Ning Xiong, Peter Funk. Building similarity metrics reflecting utility in case-based reasoning. Journal of Intelligent & Fuzzy Systems, IOS Press, pp. 407-416.

Building Similarity Metrics Reflecting Utility in Case-Based Reasoning

Ning Xiong and Peter Funk
Department of Computer Science and Engineering
Mälardalen University
SE-72123 Västerås, Sweden
{ning.xiong, peter.funk@mdh.se}

Abstract

Fundamental to case-based reasoning is the idea that similar problems have similar solutions. The meaning of the concept of "similarity" can vary in different situations and remains an issue. Since we want to identify and retrieve truly useful or relevant cases for problem solving, the metrics of similarity must be defined suitably to reflect the utility of cases for solving a particular target problem. A framework for utility-oriented similarity modeling is developed in this paper. The main idea is to exploit a case library to obtain adequate samples of utility from pairs of cases. The task of similarity modeling then becomes the customization of the parameters in a similarity metric to minimize the discrepancy between the assessed similarity values and the utility scores desired. A new structure for similarity metrics is introduced which enables the encoding of single feature impacts and more competent approximation of case utility. Preliminary experimental results have shown that the proposed approach can be used for learning with a surprisingly small case base without the risk of over-fitting and that it yields stable system performance with variations in the threshold selected for case retrieval.

-

1. Introduction

Case-based reasoning (CBR) [1] attempts to solve problems via analogy with problems previously solved. The underlying assumption is that similar problems have similar solutions and therefore it appears to be logical to consider previous similar problems and their solutions in coping with a new situation. As previous case data are utilized directly, CBR eases the knowledge acquisition bottleneck and facilitates learning from experiences without inducing an explicit knowledge model.

However, to be able to perform various CBR tasks successfully, the notion of similarity needs to be further clarified. The key concern here is to determine which kinds of cases should be considered as "similar" given a new situation. As our purpose is to retrieve truly useful or relevant cases for ultimately solving a new target problem, primary measurements such as Hamming distance, Euclidean distance and Canberra metric are insufficient as they merely consider synthetical differences between the entities to be compared. Suitable semantics of similarity must be defined in terms the specific application scenario in consideration. The goal is to ensure that the cases assessed as similar are those most useful in offering qualified or adaptable solutions to the new problem. Utility oriented similarity modeling has become a significant trend for advanced CBR research [2].

So far the main stream of the works involving similarity models has been focused on feature weighting [17]. Features are assigned with different weights in accordance with their importance, and the global similarity metric is defined as a weighted sum of the matching values in single attributes. Different approaches of interest have been proposed for identifying such weights automatically. Incremental learning attempts to modify feature weights according to success/failure feedback of retrieval results [3, 14]. Probability-based techniques, addressed in [7] and [5], utilized the conditional probabilities of classes and the probability of ranking principle respectively in the assignment of weight values to features. Case-ranking information was utilized in [4, 6,15] for weight adaptation towards similarity degrees of retrieved cases consistent with a desired order. Accuracy improvement represents another way for determining the set of weights as discussed in [11] and [13]. Nevertheless, the capability of these similarity learning methods for utility approximation is inherently constrained by their similarity structure, which excludes the

possibility of yielding a similarity assessment other than a weighted average of the local matching degrees in individual attributes.

This objective of this paper is to promote more flexible and accurate similarity assessments capturing the utility of cases with respect to a target problem. A new similarity metric is suggested which encodes single feature impacts on case utility into suitable local matching functions. The identification of parameters for a metric as such is realized by resorting to the case library. The basic idea is that the case library, with its known solutions for all the cases it contains, is an indispensable resource for providing adequate samples about the different appropriateness of solutions in various situations. With these samples at hand, similarity modeling can then be transformed to the task of parameter updating in the structured metric to imitate the desired utility values in outcomes. Preliminary case studies in data classification have shown that our proposed approach can be employed for learning with surprisingly small case bases without the risk of over-fitting and that it yields stable system performance against variations in the threshold selected for case retrieval.

The paper is organized as follows. In the next section we highlight the role of similarity assessments in a CBR cycle. A new structure for similarity metric is introduced in section 3, followed by a framework for learning concrete similarity metrics in section 4. Section 5 then presents experimental results for evaluation. Relevant previous works are outlined in section 6 and finally section 7 contains our conclusion and remarks.

2. Case-Based Reasoning and Similarity Assessments

An overview of the procedure for case-based problem solving is depicted in Fig. 1. Given a target problem P we look for its relevant counterparts in the case library. The matching of target P and a known case C in the library is guided by a similarity metric, which will be addressed in the following section. A library case is retrieved as long as its similarity value relative to P is not below a threshold, say α. As the result we get a set R of retrieved cases as:

$$R = \{C \in CL \,|\, Sim(P, C) \geq \alpha\} \tag{1}$$

where *CL* is the case library and *Sim*(*P*, *C*) denotes the degree of similarity of *C* respect to *P*. The elements in *R* along with their similarity scores are delivered to the block "decision fusion" for finalizing the solution to *P*.

