Neural-Fuzzy Approach for Maintaining Case-Bases

Simon C. K. Shiu, X. Z. Wang and Daniel S. Yeung

Abstract. In practical use of case-based systems, there are always changes in the reasoning environment. Overtime, the case library may need to be updated in order to maintain or improve the performance in response to these changes. 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 unacceptably high level. In order to maintain the size of a case-based system as well as preserving its competence, we propose an approach of selecting representative cases using soft computing techniques. The approach consists of three phases. In the first phase, we determine the degree of membership of each case record to different classes using a neural network. This phase will generate a fuzzy set defined on the cluster space for each record. The second phase is to refine these fuzzy sets by a transformation, where the transformed coefficients are determined by gradient-descent technique, such that the degrees of membership can be as crisp as possible. The last phase uses the fuzzy class membership value of each record to formulate the deletion policy. The case density of each record is computed to determine the percentage of record to be deleted. Using this approach, we could maintain a reasonable size of the case-base without loosing significant amount of information.

1. Introduction

Expert system is one of the branches of Artificial Intelligence that has successfully moved from laboratories to real life applications. Among the various expert system paradigms, case-based reasoning (CBR) is a relatively recent technique that is attracting increasing attention. The main reasons of the CBR success are [1][2][3]:

- It is closer to actual human decision processes (i.e. solving the problem by comparing similar situations in the past).
- Automation of the process of incorporating new knowledge in the knowledge base.
- Better explanation and justification of the decisions by showing previous examples.
- It does not require an explicit domain model and so knowledge elicitation becomes a task of gathering case histories.

A CBR system typically consists of four processes:

- Retrieve the most similar case.
- Reuse the case to attempt to solve the problem.
- Revise the proposed solution if necessary.
- Retain the new solution as a part of a new case.

Soft Computing in Case Based Reasoning, (Eds.) Sankar K Pal, Tharam Dillon and Daniel Yeung, Springer-Verlag (London) Ltd,2001, pp.259-273.

When the number of cases increases over time, inductive learning methods could be used to induce general rules from the cases. These rules may be in the form of a decision tree, where each leaf carries a class name, and each inner node specifies an attribute with a branch corresponding to each possible value. In analyzing the similarities among cases that involve uncertainty, fuzzy production rules are used to calculate the membership degrees for each case, or to find similar cases in the case library [4][5][6][7]. When similar cases have accumulated to warrant maintenance, anomalies may exist in the case library, such as redundant cases, conflicting cases, ambiguous cases, subsumed cases and unreachable cases [8][9]. In addition, the performance problems such as degradations of retrieval efficiency will become a real issue if uncontrolled case-base growth is allowed. Techniques that can automatically maintain the size of the case-base as well as detecting problem cases in the case library are therefore crucial to the future success of CBR technologies [10][11][12][13].

Currently, the CBR community has largely ignored the issue of maintenance although CBR is becoming a more mature knowledge-based technology. This research proposes a neural-fuzzy approach for maintaining CBR systems. The approach consists of three phases. In the first phase, we determine the degree of belonging of each case record to different classes using a neural network. This phase will generate a fuzzy set defined on the cluster space for each record. The second phase is to refine these fuzzy sets by a transformation, where the transformed coefficients are determined by gradient-descent technique, such that the degrees of membership can be as crisp as possible. This phase may be skipped according to the degree of fuzziness of training results with respect to classification, i.e., according to the number of cases in ODD-class (ODD-class consists of those cases where their degree of belonging to a class is ambiguous, e.g. 0.65 for class A, and 0.7 for class B etc). The last phase uses the fuzzy class membership value of each record to formulate their deletion policy. The case density of each record is computed to determine the percentage of record to be deleted. Using this approach, we could maintain the size of the case-base without loosing significant amount of information.

A glass classification problem consisting of 214 records is used as an illustration of our approach, the neural network software NEURALWORKS PROFESSIONAL II/PLUS© is used to develop the network. It was shown that it could reduce the size of the case library by 28% if we select those records that have an overall class membership of at least 0.8 and case density over 0.95. Future work will include integrating adaptation rules for building deletion policy. By using this approach, uncontrolled case-base growth can be avoided, hence the performance and retrieval efficiency could be maintained. This paper is organized into five sections. The first section gives the introduction. Section 2 reviews some of the current investigations on applying neural network in case based expert systems. Section 3 describes our methodology in details, and Section 4 uses a public domain database, i.e. glass database donated by Diagnostic Products Corporation, to illustrate our approach, the experimental results are also described and analyzed in this section. Finally, Section 5 provides the conclusions and scope of future work.

2. Literature Review

In traditional CBR systems, case retrieval relies mainly on algorithms such as the nearest neighbor search. It looks for cases stored in memory that consist of the greatest number of characteristics that are the same or similar as the current case. There are many limitations of such an approach:

- Determining which characteristics are more important in retrieval of cases is a difficult task.
- Matching of selected feature is an all or nothing affair.
- It requires a very large case base to cover the entire problem space.
- Features that define a case can be of different types, which must be indexed or represented in different ways.

