Improving Performance of Similarity-Based Clustering by Feature Weight Learning

D.S. Yeung, *Senior Member*, *IEEE*, and
X.Z. Wang, *Member*, *IEEE*

Abstract—Similarity-based clustering is a simple but powerful technique which usually results in a clustering graph for a partitioning of threshold values in the unit interval. The guiding principle of similarity-based clustering is "similar objects are grouped in the same cluster." To judge whether two objects are similar, a similarity measure must be given in advance. The similarity measure presented in this paper is determined in terms of the weighted distance between the features of the objects. Thus, the clustering graph and its performance (which is described by several evaluation indices defined in this paper) will depend on the feature weights. This paper shows that, by using gradient descent technique to learn the feature weights, the clustering performance can be significantly improved. It is also shown that our method helps to reduce the uncertainty (fuzziness and nonspecificity) of the similarity matrix. This enhances the quality of the similarity-based decision making.

Index Terms—Clustering, similarity-based clustering, transitive closure, fuzziness and nonspecificity, gradient-descent technique.

1 Introduction

CLUSTERING aims to determine a partition over a given set of objects. Among the existing clustering methodologies, the similarity-based clustering is a simple but powerful one.The guiding principle of similarity-based clustering is "similar objects are within the same cluster and dissimilar objects are in different clusters," and a similarity measure must be defined to compute the degree of similarity between two objects. Two objects are considered to be similar or dissimilar based on this degree. Obviously the boundary between the two terms, similar and dissimilar, is not crisp. Thus, similarity-based clustering is a type of fuzzy clustering though the generated partitions are considered crisp.

Roughly speaking, there are two types of fuzzy clustering. One is to generate a fuzzy partition on the set of objects (there is no crisp boundary among the clusters), whereas the other is to generate a set of crisp partitions such as the one generated by similarity-based clustering which was introduced, for instance, in [5], [16]. For the former, much research works have been done [1], [3], [13]. Comparatively speaking, the research on the latter is much less. However, many interesting applications of similarity-based clustering had been reported. This paper makes an attempt to improve the performance of the existing similarity-based clustering techniques via feature weight learning. More specifically, one objective is to obtain a reasonable similarity matrix with better clustering performance by learning the feature weights. Another objective is to reduce the uncertainty (fuzziness and nonspecificity) existing in the clustering results and therefore enhance the quality of decision making.

This paper has the following organization: Section 2 outlines the methodology of clustering based on a similarity matrix and its transitive closure. Section 3 discusses the feature weight learning where the gradient descent technique is used to minimize an objective function. Section 4 gives some indices such as intrasimilarity and intersimilarity for evaluating the quality of a clustering performance. Section 5 reports experimental results on databases selected from UCI machine learning repository [14] and makes a brief comparison with k-means. The conclusion of this paper is given in the last section.

- *D.S. Yeung is with the Department of Computing, Hong Kong Polytechnic University, Kowloon, Hong Kong. E-mail: csxzwang@comp.polyu.edu.hk.*
- *X.Z. Wang is with the School of Mathematics and Computer Science, Hebei University, Baoding, Hebei, China. E-mail: csxzwang@comp.polyu.edu.hk.*

Manuscript received 25 Oct. 2000; revised 04 Feb. 2001; accepted 21 June 2001.
Recommended for acceptance by P. Meer.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 113043.

0162-8828/02/$17.00 © 2002 IEEE

2 Clustering Based on a Similarity Matrix

Clustering based on a similarity matrix is a popular and practical technique which usually performs by means of transforming the similarity matrix into its transitive closure [5]. We briefly describe this technique as follows:

Let CL denote a set of objects for clustering and ρ a similarity measure defined on CL. Specifically, if $CL = \{e_1, e_2, \cdots, e_N\}$, then a matrix $S = (s_{ij})_{N\times N}$ can be defined by $s_{ji} = \rho(e_i, e_j)$.

The similarity matrix $S = (s_{ij})_{N\times N}$ is reflexive ($s_{ji} \geq 0$ and $s_{ji} = 1$ if $i = j$) and symmetric ($s_{if} = s_{ji}$), but does not necessarily satisfy the fuzzy transitive condition $s_{ij} \geq \vee_k (s_{ik} \wedge s_{jk})$, where $\vee, \wedge$ stand for $\max$ and $\min$, respectively. Usually we consider that one object is similar to another object if and only if the degree of similarity is greater than or equal to a predefined threshold α. In this way, the transitive condition states that, for any three objects i, j, and k, if object i is similar to object k ($s_{ik} \geq \alpha$) and object k is similar to object j ($s_{kj} \geq \alpha$) then object i is similar to object j ($s_{ij} \geq \alpha$). Since the transitive condition is indispensable for clustering, the similarity matrix is always transformed into its Transitive Closure (denoted by $TC(S) = (t_{ij})_{N\times N}$). $TC(S)$ is defined as a minimal reflexive, symmetric, and transitive matrix. Usually, $TC(S)$ is obtained by searching for an integer k such that, S^k is transitive.