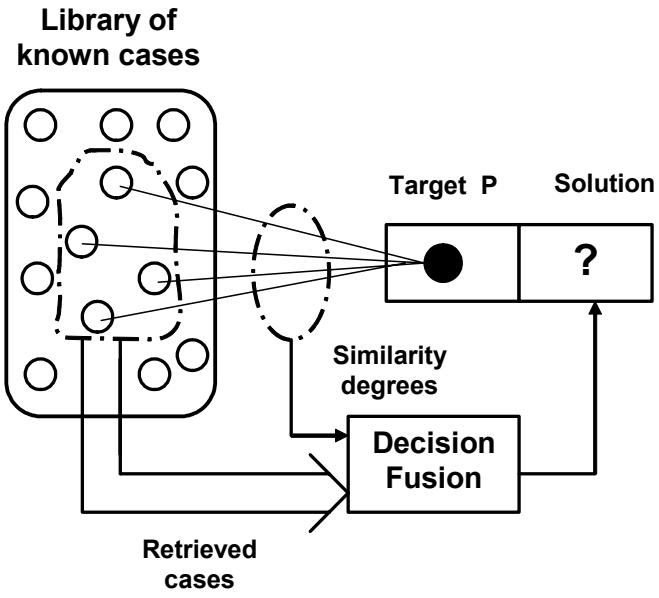

Fig. 1 An overview of case-based problem solving

The solutions of the retrieved cases are aggregated in the decision fusion step, the similarities of cases being regarded as estimates of the utility or appropriateness of their solutions for solving the new problem. Thus cases with higher similarity degrees will have more influence in determining the final solutions. For instance, for numerical prediction problems, the outcome of the target problem *P* is predicted to be a weighted average of the outcomes of the retrieved cases as given by

$$Out(P) = \frac{\sum_{C_i \in R} Sim(P, C_i) \cdot Out(C_i)}{\sum_{C_i \in R} Sim(P, C_i)} \tag{2}$$

where $Out(P)$ are $Out(C_i)$ are the output values for target P and case C_i respectively. Should the problem be a classification one, we need to launch a voting procedure to choose the most possible class from a set of candidates. The values of similarity of the retrieved cases that have the same outcome can be accumulated into a voting score (VS) for the associated class. In general, the voting score for a candidate class B is calculated by

$$VS(B) = \sum_{C_i \in R} \begin{cases} Sim(P, C_i), & if \quad Class(C_i) = B \\ 0, & otherwise \end{cases} \tag{3}$$

Finally we select the class with the largest voting score as the estimated class for target P, i.e.,

$$Class(P) = \arg\max[VS(B)] \tag{4}$$

At this point we can clearly see that similarity matching plays a central role in both case retrieval and decision fusion in a CBR task. Concretely speaking, designing accurate matching functions for similarity assessments is paramount in the following three aspects:

(i) A suitable similarity metric is required to retrieve the most relevant or useful cases.

(ii) Precise similarity assessments on retrieved cases are crucial for the final outcomes of decision fusion, as exemplified in (2) and (3).

(iii) Precise similarity assessments on all individual cases help to make system performance more stable, less sensitive to the threshold α used in (1) or the number of cases to be retrieved.

3. A New Similarity Metric for Utility Approximation

This section aims to introduce a new similarity metric with thoughts about utility. The point of departure is assuming that the appropriateness of the solution of a known case to solve a new problem is dependent upon the difference between the problem and the case's condition part. The less distinct between both, the more usable is the case solution. This motivates us to consider differences in values of attributes when addressing similarity assessments for estimating utility.

As basic notations for the rest of the paper, we suppose that there are n relevant features in the underlying domain. A case C_i in the case base is indexed by an (n+1) tuple: $C_i = (x_{i1}, x_{i2}, \cdots, x_{in}, s_i)$ where $x_{i1}, x_{i2}, \cdots, x_{in}$ denote the attribute values in this case and s_i is the corresponding solution. Similarly we use an n-tuple $(y_1, y_2, \cdots, y_n)$ to represent a target problem P with y_j referring to the value of the jth attribute in the problem. All attribute values in both the library cases and the target problem are normalized for further analysis.

The task now is to compare the condition part of case C_i and the target problem P to obtain a similarity assessment to estimate the utility for solution s_i. The key role herein is taken by the difference of values $d_{ij} = y_j - x_{ij}$ in single attributes in the sense that every difference as such contributes more or less to a degradation of the utility. In the following, we begin by discussing local matching functions taking feature differences as arguments, and then in the aggregation part we present a method that combines local matching values into a global similarity degree.

