Journal of Computer Science 6 (7): 775-784, 2010
ISSN 1549-3636

Summarizing Relational Data Using Semi-Supervised Genetic Algorithm-Based Clustering Techniques

Rayner Alfred
School of Engineering and Information Technology,
University Malaysia Sabah, Locked Bag 2073, 88999, Kota Kinabalu, Sabah, Malaysia

Abstract: Problem statement: In solving a classification problem in relational data mining, traditional methods, for example, the C4.5 and its variants, usually require data transformations from datasets stored in multiple tables into a single table. Unfortunately, we may loss some information when we join tables with a high degree of one-to-many association. Therefore, data transformation becomes a tedious trial-and-error work and the classification result is often not very promising especially when the number of tables and the degree of one-to-many association are large. **Approach:** We proposed a genetic semi-supervised clustering technique as a means of aggregating data stored in multiple tables to facilitate the task of solving a classification problem in relational database. This algorithm is suitable for classification of datasets with a high degree of one-to-many associations. It can be used in two ways. One is user-controlled clustering, where the user may control the result of clustering by varying the compactness of the spherical cluster. The other is automatic clustering, where a non-overlap clustering strategy is applied. In this study, we use the latter method to dynamically cluster multiple instances, as a means of aggregating them and illustrate the effectiveness of this method using the semi-supervised genetic algorithm-based clustering technique. **Results:** It was shown in the experimental results that using the reciprocal of Davies-Bouldin Index for cluster dispersion and the reciprocal of Gini Index for cluster purity, as the fitness function in the Genetic Algorithm (GA), finds solutions with much greater accuracy. The results obtained in this study showed that automatic clustering (seeding), by optimizing the cluster dispersion or cluster purity alone using GA, provides one with good results compared to the traditional k-means clustering. However, the best result can be achieved by optimizing the combination values of both the cluster dispersion and the cluster purity, by putting more weight on the cluster purity measurement. **Conclusion:** This study showed that semi-supervised genetic algorithm-based clustering techniques can be applied to summarize relational data with more effectively and efficiently.

Key words: Data aggregation, clustering, semi-supervised clustering, genetic algorithm, relational data mining, data pre-processing

INTRODUCTION

Relational databases require effective and efficient ways to extract patterns from contents stored in multiple tables. In this process, significant features must be extracted from datasets stored in multiple tables with one-to-many relationships. In a relational database, a record stored in a target table can be associated with one or more records stored in another table due to the one-to-many association constraint. Traditional data mining tools require data in relational databases to be transformed into attribute-value format by joining multiple tables. However, with the large volume of relational data with a high degree of one-to-many associations, this process is not efficient as the joined table can be too large to be processed and we may lose some information when the join operation is performed.

In a relational database, a record stored in the target table is often associated with one or more records stored in another non-target table. We can treat these multiple instances of a record, stored in a non-target table, as a bag of terms. There are a few ways of transforming these multiple instances into bag of terms. Once we have transformed the data representation applicable to clustering operations (Gautam and Chaudhuri, 2004; Basu *et al*., 2002), we can use any clustering techniques to aggregate these multiple instances. The most common pattern extracted from relational database is association rules. However, to extract classification rules from relational database with more effectively and efficiently, taking into consideration of multiple-instance problem, we need to aggregate these multiple instances. In this study, we use a genetic algorithm based clustering technique to aggregate multiple instances of a single record in relational database as a means of data reduction. Before a clustering technique can be applied, we transform the data to a suitable form.

Data transformation for relational data: In a relational database, a single record, R_i, stored in the target table can be associated with other records stored in the non-target table, as shown in Fig. 1. Let R denote a set of m records stored in the target table and let S denote a set of n records (T_1, T_2, T_3,...,T_n), stored in the non-target table. Let S_i be a subset of S, $S_i \subseteq S$, associated through a foreign key with a single record R_a stored in the target table, where $R_a \in R$. Thus, the association of these records can be described as $R_a \longleftarrow S_i$. In this case, we have a single record stored in the target table, T, that is associated with multiple records stored in the non-target table, NT. The target and non-target tables are defined as follows.

Definition: Target table, T, is a table that consists of rows of object where each row represents a single unique object and this is the table in which patterns are extracted.

Definition: A non-target table, NT, is a table that consists of rows of objects where a subset of these rows can be linked to a single object stored in the target table.

The records stored in the non-target table that correspond to a particular record stored in the target table can be represented as vectors of patterns. As a result, based on the vector space model (Salton and Michael, 1984), a unique record stored in non-target table can be represented as a vector of patterns.

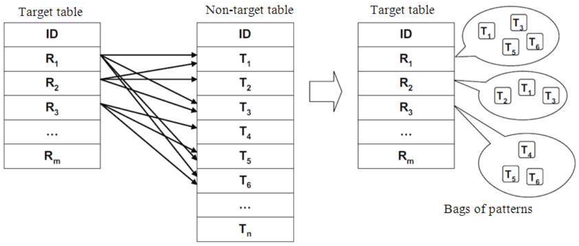


Fig. 1: A one-to-many association between target and non-target relations

In other words, a particular record stored in the target table that is related to several records stored in the non-target table can be represented as a bag of patterns, i.e., by the patterns it contains and their frequency, regardless of their order. The bag of patterns is defined as follows.

Definition: In a bag of patterns representation, each target record stored in the non-target table, NT, is represented by the set of its pattern and the pattern frequencies.

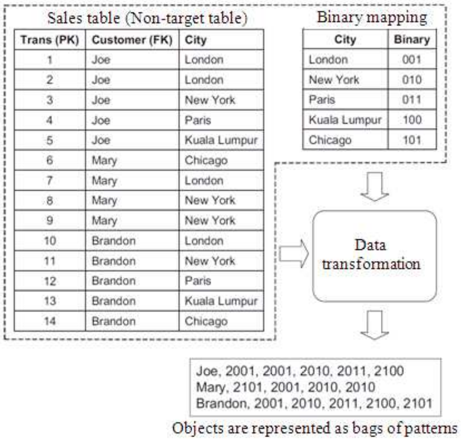
This definition follows the notion of an defined by Lachiche and Flach (2000), where the data is described as a collection of individuals and the induced rules generalize over the individuals, mapping them to a class. For instance, individual-centered domains include classification problems in molecular biology where the individuals are molecules.