Recently, fuzzy neural networks are being used for indexing and retrieval of cases. The following is a brief and incomplete review. Vasudevan et al. [14] proposed briefly to use fuzzy logic in case-based reasoning. Main et al. [15] used fuzzy feature vectors and neural network to improve the indexing and retrieval steps in case-based systems. They used a supervised neural network to accept inputs of various formats, such as Boolean, continuous, multi-valued and fuzzy. They have shown that the use of fuzzy representation for some features enhanced the accuracy of retrieval because the cases retrieved tended to match most closely on the fuzzy attributes. Egri and Underwood [16] established a knowledge extraction system, called HILDA which incorporates some aspects of rule-based reasoning and cased based reasoning to assist users in predicting case outcome and generating arguments and making case decisions. The system could use neural network to guide rule based reasoning and case based reasoning in number of ways. Jeng and Liang [17] proposed a technique of fuzzy indexing and retrieval in case-based reasoning. Liu and Yan [18] proposed a fuzzy logic-based neural network to develop a case-based system for diagnosing symptoms in electronic systems. They demonstrated through data obtained from a call-log database that the neural network is able to perform fuzzy AND/OR logic rules and to learn from examples. De and Pal [19] suggested a kind of cased-based classification using fuzziness and neural network.

In addition to the applications of fuzzy neural network to indexing and retrieval of cases, case-base maintenance is considered to be another important issue for cased based expert systems. In [13], Leak and Wilson defined the case base maintenance as the process of refining a CBR system's case-base to improve the system's performance, i.e. case-base maintenance implements policies for revising the organization or contents (representation, domain content, accounting information, or implementation) of the case-base in order to facilitate future reasoning for a particular set of performance objectives. They presented a first attempt at identifying the dimensions of case-base maintenance. In [11], Smyth and McKenna introduced the concept of competence of case-base. They considered that some case could be critical to competence while others may be largely redundant. Based on this idea, they proposed the use of case-based density and the concept of case coverage to determine the maintenance policy.

In this paper, we integrate Smyth's idea [11] with the use of neural networks to obtain a new approach to determine the deletion policy. The main idea is using the fuzzy class membership value of each record, determined by a trained neural network, to guide the record deletion. Each class is considered to contain some representatives and some redundant cases. To reduce the fuzziness of classification, this paper also incorporates the information feature weight. Learning feature weight can be via a gradient-descent technique which has been used by Pal and his colleagues [20]. It aims to acquire features' importance and eliminate irrelevant features in a given database. One important thing in the CBR community is to distinguish the salient features from all the features in the database; feature selection methods can reduce the task's dimensionality when they eliminate irrelevant features [21]. The unsupervised approach in [20] is very useful in dealing with those databases in a knowledge-poor domain and in helping to reduce the dimension size of the case-bases.

3. Methodology

Our case maintenance methodology is divided into three main phases as illustrated in Figure 1. In the first phase, we determine the degree of belonging of each case record to different classes using a neural network. This phase will generate a fuzzy set defined on the cluster space for each record. The second phase is to refine these fuzzy sets by a transformation, where the transformed coefficients are determined by gradient-descent technique, such that the degrees of membership can be as crisp as possible. This phase may be skipped according to the degree of fuzziness of training results with respect to classification, i.e., according to the number of cases in ODD-class. The last phase uses the fuzzy class membership value of each record to formulate their deletion policy. A similarity measure is used to calculate the case density of each record, and a deletion policy is then used to determine the percentage of record to be deleted. In details, we formulate these three phases in the following three subsections.

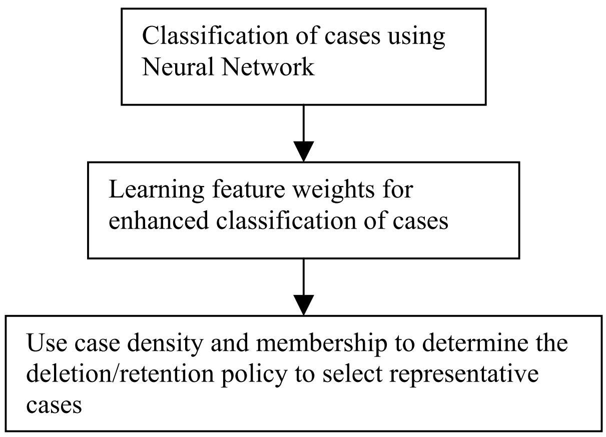

Figure 1. Case-base maintenance steps

Throughout this section, we consider a case library in which all features are supposed to take values of real number. It should be noted that the real-valued features discussed here could be, without difficulties, extended to the features that take values in a normed vector space.