According to the transitive closure $TC(S) = (t_{ij})_{N\times N}$, the N objects $\{e_1, e_2, \cdots, e_N\}$ can be categorized into several clusters in terms of the criterion "e_i and e_j belong to the same cluster if $t_{ij} \geq \alpha$," where α is a given threshold.

We now focus on the generation of a similarity matrix. Suppose that each object is identified by a collection of features $\{F_j (j = 1, \cdots, m\}$. Then, for $i = 1, 2 \cdots, N$, $e_i = (x_{i1}, x_{i2}, \cdots, x_{im})$, where x_{ij} corresponds to the value of the feature $F_j (1 \leq j \leq m)$. One may find many approaches to determine the similarity measure between two objects in terms of their feature values such as Euclidean distance, relational coefficients, cosine of angle between two vectors, etc. But, in this paper, we will consider the similarity measure associated with a weighted distance $d_{pq}^{(w)}$ which is defined as

$$d_{pq}^{(w)} = d^{(w)}(e_p, e_q) - \left(\sum_{j=1}^{m} w_j^2 (x_{pj} - x_{qj})^2\right)^{1/2}, \tag{1}$$

where $w = (w1, w2, \cdots, w_m)$ is called the feature weight vector. For each j, w_j is nonnegative and is assigned to the $j\text{th}$ feature F_j to indicate the importance of that feature. It is noted that the distance defined by (1) is just the usual Euclidean metric while all weights are equal to 1. Thus, the weighted distance defined in (1) is a generalization of the Euclidean distance. The similarity measure is then defined by the following equation:

$$\rho_{pq}^{(w)} = \frac{1}{1 + \beta \cdot d_{pq}^{(w)}}, \tag{2}$$

where β is a positive parameter determined by solving the following (2a)

$$\frac{2}{N(N-1)} \sum_{q>p} \rho_{pq}^{(1)} = 0.5, \tag{2a}$$

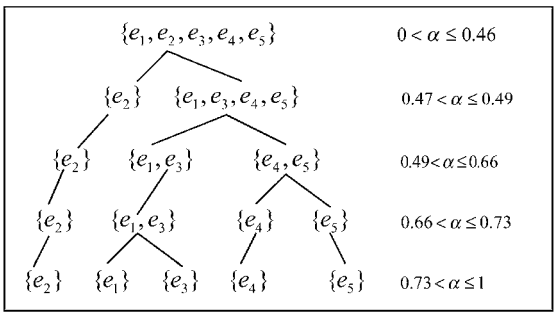

Fig. 1. Dynamic clustering graph 1 with five layers.

where N is the number of objects, $\rho_{pq}^{(1)}$ is the value of $\rho_{pq}^{(w)}$ at $w = (1, 1, \cdots, 1)$. The aim of determining β in this way is to uniformly distribute all similarity values around the value 0.5. In other words, since similarity values are user-defined, we expect that they can uniformly distribute around 0.5 when no additional information on estimating these similarity degrees is available.

Example 1. Consider five objects

$$e_1(4.8, 5.0, 3.0, 2.0),$$
$$e_3(2.0, 3.0, 4.0, 5.0),$$
$$e_3(5.0, 5.0, 2.0, 3.0),$$
$$e_4(1.0, 5.0, 3.0, 1.0), \text{ and}$$
$$e_5(1.0, 4.9, 5.0, 1.0).$$

The formula specified by (2) (where all feature weights are considered to be 1 and the parameter β is taken to be 0.26 according to the β selection criterion (2a)) is used to compute the similarity matrix S. It is easy to check that the similarity matrix does not satisfy the fuzzy transitive condition. Its transitive closure $TC(S)$ can be obtained by computing S^4.

$$S = \begin{pmatrix} 1 & 0.45 & 0.73 & 0.49 & 0.47 \\ & 1 & 0.46 & 0.45 & 0.45 \\ & & 1 & 0.46 & 0.42 \\ & & & 1 & 0.66 \\ & & & & 1 \end{pmatrix}.$$

$$TC(S) = S^4 = \begin{pmatrix} 1 & 0.46 & 0.73 & 0.49 & 0.49 \\ & 1 & 0.46 & 0.46 & 0.46 \\ & & 1 & 0.49 & 0.49 \\ & & & 1 & 0.66 \\ & & & & 1 \end{pmatrix}.$$

Based on this transitive closure, a dynamic clustering graph (Fig. 1) can be constructed when the threshold α changes from 1 to 0. From Fig. 1, one can easily see that the partition varies with the change of the threshold α. For example the partition contains three clusters when $\alpha \in (0.49, 0.66]$ and four clusters when $\alpha \in (0.66, 0.73]$. One may observe that it is difficult to make a crisp decision for selecting a specified partition since most of the nondiagonal elements of the similarity matrix or its transitive closure are close to 0.5, which is considered to be the most fuzzy value.