In our approach, an individual is represented as a bag of patterns. We use DARA algorithm (Rayner, 2008; Davies and Bouldin, 1979) to summarize data stored in non-target tables that have many-to-one relationships with data stored in the target table. In the DARA algorithm, these patterns are encoded into binary numbers. The process of encoding these patterns into binary numbers depends on the number of attributes that exist in the non-target table. For example, there are two different cases when encoding patterns for the data stored in the non-target table. In the first case (Case I), a non-target table may have a single attribute. In this case, the DARA algorithm transforms the representation of the data stored in a relational database without constructing any new feature to build the (n×p) TF-IDF (Salton and Michael, 1984) weighted frequency matrix, as only one attribute exists in the non-target table.

Case I: Table with a single attribute: In this case, it is assumed that there is exactly one attribute describing the contents of the non-target table that is associated with the target table. In Fig. 2, the Trans attribute is the Primary Key (PK) of the Sales table and the Customer attribute is the Foreign Key (FK) of the table that associates records stored in this non-target table (Sales Table) with records stored in the target table (consists of individual customer). First, the algorithm computes the cardinality of the attribute domain in the non-target table. Cardinality of an attribute is defined as the number of unique values that the attribute can take. If the data consists of continuous values, the data is discretized first and the number of bins taken as the cardinality of the attribute domain.

Next, in order to encode the values into binary numbers, the algorithm finds the appropriate number of bits, n, such that it can represent all different values of the attribute's domain, where $2^{n-1}<|\text{Attribute'sDomain}|\leq 2^{n}$. For example, if the attribute has 5 different values (London, New York, Chicago, Paris, Kuala Lumpur), then we just need 3 ($2^{2}<5\leq 2^{3}$) bits to represent each of these values (001, 010, 011, 100, 101), as shown in Fig. 2. A bag of patterns is maintained to keep track of the number of patterns encountered and their frequencies. For each encoded pattern, the counter for the corresponding pattern in the bag is incremented or the pattern is added to the bag of patterns if it is not already in the bag. The resulting bag of patterns, shown in Fig. 2, can be used to describe the characteristics of an individual record. In Fig. 2, the first digit "2" preceded the binary numbers indicates the index of attribute that the binary numbers are belong to. Since there is only one attribute exists in the datasets, all the encoded patterns produced are belong to index attribute "2".

In the other case (Case II), the non-target table may have multiple attributes exist in the table. In this case, DARA may construct new features, which results in more riched representation of each target record in the non-target table. The method used to encode the patterns derived from these attributes has some influences on the final results of the modeling task (Rayner and Dimitar, 2007).



Sales table (Non-target table)

Trans (PK)	Customer (FK)	City
1	Joe	London
2	Joe	London
3	Joe	New York
4	Joe	Paris
5	Joe	Kuala Lumpur
6	Mary	Chicago
7	Mary	London
8	Mary	New York
9	Mary	New York
10	Brandon	London
11	Brandon	New York
12	Brandon	Paris
13	Brandon	Kuala Lumpur
14	Brandon	Chicago

Binary mapping

City	Binary
London	001
New York	010
Paris	011
Kuala Lumpur	100
Chicago	101

Joe, 2001, 2001, 2010, 2011, 2100
Mary, 2101, 2001, 2010, 2010
Brandon, 2001, 2010, 2011, 2100, 2101

Objects are represented as bags of patterns

Fig. 2: Data transformation for data stored in a non-target table with a single attribute

Table 1: Number of attributes combined, p and the list of patterns produced

p	Patterns produced
1	$F_{1,a}$, $F_{2,b}$, $F_{3,c}$, $F_{4,d}$, ..., $F_{k-1,b}$, $F_{k,n}$
2	$F_{1,a}F_{2,b}$, $F_{3,c}F_{4,d}$, ..., $F_{k-1,b}F_{k,n}$, for k = even
2	$F_{1,a}F_{2,b}$, $F_{3,c}F_{4,d}$, ..., $F_{k,n}$, for k = odd
k	$F_{1,a}F_{2,b}F_{3,c}F_{4,d}...F_{k-1,b}F_{k,n}$

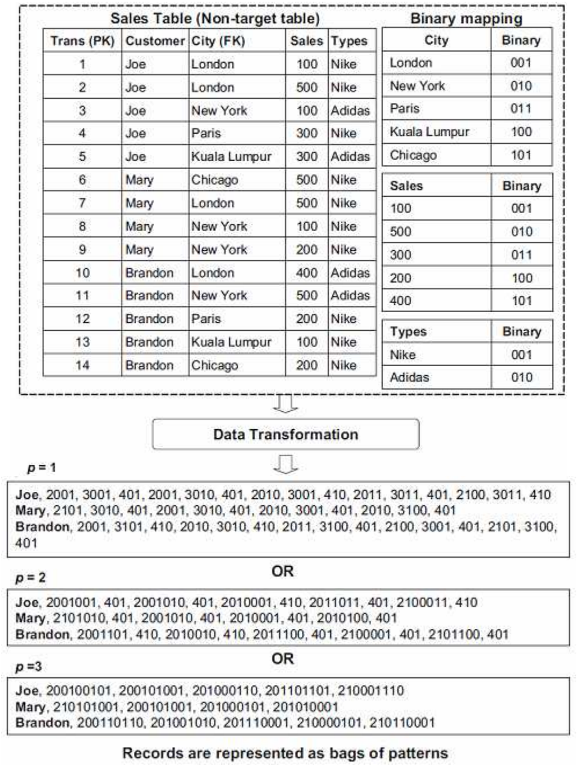
Case II: Table with multiple attributes: In this case, it is assumed that there is more than one attribute that describes the contents of the non-target table associated with the target table. All continuous values of the attributes are discretised and the number of bins is taken as the cardinality of the attribute domain. After encoding the patterns as binary numbers, the algorithm determines a subset of the attributes to be used to construct a new feature.