Let $CL = \{e_1, e_2, \cdots, e_N\}$ denote our discussed case library. Each case in the library can be identified by an index of corresponding features. In addition each case has an associated action. More formally we use a collection of features $\{F_j \ (j = 1, \cdots, n)\}$ to index the cases and a variable *V* to denote the action. The *i-th* case e_i in the library can be represented as a *n*+1-dimensional vector, *i.e.* $e_i = (x_{i1}, x_{i2}, \cdots, x_{in}, v_i)$ where x_{ij} corresponds to the value of feature $F_j (1 \le j \le n)$ and v_i corresponds to the action $(i = 1, \cdots, N)$. $v_i \ (i = 1, \cdots, N)$ are considered to be class symbol and the total number of classes is supposed to be *M*. The *M* classes are denoted by $CM = \{C_1, C_2, \cdots, C_M\}$

3. 1. Case classification using neural network

This section aims to model the classification using a neural network with three layers. The classification result will be a fuzzy set on cluster space *CM*. The key structure of the constructed neural network is described as follows.

Input layer: The number of nodes in this layer is equal to the number of features of the case base. Each node represents a feature.

Hidden layer: The number of nodes in this layer is determined according to real applications. Experimentally, the number is bigger than *n* but less than 2*n* where *n* is the number of nodes in input layer.

Output layer: This is the classification layer which contains *M* modes where *M* is the number of clusters. Each node represents a fuzzy cluster. The training result will be the form of fuzzy vector (discrete fuzzy set defined on the cluster space *CM*). The meaning of each output value is the membership value which indicates to what degree the training case belongs to the cluster corresponding to the node.

The popular Sigmoid function is selected as the activation function. According to assumptions, there are *n*, *L*, *M* nodes in the input, the hidden and the output layers respectively. For a given input case (the *m-th* case, $1 \le m \le N$), the forth-propagation process of the input vector is described as follows.

The input layer: $\{x_{mi} \mid i = 1,2,\cdots,n\}$ (the given input vector);

The hidden layer: $y_{mj} = f\left(\sum_{i=1}^{n} u_{ij} x_{mi}\right) \quad j = 1,2,\cdots,L$ (1)

The output layer: $\mu_{mk} = f\left(\sum_{j=1}^{L} v_{jk} y_{mj}\right) \quad k = 1,2,\cdots,M$ (2)

Where u_{ij} and v_{jk} are the connection weights of the neural network, and the notation f represents the Sigmoid function defined as $f(x) = \frac{1}{1+e^{-x}}$.

This is a traditional full connection network with three layers. The standard BP algorithm can be used to train this network. In other words, the popular gradient descent technique can be used to find the values of weights u_{ij} and v_{jk} such that the error function

$$E = \sum_{m=1}^{N}\left(\frac{1}{2}\sum_{k=1}^{M}(\mu_{mk} - c_{mk})^2\right) = \sum_{m=1}^{N} E_m \tag{3}$$

achieves a local minimum, where c_{mk} taking either 0 or 1 corresponds to the action of the *m-th* case, e.g., $(c_{m1}, c_{m2}, \cdots, c_{mM}) = (1,0,\cdots,0)$ if the *m-th* case belongs to the first cluster.

After finishing this training, a fuzzy set on the cluster space $\{C_1, C_2, \cdots, C_M\}$ can be given for each case according to equations (1) and (2). Denoting the fuzzy set by $(\mu_{m1}, \mu_{m2}, \cdots, \mu_{mM})$ in which each component μ_{mj} represents the degree of the *m-th* case belonging to the *j-th* cluster, we can re-classify the case base according to the following criterion (A) or (B).

Consider a case, e_m, to be re-classified. The training result of this case is supposed to be $(\mu_{m1}, \mu_{m2}, \cdots, \mu_{mM})$. α and β are two given thresholds.

Criterion (A): If $Entropy(e_m) < \beta$ and $\mu_{mk} = Max_{1\le j\le M}\,\mu_{mj} \ge \alpha$, then the case e_m is classified to the *k-th* cluster where

$$Entropy(e_m) = -\sum_{j=1}^{M} \mu_{mj} \cdot \ln \mu_{mj} \tag{4}$$

Criterion (B): If $Nonspec(e_m) < \beta$ and $\mu_{mk} = Max_{1\le j\le M}\,\mu_{mj} \ge \alpha$, then the case e_m is classified to the *k-th* cluster where

$$Nonspec\,(e_m) = \sum_{j=1}^{M}\left(\mu^*_{mj} - \mu^*_{m(j+1)}\right)\ln j\ , \tag{5}$$

in which $(\mu^*_{m1}, \mu^*_{m2}, \cdots, \mu^*_{mM})$ is the permutation of $(\mu_{m1}, \mu_{m2}, \cdots, \mu_{mM})$, sorted so that $\mu^*_{mj} \ge \mu^*_{m(j+1)}$ for j =1, 2, …, M and $\mu^*_{m(M+1)} = 0$.

