing and cloning operation in order to increase the chance that all chromosomes in a population do not contradict with the properties of a sensible solution.

It then performs the Elitist operation to find the best chromosome. In order to take the advantage of multiple streams HeMI++ then applies the neighbor information sharing component same as HeMI at a regular interval. In this study, the default value of the interval is 10 iterations. At the end of all iterations, HeMI++ applies the Global Best Selection operation same as HMI in order to find the final clustering solution.

III. Experimental Results and Discussion

A. The Data Sets and the Evaluation Criteria

We empirically compare our proposed technique called HeMI++ with six existing techniques namely K-means [5], K-means ++ [6], AGCUK [3], GAGR [4], GenClust [2] and HeMI [7] on a brain data set (CHB-MIT data set) [24, 25]. HeMI++ also compared with these existing techniques on 20 natural data sets that are available from the UCI machine learning repository [22]. HeMI++ is compared with these techniques because they are recent, and better than many other techniques as shown in the literature [2-4,7,8]. Detailed information about the data sets is presented in Table 1.

TABLE 1: A brief description of the data sets

Data set	**No. of Records**
Hepatitis (HT)	155
Glass Identification (GI)	214
Statlog Heart (STH)	270
Vertebral Column (VC)	310
Ecoli (EC)	336
Leaf (LF)	340
Liver Disorder (LD)	345
Credit Approval (CA)	690
Breast Cancer Wisconsin Original (WBC)	699
Blood Transfusion (BT)	748
Pima Indian Diabetes (PID)	768
Statlog Vehicle Silhouettes (SV)	846
Bank Note Authentication (BN)	1,372
Contraceptive Method Choice (CMC)	1,473
Yeast (YT)	1,484
Image Segmentation (IS)	2,310
Wine Quality (WQ)	4,898
Page Blocks Classification (PBC)	5,473
MAGIC Gamma Telescope (MGT)	19,020
Chess (King-Rook vs. King) (CKRK)	28,056

B. The Parameters used in the Experiments

For the experimentation of AGCUK [3], GAGR [4] GenClust [2], HeMI [7] and HeMI++ the population size is set to be 20 and the number of iterations/generations is set to be 50. In order to ensure a fair comparison among the techniques we maintain this consistency.

In the experiments, the number of iterations of K-means, K-means++, and the number of iterations of K-means in GenClust set to be 50. The number of clusters k in GAGR, K-means and K-means++ is generated randomly. The threshold value for K- means is define as 0.005.

The r_{max} and r_{min} values in AGCUK and HeMI are set to 1 and 0 respectively. For the cluster evaluation technique Tree Index [9,10], we need to build a decision tree from a data set where records are labelled on the clustering result that is being evaluated. While building the decision tree we need to assign a minimum number of records for each leaf.

In this study we assign 1% of records of a data set, as long as it stays within the range between 2 to 15. If 1% of records is less than 2 then we assign 2, and if 1% of records is more than 15 then we assign 15. On each data set, we run HeMI++ 20 times and We present the average clustering results of each technique.

C. Brain Data Set Pre-processing

We prepare the CHB-MIT Scalp EEG data set [2,25] which contains EEG recordings of 22 epileptic patients from different age groups.

Most of the cases 23 channels were used, only in some cases 24 or 26 channel were used. We divide the data in epochs of 10 seconds for each channel. We then calculate the Maximum (Max), Minimum (Min), Mean, Standard deviation (Std), Kurtosis, Skewness, Entropy, Line length and Energy for each epoch. Therefore, from each channel of one-hour data we get 360 records containing nine attributes: Max, Min, Mean, Std, Kurtosis, Skewness, Entropy, Line Length and Energy.

In this article, we prepare one-hour data of one patient. This data set has the recordings of 23 channels. Hence, from all 23 channels altogether, we get 360*23=8280 records. In this data set the patient experienced a seizure for around 40 seconds (from the 2996th second to 3036th second). During this period, we get 5 records. These records are considered as seizure records and all other records are considered as non-seizure records. Therefore, from the chb_01_03 data set altogether we get 23*5= 115 seizure records and 8165 non-seizure records.