Here is an example of a simple algorithm to construct features without using feature scoring to generate the patterns that represent the input for the DARA algorithm. Alfred has discussed in detail about the process of data summarization with a genetic-based feature construction algorithm using feature scoring (Rayner, 2008).

For each record stored in the non-target table, concatenate p number of columns' values, where p is less than or equal to the total number of attributes. For example, let F = (F_1, F_2, F_3,...,F_k) denote k field columns or attributes in the non-target table. Let dom(F_i) = ($F_{i,1}$, $F_{i,2}$, $F_{i,3}$, ..., $F_{i,n}$) denote the domain of attribute F_i, with n different values. So, one may have an instance of a record stored in the non-target table with these values ($F_{1,a}$, $F_{2,b}$, $F_{3,c}$, $F_{4,d}$, ..., $F_{k-1,b}$, $F_{k,n}$), where $F_{1,a}\in \text{dom}(F_1)$, $F_{2,b}\in \text{dom}(F_2)$, $F_{3,c}\in \text{dom}(F_3)$, $F_{4,d}\in \text{dom}(F_4)$, ..., $F_{k-1,b}\in \text{dom}(F_{k-1})$, $F_{k,n}\in \text{dom}(F_k)$. Table 1 shows the list of patterns produced with different values of p. It is not natural to have features concatenated like $F_{1,a}F_{2,b}$ but not $F_{1,a}F_{3,c}$, when we have p = 2, since the attributes do not have a natural order. However, the GA approach (Davies and Bouldin, 1979) can be applied to solve this problem.

For each record, a bag of patterns is maintained to keep track of the patterns encountered and their frequencies. For each new pattern encoded, if the pattern exists in the bag, the counter for the corresponding pattern is increased, else the pattern is added to the bag and set the counter for this particular pattern to 1. The resulting bag of patterns can be used to describe the characteristics of a record associated with them.

For instance, Fig. 3 shows the data transformation for data stored in non-target table with multiple attributes. In this example, the Trans attribute is the Primary Key (PK) of the Sales table and the Customer attribute is the Foreign Key (FK) of the table that associates records stored in this non-target table (Sales table) with records stored in the target table (consists of individual customer). Based on this example, the format of patterns produced depends on the parameter p (p = 1, p = 2 and p = k), where p is the number of attributes combined to generate these patterns and k is the total number of attributes. The algorithm is called P_{Single} when p = 1 and P_{All} when p = k respectively.



Sales Table (Non-target table)

Trans (PK)	Customer	City (FK)	Sales	Types
1	Joe	London	100	Nike
2	Joe	London	500	Nike
3	Joe	New York	100	Adidas
4	Joe	Paris	300	Nike
5	Joe	Kuala Lumpur	300	Adidas
6	Mary	Chicago	500	Nike
7	Mary	London	500	Nike
8	Mary	New York	100	Nike
9	Mary	New York	200	Nike
10	Brandon	London	400	Adidas
11	Brandon	New York	500	Adidas
12	Brandon	Paris	200	Nike
13	Brandon	Kuala Lumpur	100	Nike
14	Brandon	Chicago	200	Nike

Binary mapping

City	Binary
London	001
New York	010
Paris	011
Kuala Lumpur	100
Chicago	101

Sales	Binary
100	001
500	010
300	011
200	100
400	101

Types	Binary
Nike	001
Adidas	010

Joe, 2001, 3001, 401, 2001, 3010, 401, 2010, 3001, 410, 2011, 3011, 401, 2100, 3011, 410
Mary, 2101, 3010, 401, 2001, 3010, 401, 2010, 3001, 401, 2010, 3100, 401
Brandon, 2001, 3101, 410, 2010, 3010, 410, 2011, 3100, 401, 2100, 3001, 401, 2101, 3100, 401

Joe, 2001001, 401, 2001010, 401, 2010001, 410, 2011011, 401, 2100011, 410
Mary, 2101010, 401, 2001010, 401, 2010001, 401, 2010100, 401
Brandon, 2001101, 410, 2010010, 410, 2011100, 401, 2100001, 401, 2101100, 401

Joe, 200100101, 200101001, 201000110, 201101101, 210001110
Mary, 210101001, 200101001, 201000101, 201010001
Brandon, 200110110, 201001010, 201110001, 210000101, 210110001

Fig. 3: Illustration of data transformation for data stored in non-target table with multiple attributes

Since there are more than one attribute exist in the datasets, when p = 1, the encoded patterns produced are preceded by the index of the attribute (index "2" through k), where k is the number of attributes in the datasets, as shown in Fig. 3.

In short, the encoding process described here transforms data stored in the non-target table that has many-to-one relations with the target table, to the representation of data in a vector-space model (Salton and Michael, 1984). With this representation, the data can be conveniently clustered by using the hierarchical or partitioning clustering technique, as a means of summarizing them.

MATERIALS AND METHODS

Here, we provide an overview of a semi-supervised clustering technique based on a genetic algorithm. Since clustering (data summarization) works in an unsupervised fashion, the user has no control on the result. However, this study introduces supervision to the learning scheme through some measure of cluster impurity. The basic idea is to find a set of clusters that minimize a linear combination of the cluster dispersion and cluster purity measures.