The criterion (A) is based on the fuzzy entropy (equation 4) [22] which will tend to zero when all μ_{mj} tend either 0 or 1. The criterion (B) is based on the non-specificity (equation 5) [13] which will tend to zero when only one μ_{mj} tends 1 and the others tend to 0. The fuzzy entropy and the non-specificity provide two different kinds of uncertainty. The former is suitable probability distribution while the later the possibility distribution.

According to criterion (A) or (B), the case base can be classified to M+1 clusters. The (M+1)-*th* cluster, called ODD class, is the remaining cases which cannot be explicitly classified into certain one of the M classes. The cases in the ODD class have poor training results. Obviously we expect that the number of cases in the ODD class is as small as possible. However, this number depends strongly on the training of the neural network. When the training result (or the number of cases in ODD class) is not desirable, we attempt to improve it by the following feature weight learning.

3. 2. Learning feature weight information

Let the training results be $\{(\mu_{m1}, \mu_{m2}, \cdots, \mu_{mM}), m = 1,2,\cdots, N\}$ for N cases $\{e_m, m = 1,2,\cdots, N\}$ where $\mu_{mk} (\in [0,1])$ represents the degree of the *m-th* case belonging to the *k-th* cluster. An index evaluation function which is similar to one given in [20][24] is defined as

$$E(w_1, w_2, \cdots, w_n) = \sum_{m=1}^{N} \sum_{k=1}^{M} \left(\mu_{mk}^{(w)}(1-\mu_{mk}) + \mu_{mk}(1-\mu_{mk}^{(w)})\right) \tag{6}$$

in which μ_{mj} is computed according to equations (1) and (2), and $\mu_{mj}^{(w)}$ according to the following equations (7) and (8):

$$\mu_{mk}^{(w)} = f\left(\sum_{j=1}^{L} v_{jk} y_{mj}^{(w)}\right) \quad k = 1,2,\cdots, M \tag{7}$$

$$y_{mj}^{(w)} = f\left(\sum_{i=1}^{n} w_i u_{ij} x_{mi}\right) \quad j = 1,2,\cdots, L \tag{8}$$

where u_{ij} and v_{jk} are the weights trained already in the previous phase, and $f(x) = \dfrac{1}{1+e^{-x}}$ represents the Sigmoid function.

In equation (6), $\{w_i : w_i \in [0,1], i = 1,2,\cdots, n\}$ are called feature weights which remain to be determined. They indicate that, for the trained neural network in the previous phase, different features have different degrees of importance to the training-classification.

The evaluation function (6), which has been appllied by Pal [24] and his group to feature exttraction in [20], is designed according to a simple function $g(x, y) = x(1-y) + y(1-x) \quad (1 \le x, y \le 1)$. Noting that $\dfrac{\partial g}{\partial x} = 1-2y$, $\dfrac{\partial g}{\partial x} > 0$

if $y < 0.5$, $\frac{\partial g}{\partial x} < 0$ if $y > 0.5$, one can easily find that equation (6) has the following characteristics:

(a). If $\mu_{mk} < 0.5$ and $\mu_{mk}^{(w)} \to 0$, then $E \to 0$ (minimum);

(b). If $\mu_{mk} > 0.5$ and $\mu_{mk}^{(w)} \to 1$, then $E \to 0$ (minimum);

(c). If $\mu_{mk} = 0.5$ and $\mu_{mk}^{(w)} = 0.5$, then E attains maximum;

The main task of this phase is to minimize the index function $E(w_1, w_2, \cdots, w_n)$ with respect to the weights $w_1, w_2, \cdots, w_n$. More formally, we attempt to find $(w_1^*, w_2^*, \cdots, w_n^*)$ such that

$$E\left(w_1^*, w_2^*, \cdots, w_n^*\right) = Min\left\{E\left(w_1, w_2, \cdots, w_n\right) \mid w_i \in [0,1], i = 1,2,\cdots,n\right\} \tag{9}$$

Minimizing the index function *E* is regarded as a process of refinement of membership values to crispness. If we consider μ_{mk} and $\mu_{mk}^{(w)}$ as the membership degrees of the *m-th* case belonging to the *k-th* cluster before refinement and after refinement respectively, the minimization of equation (6) attempts to make the membership degree after refinement being more crisp than the membership degree before refinement. That is, we expect that the minimization of equation (6) can make the membership degree after refinement being close to 0 if the membership degree before refinement is less than 0.5; and the membership degree after refinement being close to 1 if the membership degree before refinement is bigger than 0.5. It is also expected that the number of cases in the ODD class determined in the previous phase can be reduced by using the new membership degrees.

