Consensus functions for cluster ensembles

Article *in* Applied Artificial Intelligence · July 2012

DOI: 10.1080/08839514.2012.687668

CITATIONS
4

READS
605

3 authors:

Gaith Manita

14 PUBLICATIONS **19** CITATIONS

SEE PROFILE

Riadh Khanchel

University of Carthage

5 PUBLICATIONS **35** CITATIONS

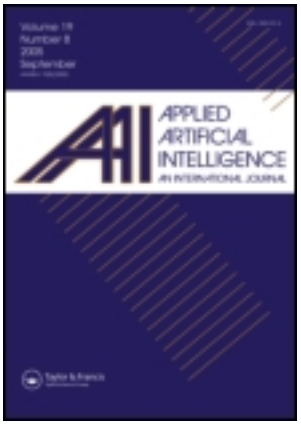
SEE PROFILE

Mohamed Limam

University Ibn Khaldoun - Tunis

146 PUBLICATIONS **1,375** CITATIONS

SEE PROFILE



Applied Artificial Intelligence: An International Journal

Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/uaai20

CONSENSUS FUNCTIONS FOR CLUSTER ENSEMBLES

Ghaith Manita [a], Riadh Khanchel [b] & Mohamed Limam [a]

[a] LARODEC, ISGT, University of Tunis, Tunis, Tunisia

[b] LARODEC, FSEGN, University of Carthage, Carthage, Tunisia

Version of record first published: 18 Jun 2012.

To cite this article: Ghaith Manita , Riadh Khanchel & Mohamed Limam (2012): CONSENSUS FUNCTIONS FOR CLUSTER ENSEMBLES, Applied Artificial Intelligence: An International Journal, 26:6, 598-614

To link to this article: http://dx.doi.org/10.1080/08839514.2012.687668

Applied Artificial Intelligence, 26:598–614, 2012

ISSN: 0883-9514 print/1087-6545 online
DOI: 10.1080/08839514.2012.687668

Taylor & Francis
Taylor & Francis Group

CONSENSUS FUNCTIONS FOR CLUSTER ENSEMBLES

Ghaith Manita[1], Riadh Khanchel[2], and Mohamed Limam[1]
[1]*LARODEC, ISGT, University of Tunis, Tunis, Tunisia*
[2]*LARODEC, FSEGN, University of Carthage, Carthage, Tunisia*

□ *The major task of clustering is to group an heterogeneous population into unknown groups based on a similarity measure. In order to enhance the robustness and the stability of unsupervised classification solutions, clustering ensembles are used; they are considered to be a powerful tool to deal with this issue. Individual clusterers consolidate the process of decision making through the concept of weighting. The aim is to determine an effective combination method that makes use of the benefits of each clusterer while avoiding its weaknesses. In this paper, we study the problem of combining multiple partitioning without accessing the original features. A genetic algorithm is proposed using three different fitness scores. Following three scenarios: Object Distributed Clustering, Feature Distributed Clustering, and Robust Centralized Clustering, the proposed consensus functions algorithm outperforms three existing ones: Cluster-based Similarity Partitioning Algorithm, HyperGraph Partitioning Algorithm and Meta-Clustering Algorithm.*

INTRODUCTION

Despite their widespread uses, clustering techniques face many challenges and difficulties such as the prediction of the right number of clusters, the scalability, the choice of the right similarity measure, and the identification of the noisy points. Robustness also represents one of the major challenges of clustering algorithms. In order to solve this issue, a combination of clustering results is considered a good alternative. This idea is inspired by the success of aggregating prediction results in the supervised case.

According to Topchy, Jain, and Punch (2004), cluster ensembles can go beyond what is typically achieved by a single-clustering algorithm, in several respects. First, cluster ensembles improve robustness by providing better performance across the domains and data sets. Second, they introduce

Address correspondence to Ghaith Manita, Higher School of Economic and Commercial Sciences, Research Laboratory: LARODEC, cité CNRPS, Menzah 6 Imm20 Appll, Ariana, Tunisia. E-mail: gaith.manita@gmail.com

novelty by having a combined solution unattainable by any single-clustering algorithm. Third, they ensure stability by providing clustering solutions with lower sensitivity to noise, outliers, and sampling variations. Finally, they guarantee parallelization and scalability by integrating solutions from multiple sources of data or attributes.

Given a data set of n instances $X=\{X_1, X_2, \ldots, X_n\}$, an ensemble combines r clusterings which is represented as $\Pi=\{\pi^1,\pi^2, \ldots, \pi^r\}$. Each clustering solution π^i is simply a partition of the data set X into K^i disjoint clusters of instances, represented as $\pi^i = \{C_1^i, C_2^i, \ldots, C_k^i\}$, where $\cup_k C_k^i = X$. These clusterings are combined using a consensus function Γ.

Cluster ensembles combine results produced by different clustering techniques through a consensus function. The information of different partition sets is merged to reach a more representative partition. Hence, the two essential components are the mechanism used to generate initial partitions and the consensus function used to combine these partitions into a final one.

This article is organized as follows: "Motivation and Related Work" reviews the current literature concerning cluster ensemble methods. We describe our work in "Proposed Algorithm." Then, in order to study its performance, we present the experimental results in the section titled "Evaluation of Proposed Consensus Functions." Finally, we provide conclusions and future researches.

MOTIVATION AND RELATED WORK

Several methods are proposed to combine results of individual clusterers. Hence, Ghaemi et al. (2009) group them into two categories: generative mechanisms and consensus functions.

The first category includes many approaches that generate multiple clusterings from a given data set by projecting various clustering algorithms, selecting different subsets of data points, projecting data into different subspaces, and using one algorithm with different built-in initialization and parameters. The second category aims to combine initial partitions into a final one using mathematical functions and algorithms named consensus functions. This category is formed by six families of algorithms: hypergraph methods, voting approaches, information theoretic methods, co-association-based methods, mixture models, and evolutionary algorithms. Our work focuses on the second category, the consensus functions.