A semi-supervised clustering technique: As a base to the semi-supervised algorithm, an unsupervised clustering method optimized with a genetic algorithm incorporating a measure of classification accuracy used in decision tree algorithm, the Gini Index (GI) (Breiman *et al*., 1984; Laura and Kilian, 2004), is used. Here, the clustering algorithm that minimizes some objective functions applied to k-cluster centers is examined. Each point is assigned to the nearest cluster centre by Euclidean distance (Srinivasan *et al*., 1996). The main objective is to choose the number of clusters that minimizes some measure of cluster quality. For instance, a cluster dispersion metric can be used, such as the Davies-Bouldin Index (DBI) (Blockeel and de Raedt, 1998), to measure the cluster quality. DBI uses both the within-cluster and between clusters distances to measure the cluster quality. Let $S(Q_k)$ denote the within-cluster distances, where x_i, $x_{i'} \in Q_k$, $i \neq i'$, N_k is the number of samples in cluster Q_k and:

$$C_k = \frac{1}{N_k}\sum\nolimits_{x_i \in Q_k} x_i$$

Then, the Centroid distance, $S_c(Q_k)$ (Eq. 1), for within cluster distance is the mean distance from each element in the cluster to the centroid of the clusters and $d_{ce}(Q_k, Q_l)$ (Eq. 2) is the centroid distance between two clusters, Q_k and Q_l, measured by the Euclidean distance between their centroids, c_k and c_l:

$$\text{Centroid Distance, } S_c(Q_k) = \frac{\sum_i \|x_i - c_k\|}{N_k} \quad (1)$$

$$\text{Centroid Linkage, } d_{ce}(Q_k, Q_l) = \|c_k - c_l\| \quad (2)$$

$$\text{DBI} = \frac{1}{C}\sum_{k=1}^{C} \max_{l \neq k} \left\{ \frac{S_c(Q_k) + S_c(Q_l)}{d_{ce}(Q_k, Q_l)} \right\} \quad (3)$$

According to Davies-Bouldin validity Index (DBI), the best clustering minimizes Eq. 3, where C is the number of clusters. The Davies-Bouldin index is well suited when using k-means partitioning because it gives low values, indicating good clustering results for spherical clusters and those with centers that are far away from each other. This cluster dispersion measure can be incorporated into any clustering algorithm to evaluate a particular segmentation of the data.

The GI has been used extensively in the literature to determine the impurity of a certain branching point in decision trees (Goldberg, 1989; Holland, 1975). Clustering using K cluster centers partitions the input space into K regions. Therefore clustering can be considered as a K-nary partitioning at a particular node in a decision tree and GI can be applied to determine the purity of such a partitioning. In this case, GI of a certain cluster, k, is computed as defined in (Eq. 4), where n is the number of class, P_{kc} is the number of points belonging to cth class in cluster k and N_k is the total number of points in cluster k:

$$\text{GiniC}_k = 1 - \sum_{c=1}^{n}\left(\frac{P_{k_c}}{N_k}\right)^2 \tag{4}$$

$$\text{Purity} = \frac{\sum_{k=1}^{K} T_{C_k} \bullet \text{GiniC}_k}{N} \tag{5}$$

The purity of a particular partitioning into K clusters is defined in Eq. 5, where N is the number of points in the dataset and T_{Ck} is the number of points in cluster k. The smaller the purity, the better the quality of clusters obtained.

Therefore, given both the cluster dispersion measure (DBI) and the cluster impurity measure (GI), by minimizing the objective function defined as a linear combination of DBI (Eq. 3) and GI (Eq. 5), the algorithm becomes semi-supervised. More specifically, given N points and K-clusters, the algorithm will select K cluster centers that minimize the objective function as shown in Eq. 6:

$$F(N, K) = \text{DBI} + \text{Purity} \tag{6}$$

Finding a clustering that is guaranteed to be optimal in terms of a chosen quality measure (e.g., Eq. 6), is in most cases an infeasible task, as it would require an exhaustive search through the space of all possible clustering. Hence, in this experiment, a genetic algorithm-based clustering technique is employed to find the best number of clusters.

A semi-supervised genetic algorithm-based clustering technique: Here, we describe how a semi-supervised genetic algorithm-based clustering technique is employed to improve the predictive accuracy of a modeling task based on a summarized data. A Genetic Algorithm (GA) is a computational abstraction from biological evolution that can be used in any optimization problems (Goldberg, 1989; Holland, 1975). In its simplest form, a GA is an iterative process applying a series of genetic operators such as selection, crossover and mutation to a population of elements. These individuals, or chromosomes, represent possible solutions to the problem. Initially, a random population is created, which represents different points in the search space. An objective or fitness function is associated with each chromosome, which represents the degree of goodness of the chromosome. Based on the principle of the survival of the fittest, some of the chromosomes are selected and each is assigned a number of copies, which go into the mating pool. Biologically inspired operators like crossover and mutation are applied to these strings to yield a new generation of strings. The process of selection, crossover and mutation continues for a fixed number of generations or till a termination condition is satisfied.

There are two phases in the proposed method for the semi-supervised genetic algorithm-based clustering algorithm. In phase I, given N points data, they are reduced by grouping all points to their nearest neighbor. The purpose of this data reduction is to speed up the process of genetic clustering and also to provide the basic platform to find the seeds for the clustering task automatically (Basu *et al*., 2002). In phase II, a genetic algorithm (Holland, 1975) is used to find seeds for clustering m data points based on the objective function (Eq. 6), where m<N. The next two subsections describe the process of finding the seeds of the clusters automatically to achieve the goal of finding the best number of clusters with respect to the objective functions described previously.

Phase I: Data reduction and seeding: The goal of the task in Phase I is to find the initial seeds of the clusters by grouping target records to their nearest neighbor. The steps are describes as follows:

1. For every target record O_i, find the distance to its nearest neighbor, $d_{NNj}(O_i) = \|O_i - O_j\|$, where O_j is the nearest neighbor to O_i and $i \neq j$
2. Compute the average distance of all target records to their nearest neighbor, $d_{AVE} = \frac{1}{N}\sum_{i=1}^{N} d_{NNj}(O_i)$
3. Let d = scale•d_{AVE}, where scale is a constant (Initial value for scale is 0.5 in this experiment). Now, view the n target records as nodes of a graph and connect all nodes that have distance less than or equal to d. Increment scale by 0.1. This is done to find seeds for the clusters
4. Repeat step 3 for as long as there is no target record chosen as the nearest neighbor for two different components of connected target records.