To solve equation (9), a neural-fuzzy method which is similar to one in [20] can be used. However, for simplicity, we do not design a fuzzy neural network but directly use a gradient-decent technique to minimize equation (6). The change in w_i (i.e. Δw_i) is computed as

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i} \tag{10}$$

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

$$\frac{\partial E}{\partial w_i} = \sum_{m=1}^{N} \sum_{k=1}^{M} \left((1 - 2\mu_{mk}) \frac{\partial \mu_{mk}^{(w)}}{\partial w_i} \right) \tag{11}$$

$$\frac{\partial \mu_{mk}^{(w)}}{\partial w_i} = f\left(\sum_{j=1}^{L} v_{jk} y_{mj}^{(w)}\right)\left(1 - f\left(\sum_{j=1}^{L} v_{jk} y_{mj}^{(w)}\right)\right)\frac{\partial y_{mj}^{(w)}}{\partial w_i} \quad (12)$$

$$\frac{\partial y_{mj}^{(w)}}{\partial w_i} = f\left(\sum_{p=1}^{n} w_p u_{pj} x_{mp}\right)\left(1 - f\left(\sum_{p=1}^{n} w_p u_{pj} x_{mp}\right)\right)\cdot u_{ij} x_{mi} \quad (13)$$

where $f(x) = \dfrac{1}{1+e^{-x}}$ represents the Sigmoid function and other notations have the same meaning as the equations (6) – (8).

The training algorithm is described as follows.

Step 1. Select the learning rate η .

Step 2. Initialize w_i with random values in [0, 1].

Step 3. Compute Δw_i for each i using equation (10).

Step 4. Update w_i with $w_i + \Delta w_i$ for each i if $w_i + \Delta w_i \in$ [0, 1].

Step 5. Repeat step 3 and step 4 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 $E(w_1, w_2, \cdots, w_n)$ attains a local minimum. We expect that, in average, the membership degrees $\{\mu_{mk}^{(w)} : m = 1, \cdots, N; k = 1, \cdots, M\}$ with trained weights are closer to 0 or 1 than $\{\mu_{mk} : m = 1, \cdots, N; k = 1, \cdots, M\}$ without trained weights.

3. 3. Determine the deletion policy

According to the membership degrees obtained by training a neural network in phase 1 or after refining them in phase 2, the case base can be classified to *M*+1 clusters by using criterion (A) or (B) mentioned previously. The (*M*+1)-*th* cluster is the ODD class in which the cases cannot be explicitly classified into certain one of the *M* classes. This phase is based on such an idea that there exist several representatives in each class of cases. Therefore, the aim of this phase is to select these representatives. Our selection strategy is mainly based on the case-density and membership degree. Before introducing our selection strategy, we briefly and incompletely review some related works for case deletion.

In [11], Smyth and Keane described a technique for measuring the local coverage of individual cases with respect to a system's retrieval and adaptation characteristics. They also suggested deleting cases based on their coverage and reachability. The coverage of a case is defined as the set of target cases that can be adapted to solve, while the reachability of a case is the set of cases in the case-base that can be adapted to solve that case. Based on these measures, Smyth et al. classified cases within the case-base into four groups: Pivotal (if its reachability is a singleton consisting of itself), Auxiliary (if its coverage is subsumed by the coverage of a case to which it is reachable), Spanning (if its coverage space links regions covered by other cases) and Support (groups of cases having the same coverage). The deletion policy (footprint policy) suggested by Smyth et al. [11] was to delete auxiliary cases first, then support cases, then spanning cases and finally pivotal cases. If more than one case is a candidate for deletion, sub-strategy is formulated when deciding on which case to delete. Two problems of the footprint deletion policy are that the coverage and reachability of a case are dependent on the adaptation knowledge available, and secondly, it is unclear why a pivotal case would ever need to be deleted [23].

Rather than using a deletion strategy, we prepare to select several representatives in each class of cases. This selection makes use of the membership degree and the concept of case density [11], defined as follows:

$$\mathrm{CaseDensity(c,G)} = \frac{\sum_{C' \in G-\{c\}} sim(c,c')}{|G|-1} \tag{14}$$

$$\mathrm{Sim(c,c')} = 1 - \frac{[\sum_i w_i (c_i - c_i')^2]^{\frac{1}{2}}}{n} \tag{15}$$

in which G denotes a class of cases, |G| is the number of cases in class G, c and $\mathrm{c'}$ are two cases, c_i and c_i' are the *i-th* component of the cases c and $\mathrm{c'}$ respectively, n is the number of features, w_i is the feature weight learned in phase 2 (all w_i will be equal to 1 if phase 2 is skipped), and $\mathrm{Sim(c,c')}$ represents the similarity between cases c and $\mathrm{c'}$.

Our selection algorithm is described as follows:
Step 1. Select two thresholds α and β for membership value and case density.
Step 2. Compute case density for every case in each class except for ODD class.
Step 3. Select all cases in ODD class.
Step 4. Select representative cases from every other class if their membership degrees (of belonging this class) are greater than or equal to α and their case densities are greater than or equal to β.