D. Evaluation of HeMI++ and other techniques on the MIT-chb01_03 data set

In this section, we empirically compere HeMI++ with K-means [5], K-means ++ [6], AGCUK [3], GAGR [4], GenClust [2] and HeMI [7] on a brain data set (MIT-chb01_03) through visual analysis of clustering results. We also compare all the techniques based on Tree Index [9,10] in order to validate the correctness of Tree Index evaluation. In this section, we use three attributes (Max, Min and Std) of the data set in order to plot the records so that we can see the records and their orientations. Such plots also help us to see clustering results and their appropriateness.

TABLE 2. Clustering results of HeMI++ and other techniques on (MIT-chb01_03) based on Tree Index

Clustering Techniques	Tree Index (lower the better)
HeMI++	**0.55**
HeMI	∞
GenClust	5.27
GAGR	19.89
AGCUK	18.19
K-means	27.41
K-means++	31.01

Fig.1 shows the clustering result of HeMI on the CHB-MIT Scalp EEG (chb01-03) data set. HeMI generates two clusters but one cluster contains only one record and all other records belong to the other cluster. Clearly, this does not appear to be a sensible clustering. From Table 2 we can see that according to Tree Index HeMI receives a poor evaluation result which is ∞. Therefore, the evaluation made by Tree Index matches with the manual evaluation (the visual analysis of the plotted records).

Fig.2 shows the clustering result of AGCUK where it generates two clusters: seizure and non-seizure. Mainly, the non-seizure records appear in Cluster 1 and a mixture of seizure and non-seizure records are found in Cluster 2. Cluster 1 has 2836 non-seizure records (dots in Fig.2) and 0 seizure records (plus signs in Fig.2), while Cluster 2 has 5389 non-seizure records (triangles in Fig.2) and 55 seizure records (circles in Fig.2). We can clearly see that while the clustering result is more sensible than the clustering result of HeMI (see Fig.1), it is still not a good clustering result.

Fig.3, Fig.4, Fig.5 and Fig.6 show the clustering results of GAGR, GenClust, K-means and K-means++ where GAGR, GenClust, K-means and K-means++ produce 56, 477, 28 and 13 clusters, respectively. Considering that the data set has only two types of records: Seizure and Non-seizure these clustering results with so many clusters also do not seem appropriate. This is also identified by Tree Index evaluation technique as shown in Table 2.

As we can see in Fig.7, HeMI++ produces a sensible clustering solution as it matches with the original orientation of records in the data set. It produces two clusters: Cluster 1 and Cluster 2. Cluster 1 contains 8219 non-seizure records and 38 seizure records, while Cluster 2 contains 6 non-seizure records and 17 seizure records. As a result, HeMI++ also achieves a good evaluation value based on *Tree Index* as shown in Table 2. This re-confirms that Tree Index produces better evaluation value for better clustering solutions.

E. Experimental Results on All Techniques on 20 Real Life Data Sets based on Tree Index

We now experimentally evaluate the performance of HeMI++ by comparing it with K-means, K-means++, GAGR, AGCUK, GenClust and HeMI on 20 other real-life data sets. For each data set, we run each technique 20 times.