This is to ensure that all connected target records are close enough to one another

5. Find all connected nodes and let the data sets represented by these connected nodes be denoted by $(B_1,B_2,B_3,...,B_{m-1},B_m)$ where m is the number of connected nodes and m <N, since B_i consists of 1 or more connected nodes, $i \leq m$
6. Compute m cluster centers $(z_1, z_2, z_3, ..., z_m)$ from all connected components $(B_1,B_2,B_3,...,B_{m-1},B_m)$ from Step (5), where:

$$z_i = \frac{1}{N_i}\sum_{x_j \in B_i} x_j, \quad i = 1,2,3,...,m$$

where, N_i is the number of nodes connected in B_i.

After reducing N points into m points by grouping them to their nearest neighbors, a genetic algorithm can be applied by treating the m points as the string of chromosomes in the initial population initialization.

Phase II: Genetic-based clustering algorithm: In Phase II, we perform the clustering task based on the cluster seeds obtained in Phase I. Here, we describe the initialization of the population set, the computation of the fitness function and the selection, crossover and mutation processes.

Population initialization step: A population of X strings of length m is randomly generated, where m is the number of the sets (connected components) obtained from the first part (Phase I). X strings are generated with the number of 1's in the strings uniformly distributed within [1, m]. Each string represents a subset of $(B_1, B_2, B_3,...,B_{m-1}, B_m)$. If B_i is in this subset S, the ith position of the string will be 1; otherwise, it will be 0, where $1 \leq i \leq m$. Each B_i in the subset S is used as a seed to generate a cluster. If B_j is not in the subset, they will be merged to the nearest B_k in the subset S, where j, k = 1, 2, 3,...,m and $j \neq k$. The merging of these two components, B_j and B_k, is based on the distance between their centers and this forms a new cluster. After merging, the size and the centre of the new cluster will be recomputed. The merging process for all components that are not listed in the subset S will be repeated until all of them are assigned to the nearest cluster.

Fitness computation: The objective or fitness function has two components (Eq. 6); cluster dispersion and cluster purity. In order to get the best number of clusters, one needs to minimize the DBI (Davies and Bouldin, 1979). On the other hand, in order to group the same type of target records together in a cluster, the purity function, GI (Eq. 5) needs to be minimized. Since the objective fitness function needs to be maximized in GA, the Objective Fitness Function (OFF) that needs to be maximized will be the accumulative value of the reciprocal of cluster dispersion and the reciprocal of cluster purity as defined in Eq. 7:

$$\text{OFF} = \frac{1}{\text{DBI}} + \frac{1}{\text{Purity}} \tag{7}$$

$$\text{OFF} = \beta \cdot \frac{1}{\text{DBI}} + \alpha \cdot \frac{1}{\text{Purity}} \tag{8}$$

In this study, two scalars, β and α (Eq. 8), are introduced that carry the weights of the cluster dispersion and cluster purity parameters. If β = 1 and α = 0, the algorithm becomes an unsupervised GA-based clustering algorithm that will optimize the value of cluster dispersion to get the best number of clusters (represented by DBI-GA-DARA as shown in Table 2). On the other hand, if β = 1 and α = 1, the algorithm becomes a semi-supervised GA-based clustering algorithm that will optimize the values of cluster dispersion and cluster purity to get the best number of clusters while ensuring the purity of the clusters (represented as SS-GA-DARA in Table 2). Finally, if β = 0 and α = 1, the algorithm will optimize the cluster purity only in the process of finding the best number of clusters (represented as GI-GA-DARA as shown in Table 2). In this study, the behavior of the clustering algorithm with the rest of the combinations of values for β and α is examined, as shown in Table 2.

Selection process: For the selection process, either a roulette wheel with slots sized according to the fitness or a tournament selection can be used to sample from the distribution.

Crossover process: A pair of chromosomes, c_i and c_j, are chosen for applying the crossover operator. One of the parameters of a genetic system is probability of crossover, p_c. In our experiment, the probability of crossover, p_c, is set to 0.25. This probability gives the expected number $p_c \bullet X$ of chromosomes, which undergo the crossover operation.

Table 2: Setting and weights of scalars β and α

	Scalar	
Setting	β	α
K-DARA (k-means clustering only)	-	-
DBI-GA-DARA (GDBI)	1.00	0.00
MORE-DBI-GA-DARA (GMDBI)	0.75	0.25
SS-GA-DARA (GSS)	1.00	1.00
MORE-GI-GA-DARA (GMGI)	0.25	0.75
GI-GA-DARA (GGI)	0.00	1.00

Mutation process: The mutation operator performs on a bit-by-bit basis. Another parameter of the genetic system, the probability of permutation p_m, gives the expected number of mutated bits p_m•m•X. In our experiment, the probability of permutation p_m is set to 0.01.

Following selection, crossover and mutation, the new population is ready for its next generation. This evaluation is used to build the probability distribution for the construction of a roulette wheel with slots sized according to current fitness values. The rest of the evolution is just a cyclical repetition of selection, crossover and mutation until a number of specified generations or a specific threshold has been achieved. Once the generation of new chromosomes stops, clusters with only few target records (less than 3 target records) will be removed and its members are moved to the nearest cluster (based on the distance between centers of the clusters).

RESULTS

These experiments are designed to investigate four main factors:

- The feasibility of using data summarization to support the data mining task (e.g., classification) in a multiple tables environment
- The effects of adjusting the weights of the cluster dispersion and purity on the classification task
- The performance gain of a semi-supervised genetic algorithm-based clustering technique over the traditional clustering technique achieved by adjusting the weights of the cluster dispersion and purity and selecting seeds for clustering
- The performance of the DARA algorithm compared to other relational data mining approaches including Progol (Srinivasan *et al*., 1996), Tilde (Blockeel and de Raedt, 1998), Foil (Finn *et al*., 1998), RDBC (Kirsten and Wrobel, 1998; 2000), RElaggs (Krogel and Wrobel, 2001)

These experiments use datasets from the Mutagenesis database (B1, B2, B3) (Kirsten and Wrobel, 2000), Financial database (Discovery Challenge PKDD 1999) and Hepatitis database (PKDD 2005).