4. Glass Identification Databbase

In order to illustrate the effectiveness of our approach, we apply it to a glass identification problem. The glass identification database downloaded from UML [25], as shown in Table 1, consists of 214 records, and each record has 11 attributes as follows:

1. Id number: 1 to 214
2. RI: refractive index
3. Na: Sodium (unit measurement is weight percent in corresponding oxide, as are attributes 4 to 10)
4. Mg: Magnesium
5. Al: Aluminum
6. Si: Silicon
7. K: Potassium
8. Ca: Calcium
9. Ba: Barium
10. Fe: Iron
11. Type of glass: window glass / non-window glass

Table 1 Sample Cases

RI	Na	Mg	Al	Si	K	Ca	Ba	Fe	Type
1.51215	12.99	3.47	1.12	72.98	0.62	8.35	0	0.31	W
1.51768	12.56	3.52	1.43	73.15	0.57	8.54	0	0	W
1.51652	13.56	3.57	1.47	72.45	0.64	7.96	0	0	W
1.51969	12.64	0	1.65	73.75	0.38	11.53	0	0	N
1.51754	13.39	3.66	1.19	72.79	0.57	8.27	0	0.11	W
1.51911	13.9	3.73	1.18	72.12	0.06	8.89	0	0	W

We used the neural network package NERUALWORKS PROFESSIONAL II/PLUS© version 5.3 for testing various types of neural network's performance on fuzzy classification of data. We have tried the following network models: back-propagation model, probabilistic model and SOM model. All of them are suitable for our purpose, and we have finally chosen the back-propagation network to perform detail analysis of the case base. The learning rule algorithm is delta-rule, and the transfer function is a sigmoid function. After about 50,000 cycles of training, the network converged and the RMS error was 0.1322, which was considered to be successfully trained.

Furthermore, the confusion matrix graph shown in Figure 2 represents the following results: the x-axis representing the desired output and the y-axis representing the actual output. The confusion matrix breaks the diagram into a grid. If the probe point produce an output of 0.7 and the desired output was 0.5, then the bin around the intersection of 0.7 from the y-axis and 0.5 from the x-axis receives a count. A bar within the bin displays counts, and the bar grows as counts accumulated. The bin that received the most counts is shown at full height, while all of the other bins are scaled in relation to it. The confusion matrix is also equipped with a pair of histograms.

The histogram that runs across the top of the instrument shows the distribution of the desired outputs. The histogram along the right shows the distribution of the actual outputs. Any actual outputs that lay outside the range of the graph will be added to the top or bottom bins along the right (depending on their value). By looking at the two confusion matrix, the desired outputs and actual outputs intercepts quite well and this also indicates that the network was trained satisfactorily.

We then tested the network by the original set of data, and only accept correct classification if the fuzzy membership value is higher than 0.8, even with such a high membership degree, the overall accuracy of correct classification is 94%. We expected that it would go up to as high as 99% if more tolerance of fuzziness were allowed. The typical output after training is shown in Table 2 and the network architecture is shown in Figure 2.

Table 2 Sample output from training

Expected Value 1=Window Glass 0=Non Windows Glass		Fuzzy membership for Window Glass	Fuzzy Membership for Non Window Glass
1	0	0.999998	0.000002
1	0	0.997225	0.002775
1	0	0.99802	0.00198
0	1	0.007992	0.992008
1	0	0.999834	0.000166
1	0	0.999047	0.000954
0	1	0.457469	0.542531
1	0	0.999737	0.000263

We select those records having WINDOW-GLASS membership value higher than 0.8 and label them as WINDOW-GLASS class and those records having a NON-WINDOW-GLASS membership value higher than 0.8 and label them as NON-WINDOW-GLASS class. The remaining records are labeled as ODD class, i.e. they neither belong to WINDOW-GLASS nor NON-WINDOW-GLASS to a satisfactory degree. The results are as follows:

Table 3 Classification of the records

Class Name	**No. of cases in this class**
WINDOW-GLASS	158
NON-WINDOW-GLASS	43
ODD(the remaining)	13

Noting that the number of cases in the ODD class is 13, we consider that the feature weight learning is unnecessary (i.e., the phase 2 can be skipped). For each class, we could use the similarity measurement and case density calculation to compute every case's case density according to equations (14) and (15) where the feature weights are considered to be 1, as shown in Table 4.