3 Learning Feature Weights

The last section shows that the similarity-based clustering result is a dynamic clustering graph with the change of the threshold, i.e., a set of crisp partitions. According to the guiding principle of similarity-based clustering, the degree of similarity between objects can be regarded as the degree to which the two objects belong to the same cluster. It is difficult to say that two objects definitely belong to the same cluster except when they are identical. One reason is that two objects may belong to the same cluster for a given threshold and they could possibly belong to different clusters for another given threshold. This indicates that uncertainty exists when judging whether two objects belong to the same cluster. This uncertainty results from the fuzziness of the similarity matrix. The bigger the fuzziness of the similarity matrix is, the more difficult it is to determine the clustering. Feature weight is an important concept which has been successfully applied to fuzzy production rules [15]. Noting that the similarity matrix depends on the feature weight, it is possible to reduce its fuzziness by adjusting the values of the feature weights. This would improve the decision making performance.

For the purpose of reducing fuzziness of the similarity matrix, we consider the minimization of the following evaluation function [2]:

$$E(w) = \frac{2}{N(N-1)} \sum_{q<p} \frac{1}{2} \left(\rho_{pq}^{(w)} \left(1 - \rho_{pq}^{(1)}\right) + \rho_{pq}^{(1)} \left(1 - \rho_{pq}^{(w)}\right) \right) \quad (3)$$

in which N is the number of objects in a partition, $w = (w_1, w_2, \cdots, w_n)$ represents the feature weight vector, $\rho_{pq}^{(w)}$ specified by (2) is the similarity between objects e_p and e_q, and $\rho_{pq}^{(1)}$ is defined in (2a).

This evaluation function $E(w)$, which was formulated in [2], is constructed based on a simple function $f(x, y) = x(1-y) + y(1-x)$ $(1 \le x, y \le 1)$.

Noting that $\frac{\partial f}{\partial x} = 1 - 2y, \frac{\partial f}{\partial x} > 0$ if $y < 0.5, \frac{\partial f}{\partial x} < 0$ if $y > 0.5$, we have from (3) the following equality:

$$Lim_{[\rho_{pq}^{(w)} \to 0, \rho_{pq}^{(1)} < 0.5] \text{ or } [\rho_{pq}^{(w)} \to 1, \rho_{pq}^{(1)} > 0.5]} E(w) = \min E(w). \quad (3a)$$

Assuming all feature weights be equal to 1, we can compute the similarity between two objects by (2), which can be regarded as the "old similarity." If one allows the feature weights to vary in the interval $[0, \infty)$, then the similarity computed by (2) may be viewed as the "new similarity," which depends on the selection of the feature weights. It is observed from (3a) that minimizing (3) implies that the new similarity tends to 1 (0) if the old similarity is greater (less) than 0.5. It is also true that by minimizing (3), one could improve the intrasimilarity and intersimilarity (defined in the next section). That is, the average similarity within the same cluster will increase and the average similarity among diverse clusters will decrease. Moreover, according to [4], the fuzziness of the new similarity matrix $S^{(w)}$ can be defined as

$$Fuzziness\left(S^{(w)}\right) = -\frac{1}{N(N-1)} \sum_{q<p} \left(\rho_{pq}^{(w)} \log \rho_{pq}^{(w)} + \left(1 - \rho_{pq}^{(w)}\right) \log\left(1 - \rho_{pq}^{(w)}\right) \right).$$

From (3a), we know that minimizing $E(w)$ possibly leads to $\rho_{pq}^{(w)} \to 0$ or 1. In either case, one can easily see that the fuzziness of $S^{(w)}$ will be very small. Due to the reduction of fuzziness (i.e., since the new similarity departs from 0.5 more than the old similarity), the similarity matrix with feature weights offers a better decision making capability for clustering results than the one without the feature weights.

According to [6], fuzzy sets mainly have three semantics, namely, similarity, preference, and uncertainty. Similarity is exploited in clustering analysis and fuzzy control. Since the similarity depends on the feature weights, the degree to which a pair of objects belongs to the same model could be changed by

adjusting (learning) the feature weights. We train the feature weights by minimizing (3).

For training the feature weights, a three-layered neural network ([2]) can be designed. We omit the network figure and directly establish the training equations. To minimize (3), the gradient-descent technique is used. We derive the training equations as follows:

The change in w_j, $\triangle w_j$ is computed as

$$\triangle w_j = -\eta \frac{\partial E(w)}{\partial w_j}, \tag{4}$$

for $j = 1, \cdots, n$, where η is the learning rate. For the computation of $\frac{\partial E(w)}{\partial w_j}$, the following expressions are used:

$$\frac{\partial E(w)}{\partial w_j} = \frac{1}{N(N-1)} \sum_{q<p} \left(1 - 2 \cdot \rho_{pq}^{(1)}\right) \cdot \frac{\partial \rho_{pq}^{(w)}}{\partial d_{pq}^{(w)}} \cdot \frac{\partial d_{pq}^{(w)}}{\partial w_j}, \tag{5}$$

$$\frac{\partial \rho_{pq}^{(w)}}{\partial d_{pq}^{(w)}} = \frac{-\beta}{\left(1 + \beta \cdot d_{pq}^{(w)}\right)^2}, \tag{6}$$

$$\frac{\partial d_{pq}^{(w)}}{\partial w_j} = w_j (x_{pj} - x_{qj})^2 / d_{pq}^{(w)}. \tag{7}$$