The experiments are performed with five different combinations of values for β and α, shown in Table 2. They are referred to as GDBI, GGI, GSS, GMDBI and GMGI as shown in Table 2. For all these settings, we apply a semi-supervised genetic-based algorithm to find the best number of clusters and the Objective Fitness Function (OFF) that needs to be maximized will be the accumulative value of the reciprocal of cluster dispersion and the reciprocal of cluster purity as defined in Eq. 8 with two scalars, β and α. These scalars are introduced that carry the weights of the cluster dispersion and cluster purity parameters respectively. On the other hand, a non-genetic based algorithm is used in the K-DARA setting to cluster the data for a given k number of clusters. For instance, there are two different ranges of k, small (from 2-20 clusters) and large (from 22-40 clusters) numbers of clusters. In K-$\text{DARA}_{\text{small}}$ (small number of clusters) and K-DARA_{big} (large number of clusters), records are clustered based on K_K number of clusters, where K_K has a range from 2 -20 inclusively and from 22-40 respectively, which is manually defined by user. In these experiments, we use the partitioning clustering technique, k-means, to cluster the records. For each different setting, the experiments are repeated for ten different values of K_K and the average of the performance accuracies of the J48 classifiers, implemented in WEKA (Witten and Frank, 1999), are recorded.

In the other settings (GDBI, GGI, GSS, GMDBI, GMGI), the clustering tasks are performed with different values of β and α and the number of clusters k is optimized automatically during the clustering process to maximize the fitness function defined in Eq. 8. Other parameters were set to p_c = 0.80 (crossover probability) and p_m = 0.50 (permutation probability).

Table 3 and 4 show the results of DARA-based performance accuracy, in which seven different settings for the algorithms are compared: K-$\text{DARA}_{\text{small}}$ (small number of clusters), K-DARA_{big} (large number of clusters), GDBI, GGI, GSS, GMDBI and GMGI.

Table 3: 10-fold Cross-Validation performance of the J48 classifier on financial PKDD 1999 and mutagenesis datasets

		Mutagenesis		
Setting	Financial	B1	B2	B3
K-$\text{DARA}_{\text{small}}$	76.3±2.7	80.0±2.0	79.2±3.0	79.2±5.7
K-DARA_{big}	72.3±2.5	81.1±1.7	80.0±2.9	78.4±5.6
GDBI	93.2±2.2	88.8±2.4	88.3±2.2	88.7±1.9
GMDBI	94.8±1.4	95.0±0.6	90.6±1.9	91.4±1.7
GSS	93.2±1.3	95.3±0.6	92.8±1.3	92.4±1.6
GMGI	95.1±1.2	95.3±0.6	96.9±2.8	92.0±0.9
GGI	95.1±1.2	86.6±3.5	84.2±1.9	91.9±1.5

Table 4: 10-fold cross-validation performance of the J48 classifier on Hepatitis PKDD 2005 dataset

	Hepatitis		
Setting	H1	H2	H3
K-$\text{DARA}_{\text{small}}$	72.3±1.7	74.7±1.3	74.8±1.3
K-DARA_{big}	72.7±3.7	73.2±2.2	73.8±2.2
GDBI	76.1±1.8	74.0±1.7	74.1±1.7
GMDBI	87.5±1.6	88.0±1.8	88.0±1.7
GSS	86.8±1.9	86.0±1.9	86.4±2.0
GMGI	84.6±1.9	85.2±1.9	86.2±1.9
GGI	88.3±2.2	88.3±1.8	88.8±1.8

For GSS, setting an equal weight for both values of the reciprocal of cluster dispersion, β and the reciprocal of cluster purity, α, as shown in Eq. 8, provides one with good results. However, for Financial and Mutagenesis datasets, the best result is obtained when more weight is set to the reciprocal of cluster purity, α (GMGI), in the GA fitness function ($\beta = 0.25$ and $\alpha = 0.75$). On the other hand, setting more weight to the reciprocal of cluster dispersion, β (GMDBI), does not provide better results for all three datasets, as shown in Table 3.

For the Hepatitis dataset, the results obtained for all GGI, GSS, GMDBI and GMGI are virtually identical, as shown in Table 4. This indication shows that the different weights for the reciprocal of cluster dispersion, β and the reciprocal of cluster purity, α, in Eq. 8 have no effects on the results, provided that $\alpha \neq 0$.

The accuracy estimations from the 10-fold cross-validation tests the classification of the transformed Mutagenesis datasets (B1, B2, B3), the Financial dataset and the Hepatitis datasets (H1, H2, H3), are much lower when the algorithm uses the reciprocal of cluster dispersion only ($\beta = 1$, $\alpha = 0$ for GDBI). When setting $\beta = 0$ and $\alpha = 0$ (in K-DARA setting), the accuracy estimations, from 10-fold cross-validation performance results for the classification of the transformed Financial, Mutagenesis and Hepatitis datasets, show a drop in performance for all three datasets, compared to the accuracy estimations obtained in the GSS ($\beta = 1$ and $\alpha = 1$). It is not surprising that in the K-DARA setting, the clustering task did poorly, since neither the cluster dispersion nor the cluster purity are considered. With a smaller number of clusters, k, the K-DARA$_{small}$ algorithm (2-20 clusters) performs equally the same compared to the K-DARA$_{big}$ (22-40 clusters) algorithm as shown in Table 3.

However, the algorithm with the GDBI setting still shows an improvement in the performance accuracy compared to the algorithm with the K-DARA setting, as all centers of the clusters are chosen automatically in order to maximize the fitness function of the genetic algorithm-based clustering algorithm. In contrast, with the same number of clusters for the algorithm with the K-DARA setting (non-genetic based algorithm), the task of choosing all centers of k clusters is done by taking the first k points in the datasets, which is not very efficient. As a result, the algorithm with the K-DARA setting produces clusters that may not be distinguishable from each other. In other words, the differences between clusters are not clear when using the clustering algorithm with the K-DARA setting. In short, by transforming the representation of data for records stored in the non-target table with one-to-many relations into a vector space model using DARA, the automatic clustering method that uses a semi-supervised genetic algorithm-based clustering technique proved particularly successful on datasets with one-to-many relationships.