In the hypergraph methods approach, objects to be clustered are represented as vertices of a graph. The hyperedges correspond to the set of objects that form the clusters. The aim of this approach is to find the partitioning of a hypergraph by minimizing a cost function. In the work of

Strehl and Ghosh (2002), the best solution consists of finding the minimum cut of the hypergraph into k components.

The voting approach aims to solve the correspondence problem between the labels of known and derived clusters. The basic idea is to switch cluster labels to meet the best agreement between the labels of two partitions. Then, the generated partitions have to be relabeled according to a predefined partition selected from either the ensemble or a new clustering of the data set known as the reference partition. Also, in the voting approach, the number of clusters in each given partition is the same as in the target partition. Fischer and Buhmann (2003) and Dudoit and Fridlyand (2003) have focused on this approach.

In the information theory approach, the objective function of a clustering ensemble is to construct a mutual information (MI) between the experimental probability distribution of labels in the consensus partition and labels in the ensemble. Consequently, the MI value for a candidate partition solution and the ensemble is calculated as the sum of pair-wise MIs between target and given partitions. Using the information theoretic principles, Topchy, Jain, and Punch (2003) have developed a new, different consensus function.

The coassociation-based functions approach uses a similarity matrix called coassociation matrix. These coassociation values determine the probability that each pair belongs to the same cluster. In order to obtain the final clustering, various hierarchical agglomerative algorithms are applied. To link objects whose coassociation values exceed a certain threshold, Fred (2001) uses voting k-means whereas Fred and Jain (2002) use single link (SL).

The finite mixture model approach manipulates probabilistic models for density estimation using a mixture distribution. It considers the labels as random variables derived from a probability distribution. Therefore, the objective of consensus clustering is described as a maximum likelihood estimation problem where the expectation maximization algorithm (EM) is used to solve it by computing the parameters of the mixture model distribution. Topchy, Jain, and Punch (2004) use the EM consensus function to handle missing data, equivalently missing cluster labels. Then, the Hungarian method for minimal weight bipartite matching problem is used to generate the optimal correspondence.

In the evolutionary algorithms approach, the searching capability of genetic algorithms is used to derive a consensus clustering from clustering ensembles. For that, Luo, Jing, and Xie (2006) propose a new consensus function based on genetic algorithms to find an almost median partition. The clustering metric used by this function is the sum of the entropy-based dissimilarity of the consensus clustering from the component clusterings in the ensemble.

As noticed, consensus functions represent a difficult combinatorial optimization problem. In this article, we focus on genetic algorithms to solve this problem. It is worth noticing that the data set size depends on the number of distributed subsets *K*.

PROPOSED ALGORITHM

Genetic algorithms (GA) are optimization algorithms based on techniques derived from genetics and natural evolution: crosses, mutations, selection, etc. A GA searches for extremes of a function defined on a data space.

Based on the same concepts as GAs, we define the following elements in order to propose an efficient cluster ensemble algorithm that deals with both small and large data sets *X*.

- String representation: For a given data set composed with n instances, $X = \{X_1, X_2, \ldots, X_n\}$, each chromosome P is a string sequence of integer numbers representing the class label of the n objects where the i-th position (or gene) represents the class label of X_i.
- Population initialization: The chromosomal population $P(\mathrm{t}) = \{P_1, P_2, \ldots, P_\mathrm{t}\}$ consists of t chromosomes. Each chromosome can be regarded as an integer sequence of length $|\mathrm{X}|$ representing a possible clustering of the data. The chromosomes are representations of initial partitions. This initialization has a major effect on results. The first initialization mechanism proposed by Luo, Jing, and Xie (2006) is limited by clusterings generated on distributed data sets. Then, if there are few distributed data sets, results will be poor. Hence, our first interest is to define a new mechanism.

  The proposed mechanism consists of generating new clusterings from existing ones in order to create a dense population able to provide good results. New clusterings are the results of a crossover operation. This mechanism is applied when the number of distributed data sets is fewer than 20. This choice is made based on the work of Luo, Jing, and Xie (2006). In their experimental results, they proved that GAs generate good results only if the number of distributed data sets is greater or equal to 20.
- Fitness computation: In this part, we define the fitness measure associated with each chromosome P_k, as the following:

$$fitness\_score\ (P_k) = \sum_{i=1, k \neq i}^{n} CF(P_i, P_k), \tag{1}$$

where *CF* is a measure to compute the mutual information shared between two clusterings.

In a cluster ensemble, the objective function is an important component that computes the shared information between clusterings. Hence, finding the best objective function is the most difficult task in generating cluster ensembles. In GAs, the objective function is represented by the fitness score. So, defining the fitness score to use is an important step in developing a good consensus function. In the literature, many measures are implemented to compute shared information between clusterings. However, none can be qualified as the best. For that, we choose to use three different fitness scores: normalized mutual information (NMI), adjusted mutual information (AMI) and generalized conditional entropy (GCE).

Normalized mutual information quantifies the statistical information shared between two distributions. NMI is given in the following:

$$NMI(\pi^1, \pi^2) = \frac{I(\pi^1, \pi^2)}{\sqrt{H(\pi^1)H(\pi^2)}}, \tag{2}$$

where I(π^1, π^2) is the mutual information between π^1 and π^2, and H (π^i) is the entropy of π^i.

Adjusted mutual information is a derivative of NMI used to correct the negative effect of the random generation of partitions. AMI is given by:

$$AMI(\pi^1, \pi^2) = \frac{I(\pi^1, \pi^2) - E\{I(M)|a, b\}}{\sqrt{H(\pi^1)H(\pi^2) - E\{I(M)|a, b\}}}, \tag{3}$$

where E is the expected mutual information, $I(M)$ denote the mutual information between any two clusterings associated with the contingency table M and a,b are the marginals of the contingency table of π^1 and π^2. The NMI and AMI functions reach their maximum values when the two clusterings are similar.