The learning rate η for each epoch is determined by

$$\begin{aligned} &E\left(w_1 - \eta \frac{\partial E(w)}{\partial w_1}, \cdots, w_n - \eta \frac{\partial E(w)}{\partial w_n}\right) = \min_{\lambda > 0} \\ &E\left(w_1 - \lambda \frac{\partial E(w)}{\partial w_1}, \cdots, w_n - \lambda \frac{\partial E(w)}{\partial w_n}\right). \end{aligned} \tag{8}$$

Equation (8) aims to search for an appropriate step-length (since too small an η leads to low computational efficiency but too big an η usually results in divergence of the algorithm). This is a one-dimensional search (E is considered as a function with respect to the nonnegative variable λ), therefore, many standard one-dimensional search techniques such as Fibonacci method [12] can be used.

The training algorithm is described as follows:

Step 1. Determine the value of β according to (2a).

Step 2. Initialize w_j with random values in $[0, \infty)$.

Step 3. Compute $\frac{\partial E(w)}{\partial w_j}$ for each j using (5), (6), and (7).

Step 4. Searching for η according to (8).

Step 5. Update w_j with $w_j + \triangle w_j$ for each j if $w_j + \triangle w_j > 0$.

Step 6. Repeat steps 3, 4, and 5 until convergence, i.e., until the value of E becomes less than or equal to a given threshold, or until the number of iterations exceeds a certain predefined number.

After training, the function $E(w)$ should attain a local minimum. We expect that, on average, the similarity values $\{\rho_{pq}^{(w)}, p = 1, \cdots, N, q < p\}$ with trained weights are closer to 0 or 1 than that without trained weights such as $\{\rho_{pq}^{(1)}, p = 1, \cdots, N, q < p\}$. That is, we expect the fuzziness of the similarity matrix with trained weights to be much smaller than the fuzziness of the similarity matrix without weights.

Example 2. Consider the five objects in Example 1 again. We want to learn the four corresponding feature weights such that the evaluation function (3) attains a local minimum. The parameters β are taken to be 0.26 and all initial values of feature weights are taken to be 1. After 1,000 epochs, the evaluation function specified in (3) converges with the feature weight vector $w = [3.8210, 8.1342, 0.0000, 0.0000]$. Using this vector and (1) and (2), one can determine the similarity matrix S and its transitive closure S^4:

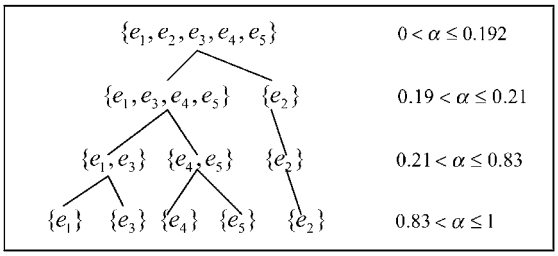

Fig. 2. Dynamic clustering graph 2 with four layers.

$$S = \begin{pmatrix} 1 & 0.16 & 0.83 & 0.21 & 0.21 \\ & 1 & 0.16 & 0.19 & 0.19 \\ & & 1 & 0.20 & 0.20 \\ & & & 1 & 0.83 \\ & & & & 1 \end{pmatrix}$$

$$TC(S) = S^4 = \begin{pmatrix} 1 & 0.19 & 0.83 & 0.21 & 0.21 \\ & 1 & 0.19 & 0.19 & 0.19 \\ & & 1 & 0.21 & 0.21 \\ & & & 1 & 0.83 \\ & & & & 1 \end{pmatrix}.$$

Based on this transitive closure, a dynamic clustering graph (Fig. 2) is formed when the threshold α changes from 1 to 0. By comparing the similarity matrices of Examples 1 and 2, it is found that the degrees of similarity depart from 0.5 in Example 2 more than those in Example 1. The decision making based on the similarity matrices in Example 2 is easier than the case in Example 1 for clustering. For Example 2, an intuitive decision of three clusters can be made, i.e., $\{e_1, e_3\}$, $\{e_4, e_5\}$, and $\{e_2\}$, corresponding to any threshold selected in $(0.21, 0.83]$. This intuitive decision will be validated according to the evaluation indices defined in the next section.

It is worth noting that, as a result of the feature weight learning, the last two feature weights vanish. In this situation, the feature weight learning coincides with the feature selection in [2]. The learned feature weight vector indicates that the first two features are significant for clustering.

