Organizing Large Case Library by Linear Programming

Caihong Sun[1], Simon Chi Keung Shiu[2], and Xizhao Wang[3]

[1] Information School, Renmin University of China, Beijing, 100872, P.R. China
caihongsun@vip.sina.com
[2] Computing Department, The Hong Kong Polytechnic University,
Hung Hum, Kowloon, Hong Kong
csckshiu@comp.polyu.edu.hk
[3] College of Mathematics and Computer, Hebei University,
Baoding City, Hebei Province, 071002, P.R. China
wangxz@mail.hbu.edu.cn

Abstract. In this paper we proposed an approach to maintain large case library, which based on the idea that a large case library can be transformed to a compact one by using a set of case-specific weights. A linear programming technique is being used to obtain case-specific weights. By learning such local weights knowledge, many of redundant or similar cases can be removed from the original case library or stored in a secondary case library. This approach is useful for case library with a large number of redundant or similar cases and the retrieval efficiency is a real concern of the user. This method of maintaining case library from scratch, as proposed in this paper, consists of two main steps. First, a linear programming technique for learning case-specific weights is used to evaluate the importance of different features for each case. Second, a case selection strategy based on the concepts of case coverage and reachability is carried out to select representative cases. Furthermore, a case retrieval strategy of the compact case library we built is discussed. The effectiveness of the approach is demonstrated experimentally by using two sets of testing data, and the results are promising.

1 Introduction

Nowadays, with information exploding, how to help users to locate the information what they need is a big challenge for the information providers, educators. Recently, how to build an efficient case library are proposed and discussed a lot by many researchers, especially with the rapid growth of case-based reasoning (CBR) systems both in research area and commercial use in e-business. Maintaining case libraries become more important when case-based reasoning (CBR) systems are being used to solve wide range of problems. Large-scale CBR systems are becoming more prevalent, with case library sizes ranging from thousands [1] to millions of cases [2]. With the increasing growth of the size of a case base, case retrieval takes longer time and case base maintenance, which is defined as the process of refining a case base to improve the system's performance [3], has become an active research topic. Various case base maintenance problems have been addressed in the past few years. To provide maintenance support at the case level, [4] suggested a competence preserving

© Springer-Verlag Berlin Heidelberg 2005

deletion approach. Competence (or coverage) is the range of target problems that a given system can solve, and is also a fundamental evaluation criterion of CBR system performance. In [5], the authors presented a new model of case competence, and demonstrated a way in which the proposed model of competence can be used to assist case authors. Authors in [6] used data mining techniques in a novel role of a back-end technology for CBR systems, i.e., the acquisition of cases and discovery of adaptation knowledge. [7] used advanced AI techniques such as neural networks and fuzzy methods to acquire features' importance and eliminate irrelevant features in a given dataset. One important concept in the CBR community is to distinguish the salient features from all the features in the dataset; feature selection methods can reduce the task's dimensionality when they eliminate irrelevant features [8]. In [7,8,9], for each feature, a value can be assigned to indicate the degree of importance of this feature. In [10] the authors introduced the local weight concept while investigating weighted fuzzy production rules in which a local weight is assigned to each proposition of a rule to indicate the degree of importance of the proposition in the antecedent contributing to its consequent. In this paper, we use the local weight concept to select representative cases from a large case library. A local weight called case-specific weight is assigned to each feature of each case to indicate the degree of importance of the feature contributing to its case.

Nearest neighbor (NN) algorithms are techniques used to solve Pattern Recognition and Classification problems. Nowadays NNs is used for case retrieval in CBR systems 12]. [11] demonstrated that even if cases are not explicitly classified into a set of finite groups (classes), the solution space can often be clustered into a collection of sets and each set contains similar solutions. The NN classification procedure is very straightforward: given a set of classified examples, which are described as points in an input space, a new unclassified example is assigned to the known class of the nearest example. Many researchers [13,14,15] use local metrics to compute the "nearest" relation while others use global metrics. In our paper, we use case-specific weight to compute the "nearest" relation. In order to ignore the noisy data and improve the retrieval efficiency of CBR systems, we propose a method to maintain the case base by selecting representative cases based on the coverage and reachability concepts. After applying the maintenance process, the case base contains fewer cases, and many noisy cases were deleted.