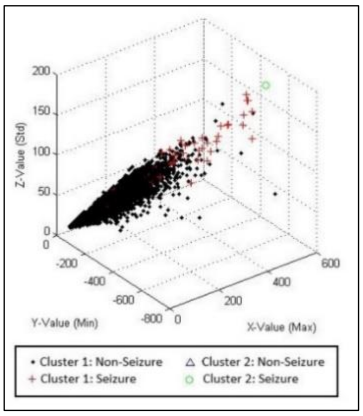

Fig.1. Clustering result of HeMI on the CHB-MIT Scalp EEG (chb01-03) data set

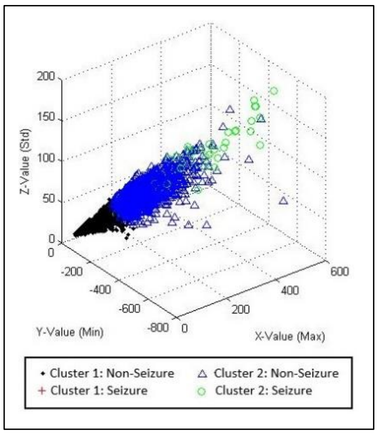

Fig. 2. Clustering result of AGCUK on the CHB-MIT Scalp EEG (chb01-03) data set

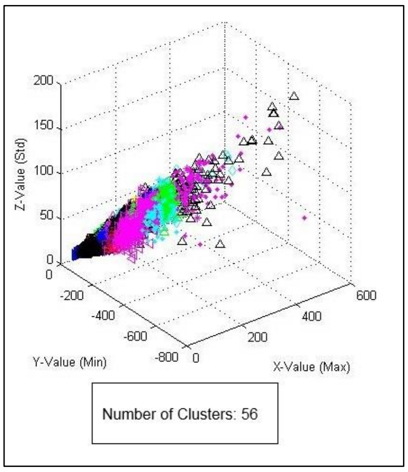

Fig. 3. Clustering result of GAGR on the CHB-MIT Scalp EEG (chb01-03) data set

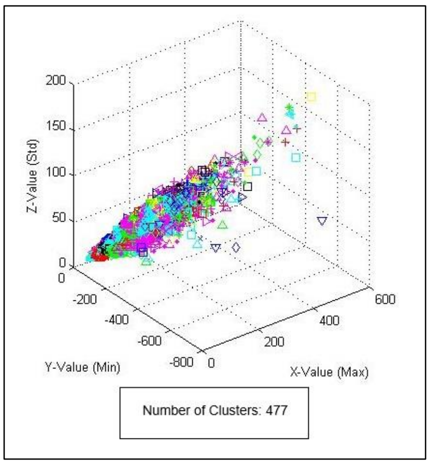

Fig. 4. Clustering result of GenClust on the CHB-MIT Scalp EEG (chb01-03) data set

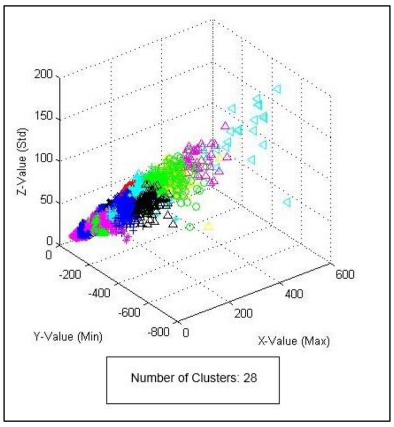

Fig. 5. Clustering result of K-means on the CHB-MIT Scalp EEG (chb01-03) data set

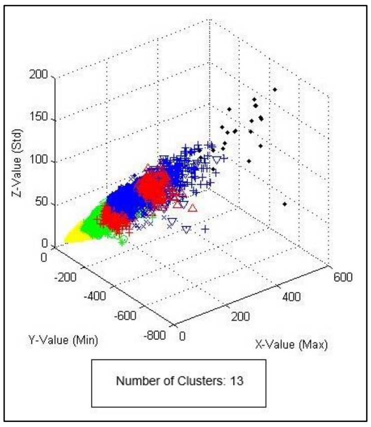

Fig. 6**.** Clustering result of K-means++ on the CHB-MIT Scalp EEG (chb01-03) data set

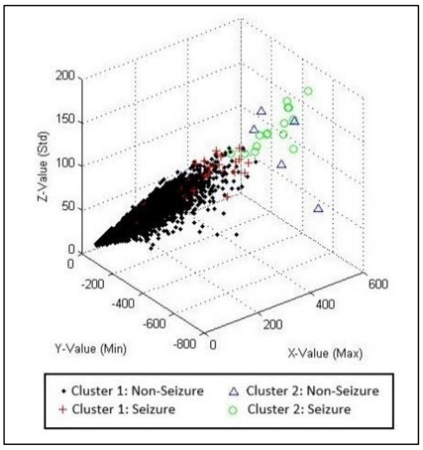

Fig.7. Clustering result of our proposed technique, HeMI++ on the CHB-MIT Scalp EEG (chb01-03) data set

Fig.8 shows the total score of all techniques on 15 numerical data sets based on Tree Index [9,10]. In this scoring system, the technique with the best clustering result gets 7 points and the technique with the worst result get 1 point - for each data set. Fig.8 shows the total scores of a technique over all data sets. The bar graph shows that HeMI++ achieves higher score than all other techniques.