Generalized conditional entropy is a generalization of the classical notion of entropy and conditional entropy. It is given by:

$$GCE = \sum_{i=1}^{r} d^f(\pi^*, \pi^i) \tag{4}$$

$$= \sum_{i=1}^{r} H^f(\pi^*|\pi^i) + H^f(\pi^i|\pi^*), \tag{5}$$

where d^f is the dissimilarity measure between two clusterings, H^f ($\pi^1|\pi^2$) is the conditional f-entropy and f is the entropy generator. The GCE function reaches its minimum value when the two clusterings are similar.

By combining these fitness scores with a GA, we obtain three new consensus functions: GA-NMI, GA-AMI and GA-GCE.

- Crossover: Crossover is an operation used to generate a new string (i.e., child) from two parent strings. In this study, the maximal preservative crossover (MPX) operator is used as a crossover operator. This crossing operator has been proposed by Muhlenbein (1993) for the traveling salesman problem. The idea of this operator is to insert a part chromosome of one parent in the chromosome of the other one such that the resulting intersection is the closest to his parents. This is a meet at two points. Both childs are obtained in a symmetrical way. We illustrate crossover in Figure 1.
- Mutation: In order to maintain genetic diversity through generations, mutation is used. It is analogous to biological mutation. In this operation we choose to randomly change a sequence of positions in the chromosomes. The new value of chromosome position is incremented by one.

Algorithm 1 **Proposed algorithm**

Require: k:number of distributed sub data sets.
Ensure: final clustering π^*
  Population initialization
  Compute fitness score
  while maximum fitness score <1 and !stop condition **do**
    Childl, Child2 = crossover(parent1,parent2)
    Child3 = mutation(parent)
    if number of chromosomes < maximum population size **then**
      Add Child1, Child2 and Child3 to the population
    else if Child chromosome fitness score > the worst chromosome fitness score **then**
      Swap Childl, Child2 and Child3 with the worst chromosomes
    end if
    Compute fitness score
  end while
  return best chromosome

Given K distributed subsets, we apply to each of them an individual clusterer in order to obtain K different clusterings. In fact, the same single-clustering algorithm is applied to all data sets. However, several clustering algorithms can be applied. In the first step, we initialize the population using the mechanism mentioned previously, then we compute the fitness score of the generated population. The next step consists of an iterative procedure in order to obtain the final solution. To begin this procedure, we should verify that the fitness score of the generated

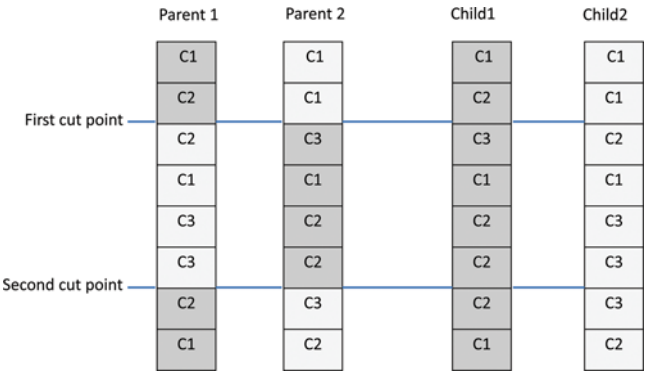


FIGURE 1 The maximal preservative crossover operation. (Color figure available online.)

population is less than one. After that, two clusterings are created by the mean of the crossover operator. Then, a mutation operator is applied to form a third clustering. Two cases are revealed. In the first, if the number of the actual population is inferior to the maximum size of the population, then the three generated clusterings are added to the population. In the second, if number of the actual population is equal to the maximum size of the population, then the generated clusterings are swapped with the worst clusterings. This procedure is stopped when n successive iterations do not provide any improvement. The pseudocode of this consensus function is detailed in Algorithm (1).

EVALUATION OF PROPOSED CONSENSUS FUNCTIONS

Cluster ensemble can be used in several scenarios. We evaluate the effectiveness of the application of our proposed consensus function in three different scenarios which are object distributed clustering (ODC), feature distributed clustering (FDC) and robust centralized clustering (RCC).

In ODC, objects are stored in distributed data sets. They are characterized by the same features, and each scattered data set is constructed of a specific collection of objects, which they may overlap. In FDC, each clusterer has a partial view of the data because it has access to only a small subset of features (subspace). Each clusterer has access to all objects. The clusterers find groups in their views/subspaces using the same clustering technique. In the combination stage, individual results are integrated using a supraconsensus function to recover the full structure of the data. In RCC, each clusterer has access to all features and to all objects. However, each clusterer might take a different approach. In fact, approaches should be

very diverse for best results. Therefore, different distance/similarity measures or techniques can be used. The ensemble clusterings are then integrated using the consensus function without having access to the original features.

In order to study the performance of our proposed consensus functions, we have carried out an experimental study on five data sets, namely: Iris, Pima, Heart, Transfusion and Vehicle, and the UCI Repository (Blake and Merz 1998). These data sets have different numbers of instances and features.

In this section, we compare our consensus functions with three consensus functions proposed by Strehl and Ghosh(2002): cluster-based similarity partitioning algorithm (CSPA), hypergraph-partitioning algorithm (HGPA) and meta-clustering algorithm (MCLA). The CSPA induces a graph from a coassociation matrix and clusters it using the (METIS) algorithm (Karypis and Kumar 1998). The HGPA represents each cluster by a hyperedge in a graph where nodes correspond to a given set of objects. Good hypergraph partitions are found using minimal cut algorithms such as (HMETIS), developed by Karypis et al. (1997), coupled with the proper objective functions, which also control partition's size. The MCLA uses hyperedge collapsing operations to determine soft cluster membership values for each object. In order to select the best solution, they define a mutual-information-based objective function and build a supraconsensus function as well.

Because these proposed functions are based on GAs, it is necessary to compare them with the solution proposed by Luo, Jing and Xie (2006), which we named OLD GA. Throughout the study, different numbers of distributed data sets have been used $K = \{5, 10, 20, 50\}$, with the aim to evaluate these consensus functions on small and large data sets as well.