We establish an approach to transfer the original large case library to a small one with the purpose that it can significantly improve the retrieval efficiency and the performance of the system. Furthermore, the computational complexity of acquiring local specified feature weights is relative small when comparing with neuro-fuzzy feature learning. The approach integrates learning case-specific weight, computing case competence and selecting seed cases into a framework of case base maintenance. The maintaining methodology has two main steps. First, a linear programming technique is used to learn case-specific weights and evaluate the importance of different features for each case. Second, a case selection strategy based on the concepts of case coverage and reachability is used to select representative cases. Furthermore, based on the framework of case maintenance, we discuss a case retrieval strategy and a case addition strategy. In order to demonstrate the effectiveness of this approach, two experiments using the Pima Indians Diabetes and the Australian Credit Approval are carried out. The results show that the two testing case bases can be reduced by 89.96% and 90.04% respectively. The training set overall accuracy of the two smaller case bases is 100%, and the testing set overall accuracy is 73.48% and 79.71% respectively.

2 Approach of Maintaining a Case Library

Consider a case library where an individual case is represented as (Problem, Solution). The problem is assumed to be an n-dimensional vector $(p_1, p_2, \cdots, p_n)$ where each component corresponds to a feature (attribute) of the case library. The solution, without losing generality, is regarded as a cluster symbol taking values of 0 and 1 where 0 denotes positive class and 1 negative class. In other words, all cases in the case library are categorized into two classes. They are the positive class X with the individual case $x = (x_1, x_2, \cdots, x_n)$, and the negative class Y with the individual case $y = (y_1, y_2, \cdots, y_n)$.

Assume that, for each feature, a distance measure has been defined. The distance measure for the j-th feature is denoted by ρ_j , i.e., ρ_j is a mapping from $F_j \times F_j$ to $[0, \infty)$ (where F_j denotes the range of the j-th feature) with the properties:

(1) $\rho_j(a,b) = 0$ if and only if a = b;

(2) $\rho_j(a,b) = \rho_j(b,a)$ and

(3) $\rho_j(a,b) \le \rho_j(a,c) + \rho_j(c,b)$

Usually the distance measure depends on the specific domains. In this paper, we define the distance measure as follows:

(1) $\rho_j(a,b) = |a-b|$ if a and b are real numbers;

(2) $\rho_j(a,b) = \begin{cases} 1\ if\ \ a \ne b \\ 0\ if\ \ a = b \end{cases}$ if a and b are symbols.

We now define the distance measure between two cases. For any pair of cases $x = (x_1, x_2, \cdots, x_n)$ and $y = (y_1, y_2, \cdots, y_n)$, the distance measure is defined as

$$d(x,y) = \frac{1}{M} \sum_{j=1}^{n} \rho_j(x_j, y_j) \tag{1}$$

where M is such a positive number to scale that the value range of the distance measure into [0, 1]. Usually, the number M can be determined by

$$M = \max_{x \in X, y \in Y} od(x,y) . \tag{2}$$

and $od(x,y) = \sum_{j=1}^{n} \rho_j(x_j, y_j)$.

After introducing the distance measure, our methodology, which consists of two major phases, will be described in sub-sections 2.1 and 2.2 respectively. Furthermore, based on maintenance framework, section 2.3 will discuss about case retrieval strategy.

2.1 Phase One - Learning Case-Specific Weights

In this section, a feature evaluation function is defined. The smaller the evaluation value, the better the corresponding case-specific weights. Thus we would like to find the weights such that the evaluation function attains its minimum by using a linear programming technique. We formulate this optimization problem as follows. We first introduce a new distance measure based on case-specific weight, called pseudo-distance.

For any given case $p=(p_1,p_2,\cdots,p_n)$, its case-specific weight refers to a vector $w^{(p)}=(w_1^{(p)},w_2^{(p)},\cdots,w_n^{(p)})$ where each component is non-negative real number. When no confusion occurs, the superscript (p) can be omitted.

By incorporating the case-specific weight into the distance measure, a pseudo-distance measure for a pair of cases $x=(x_1,x_2,\cdots,x_n)$ and $y=(y_1,y_2,\cdots,y_n)$ can be defined as

$$d_x(y)=\sum\nolimits_{j=1}^{n} w_j^{(x)}\rho_j(x_j,y_j) \tag{3}$$

Where $w_j^{(x)}$ is the case-specific weight corresponding to the j-th component of case x. This measure is called pseudo-distance because the symmetry $d_x(y)=d_y(x)$ generally does not hold. Here we denote $d_x(y)$ as the distance from y to x.