4 Indices of Evaluation on Clustering Based on a Similarity Matrix

For a given similarity matrix, the clustering results based on its transitive closure will be different with the change of the threshold α. All objects will constitute one cluster if α is taken to be 0. While the threshold α changes from 0 to 1, the number of clusters generated in the clustering will gradually increase from 1 to a maximum number. Thus, a dynamic clustering graph is obtained (see Figs. 1 and 2). One problem is the appropriate evaluation of the clustering graph. In other words, if two dynamic clustering graphs are generated for a given data set, which graph is relatively better? In what follows, we suggest to use some evaluation indices to measure the performance of the clustering. We expect that the learning of the feature weights specified in the previous section will result in a better performance of the clustering. However, due to the interaction among these indexes, it is difficult to achieve the optimal values of all indices simultaneously. Usually a trade-off must be made.

1. **Fuzziness of a similarity matrix.** For a given similarity-matrix $S = (s_{ij})_{N \times N}$, the fuzziness of S is defined according to [4], i.e.,

$$\begin{aligned} &Fuzziness(S) = \\ &\frac{-1}{N(N-1)} \sum_{j<i} \left(s_{ij} \log s_{ij} + (1 - s_{ij}) \log(1 - s_{ij})\right) \end{aligned} \tag{9}$$

Reducing the fuzziness will make decision making easier. This evaluation index is for a similarity matrix, i.e., for the entire clustering graph.

The following evaluation indices 2, 3, 4, and 5 are defined for a specified partition. While these indices are needed for evaluating the entire clustering performance, the simple average for different levels can be used.

2. **Intrasimilarity**. For a given cluster L, the intrasimilarity of L is defined as the average similarity of all pairs in L. For a clustering with m clusters $\{L_1, L_2, \cdots, L_m\}$ corresponding to a threshold α, the intrasimilarity $SM^{(w)}_{Intra}(\alpha)$ is defined as the average of all its cluster intrasimilarities. It is clear that the value of $SM^{(w)}_{Intra}$ is in $[0,1]$. The bigger the value of $SM^{(w)}_{Intra}$, the better the performance of clustering.
3. **Intersimilarity.** For a pair of clusters L_1 and L_2, the intersimilarity is defined as

$$SM^{(w)}_{L_1,L_2} = \frac{1}{\boldsymbol{r}_1 \cdot \boldsymbol{r}_2} \sum_{p \in L_1, q \in L_2} \rho^{(w)}_{pq},$$

where $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are numbers of objects in L_1 and L_2, respectively. For a clustering with m clusters $\{L_1, L_2, \cdots, L_m\}$ corresponding to a threshold α, the intersimilarity $SM^{(w)}_{Inter}(\alpha)$ is defined as the average of all pairs of intersimilarities. Obviously, the value of $SM^{(w)}_{\text{int}\,er}$ is in $[0,1]$. The smaller is the value of $SM^{(w)}_{\text{int}\,er}$, the better is the performance of clustering.

Sometimes the clustering performance is evaluated in terms of the intersimilarity matrix $(s_{ij})_{m\times m}$ which is defined as $\left(SM^{(w)}_{L_i,L_j}\right)_{m\times m}$.

4. **Ratio of intrasimilarity to intersimilarity.** According to (2) and (3), the intrasimilarity and the intersimilarity are two important indices to describe the quality of a partition. But, in many practical problems, their optimal values, i.e., the maximum intrasimilarity and the minimum intersimilarity, are not achieved at the same partition. Thus, a trade-off between them should be given to evaluate the overall quality of the partition. It is suggested to use the ratio of intrasimilarity to intersimilarity to evaluate the overall similarity of a partition. The maximal threshold which maximizes the ratio of intrasimilarity to intersimilarity is called the optimal threshold and it is denoted by α_{opt}. More specifically, the optimal threshold value α_{opt} is defined as $R(\alpha_{opt}) = \max_{0<\alpha<1} R(\alpha)$, where

$$R(\alpha) = \text{Intrasimilarity}(\alpha)/\text{Intersimilarity}(\alpha). \tag{10}$$

When a crisp decision for the clustering result needs to be made, we suggest considering the crisp partition corresponding to the optimal threshold.

5. **Nonspecificity.** Suppose that the clustering graph contains $m+1$ partitions corresponding to $m+1$ thresholds $\alpha_0, \alpha_1, \alpha_2 \cdots \alpha_m$ with the order $\alpha_0 < \alpha_1 < \alpha_2 < \cdots < \alpha_m$. We neglect the partition corresponding to α_0 (because the intersimilarity does not exist) and consider the remaining m partitions. When a crisp decision for the clustering graph is needed, one crisp partition will be selected from the m partitions. The index 4 suggests using the maximum ratio of intrasimilarity to intersimilarity to make the selection. Obviously, the nonspecificity exists in the selection process. We model the nonspecificity as follows:

Consider the vector

$$V = (R_1, R_2, \cdots, R_m),$$

where $R_1 = R(\alpha_i)/(\max_{1\le i\le m} R(\alpha_i))$ and $R(\alpha_i)$ is defined by (10). Referring to index 4, we can consider R_j as the possibility with which the jth partition is selected as the crisp decision $(1 \le j \le m)$. Thus, the vector V can be considered as a possibility distribution. The nonspecificity (or ambiguity) [8] is defined as

$$Ambig(V) = \sum_{i=1}^{m} (\pi_i - \pi_{i+1}) \ln i, \tag{11}$$

where $(\pi_1, \pi_2, \cdots, \pi_m)$ is the permutation of the normalized possibility distribution $(R_1, R_2, \cdots, R_m)$, sorted such that, $\pi_i \ge \pi_{i+1}$ for $i = 1, 2, \cdots, m$, and $\pi_{m+1} = 0$

In [10], this specificity is called U-uncertainty. It is clear that the distribution $(1, 0, \cdots, 0)$ has the minimum nonspecificity 0, which means the first component can be fully specified.

Example 3. Consider Examples 1 and 2. Based on the similarity matrix and its transitive closure in Example 1 where no feature weights are used, the values of the evaluation indices are computed as follows: Optimal threshold: $\alpha_{opt} = 1.0000$; Corresponding partition: $\{\{e_1\}, \{e_2\}, \{e_3\}, \{e_4\}, \{e_5\}\}$; Intrasimilarity = 1.0000; Intersimilarity = 0.5031; Ratio of intrasimilarity to intersimilarity = 1.9876; Fuzziness = 0.3367; Nonspecificity = 1.4707.

Similarly, the values of those for Example 2 with the learned feature weight vector

$$w = [3.8210, 8.1342, 0.0000, 0.0000]$$

are: Optimal threshold $\alpha_{opt} = 0.8300$; Corresponding partition: $\{\{e_1\}, \{e_2\}, \{e_3\}, \{e_4\}, \{e_5\}\}$; Intrasimilarity = 0.8866; Intersimilarity = 0.1865; Ratio of intrasimilarity to intersimilarity = 4.7545; Fuzziness = 0.2404; Nonspecificity = 0.8187.

By comparing these two results, one can see that the ratio of intrasimilarity to intersimilarity of the first result is far less than that of the second result and the first fuzziness (nonspecificity) is bigger than the second one. It indicates that the second result (with learned feature weights) is overall better than the first result (without feature weights). In fact, the first result corresponding to the threshold 1 makes no sense.

5 Experimental Demonstration

To further verify the advantages of clustering with feature weight learning, we select eight databases. For these databases, the performance based on the five evaluation indices defined in Section 4 will be computed and compared.

The eight databases employed for experiments are obtained from various sources. They are

1. Rice taste data [11],
2. Iris data [7],
3. Servo data [14],
4. Thyroid gland data [14],
5. BUPA liver disorders [14],
6. MPG data [14],
7. Boston Housing Data [14], and
8. Pima India diabetes data [14]; where Servo data have four symbolic attributes.

Due to the symbolic attributes, we revise the distance formula (1) as the following:

$$d^{(w)}_{pq} = d^{(w)}(e_p, e_q) = \left(\sum_{j=1}^{n} w_j^2 \rho_j^2(x_j, y_j) \right)^{1/2}, \tag{1a}$$

TABLE 1
Clustering Performance on Several Data Sets

Data-Sets	Weight Information	#. of layers	Avg #. of classes per layer	Avg #. Of singletons per layer	Intra-Similarity	Inter-Similarity	Ratio	Maximum ratio	Fuzziness	Non-specificity
Rice	Without weights	37	48.71	44.11	0.9492	0.3995	2.4169	3.0024	0.3346	4.2799
	With 5 learned weights	32	43.81	34.13	0.9684	0.3897	2.4552	3.2826	0.2984	3.3373
Iris	Without weights	21	50.43	40.52	0.9151	0.4600	1.9945	2.1146	0.3115	3.2360
	With 4 learned weights	4	14.00	3.25	0.8536	0.3211	2.7186	3.2248	0.2691	1.4455
	With weights (1,1,0,0)	10	25.70	16.90	0.9182	0.5555	1.6576	1.8235	0.2992	2.4642
	With weights (0,0,1,1)	7	26.86	18.57	0.9210	0.5118	1.8201	2.1960	0.2894	2.0110
Servo	Without weights	1	167.00	167.00	1.0000	0.5018	1.9927	1.9927	0.3446	0
	With 4 learned weights	3	39.00	14.00	0.9216	0.3479	2.7321	3.3747	0.2587	1.1203
Thyroid	Without weights	38	70.53	61.68	0.9173	0.3542	2.6500	3.0913	0.3119	4.6893
	With 5 learned weights	32	34.19	26.03	0.9196	0.2370	4.4331	13.479	0.2934	4.3005
Bupa	Without weights	42	117.10	107.21	0.9456	0.3626	2.7588	5.3601	0.3203	4.5662
	With 6 learned weights	36	101.40	89.54	0.9599	0.3859	2.9886	5.4428	0.2831	4.0206
MPG	Without weights	15	86.74	67.87	0.9174	0.4496	2.0630	2.4297	0.2983	3.3936
	With 9 learned weights	15	41.53	17.53	0.8986	0.3396	2.6210	3.1715	0.2831	3.2376
Boston	Without weights	25	113.84	89.24	0.9169	0.4736	1.9412	2.0580	0.3046	3.4791
	With 14 learned weights	19	36.48	18.5	0.9176	0.4569	2.0144	2.1731	0.2838	3.3353
Pima	Without weights	36	237.08	218.44	0.9417	0.3730	2.6879	6.2076	0.3183	4.1709
	With 8 learned weights	24	38.59	19.99	0.8966	0.1973	5.0897	17.106	0.1970	3.4962