First experimental results of proposed consensus functions show that the computational time is too high when using 100 as the maximum population size. The behavior of its fitness score on Iris data is illustrated in Figure 2. For the first 50 iterations, the fitness improvement increases from 0.88 to 0.99 and thus achieves an improvement rate of 12.5% over 23 seconds. From the 51th iteration to the 90th, we notice a slight improvement over 103 seconds. For the last 5 iterations, the curve reaches its maximum of 1. At this stage the algorithm becomes expensive in terms of time, when we have 80 seconds for an improvement rate of 0.001%. Based on this, we decide to set the maximum population size to 50.

The number of tolerated iterations with no improvement before stopping our algorithm is fixed at 15. We can say that this choice is not totally random because our experimental results show that this number should belong to the interval [10–20]. Our parameters settings are given in Table 1.

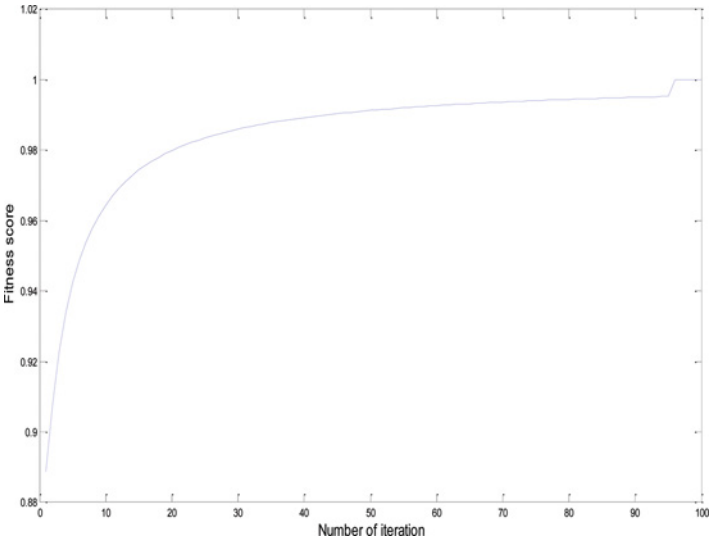


FIGURE 2 Variation of fitness score when applying GA to Iris data set. (Color figure available online.)

Object Distributed Clustering

In these experimental results, we generate artificial distributed subsets by random sampling from an initial data set. In order to produce overlapped subsets, we use sampling with replacement. Evaluation is based on error rate averaged over all subsets. Table 2 illustrates the average error rates of the seven consensus functions applied to five data sets using, respectively, five, ten, twenty, and fifty distributed subsets over 50 runs. For a fixed number of distributed subsets, the lowest error rate for each data set is marked in bold.

Results show that when the number of distributed data sets is equal to five, CSPA and HGPA outperform almost all other consensus functions.

Based on the average error rates of the GA-based algorithms, we conclude on the one hand that all of these consensus functions have a competitive behavior because they offer close average error rates.

On the other hand, comparing the average error rates of the proposed consensus functions and those of the OLD GA, we notice that, in small

TABLE 1 Parameter Settings

Parameters	Value
Max population size	50
Number of succsessive iterations without amelioration	15
Crossover rate	0.2
Mutation rate	0.2

TABLE 2 Average Error Rate of 7 Consensus Functions Using K = 5, 10, 20, 50 Distributed Subsets

Number of distributed subsets	Data set	CSPA	HGPA	MCLA	OLD GA	GA-NMI	GA-AMI	GA-GCE
5	Iris	**0.4377**	0.5463	0.5370	0.6423	0.6220	0.6310	0.6560
	Pima	**0.4590**	0.4882	0.5739	0.5016	0.4987	0.5051	0.4856
	Heart	0.4965	**0.4719**	0.5154	0.5056	0.4909	0.4924	0.4967
	Transfusion	**0.4815**	0.4913	0.5909	0.5738	0.5597	0.5517	0.5419
	Vehicle	**0.7294**	0.7471	0.7676	0.7618	0.7588	0.7529	0.7471
10	Iris	**0.4597**	0.5597	0.5223	0.6943	0.6570	0.6570	0.6760
	Pima	**0.4798**	0.4914	0.4837	0.5259	0.4868	0.5007	0.5057
	Heart	**0.4567**	0.4648	0.4837	0.5161	0.5059	0.5053	0.5067
	Transfusion	0.5252	0.4963	0.5685	0.5030	**0.4829**	0.5057	0.5282
	Vehicle	0.7353	**0.7265**	0.7706	0.7294	0.7471	0.7471	0.7294
20	Iris	**0.4060**	0.5820	0.5657	0.6457	0.6523	0.6303	0.6483
	Pima	**0.4824**	0.4942	0.6173	0.5078	0.5041	0.5097	0.5024
	Heart	**0.4659**	0.4719	0.5289	0.5098	0.5019	0.5052	0.4885
	Transfusion	0.5023	0.4909	0.6899	0.4822	0.4953	**0.4752**	0.5419
	Vehicle	0.7971	0.7647	0.8	**0.7441**	**0.7441**	**0.7441**	**0.7441**
50	Iris	0.4867	**0.4773**	0.5650	0.6433	0.6433	0.6433	0.6433
	Pima	0.4947	**0.4903**	0.5999	0.5148	0.5022	0.5022	0.5022
	Heart	**0.4663**	0.4869	0.5394	0.4893	0.5126	0.5126	0.5126
	Transfusion	**0.4775**	0.4822	0.6282	0.4829	0.5047	0.5047	0.5047
	Vehicle	0.7529	0.7647	0.75	**0.7471**	**0.7471**	**0.7471**	**0.7471**

data sets (k = 5 or k = 10), our three consensus functions provide the lowest error rates. Consequently, GA-NMI, GA-AMI, and GA-GCE outperform the OLD GA in small data sets, which meets one of our motivations.