For a fixed case $p=(p_1,p_2,\cdots,p_n)\in X$ with case-specific weight vector $w=(w_1,w_2,\cdots,w_n)$, we consider the following function:

$$E_p(w_1,w_2,\cdots,w_n)=\Sigma_{x\in X}\left(\frac{\beta}{1-\beta}d_p(x)\big(1-d(p,x)\big)-d(p,x)d_p(x)\right) \tag{4}$$

Where β is a constant in (0,1). The two measures, $d(p,x)$ and $d_p(x)$ are regarded as the old and new distance from case x to case p respectively. Noting that

$$\frac{\partial}{\partial d_p(x)}E_p(w_1,w_2,\cdots,w_n)=\sum\nolimits_{x\in X}\left(\frac{\beta}{1-\beta}-\frac{1}{1-\beta}d(p,x)\right) \tag{5}$$

We have

(1) if $d(p,x)<\beta$ then $E_p(w_1,w_2,\cdots,w_n)$ monotonically increases with respect to $d_p(x)$; and

(2) if $d(p,x)>\beta$ then $E_p(w_1,w_2,\cdots,w_n)$ monotonically decreases with respect to $d_p(x)$.

It implies that, when $d(p,x) < \beta$, $d_p(x)$ approaches its minimum (or when $d(p,x) > \beta$, $d_p(x)$ approaches its maximum), the evaluation function (4) monotonically approaches its minimum. Thus, we expect that, by minimizing the evaluation function (4), $d_p(x)$ becomes small (when $d(p,x) < \beta$) or big (when $d(p,x) > \beta$). If the parameter β is regarded as a threshold, then $d(p,x) < \beta$ ($d(p,x) > \beta$ resp.) can be interpreted as that the old distance from x to p is relatively "small" ("big" resp.). By minimizing function (4), we expect that the new distance from x to p, $d_p(x)$, becomes smaller (bigger resp.). It is clear that the objective of minimizing function (4) is to make the case classification decision easy. In other words, under the principle that "The smaller the distance between cases, the more possibly the two cases belonging to the same cluster", the decision for determining whether two cases belonging to the same cluster in the new distance measure is easier than in the old distance measure.

We now focus on the minimization of function (4) under certain conditions. Let $c = d(p,x)$ and

$$\chi_j = \rho_j(x_j, p_j) \tag{5*}$$

Then, from equation (4), (1) and (3), we have

$$E_p(w_1, w_2, \cdots, w_n) = \sum_{x\in X}\left(\frac{\beta}{1-\beta}\left(\sum_{j=1}^{n} w_j\chi_j\right)(1-c) - \left(\sum_{j=1}^{n} w_j\chi_j\right)c\right)$$

$$= \sum_{x\in X}\left(\sum_{j=1}^{n}\left(\frac{\beta}{1-\beta}(1-c)\chi_j - c\chi_j\right)\right)w_j = \sum_{j=1}^{n} k_j w_j \tag{6}$$

Where $k_j = \sum_{x\in X}\left(\left(\frac{\beta}{1-\beta}(1-c)-c\right)\chi_j\right), j = 1,2,\cdots,n$. (6*)

It is no meaning minimizing (6) if without any constraints. Suppose that the case library $X \cup Y$ has no conflicted cases, i.e., there exist no two cases $x \in X$ and $y \in Y$ such that $x = y$. The shortest distance (in old measure) from the positive case p to the negative set Y is given by:

$$\varepsilon_p = \min_{y\in Y} d(p,y) \tag{7}$$

Thus, the inequality $\varepsilon_p > 0$ holds. We hope that, in new pseudo-distance measure, the distance from p to each negative y is not less than ε_p. That is,

$$\min_{y\in Y} d_p(y) \geq \min_{y\in Y} d(p,y) \tag{8}$$

We attempt to find a set of case-specific weights for the case p such that the function (6) achieves minimum subject to the constraint (8). This is an optimization problem with constraints. Noting that equation (8) is equivalent to $d_p(y) \ge \varepsilon_p$ for all cases $y \in Y$, and $d_p(y) = \sum_{j=1}^{n} w_j \rho_j(p_j, y_j)$, the optimization problem becomes the following linear programming problem:

$$\left.\begin{aligned} &\min\ E_p(w_1, w_2, \cdots, w_n) = \sum_{j=1}^{n} k_j w_j \\ &\qquad \text{s.t. } d_p(y) = \sum_{j=1}^{n} \alpha_j w_j \ge \varepsilon_p,\ y \in Y \\ &\qquad\qquad 0 \le w_j \le 1 \quad j = 1, 2, \cdots, n \end{aligned}\right\} \tag{9}$$

Where k_j is given by equations (6*), α_j is defined as $\alpha_j = \rho_j(p_j, y_j)$, and ε_p is specified by equation (7).