DISCUSSION

Table 5 shows the results of paired t-test (p = 0.05) for mutagenesis, financial and hepatitis datasets. In this table, the symbol '⊕' indicates significant improvement in performance by method in row over method in column and the symbol '∅' indicates no significant improvement in performance by method in row over method in column, on the three datasets.

Table 5: Results of paired t-test (p = 0.05) for mutagenesis, financial PKDD 1999 and hepatitis PKDD 2005 datasets

Mutagenesis (B1, B2, B3)					
Method	GSS	GDBI	GGI	GMDBI	GMGI	K-DARA
GSS	-	⊕,⊕,⊕	⊕,⊕,⊕	⊕,⊕,⊕	∅,∅,∅	⊕,⊕,⊕
GDBI	∅,∅,∅	-	⊕,⊕,⊕	∅,∅,∅	∅,∅,∅	⊕,⊕,⊕
GGI	∅,∅,∅	∅,∅,∅	-	∅,∅,∅	∅,∅,∅	⊕,⊕,⊕
GMDBI	∅,∅,∅	⊕,⊕,⊕	⊕,⊕,⊕	-	∅,∅,∅	⊕,⊕,⊕
GMGI	∅,∅,∅	⊕,⊕,⊕	⊕,⊕,⊕	∅,⊕,⊕	-	⊕,⊕,⊕
K-DARA	∅,∅,∅	∅,∅,∅	∅,∅,∅	∅,∅,∅	∅,∅,∅	-
Financial						
Method	GSS	GDBI	GGI	GMDBI	GMGI	K-DARA
GSS	-	∅	∅	⊕	∅	⊕
GDBI	∅	-	∅	∅	∅	⊕
GGI	⊕	⊕	-	∅	∅	⊕
GMDBI	∅	⊕	∅	-	∅	⊕
GMGI	∅	⊕	∅	∅	-	⊕
K-DARA	∅	∅	∅	∅	∅	-
Hepatitis (H1, H2, H3)						
Method	GSS	GDBI	GGI	GMDBI	GMGI	K-DARA
GSS	-	⊕,⊕,⊕	∅,∅,∅	∅,∅,∅	∅,∅,∅	⊕,⊕,⊕
GDBI	∅,∅,∅	-	∅,∅,∅	∅,∅,∅	∅,∅,∅	∅,∅,∅
GGI	∅,∅,∅	⊕,⊕,⊕	-	∅,∅,∅	∅,∅,∅	⊕,⊕,⊕
GMDBI	∅,∅,∅	⊕,⊕,⊕	∅,∅,∅	-	∅,∅,∅	⊕,⊕,⊕
GMGI	∅,∅,∅	⊕,⊕,⊕	∅,∅,∅	∅,∅,∅	-	⊕,⊕,⊕
K-DARA	∅,∅,∅	∅,∅,∅	∅,∅,∅	∅,∅,∅	∅,∅,∅	-

Table 6: Results previously published of mutagenesis (B1, B2, B3) and financial (PKDD 1999) datasets

		Mutagenesis		
Setting	Financial	B1	B2	B3
GDBI	93.2	88.8	88.3	88.7
Progol (Srinivasan *et al*., 1996)	-	76.0	81.0	83.0
Foil (Finn *et al*., 1998)	74.0	83.0	75.0	83.0
Tilde (Blockeel and de Raedt, 1998)	81.3	75.0	75.0	85.0
RDBC (Kirsten and Wrobel, 1998; 2000)	-	83.0	84.0	82.0
Relaggs (Krogel and Wrobel, 2001)	99.9	86.7	87.8	86.7

Significant improvements in predictive accuracy for the J48 classifier are recorded for both GSS and GMGI methods over the rest of the methods but not each other. There is no significant improvement in predictive accuracy when using the GSS method over the GMGI method and vice-versa. Finally, Table 6 also shows the comparison between the results obtained in these experiments and the other previously published results on Mutagenesis and Financial datasets, such as Progol (Srinivasan *et al*., 1996), Tilde (Blockeel and de Raedt, 1998), Foil (Finn *et al*., 1998), RDBC (Kirsten and Wrobel, 1998; 2000), RElaggs (Krogel and Wrobel, 2001). The algorithm with the GDBI setting is chosen to compare the accuracy estimations with other published results since the class information is not utilized in this setting. In Table 6, the algorithm GDBI produces better results compared to the other approaches on relational data mining. However, the algorithm with the K-DARA setting produces no improvement in the classification task compared to the other published results, simply because the centers of the clusters chosen are not the best centers that can distinguish all the clusters from each other. In other words, the DARA algorithm can use the cluster seeds to improve the k-means clustering in order to summarize datasets in a multi-relational environment. In short, some of the findings that can be concluded from these experiments are outlined as follows:

- Data summarization for multiple tables with a high number of one-to-many relationship is feasible in order to get higher accuracy estimations
- Using automatic seeds for clustering has improved the accuracy estimations for the DARA algorithm
- Adjusting the weights of cluster dispersion and purity has influenced the accuracy estimations, in which using the DARA transformation process with the GSS or GMGI settings for clustering produced a better result
- Without considering the class information, the DARA algorithm with the GDBI setting produced higher accuracy estimation results compared to the other relational data mining approaches

CONCLUSION

This study introduced the concept of data summarization that adopts the TF-IDF weighted frequency matrix concept borrowed from the information retrieval theory (Salton and Michael, 1984) to summarize data stored in relational databases with a high number of one-to-many relationships among entities, through the use of a clustering technique. Clustering algorithms can be used to generate summaries based on the information contained in the datasets that are stored in a multi-relational environment. This study outlined the data transformation process performed by the Dynamic Aggregation of Relational Attributes (DARA) algorithm that transforms the representation of data stored in relational databases into a vector space format data representation that is suitable in clustering operations. By clustering these multi-association occurrences of an individual record in the multi-relational database, the characteristics of records stored in non-target tables are summarized by putting them into groups that share similar characteristics.