Varying the number of distributed subsets, we observe a slight increase in many consensus function rates for each data set. The average error rates of CSPA applied on PIMA for, respectively, 5, 10, 20, and 50 data sets are: 0.4590, 0.4798, 0.4824, and 0.4947. This decreasing performance of the consensus functions is a result of the complexity growth, which is proportional to the number of distributed data sets.

First, we notice that by varying the number of distributed data sets, our three consensus functions provide stable performance. Second, we show that the proposed consensus functions improve the performance of the algorithm proposed by Luo, Jing, and Xie (2006), especially with a small number of data sets.

These results are based on an average error rate that is sensitive to outliers and does not take into account the logical identical clusterings problem. We can explain this through this example: let $\pi^1 = (1; 1; 1; 2; 2; 3; 3)$ and $\pi^2 = (2; 2; 2; 3; 3; 1; 1)$, at a first sight, π^1 and π^2 seem totally different, but in fact the difference resides only in their labeling. This example presents the problem of logical identical clusterings.

In order to reassess the performance of the consensus functions cited above, different evaluation criteria such as F-measure and generalized

conditional entropy (GCE) will be used. F-measure is an evaluation criterion used to assess the results of clusterings techniques. This measure is based on recall and precision in order to penalize errors type I and type II. It reaches its best value at 1 and worst value at 0. Therefore, the best consensus function must provide the greatest F-measure rate. The GCE is used as an evaluation measure because it is able to quantify the shared information between two clusterings. The lower the GCE rate is, the better is the performance of the consensus function. The mathematical formulation of this criterion is given in the "Proposed Algorithm" section.

Tables 3 and 4 present the average F-measure and GCE rates of 50 runs for the studied consensus functions applied to five data sets using 5, 10, 20, and 50 distributed subsets. Based on both measures for small subsets, the best performances are provided by the three proposed algorithms for Iris and vehicle data sets and by MCLA for the rest of data sets. Consequently, the new consensus functions are better than the solution proposed by Luo, Jing and Xie (2006) when applied on small subsets.

In the case of large data sets, we notice that the performances of the proposed algorithms increase because they outperform all the others for all data sets, because in large subsets the population initialization becomes more robust as a result of the large number of subsets.

TABLE 3 Average Accuracy Rate Using F-Measure of 7 Consensus Functions Using K = 5, 10, 20, 50 Distributed Subsets for ODC Application

Number of distributed subsets	Data set	CSPA	HGPA	MCLA	OLD GA	GA-NMI	GA-AMI	GA-GCE
5	Iris	0.7199	0.6903	0.7755	0.7814	0.7845	**0.7851**	**0.7851**
	Pima	0.5839	0.5670	**0.6865**	0.6001	0.6065	0.6119	0.6162
	Heart	0.6010	0.5631	**0.6199**	0.5215	0.5693	0.5680	0.5698
	Transfusion	0.6054	0.6048	**0.7376**	0.6484	0.6464	0.6510	0.6727
	Vehicle	0.5442	0.5585	0.5740	0.5865	0.5864	0.5868	**0.5872**
10	Iris	0.7179	0.7075	0.7803	0.7882	**0.7887**	0.7884	0.7885
	Pima	0.5796	0.5667	**0.6849**	0.5968	0.5957	0.5956	0.5954
	Heart	0.5989	0.5763	**0.6220**	0.5196	0.5684	0.5708	0.5690
	Transfusion	0.6053	0.6048	**0.7151**	0.6295	0.6615	0.6502	0.6573
	Vehicle	0.5400	0.5432	0.5628	**0.5791**	0.5752	0.5737	**0.5791**
20	Iris	0.7339	0.7094	0.7824	**0.7882**	0.7878	0.7881	0.7878
	Pima	0.5806	0.5672	**0.6814**	0.6248	0.6029	0.6037	0.6057
	Heart	0.5989	0.5819	**0.6172**	0.5192	0.5696	0.5740	0.5727
	Transfusion	0.6052	0.6047	**0.7317**	0.6454	0.6340	0.6303	0.6241
	Vehicle	0.5356	0.5156	0.5812	**0.5864**	**0.5864**	**0.5864**	**0.5864**
50	Iris	0.7369	0.7254	0.7871	**0.7883**	**0.7883**	**0.7883**	**0.7883**
	Pima	0.5835	0.5670	0.6832	0.6879	**0.7295**	**0.7295**	**0.7295**
	Heart	0.5922	0.5717	0.6154	0.6174	**0.6183**	**0.6183**	**0.6183**
	Transfusion	0.6063	0.6047	0.7321	0.7459	**0.8174**	**0.8174**	**0.8174**
	Vehicle	0.5412	0.5536	0.5672	0.5883	**0.5885**	**0.5885**	**0.5885**

TABLE 4 Average Accuracy Rate Using GCE Measure of 7 Consensus Functions Using K = 5, 10, 20, 50 Distributed Subsets for ODC Application