2.2 Phase Two - Representative Cases Selection Strategy

This phase aims to select representative cases from positive class X according to the case-specific weights obtained in phase one. Our selection strategy is based on a coverage concept.

After solving the linear programming (9), we can obtain a set of case-specific weights $(w_1^{(p)}, w_2^{(p)}, \cdots, w_n^{(p)})$ for each positive $p = (p_1, p_2, \cdots, p_n) \in X$. Using this set of case-specific weights, we define the coverage of a positive case. Let p be a given positive case, q be an arbitrary case, we say p covers q if

$$d_p(q) < \varepsilon_p \tag{10}$$

where ε_p is specified by equation (7). The coverage of p, Coverage(p), is defined as

$$Coverage(p) = \{e \mid d_p(e) < \varepsilon_p\} \tag{11}$$

From equation (7) we know that a positive example covers a negative example is impossible. In other words, the coverage of a positive case is a subset of X. Similarly, we can define the reachablity of p as

$$\mathrm{Re}\,achibillity(p) = \{e \mid d_e(p) < \varepsilon_p\} \tag{12}$$

The coverage of a case p represents the generalization capability of this case. The bigger the number of cases in its coverage, the more representative the selected case p. On the other hand, the reachability of a case p represents the degree to which p can be covered by another case. From equation (11) and (12), we can determine both the coverage and the reachability for each positive case p. The current objective is to select a set of positive cases with their case-specific weights such that the selected cases can cover the entire positive set X. In fact, the selected cases can exclude the entire negative set Y.

There is always a trade-off between the number of cases to be stored in the case library of a Case-Based Expert System and the performance of retrieval efficiency. The larger the case library, the more the problem space covered. However, it would also downgrade the system performance if the number of cases grows to an unacceptable high level. We expect the number of the selected representatives to be as small as possible. Thus our selection strategy is described as:

"Finding a set of positive cases which can cover the entire positive set X such that the number of this set of cases attains minimum."

Finding an exact algorithm for our optimal selection problem is not realistic, since it is a NP-hard problem. An intuitive and powerful heuristic algorithm is described as follows.
The set R is initialized to be empty.

1. For each case p in X, determine Coverage(p) and Reachability(p) by equations (11) and (12).
2. Find case p* such that $|Coverage(p^*)| = Max_{p \in L} |Coverage(p)|$. If there exists more than one case such that the maximum is reached, then select a case p** from them such that $|\mathrm{Re}\,achability(p^{**})| = Min_{p^*} |\mathrm{Re}\,achibility(p^*)|$. If there exists more than one case such that the minimum is reached, select one randomly.
3. Put $R = R \cup \{p^*\}$ and $L = L - Coverage(p^*)$, if $|L| = 0$ then stop else goto 1.

Consequently, the set R is approximately regarded as the set of selected representative cases.

2.3 Case Retrieval Strategy

After the above two phases, we obtain a case library with a group of representative positive cases and a group of representative negative cases, together with a case specific weight with each case. When a test case comes, we use Nearest Neighbor (NN) algorithm to retrieve cases in the new case library. Here we modify the NN algorithm a little, i.e. use pseudo-distance given in equation (3) to compute distance between the test case and each representative cases (with case specific weights) in the obtained compact case library.

2.4 An Example

There is a case base with 10 cases, each case has 4 attributes, 6 cases belong to positive class X, and the other 4 cases belong to negative class Y. Without losing generality, we consider case 1 to 6 as class X, and case 7 to 10 as class Y. Now we use this sample case base to demonstrate our approach.

Learning the specific weight of case p (a positive case). First we use Equation (7) to compute ε_p, then for a given constant β in (0,1), apply case p (here p=1,2,…,6) and negative cases (i.e. case 7 – case 10) in class Y to linear programming problem specified as equation (9). After solving this linear programming problem, we get a specific weight for case p.