Table 4 Sample Result of Case density

Fuzzy membership for Window Glass	Fuzzy Membership for Non Window Glass	Class we defined according to fuzzy membership value	Case density
0.999834	0.000166	Window Glass	0.963236
0.999047	0.000954	Window Glass	0.959489
0.999737	0.000263	Window Glass	0.964335
0.412578	0.587422	Not Sure	0.940094
0.999975	0.000025	Window Glass	0.939792
0.999985	0.000015	Window Glass	0.939759
0.995205	0.004795	Window Glass	0.9396
0.000004	0.999996	Non Window Glass	0.938816
0.999962	0.000038	Window Glass	0.93853
0.000023	0.999977	Non Window Glass	0.938372

From the result, in WINDOW-CLASS class, we select those cases whose case density is greater than 0.95 and has a WINDOW-CLASS membership value greater than 0.95 as the representative cases, and delete the others in this class. Similarly, in NON-WINDOW-GLASS class, we select those cases whose case density is greater than 0.95 and has a NON-WINDOW-GLASS membership value greater than 0.95 as the representative cases, and delete the others in this class. In addition, we retain all the cases in ODD class. We could also use the case density to select some cases as representative cases in the ODD class as well. (In this experiment, we choose to retain all the cases in the ODD class). The final case maintenance result is as follows:

Table 5 Case Maintenance Result

Class Name	No. of cases originally	No. of cases remain
WINDOW-GLASS	158	112
NON-WINDOW-GLASS	43	29
ODD class	13	13
TOTAL	214	154

Using the above approach, we could delete 60 cases, having a 28% decrease in the size of the case base.

5. Conclusions

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 retrieval efficiency encountered. In this paper, we have developed an approach of maintaining the size of a case-based expert system. This involves computation of the fuzzy class membership values of each record determined by a trained neural network. The case base is classified into several clusters according to the membership values and each cluster is considered to contain some representative cases and several redundant cases. A similarity measure is used to calculate the case density of each record. A deletion strategy is then used to determine to percentage of records to be deleted. A testing Glass/Non-Glass case-base consisting of 214 records is used as an illustration, and NEURALWORKS PROFESSIONAL II/PLUS© is used for implementation. It was shown that we could reduce the size of the case library by 28% by selecting representative cases that have an overall class membership of over 0.8 and case density of over 0.95. Future work includes extension of the fuzzy feature selection concepts for identifying important case features for ranking, development of a neural network with unsupervised learning to determine the class membership, integration of data mining techniques, such as discovering of adaptation rules, for guiding the deletion policy, and use of a classifier to validate the results.

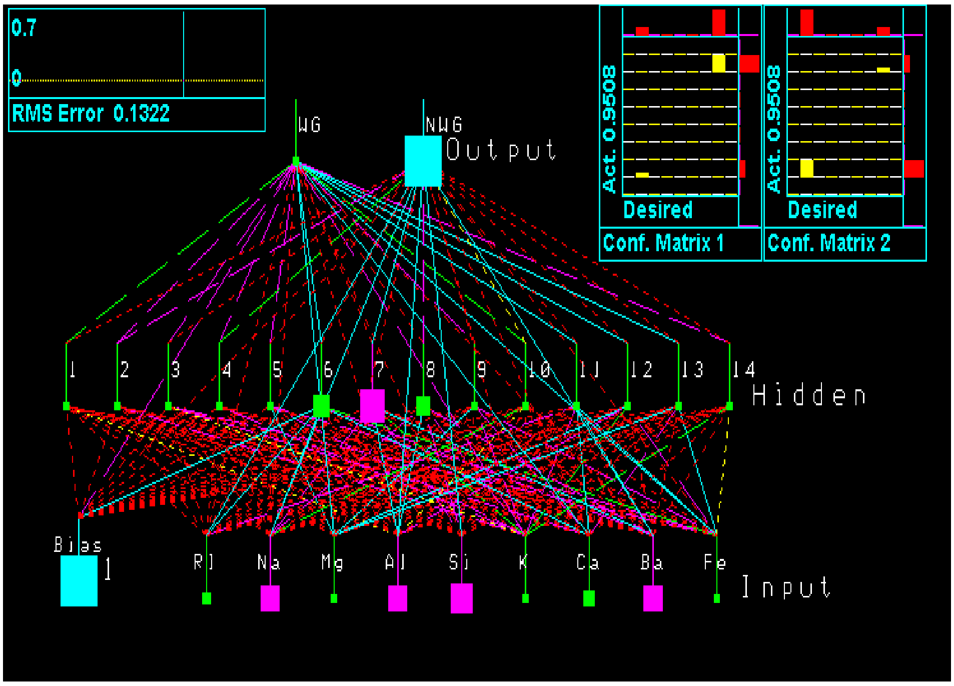

Figure 2. Neural Network Architecture for classification

6. Acknowledgement

This project is supported by a Hong Kong Polytechnic University grant PA25

7. References

[1] Ketler, K., "Case-Based Reasoning: An Introduction," Expert Systems With Applications, Vol. 6, pp. 3-8, 1993.

[2] Marir, F. & Watson, I., "Case-Based Reasoning: A Categorised Bibliography," The Knowledge Engineering Review, Vol. 9:4, 1994, pp.355-381.