where $\rho_j(a,b) = |a-b|$ if a and b are real and

$$\rho_j(a,b) = \begin{cases} 1 & if \quad a \neq b \\ 0 & if \quad a = b \end{cases}$$

if a and b are symbols.

Instead of using (1), one can use (1a) to derive a set of equations similar to (2)-(14). For each selected database, the learning weight procedure proposed in Section 3 can be used to obtain the values of the feature weights. The learning rate η for each epoch is determined by (8) and the well-known Fibonacci one dimensional search technique is used [12]. Compared with the fixed learning rate, the one-dimensional search for learning rate can expedite the convergence of the gradient-descent algorithm.

Using the Transitive Closure method of similarity-based clustering, two clustering graphs for each selected database are obtained, with the thresholds ranging from 0 to 1. One graph is generated by using a similarity measure with feature weights and the other is without feature weights (i.e., all weights are equal to 1). For every graph of the given databases, we first count the number of "layers," i.e., the number of partitions and, then, compute the values of evaluation indices given in Section 4. The results are shown in Table 1 the nine indices shown in Table 1 are used to evaluate the performance of a clustering graph. One can make the following observations from Table 1.

1. The proposed feature weight leaning technique can result in an improvement for almost all evaluation indices individually, but not necessarily simultaneously. For example, in the Bupa database, the intrasimilarity and the intersimilarity may not be simultaneously improved due to the interaction that exists between them.
2. Three important evaluation indices, i.e., the ratio of intrasimilarity to intersimilarity, the fuzziness, and the nonspecificity, are improved for all databases. That will reduce or overcome the difficulties (which results from the uncertainty of similarity-based clustering) in the decision—making for choosing a crisp partition.
3. The amount of improvement for these evaluation indices is dependent on the specified database and the specified features. For example, the improvement for the Iris database is more significant than that for the Bupa database. For all databases, the reduction of number of clusters is significant.
4. Overall, Table 1 shows that the clustering graph after feature weight learning is better than that before learning. It verifies the validity and advantages of similarity-based clustering with feature weight learning.

For the eight selected databases, Table 2 provides a brief comparison between the similarity-based clustering and the popular clustering algorithm k-means method [9]. Since k-means method is a kind of distance-based clustering, it can also be regarded as similarity-based clustering method according to (2). By using the Euclidean distance and weighted distance in k-means, we compare the three indices: intrasimilarity, intersimilarity, and their ratio. The cluster number for the k-means method is specified such that it corresponds to the similarity-based one. It is hard to see from Table 2 which method has a better performance. But, one thing is obvious: The learning weight indeed can improve the performance of both Transitive Closure (similarity-based method) and k-means (distance-based method). But, the improvement for Transitive Closure seems to be more significant than for k-means.

It is necessary to discuss the complexity of the feature weight learning algorithm since the performance improvement is at the price of the learning weights. We first analyze the time complexity for each epoch. One part is (5)-(7). We can estimate the times of applications and divisions, which are $O(N^2)$. The other part is the

TABLE 2
A Brief Comparison between Transitive Closure Method and K-Means

Databases	Transitive closure $(\alpha, \beta, \gamma, t)$ without weights	K-means $(\alpha, \beta, \gamma, t)$ without weights	Transitive closure $(\alpha, \beta, \gamma, t)$ with weights	K-means $(\alpha, \beta, \gamma, t)$ with weights
Rice	(0.95, 0.40, 3.00, 002)	(0.77, 0.36, 2.17, 004)	(0.97, 0.39, 3.28, 003)	(0.88, 0.38, 2.34, 006)
Iris	(0.92, 0.46, 2.12, 002)	(0.91, 0.44, 2.06, 006)	(0.85, 0.32, 3.23, 004)	(0.90, 0.39, 2.41, 007)
Servo	(1.00, 0.50, 2.00, 001)	(1.00, 0.50, 2.02, 003)	(0.92, 0.35, 3.38, 004)	(0.91, 0.40, 2.38, 006)
Thyroid	(0.92, 0.36, 3.09, 080)	(0.84, 0.35, 2.38, 079)	(0.92, 0.24, 13.5, 135)	(0.85, 0.38, 2.68, 143)
Bupa	(0.95, 0.36, 5.36, 121)	(0.93, 0.36, 2.58, 136)	(0.96, 0.39, 5.44, 181)	(0.97, 0.37, 2.62, 180)
MPG	(0.92, 0.45, 2.43, 158)	(0.88, 0.41, 2.14, 191)	(0.90, 0.34, 3.17, 251)	(0.90, 0.32, 3.28, 295)
Boston	(0.92, 0.47, 2.06, 185)	(0.89, 0.49, 1.84, 203)	(0.92, 0.46, 2.17, 335)	(0.90, 0.48, 1.88, 362)
Pima	(0.94, 0.37, 6.21, 145)	(0.77, 0.30, 2.55, 158)	(0.90, 0.20, 17.1, 258)	(0.70, 0.19, 8.68, 269)