Now we've got a set of specific weights for each positive case. Then we apply these weights to the heuristic algorithm described in section 2.2 to obtain the selected representative cases.

3 Experimental Analyses

This section presents the experimental analysis of our methodology on two real-world problems, i.e. the Pima Indians Diabetes (PID)[1] and the Australian Credit Approval (ACA)[2] problems. The PID data consist of 8 attributes with numeric values (The 8 attributes are " Number of times pregnant", "Plasma glucose concentration a 2 hours in an oral glucose tolerance test", "Diastolic blood pressure (mm Hg)", "Triceps skin fold thickness (mm)", "2-Hour serum insulin (mu U/ml)", "Body mass index (weight in kg/(height in m)^2)", "Diabetes pedigree function" and "Age (years)".), two classes (one is positive and the other is negative), and 768 instances. The ACA data consists of 14 attributes, two classes (one is positive and the other is negative) and 690 instances. Among the 14 attributes, 6 are numerical and 8 are categorical.

For the whole PID database we randomly select 70% for training and the other 30% for testing, for the whole ACA database we randomly select 80% for training and other 20% for testing. Table 1 shows the distribution of training and testing data of the PID and ACA datasets.

Table 1. Distribution of Training and Testing for PID and ACA

	PID database			ACA database		
	Total	Training	Testing	Total	Training	Testing
Positive	538	346	154	383	308	75
Negative	230	192	76	307	244	63
Total	768	538	230	690	552	138

After applying our approach mentioned in section 2 to PID database (β=0.4) and ACA database ((β=0.3), the training and testing accuracy are shown in Table 2.

Table 2. PID and ACA Experimental Results

	Representatives	Deletion Rate	Training Accuracy	Testing Accuracy
PID	54	89.96%	100%	73.48%
ACA	55	90.04%	100%	79.71%

[1] Taken from UCI Machine Learning Repository, see website http://www.ics.uci.edu/~mlearn/MLRepository.html

[2] The dataset can be downloaded from http://www.liacc.up.pt/ML/statlog/datasets/australian/australian.doc.html

We also compare the number of selected representatives and training/testing accuracy by using case-specific weights and without using case-specific weights. We also test with different threshold β values. The comparison results are shown in Figures 1 and 2 for PID and ACA respectively. From the experimental results, one may see that by introducing the case-specific weights, the deletion rate and testing accuracy are better than without using weights. In addition, the selection of a good threshold β can base on the distribution of the old distance in the training sets, i.e., identifying a threshold to indicate if the old distance is small or big.

The result shows that the size of case bases after using our maintenance process can be reduced by almost 90% if case-specific weights are introduced. The training accuracy could be 100%, and the testing accuracy could attain 75% or more. Since the size of a case base is significantly reduced, the retrieval efficiency of the case base system could greatly be increased.