[3] Watson, I. & Marir, F., "Case-Based Reasoning: A Review," The Knowledge Engineering Review, Vol. 9:4, 1994, pp.327-354.

[4] Dubitzky W. et al., "An Advanced Case-knowledge Architecture Based on Fuzzy Objects," Applied Intelligence, Vol. 7, No.3, July, 1997, pp. 187-204.

[5] Petersen, J., "Similarity of fuzzy data in a case-based fuzzy system in anaesthesia," Fuzzy Sets and Systems, Vol. 85 (1997), pp. 247-262.

[6] Yeung, D.S. & Tsang, E.C.C., "Improved Fuzzy Knowledge Representation and Rule Evaluation Using Fuzzy Petri Nets and Degree of Subsethood," International J. of Intelligent Systems, Vol. 9, pp.1083-1100, 1994.

[7] Yeung, D.S. & Tsang, E.C.C., "A Comparative Study on Similarity-Based Fuzzy Reasoning Methods," IEEE Transactions on SMC, Vol. 27, No. 2, April, 1997, pp. 216-227.

[8] O'Leary, D.E., "Verification and Validation of Case-Based Systems," Expert Systems With Applications, Vol. 6, pp.57-66, 1993.

[9] Shiu, S.C.K. "Formal Description Techniques for the Verification of Expert Systems," PhD Thesis, Department of Computing, HK Polytechnic University, 1997.

[10] Shiu, S.C.K., Tsang, E.C.C., Yeung, D.S., "Maintaining Case-Based Expert Systems Using Fuzzy Neural Network," in Proc. of 1999 IEEE SMC, Vol. III, pp.424-428.

[11] Smyth, B. and McKenna E., "Modelling the Competence of Case-Bases," in Proceedings of 4th European Workshop, EXCBR-98, pp.207-220.

[12] Heister F. and Wilke W., "An Architecture for Maintaining Case-based Reasoning Systems," in Proceedings of EWCBR-98, pp.221-232.

[13] Higashi, M. and Klir, G. J., “Measures on uncertainty and information based on possibility distribution,” International J. of General Systems, vol. 9, pp43-58, 1983

[13] Leake, D. B. and Wilson, D. C., "Categorizing Case-Base Maintenance: Dimensions and Directions," in Proceedings of 4th European Workshop, EXCBR-98, pp.196-207.

[14] Vasudevan, C., Smith, S.M. and Ganesan, K., "Fuzzy Logic in Case-Based Reasoning," NASA Joint Technology Workshop on Neural Network and Fuzzy Logic, NAFIPS/IFIS/NASA'94, pp.301-302

[15] Main, J., Dillon T.S. and Khosla R. "Use of Neural Networks for Case-Retrieval in a System For Fashion Shoe Design," in Proceedings of Eight International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, Melbourne, Australia, June 1995, pp.151-158.

[16] Egri, P. A. and Underwood, P. F., "HILDA: Knowledge Extraction From Neural Networks in Legal Rule Based and Case Based Reasoning," in IEEE Int. Conf. on Neural Networks, 1995, Vol.4. pp.1800-1805.

[17] Jeng, B.C. & Liang, T.P., "Fuzzy Indexing and Retrieval in Case-Based Systems," Expert Systems With Applications, Vol. 8, No. 1, pp.135-142, 1995.

[18] Liu, Zhi Qiang and Yan, Francis, "Fuzzy Neural Network in Case-Based Diagnostic System," IEEE Trans on Fuzzy Systems, Vol.5., No.2., May, 1997.

[19] De Rajat K. and Pal Sankar K., "Case-based Classification using Fuzziness and Neural Networks," Knowledge Discovery and Data Mining (Digest No. 1998/310), IEE Colloquium, 1998, pp. 6/1-6/3.

[20] Basak, J., De, R. K. and.Pal, S.K., Unsupervised feature selection using a neuro-fuzzy approach, Pattern Recognition Letters 19 (1998), pp998-1006,1998.

[21] Wettscherck, D. and Aha, D.W., Weighting Features, Case-based Reasoning Research and Development, First International Conference, ICCBR-95, Sesimbra, Portugal, pp347-358, 1995.

[22] De Luca, A. and Termin, S., "A definition of a nonprobabilistic entropy in the setting of fuzzy set theory," Inform. and Control, vol. 20, pp301-312, 1972
[23] Anand, S. S., Patterson, D., Hughes, J.G. , Bell, D.A., Discovering Case Knowledge using Data Mining, Second Pacific Asia Conference, PAKDD-98, Australia, pp25-35, 1998.
[24] Pal S.K. and Mitra S., Neuro-Fuzzy Pattern Recognition: Methods in Soft Computing, Wiley, New York, 1999.
[25] Glass Identification Database, donated by Diagnostic Products Corporation. UML machine learning repository database.