Number of distributed subsets	Data set	CSPA	HGPA	MCLA	OLD GA	GA-NMI	GA-AMI	GA-GCE
5	Iris	0.4481	0.4919	0.3184	0.2995	**0.2961**	0.2987	0.2967
	Pima	0.5672	0.5787	**0.4947**	0.5897	0.5483	0.5488	0.5458
	Heart	0.5762	0.5884	**0.5599**	0.5672	0.5754	0.575	0.5757
	Transfusion	0.5298	0.5347	**0.4272**	0.4997	0.4952	0.4928	0.4714
	Vehicle	0.7638	0.7507	0.7003	0.6246	**0.6149**	0.643	0.6176
10	Iris	0.4459	0.4682	0.3154	0.3035	0.2996	**0.2977**	0.2983
	Pima	0.5693	0.5791	**0.4978**	0.5715	0.5597	0.559	0.5597
	Heart	0.5748	0.5829	0.5573	**0.5558**	0.5753	0.5744	0.5725
	Transfusion	0.5267	0.5365	**0.4416**	0.4777	0,4755	0.4829	0.4813
	Vehicle	0.776	0.7600	0.6525	**0.6257**	0.6321	0.6265	**0.6257**
20	Iris	0.4304	0.466	0.3076	0.2961	**0.2916**	0.2949	**0.2916**
	Pima	0.5690	0.5786	**0.5006**	0.5652	0.5544	0.5336	0.5537
	Heart	0.5770	0.5824	0.5592	**0.5555**	0.5727	0.5751	0.5747
	Transfusion	0.5306	0.5353	**0.4260**	0.4524	0.4987	0.5082	0.5035
	Vehicle	0.7810	0.7931	0.6982	**0.6299**	**0.6299**	**0.6299**	**0.6299**
50	Iris	0.4295	0.4374	0.3028	**0.2964**	**0.2964**	**0.2964**	**0.2964**
	Pima	0.5676	0.5786	0.498	0.4924	**0.413**	**0.413**	**0.413**
	Heart	0.5792	0.587	0.5622	0.5566	**0.5081**	**0.5081**	**0.5081**
	Transfusion	0.5288	0.5317	0.4288	0.4147	**0.3173**	**0.3173**	**0.3173**
	Vehicle	0.7513	0.7272	0.6932	**0.6031**	**0.6031**	**0.6031**	**0.6031**

To summarize the performance of the studied algorithms in the ODC application, we notice that when using F-measure and GCE for GA-based consensus functions, the variance of the accuracy is small compared with that of the error rate because the former can handle the problem of illogical identical clusterings. We also notice that GCE makes all consensus functions less sensible to the variation of *K*.

Feature Distributed Clustering

In FDC, we show how cluster ensembles can be used to boost quality of results by combining a set of clusterings obtained from partial views of the data. Such scenarios result in distributed databases and federated systems that cannot be pooled together because of proprietary data aspects, privacy concerns, performance issues, and so forth.

Tables 5 and 6 illustrate accuracy rates using F-measure and GCE for CSPA, HGPA, MCLA, OLD GA, and our proposed consensus functions applied to five data sets using, respectively, 5, 10, 20 and 50 distributed subsets.

When K = 5, according the F-measure results presented in Table 5, the greatest accuracy rates are given by CSPA. However, based on GCE measure,

TABLE 5 Average Accuracy Rate Using F-measure of 7 Consensus Functions Using K = 5, 10, 20 and 50 Distributed Subsets for FDC Application

Number of distributed subsets	Data set	CSPA	HGPA	MCLA	OLD GA	GA-NMI	GA-AMI	GA-GCE
5	Iris	**0.6020**	0.5417	0.5725	0.5099	0.5142	0.5000	0.5195
	Pima	0.6045	0.5665	0.5494	0.7389	0.7574	0.7297	**0.7817**
	Heart	0.5786	0.5765	**0.6837**	0.6143	0.681	0.6459	0.6556
	Transfusion	0.6062	0.6048	0.723	**0.8095**	0.8056	0.8053	0.8211
	Vehicle	**0.5419**	0.4909	0.4378	0.4545	0.4545	0.4545	0.4545
10	Iris	**0.6425**	0.6169	0.5684	0.5000	0.5000	0.5000	0.5000
	Pima	0.6335	0.5697	0.6968	0.7205	0.7294	**0.7323**	0.7254
	Heart	0.624	0.5867	0.6328	0.6343	0.6738	0.6703	**0.6782**
	Transfusion	0.6173	0.6047	0.6102	0.7671	**0.7916**	0.7712	0.7780
	Vehicle	0.5156	0.5346	**0.5378**	**0.5378**	**0.5378**	**0.5378**	**0.5378**
20	Iris	**0.6425**	0.6169	0.5684	0.5000	0.5000	0.5000	0.5000
	Pima	0.6335	0.5697	0.6968	0.7205	0.7294	**0.7323**	0.7254
	Heart	0.6240	0.5867	0.6328	0.6343	0.6738	0.6703	**0.6782**
	Transfusion	0.6173	0.6047	0.6102	0.7671	**0.7916**	0.7712	0.7780
	Vehicle	0.5156	**0.5346**	0.4378	0.4378	0.4378	0.4378	0.4378
50	Iris	**0.6479**	0.5811	0.541	0.5792	0.5792	0.5792	0.5792
	Pima	0.6533	0.5781	0.7006	0.7105	**0.7144**	**0.7144**	**0.7144**
	Heart	0.6369	0.5672	0.6143	0.6143	**0.6505**	**0.6505**	**0.6505**
	Transfusion	0.6111	0.6048	0.6055	0.7690	**0.7702**	**0.7702**	**0.7702**
	Vehicle	**0.5451**	0.5265	0.4931	0.4931	0.4931	0.4931	0.4931

TABLE 6 Average Accuracy Rate Using GCE of 7 Consensus Functions Using K = 5, 10, 20 and 50 Distributed Subsets for FDC Application