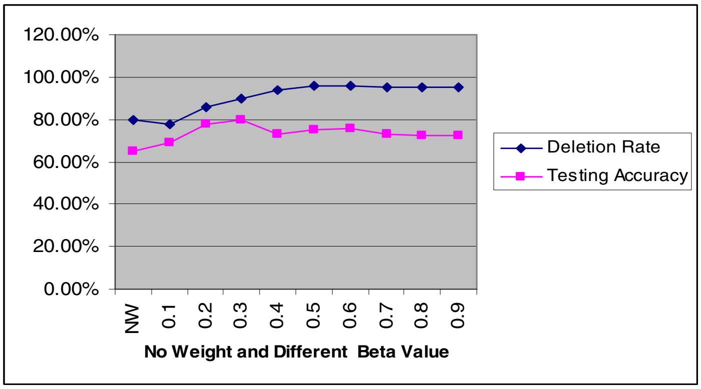

Fig. 1. PID Experiment Analysis on no weights and different Beta value

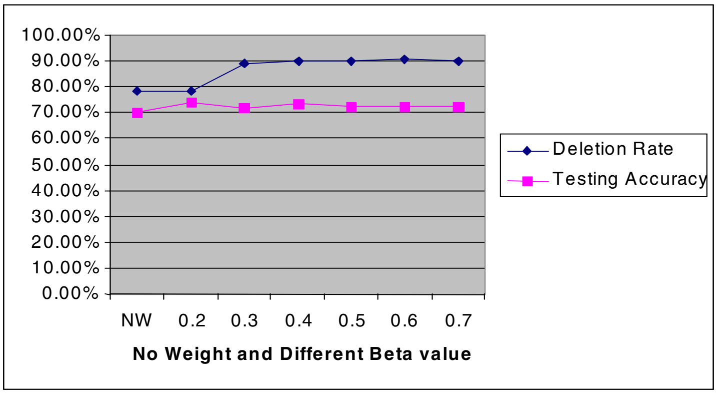

Fig. 2. ACA Experiment Analysis on no weights and different Beta value

4 Summary and Future Works

In this paper, we have developed a methodology of maintaining CBR systems by introducing the case-specific weight concept. The main idea is to transform a large case library to a small one by using a set of case-specific weights, which are obtained by solving linear programming problems. The experimental results show that both testing accuracy and retrieval efficiency of the CBR systems are increased. Besides, it shows that the computational cost of maintenance process is not high. Future works include (1) performing a computational complexity analysis and compare computational costs with other case base maintenance approaches, (2) extending this approach to solve multiple classes problems and non-classification problems (such as predication problem), (3) comparing the linear programming case-specific weight learning algorithm with weight learning algorithms, and (4) investigating the case addition strategy of our approach for on-line or periodic updates, and (5) comparing the retrieval efficiency with C4.5.

References

1. Cheetham, W. , Graf, J.: Case-Based Reasoning in Color Matching, Advances in Case-Based Reasoning, the second International Conference on Case-Based Reasoning, ICCBR(1997) 1-12.
2. Deangdej, J., Lukose, D., Tsui, E., Beinat, P. , Prophet, L. : Dynamically creating indices for two million cases: A real world problem. Advances in Case-Based Reasoning, the third European Workshop, EWCBR (1996) 105-119.
3. Leake, D.B., Wilson, C.: Categorizing Case-Base Maintenance: Dimensions and Directions. Advances in Case-Based Reasoning, 4th European Workshop, EWCBR (1998) 196-207.
4. Smyth, B. , Keane, M.T.: Remembering to Forget: A Competence-Preserving Case Deletion Policy for Case-based Reasoning systems. Proceedings of the fourteenth International Joint Conference on Artificial Intelligence, IJCAI (1995) 377-382.
5. Smyth, B., Mckenna, E.: Modeling the Competence of Case-bases. Advances in Case-Based Reasoning, 4th European Workshop, EWCBR(1998) 208-220.
6. Anand, S.S, Patterson, D., Hughes, J. , Bell, G.: Discovering Case Knowledge using Data Mining. in D. A. Second Pacific Asia Conference, Australia, PAKDD (1998) 25-35.
7. Basak, J., De, R. K. ,Pal, S. K: Unsupervised feature selection using a neuro-fuzzy approach. Pattern Recognition Letters, 19 (1998) 998-1006.
8. Wettscherck, D., Aha, D.W.: Weighting Features. Case-based Reasoning Research and Development, First International Conference, ICCBR (1995) 347-358.
9. Aha, D. W.: Feature weighting for lazy learning algorithms. In H. Liu & H. Motoda (Eds.) Feature Extraction, Construction and Selection: A Data Mining Perspective, Norwell MA: Kluwer (1998).
10. Yeung, D. S. , Tsang, E. C. C.: Weighted Fuzzy Production Rules. Fuzzy Sets and Systems 88 (1997) 299-313.
11. Avesani, P., Perini, A. , Ricci, F.: Interactive case-based planning for forest fire management, Applied Artificial Intelligence (1998).

12. Blanzieri, E., Ricci, F.: Probability Based Metrics for Nearest Neighbor Classification and Case-Based Reasoning. Advances in Case-Based Reasoning, the third European Workshop, EWCBR (1996) 14-28.
13. Aha, D. W., Goldstone, R. L.: Learning attribute relevance in context in instance-based learning algorithms. In proceedings of the Twelfth Annual Conference of the Cognitive Science Society, pp141-148, Cambridge, MA, (1990).
14. Short, R. D. , Fukunaga K.: A new nearest neighbor distance measure. In proceedings of the fifth IEEE International Conference on Pattern Recognition,, Miami veach, FL(1980) 81-86.
15. Racci, F. ,Avesani P.: Learning a local similarity metric for case-based reasoning. In the first international Conference on Case-Based Reasoning, ICCBR (1995) 301-312.