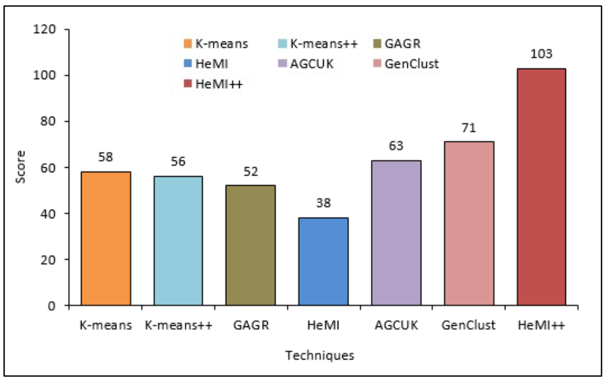

Fig. 8. Scores of the techniques on 15 numerical data sets based on Tree Index

TABLE 3. Clustering results of HeMI ++ and other techniques on 5 categorical data sets based on Tree Index

Data set	GenCslust	HeMI	HeMI++
HT	1.94	0.59	**0.46**
STH	1.75	∞	**1.38**
CA	∞	∞	**1.57**
CMC	**0.78**	∞	2.51
CKRK	60.39	2.23	**1.39**

We compare HeMI++ with GenClust and HeMI on 5 categorical data sets (data sets that have only categorical attributes or both the categorical and numerical attributes). For each data set we run each technique 20 times and present the average clustering result. Table 3 shows that HeMI++ achieves better results in 4 out of 5 data sets than GenClust. HeMI++ performs better than HeMI in 5 out of 5 categorical data set.

F. *Statistical Friedman Test*

We now carry out statistical Friedman test [23] in order to evaluate the superiority of the results (Silhouette Coefficient) obtained by HeMI++ over the results obtained by the existing techniques and our proposed technique HeMI. We compute the Silhouette Coefficient for each algorithm according to rank-ordering as it is used in the Friedman Test [23]. Among the 7 competing algorithms, the one providing the best Silhouette Coefficient is assigned a Rank of 1, the second best to the Silhouette Coefficient receives a Rank of 2 and so on (hence, the lower the average rank the better result). The result of ties is resolved by assigning the average of the sequential Silhouette Coefficient ranks they would have received. The Silhouette Coefficient Rank of each competing algorithm for each data set is presented within parentheses. The bottom row of Table 4 presents (within parentheses) the average of Silhouette Coefficient Rank (in short, Rank) of each competing algorithm from all data sets considered.

From Table 4, we can see that K-means provides the best Silhouette Coefficient for no data set (Rank: 4.26), K-means++ for no data sets (Rank: 4.40), GAGR for no data sets (Rank: 4.60), AGCUK for no data sets (Rank: 4.20), GenClust for 2 data sets (Rank: 3.50), HeMI for no data sets (Rank: 5.90) whereas HeMI++ achieves the best Silhouette

TABLE 4: Silhouette Coefficient rank of the techniques based on Friedman Test [23]

Data set	Tree Index (lower the better)						
	K-means	K-means++	GAGR	AGCUK	GenClust	HeMI	HeMI++
GI	2.11 (4)	1.87 (3)	3.47 (6)	0.92 (2)	2.47 (5)	∞ (7)	**0.31 (1)**
VC	4.94 (4)	4.11 (3)	5.37 (5)	∞ (6.5)	3.55 (2)	∞ (6.5)	**1.53 (1)**
EC	∞ (5)	∞ (5)	6.12 (2)	∞ (5)	∞ (5)	∞ (5)	**2.94 (1)**
LF	2.64 (3)	3.05 (4)	3.53 (5)	1.32 (2)	3.71 (6)	∞ (7)	**0.95 (1)**
LD	6.31 (6)	4.85 (5)	7.52 (7)	1.21 (3)	4.28 (4)	0.46 (2)	**0.24 (1)**
WBC	5.92 (5)	7.07 (6)	8.24 (7)	3.21 (3)	5.58 (4)	2.38 (2)	**1.28 (1)**
BT	5.81 (4)	5.73 (3)	0.47 (2)	∞ (6)	∞ (6)	∞ (6)	**0.27 (1)**
PID	13.40 (4)	14.19 (5)	6.20 (3)	∞ (6.5)	**3.72 (1)**	∞ (6.5)	6.19 (2)
SV	5.18 (6)	3.25 (4)	4.46 (5)	1.2 (2)	3.05 (3)	∞ (7)	**0.00 (1)**
BN	4.25 (4)	5.59 (6)	4.64 (5)	1.89 (2)	2.51 (3)	∞ (7)	**0.77 (1)**
YT	13.78 (4)	12.98 (3)	∞ (6)	∞ (6)	4.87 (2)	∞ (6)	**2.44 (1)**
IS	3.12 (4)	2.48 (3)	5.19 (5)	2.11 (2)	∞ (6.5)	∞ (6.5)	**1.53 (1)**
WQ	32.64 (4)	47.09 (5)	15.37 (3)	∞ (6.5)	**7.98 (1)**	∞ (6.5)	13.26 (2)
PBC	13.15 (4)	14.18 (5)	10.21 (3)	∞ (6.5)	4.77 (2)	∞ (6.5)	**0.44 (1)**
MGT	62.06 (3)	128.61 (6)	100.09 (5)	72.92 (4)	30.67 (2)	∞ (7)	**18.89 (1)**
Average rank	(4.26)	(4.40)	(4.60)	(4.20)	(3.50)	(5.90)	**(1.13)**