Number of distributed subsets	Data set	CSPA	HGPA	MCLA	OLD GA	GA-NMI	GA-AMI	GA-GCE
5	Iris	0.703	0.7589	0.5556	0.4966	0.4946	0.4771	**0.4743**
	Pima	0.5572	0.5796	0.3756	0.3631	0.3471	0.3696	**0.3156**
	Heart	0.5839	0.5839	**0.3535**	0.3983	0.3670	0.4085	0.3978
	Transfusion	0.5057	0.5371	0.3761	0.3265	0.3125	0.2994	**0.2984**
	Vehicle	0.7778	0.871	0.7941	**0.5999**	**0.5999**	**0.5999**	**0.5999**
10	Iris	0.741	0.7718	0.5799	**0.4771**	**0.4771**	**0.4771**	**0.4771**
	Pima	0.5526	0.5790	0.4245	0.3802	0.361	0.3584	**0.3372**
	Heart	0.5847	0.5948	**0.2983**	0.3983	0.3888	0.4014	0.384
	Transfusion	0.4942	0.5360	0.4180	0.3179	**0.3045**	0.3146	0.3447
	Vehicle	0.7981	0.8017	0.7193	**0.5999**	**0.5999**	**0.5999**	**0.5999**
20	Iris	0.6512	0.6694	0.5381	**0.4771**	**0.4771**	**0.4771**	**0.4771**
	Pima	0.5389	0.5773	0.5020	0.3820	**0.3571**	0.3583	0.3869
	Heart	0.5661	0.5820	0.4903	**0.3983**	0.3964	0.3988	0.4146
	Transfusion	0.4863	0.5366	0.4876	**0.3343**	0.3355	0.3446	0.3406
	Vehicle	0.8486	0.8153	0.7180	**0.5999**	**0.5999**	**0.5999**	**0.5999**
50	Iris	0.6387	0.7128	0.6791	**0.4771**	**0.4771**	**0.4771**	**0.4771**
	Pima	0.5287	0.5725	0.4683	0.4802	**0.4090**	**0.4090**	**0.4090**
	Heart	0.5572	0.5889	0.4983	0.4983	**0.4897**	**0.4897**	**0.4897**
	Transfusion	0.4960	0.5366	0.4961	**0.3149**	0.3292	0.3292	0.3292
	Vehicle	0.7353	0.7916	0.7599	**0.5999**	**0.5999**	**0.5999**	**0.5999**

GA using GCE consensus function is the most efficient. In the case of a large data set where K = 50 and based on F-measure and GCE criteria, our proposed consensus functions significantly outperform all other functions with, in some cases, an improvement greater than 0.2. However, in some cases, OLD GA competes with them. Moreover, it is clear that the GA-based algorithms adopt a stable behavior because they all converge to the same values.

After comparing the performances of the consensus functions through F-measure and GCE rates, we notice that GCE criterion favors GA-based algorithms. However, based on both evaluation criteria, the best performances of GA-based algorithms are provided by the Transfusion data set. Although the Iris data set has the same structure and attribute number as Transfusion, performance of Iris is not as good as Transfusion. This is explained by the small number of attributes (N), which is four. Because GAs use a fully random selection of N-1 attributes, the most informative attribute may be excluded, such as the case of the Iris data set.

Robust Centralized Clustering

In RCC, a consensus function may introduce redundancy and robustness when, instead ofchoos-ing or fine-tuning a single clusterer, an ensemble of clusterers is employed and their results are combined. This is particularly useful when clustering has to be performed in a closed loop without human interaction. The goal of RCC is to perform well for a wide variety of data distributions with a fixed ensemble of clusterers.

Tables 7 and 8 summarize the F-measure and GCE rates of seven consensus functions using different numbers of distributed data sets $K = \{5, 10, 20 \text{ and } 50\}$ in RCC application. Contrariwise to ODC and FDC, these results reveal that the three proposed algorithms provide good performance in small data sets especially for GA using AMI. However, when K is set to 50, evaluation criteria rates of GA-based algorithm improve and converge to the same value. The performance of OLD GA competes with them only with large data sets. From the comparison of the performances of consensus functions through F-measure and GCE rates, we conclude that the variances presented in GCE rates are more important than those of F-measure. Moreover, as for FDC application, GCE criterion favors GA-based algorithms.

To summarize the empirical results, we conclude that there is not an algorithm that suits for all data sets. The performance of each consensus function depends on the data set and on the number of distributed subsets. Through general assessment, in the ODC, FDC, and RCC applications, GA-based consensus functions provide good results. In order to evaluate

TABLE 7 Average Accuracy Rate Using F-measure of 7 Consensus Functions Using K = 5, 10, 20 and 50 Distributed Subsets for RCC Application

Number of distributed subsets	Data set	CSPA	HGPA	MCLA	OLD GA	GA-NMI	GA-AMI	GA-GCE
5	Iris	0.6612	0.6612	**0.6893**	0.5050	0.5293	0.5557	0.5029
	Pima	0.5833	0.5666	0.6294	0.7120	0.6715	**0.7786**	0.6607
	Heart	0.6053	0.5913	0.6280	0.6011	0.5989	**0.6287**	0.6037
	Transfusion	0.8051	0.6048	0.7501	0.8066	**0.8101**	0.8025	0.8099
	Vehicle	0.5471	**0.5601**	0.5196	0.508	0.4833	0.4730	0.5045
10	Iris	**0.7398**	0.6784	0.6376	0.5283	0.6140	0.5833	0.5862
	Pima	0.5845	0.5666	0.6380	0.7407	0.6782	**0.7578**	0.7407
	Heart	0.6100	0.5755	**0.6376**	0.6086	0.6155	0.6312	0.6293
	Transfusion	0.6050	0.6048	0.7556	0.8051	0.7977	**0.8177**	0.8162
	Vehicle	0.5066	**0.5193**	0.5062	0.5011	0.4974	0.5167	0.4980
20	Iris	**0.7373**	0.7313	0.7429	0.5178	0.5000	0.5297	0.5292
	Pima	0.5823	0.5665	0.6734	**0.7736**	0.7029	0.7423	0.7597
	Heart	0.6072	0.5914	**0.6361**	0.6061	0.6168	0.6015	0.6170
	Transfusion	0.6054	0.6048	0.7326	**0.8069**	0.7925	0.7969	0.7855
	Vehicle	**0.5809**	0.5670	0.5493	0.4966	0.4809	0.4769	0.5101
50	Iris	0.718	**0.7349**	0.6584	0.5292	0.5292	0.5292	0.5292
	Pima	0.5763	0.5743	0.6564	**0.7760**	0.7332	0.7332	0.7332
	Heart	0.5914	0.5578	0.6146	0.608	**0.6376**	**0.6376**	**0.6376**
	Transfusion	0.6059	0.6047	0.7437	0.8069	**0.8388**	**0.8388**	**0.8388**
	Vehicle	0.5245	**0.5705**	0.5378	0.4849	0.4849	0.4849	0.4849