$(\alpha, \beta, \gamma, t)$ *represents (intrasimilarity, intersimilarity, maximum ration, and the running time [seconds]).*

one-dimensional search indicated by (8). The search algorithm is the Fibonacci method with the time complexity $O(N)$. Thus, the time complexity for each epoch is $O(N^2)$. The time complexity of the entire learning algorithm depends on the convergence of the algorithm. In fact, the learning algorithm we used is the traditional gradient-descent method. Noting that the evaluation function (3) is infinitely differentiable with respect to feature weights $w_1, w_2, \cdots, w_n$, the gradient-descent algorithm with learning rate searched by Fibonacci method is convergent [12]. Our experiments demonstrate this result. It is found in our experiments that the convergence rate of the learning algorithm is very fast. Table 2 indicates the running time for each database.

One problem is the local minimum. The learning algorithm may converge to a local, not the global minimum point. That is a drawback but is unavoidable for any gradient-descent-like algorithm. It is worth pointing out that, even if the feature weight learning algorithm converges to a local minimum point, performance improvements of clustering graphs can still be achieved. These improvements have already been demonstrated in our experiments.

6 Conclusions

When one uses the similarity-based clustering techniques such as the Transitive Closure method to construct a clustering graph and further to make a crisp decision for the clustering graph, it is often mentioned that the graph performance is poor and has much uncertainty (fuzziness and no-specificity). Hence, the decision making based on the clustering graph is rather difficult. This paper proposes that the clustering graph performance can be improved by feature weight learning. The feature weight learning results in a reduction of the uncertainty existing in the clustering process and, therefore, makes the decision making for choosing a crisp partition easier.

References

[1] A. Baraldi and P. Blonda, "A Survey of Fuzzy Clustering Algorithms for Pattern Recognition I," *IEEE Trans. Systems, Man and Cybernetics, Part B,* vol. 29, no. 6, pp. 778-785, 1999.

[2] J. Basak, R.K. De, and S.K. Pal, "Unsupervised Feature Selection Using a Neuro-Fuzzy Approach," *Pattern Recognition Letters,* vol. 19 pp. 997-1006, 1998.

[3] J.C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms.* New York: Plenum Press, 1981.

[4] A. De Luca and S. Termini, "A Definition of a Nonprobabilistic Entropy in the Setting of Fuzzy Set Theory," *Information and Control,* vol. 20, pp. 301-312, 1972.

[5] D. Dubois, *Fuzzy Sets and Systems: Theory and Applications.* New York, Boston: Academic Press, 1980.

[6] D. Dubois and H. Prade, "Three Semantics of Fuzzy Sets," *Fuzzy Sets and Systems,* vol. 90, pp. 141-150, 1997.

[7] R. Fisher, "The Use of Multiple Measurements in Taxonomic Problems," *Ann. Eugenics,* vol. 7, pp. 179-188, 1936.

[8] M. Higashi and G.J. Klir, "Measures on Uncertainty and Information Based on Possibility Distribution," *Int'l J. General Systems,* vol. 9, pp. 43-58, 1983.

[9] A.K. Jain and R.C. Dubes, *Algorithms for Clustering Data.* Englewood, Cliffs, NJ: Prentice-Hall, 1989.

[10] G.J. Klir and T.A. Folger, *Fuzzy Sets, Uncertainty, and Information.* Englewood Cliffs, NJ: Prentice-Hall, 1988.

[11] K. Nozaki, H. Ishibuchi, and H. Tanaka, "A Simple but Powerful Heuristic Method for Generating Fuzzy Rules from Numerical Data," *Fuzzy Sets and Systems,* vol. 86, pp. 251-270, 1997.

[12] S.S. Rao, *Optimization Theory and Applications.* Wiley Eastern Limited, 1985.

[13] R. Duda and P. Hart, *Pattern Classification and Scene Analysis.* New York: Wiley, 1973.

[14] UCI Repository of Machine Learning Databases and Domain Theories. FTP address: ftp://ftp.ics.uci.edu/pub/machine-learning-databases. 1995.

[15] D.S. Yeung and E.C.C. Tsang, "Weighted Fuzzy Production Rules," *Fuzzy Sets and Systems,* vol. 88, pp. 299-313, 1997.

[16] L.A. Zadeh, "Similarity Relations and Fuzzy Orderings," *Information Science,* vol. 3, pp. 177-200, 1971.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** http://computer.org/publications/dilb.