In this study, a method for semi-supervised learning that combines supervised and unsupervised learning techniques has also been introduced to get the optimum number of clusters to cluster these records. The basic idea is to treat a series of records, associated with a single record in the target table, as a bag of patterns and take an unsupervised clustering method and simultaneously optimize the misclassification error of the resulting clusters. Experimental results show that using the reciprocal of DBI for cluster dispersion and the reciprocal of GI for cluster purity as the fitness function in the GA algorithm finds solutions with much greater accuracy. The results obtained in this study show that automatic clustering (seeding), by optimizing the cluster dispersion or cluster purity alone using GA, provides one with good results compared to the traditional k clustering. However, the best result can be achieved by optimizing the combination values of both the cluster dispersion and the cluster purity, by putting more weight on the cluster purity measurement

(GMGI). The basic idea of this experiment is to incorporate classification information into an unsupervised algorithm to aggregate records with multi-association in multi-relational datasets. The experiments show that data summarization improves the performance accuracy of the prediction task. These results also support the issue stated by Blockeel and Sebag (2003), in their discussion about the concept of individual-centered representation (Lachiche and Flach, 2000), where the use of individual-centered representations has a positive effect on the theoretical learnability of concepts. By clustering these records based on the multi-instances that are related to them, the records can be summarized by putting them into groups that share similar characteristics.

REFERENCES

Basu, B., A. Banerjee and R. Mooney, 2002. Semi-supervised clustering by seeding. Proceedings of the 19th International Conference on Machine Learning, July 2002, Morgan Kaufmann Publishers Inc., San Francisco, CA., USA., pp: 27-34.

Blockeel, H. and L. de Raedt, 1998. Top-down induction of first-order logical decision trees. Artif. Intell., 101: 285-297. DOI: 10.1016/S0004-3702(98)00034-4

Blockeel, H. and M. Sebag, 2003. Scalability and efficiency in multi-relational data mining. SIGKDD Explorat., 5: 17-30.

Breiman, L., J. Friedman, T. Olshen and C. Stone, 1984. Classification and Regression Trees. 1st Edn., Wadsworth International, California, ISBN: 10: 0412048418, pp: 368.

Davies, D.L. and D.W. Bouldin, 1979. A cluster separation measure. IEEE Trans. Pattern Anal. Mach. Intel., PAMI, 1: 24-227. DOI: 10.1109/TPAMI.1979.4766909

Finn, P.W., S. Muggleton, D. Page and A. Srinivasan, 1998. Pharmacophore discovery using the inductive logic programming system Progol. Mach. Learn., 30: 241-270. DOI: 10.1023/A:1007460424845

Gautam, G. and B.B. Chaudhuri, 2004. A novel genetic algorithm for automatic clustering. Patt. Recogn. Lett., 25: 173-187. DOI: 10.1016/j.patrec.2003.09. 012

Goldberg, D.E., 1989. Genetic Algorithms-in Search, Optimization and Machine Learning. 1st Edn., Addison-Wesley Publishing Company Inc., ISBN: 0201157675, pp: 432.

Holland, J., 1975. Adaptation in Natural and Artificial Systems. 1st Edn., University of Michigan Press, ISBN: 10: 0262581116, pp: 228.

Kirsten, M. and S. Wrobel, 1998. Relational distance-based clustering. Proceeding of the 8th International Conference on Inductive Logic Programming, July 22-24, Springer-Verlag, London, UK., pp: 261-270. http://portal.acm.org/citation.cfm?id=742767

Kirsten, M. and S. Wrobel, 2000. Extending K-means clustering to first-order representations. Proceeding of the 10th International Conference on Inductive Logic Programming, July 24-27, Springer-Verlag, London, UK., pp: 112-129. http://portal.acm.org/citation.cfm?id=648000.742935

Krogel, M.A. and S. Wrobel, 2001. Transformation-based learning using multirelational aggregation. Lecturere Notes Comput. Sci., 2157: 142-155. DOI: 10.1007/3-540-44797-0

Lachiche, N. and P. Flach, 2000. A First-Order Representation for Knowledge Discovery and Bayesian Classification on Relational Data. In: Mining, decision Support, Meta-learning and ILP: Forum for Practical Problem Presentation and Prospective Solutions, Pavel, B. and J. Alipio (Eds.). Citeseerx, pp: 49-60.

Laura, E.R. and S. Kilian, 2004. Theoretical comparison between the Gini index and information gain criteria. Ann. Math. Artif. Intell., 41: 77-93.

Rayner, A. and K. Dimitar, 2007. Clustering approach to generalized pattern identification based on multi-instanced objects with DARA. Proceeding of the Communications of the 11th East-European Conference on Advances in Databases and Information Systems, Sept 2007, Technical University of Varna, pp: 1-12.

Rayner, A., 2008. A genetic-based feature construction method for data summarization. Proceeding of the 4th International Conference on Advanced Data Mining and Applications, Oct. 8-10, ACM Press, Chengdu, China, pp: 39-50. http://portal.acm.org/citation.cfm?id=1428392.1428400

Salton, G. and M. Michael, 1984. Introduction to Modern Information Retrieval. McGraw-Hill, Inc., New York, USA., ISBN: 0070544840.

Srinivasan, A., S. Muggleton, M.J.E. Sternberg and R.D. King, 1996. Theories for mutagenicity: Study in first-order and feature-based induction. Artif. Intell., 85: 277-299. DOI: 10.1016/0004-3702(95)00122-0

Witten, I.H. and E. Frank, 1999. Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. 1st Edn., Morgan Kaufmann, ISBN: 10: 1558605525, pp: 371.