Coefficient in 13 out of 15 data sets (Rank 1.13). We now conduct a statistical significance test [23] in order to assess the superiority of HeMI++ over the existing techniques The Friedman [23] test is a non-parametric test used to compare multiple algorithms on multiple data sets. For our instance of the Friedman test, the null hypothesis is that all algorithms are equivalent. If the null hypothesis is rejected, we can proceed with a post-hoc test such as the Bonferroni-Dunn test [23]. The Friedman statistics is distributed according to X_F^2 with $(k-1)$ degrees of freedom when k (the number of competing algorithms) and N (the number of data sets) are big enough (as a rule of a thumb, $k > 5$ and $N > 10$) [62]. Iman and Davenport [64] demonstrated that Friedman's X_F^2 is undesirably conservative and derived a better statistic $F_F = \frac{(N-1)X_F^2}{(k-1)-X_F^2}$. With 7 algorithms and 15 data sets, the value of F_F is calculated to be11.64. With α =0.05 the critical value of F_F is calculated to be 2.13.

We can see that the critical value remains lower than the pair wise differences of ranks between the control clustering algorithm (HeMI++) and all other contending algorithms (K-means vs HeMI++: 3.13, K-means++ vs HeMI++: 3.26, GAGR vs HeMI++: 3.46, AGCUK vs HeMI++: 3.06, GenClust vs HeMI++: 2.36 and HeMI vs HeMI++: 4.76) indicating that HeMI++ performs better (in terms of Silhouette Coefficient) than all other algorithms on 15 real-life data sets.

IV. CONCLUSION

In this paper we propose a new clustering technique HeMI++ that first learns important properties of sensible clustering solutions and then applies the information in producing its clustering solutions. When we apply HeMI++ on a brain data set we find that the proposed clustering technique overcomes the existing problem and produces sensible clustering solutions.

We empirically compared our proposed clustering technique (HeMI++) with six existing techniques on 20 publicly available data sets in terms of our Tree Index evaluation technique. We find that HeMI++ achieves the best clustering solutions in 17 out of 20 data sets. Moreover, we graphically visualise the clustering results of HeMI++ on a brain data set and find the results to be more sensible than others

REFERENCES

[1] P.-N. Tan, M. Steinbach, V. Kumar, Introduction to Data Mining, first ed., Pearson Addison Wessley, 2005.

[2] M. A. Rahman, & M. Z. Islam, "A hybrid clustering technique combining a novel genetic algorithm with K-Means," Knowledge-Based Systems. 71 (2014) 345-365.

[3] Y. Liu, X. Wu, Y. Shen, "Automatic clustering using genetic algorithms," Applied Mathematics and Computation. 218 (2011) 267-1279.

[4] D. Chang, X. Zhang, C. Zheng, "A genetic algorithm with gene rearrangement for K-means clustering," Pattern Recognition. 42 (2009) 1210-1222.

[5] S. P. Lloyd, "Least squares quantization in PCM," IEEE Transactions on Information Theory. 28 (1982) 129-13.

[6] D. Arthur, and S. Vassilvitskii, "k-means++: The Advantages of Careful Seeding," SODA '07 Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, 2007, pp.1027-1035.