TABLE 8 Average Accuracy Rate Using GCE of 7 Consensus Functions Using K = 5, 10, 20 and 50 Distributed Subsets for RCC Application

Number of distributed subsets	Data set	CSPA	HGPA	MCLA	OLD GA	GA-NMI	GA-AMI	GA-GCE
5	Iris	0.5321	0.5470	0.4088	0.4602	0.4564	**0.4249**	0.4879
	Pima	0.5656	0.5787	0.5297	0.4715	0.4519	**0.4050**	0.4523
	Heart	0.5763	0.5752	0.5402	0.4852	0.5013	**0.4091**	0.4941
	Transfusion	0.5326	0.5366	0.398	0.3109	**0.3097**	0.3229	0.3065
	Vehicle	0.7614	0.7243	0.7239	0.6142	0.6105	**0.5985**	0.6161
10	Iris	0.4159	0.5189	0.4413	0.4161	**0.3924**	**0.3924**	0.4200
	Pima	0.5672	0.5771	0.5288	0.3891	0.4357	**0.3581**	0.3676
	Heart	0.5704	0.5857	0.5317	**0.4697**	0.4590	0.4696	0.4536
	Transfusion	0.5296	0.5374	0.3911	0.3143	0.3199	**0.3076**	0.3086
	Vehicle	0.7993	0.8190	0.7237	0.6447	0.6216	**0.5971**	0.6257
20	Iris	0.4221	0.4424	**0.3578**	0.4736	0.4771	0.4651	0.4624
	Pima	0.5678	0.5780	0.5018	0.3732	0.3900	0.3581	**0.3406**
	Heart	0.5733	0.5806	0.5467	0.4931	**0.4749**	0.5263	0.4573
	Transfusion	0.5262	0.5364	0.4281	0.5367	0.3350	**0.3297**	0.3399
	Vehicle	0.7277	0.7047	0.6631	0.6347	0.6203	**0.6133**	0.6326
50	Iris	**0.4242**	0.4361	0.4307	0.4624	0.4624	0.4624	0.4624
	Pima	0.5735	0.5742	0.5160	0.3704	**0.3624**	**0.3624**	**0.3624**
	Heart	0.5790	0.5890	0.5530	0.4208	**0.4125**	**0.4125**	**0.4125**
	Transfusion	0.5245	0.5341	0.4235	**0.2667**	0.2762	0.2762	0.2762
	Vehicle	0.8048	0.7271	0.6944	**0.6156**	**0.6156**	**0.6156**	**0.6156**

the effectiveness of the seven algorithms, we chose three evaluation criteria: error rate, F-measure, and GCE. In the literature several criteria are proposed and interpretations may vary depending on these criteria. Furthermore, the use of our proposed consensus functions improve the robustness of the GA in small data sets and make the later more competitive and considerably increase the accuracy rates of large data sets. As our algorithms are based on GAs, they still suffer from the instability of results because of the random aspect.

CONCLUSION

This paper proposes three new consensus functions for cluster ensembles and applies them on three different scenarios: object distributed clustering, feature distributed clustering and robust centralized clustering. In order to find an almost median partition, these consensus functions are based on genetic algorithms.

In the experimental study, we compare the proposed consensus functions with four existing ones. Three of them are based on partitioning algorithms: CSPA, HGPA, and MCLA, whereas the last one is based on genetic algorithms. Based on three evaluation criteria, namely: error rate, F-measure, and the generalized conditional entropy, experimental results show that the proposed consensus functions provide good performance, especially with large data sets. However, we notice some random behavior as a result of the weakness of genetic algorithms in finding exact solutions.

For the proposed consensus functions, we focused on redundant input objects. However, we did not take into account the case where individual clusterers provide overlapping clusterings. It will be of interest to investigate the extension of our proposed consensus functions to fuzzy cluster ensemble.

REFERENCES

Blake, C. L. and C. J. Merz. 1998. Uci repository of machine learning databases http://www.ics.uci.edu/mlearn/mlrepository.html.

Dudoit, S. and J. Fridlyand. 2003. Bagging to improve the accuracy of a clustering procedure. *Bioinformatics* 19:1090–1099.

Fischer, B. and J. M. Buhmann. 2003. Path-based clustering for grouping of smooth curves and texture segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25:513–518.

Fred, A. L. N. 2001. Finding consistent clusters in data partitions. In *Proceedings of the Second International Workshop on Multiple Classifier Systems*, 309–318. Cambridge, UK: Springer-Verlag.

Fred, A. L. N. and A. K. Jain. 2002. Data clustering using evidence accumulation. *International Conference on Pattern Recognition* 4:40276.

Ghaemi, R., N. Sulaiman, H. Ibrahim, and N. Mustapha. 2009. A survey: Clustering ensembles techniques. In *Proceedings of World Academy of Science, Engineering and Technology* 38:148–156.

Karypis, G., R. Aggarwal, V. Kumar, and S. Shekhar. 1997. Multilevel hypergraph partitioning: Applications in VLSI domain. In *Proceedings of the design and automation conference*, 526–529. New York: ACM.

Karypis, G., and V. Kumar. 1998. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing* 20:359–392.

Luo, H., F. Jing, and X. Xie. 2006. Combining multiple clusterings using information theory based genetic algorithm. IEEE International Conference on Computational Intelligence and Security 4881:84–89.

Mühlenbein, H. 1993. Evolutionary algorithms: Theory and applications. *Local Search in Combinatorial Optimization*. New York: John Wiley & Sons.

Strehl, A., and J. Ghosh. 2002. Cluster ensembles: A knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research* 3: 583–617.

Topchy, A., A. K. Jain, and W. Punch. 2003. Combining multiple weak clusterings. In *Third IEEE International Conference on Data Mining*, 331–338. Melbourne, FL, USA.

Topchy, A., A. K. Jain, and W. Punch. 2004. A mixture model for clustering ensembles. In *Proceedings of the SIAM International Conference on Data Mining*, 379–390. Orlando, FL, USA.