[7] A.H. Beg, and M.Z. Islam, "Genetic algorithm with healthy population and multiple streams sharing infrmaton for Clustering," Knowledge-Based Systems, 114 (2016) 61-78.

[8] A. H. Beg, and M. Z. Islam, "A Novel Genetic Algorithm-Based Clustering Technique and its Suitability for Knowledge Discovery from a Brain Dataset," In Proc. of the IEEE Congress on Evolutionary Computation (IEEE CEC 2016), Vancouver, Canada, July 24 - 29, 2016, pp. 948-956.

[9] A. H. Beg, M. Z. Islam and V.Castro, "Tree Index: A New Cluster Evaluation Technique," https://arxiv.org/abs/2003.10841[cs.LG].

[10] A. H. Beg, A Novel Genetic Algorithm based Clustering and Tree based validation in Producing and Evaluating Sensible Clusters, PhD thesis in Computer Science, Charles Sturt University, 2017.

[11] S. Montani, and G. Leonardi, "Retrieval and clustering for supporting business process adjustment and analysis," Information Systems. 40(2014) 128-141.

[12] G. Gan, "Application of data clustering and machine learning in variable annuity valuation. Insurance," Mathematics and Economics. 53 (2013) 795-801.

[13] M. Girvan, and M.E.J. Newman, "Community Structure in Social and Biological Networks," Proceedings of the National Academy of Sciences of the United States of America. 99 (2002) 7821-7826.

[14] G. Stockman, and L.G. Shapiro, Computer Vision. (1st ed.), Prentice-Hall, ISBN 0-13-030796-3, New Jersey, 2001,pp. 279-325.

[15] Wu, X., Kumar, V., Ross Quinlan, J. et al. Knowl Inf Syst (2008) 14: 1.

[16] W. M. B. W. Mohd, A.H. Beg, T. Herawan and K. F. Rabbi, "An Improved Parameter less Data Clustering Technique based on Maximum Distance of Data and Lioyd k-means Algorithm," Procedia Technology. Vol. 1, pp. 367-371, 2012.

[17] A. K. Jain, "Data clustering: 50 years beyond K-Means," Pattern Recognition Letters. 31 (2010) 651-666.

[18] C. G. E. Boender, A. H. G. Rinnooy Kan, G. T. Timmer, L. Stougie, "A stochastic method for global optimization," Mathematical Programming. 22 (1982) 125-140.

[19] S. Bandyopadhyay, U. Maulik, M. K. Pakhira, "Clustering using simulated annealing with probabilistic redistribution," International Journal of Pattern Recognition and Artificial Intelligence. 15 (2001) 269- 285.

[20] M. Z. Islam, V. Estivill-Castro, Rahman, M. A. and T. Bossomaier, "Combining K-Means and a Genetic Algorithm through a Novel Arrangement of Genetic Operators for High Quality Clustering," Expert Systems with Applications (ESWA), 91 (2018) 402-417.

[21] P. Peng, O. Addam, M. Elzohbi, S. T. Özyer, A. Elhajj, S. Gao, Y. Liu, T. Özyer, M. Kaya, M. Ridley, J. Rokne, R. Alhajj, "Reporting and analyzing alternative clustering solutions by employing multi-objective genetic algorithm and conducting experiments on cancer data," Knowledge-Based Systems. 56 (2014)108-122.

[22] UCI Machine Learning Repository. <http://archive.ics.uci.edu/ml/data sets.html/> (accessed 22.06.13).

[23] M. Friedman, "A Comparison of Alternative Tests of Significance for the Problem of m Rankings," *Source: The Annals of Mathematical Statistics*, 11(1940)86–92

[24] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.K. Peng, H. E. Stanley, PhysioBank, PhysioToolkit, PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. *Circulation* 101(23):e215-e220 [Circulation Electronic Pages; http://circ.ahajournals.org/cgi/content/full/101/23/e215]; 2000 (June 13).

[25] A. Shoeb. *Application of Machine Learning to Epileptic Seizure Onset Detection and Treatment*. PhD Thesis, Massachusetts Institute of Technology, September 2009.

[26] D. L. Davies,, and D. W. Bouldin, "A Cluster Separation Measure," IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-1, 2 (1979)224–227.