3.1 Compatibility Measure on a Single Attribute

At the first step we perform matching on single attributes to see how the values in the condition part of the case are compatible with their counterparts in the target problem. The extent of compatibility is assessed by a compatibility measure m_{ij} for each attribute. Intuitively the value of m_{ij} depends on the feature difference d_{ij}: it has its maximum value of unity with identical feature values. It is otherwise reduced by a magnitude approximately proportional to the feature difference. This leads to modeling of the compatibility measure m_{ij} by a triangular function as:

$$m_{ij}(d_{ij}) = \begin{cases} (d_{ij} + w_{1,j})/w_{1,j}, & if \quad d_{ij} \in (-w_{1,j}, 0] \\ 0, & if \quad d_{ij} \notin (-w_{1,j}, w_{2,j}) \\ 1 - d_{ij}/w_{2,j}, & if \quad d_{ij} \in (0, w_{2,j}) \end{cases} \tag{5}$$

where $w_{1,j}$ and $w_{2,j}$ are function parameters that must be specified appropriately in advance.

-

It is worth noting that there is no absolute/independent compatibility criterion for an attribute and any measure for this must be defined relative to a specific application domain. The parameters $w_{1,j}$ and $w_{2,j}$ in equation (5) reflect the information about the importance of the associated attribute. This is illustrated in Figs. 2a, 2b, and 2c where a variation of compatibility measures is created by three distinct sets of parameters. Fig. 2a shows a compatibility measure characterized by $w_{1,j}=w_{2,j}=1$, which means that the value of compatibility becomes zero only when the largest feature difference occurs. Reducing the values for both $w_{1,j}$ and $w_{2,j}$ to less than unity, we arrive at the matching function in Fig. 2b which can be considered as related to some more critical attribute. The measure in that figure decreases more quickly with feature differences and thus reaches a zero degree earlier than the function in Fig. 2a. In contrast, a situation with a relatively weak attribute can be modeled by setting the parameters greater than unity as shown in Fig. 2c. The function drawn there appears less sensitive to the feature difference and remains non-zero even when the largest difference is encountered.

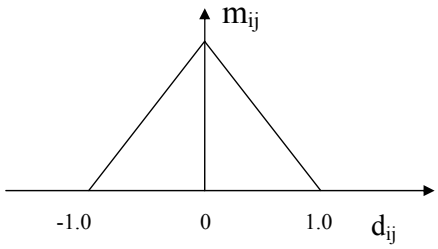

Fig. 2a. The compatibility measure with its parameters set to unity

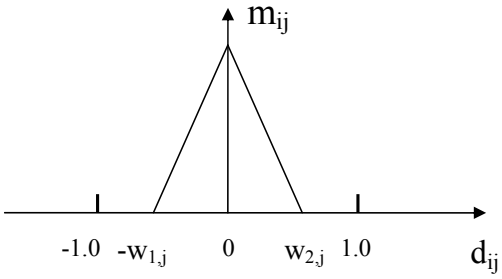

Fig. 2b. The compatibility measure with its parameters less than unity

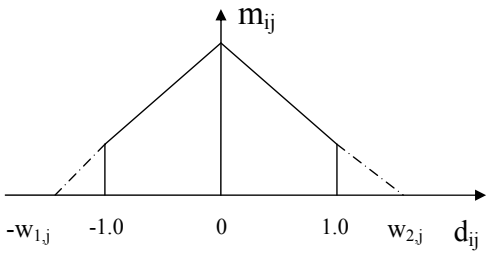

Fig. 2c. The compatibility measure with its parameters greater than unity

In addition, although the compatibility functions described above are explicitly targeted to handle numerical attributes, they are also applicable to symbolic attributes as long as the discrete symbolic values can be ordered on a scale or numerical quantities assigned for each.

3.2 Global Similarity in View of Multi-criteria Satisfaction

After performing feature matching on every relevant feature between case C_i and target problem P, we get a collection of compatibility values (m_{i1}, m_{i2}, …, m_{in}). One can consider that the need of compatibility in an individual attribute is one criterion to be satisfied for the case to be relevant for the target. In particular, the score m_{ij} can be seen as a degree of satisfaction of the criterion G_j:

> *The jth attribute in the target is compatible with the jth attribute in the library case*

The next step is to aggregate the satisfaction degrees for those criteria in single attributes to yield an overall similarity (utility) assessment.

Since the similarity between a case and the target problem is contingent upon the compatibility in each of the attributes, satisfying all the criteria G_j (j=1…n) is required for the case to be identified as relevant or usable. This leads to expressing the global similarity metric as equivalent to a logical statement as follows:

$$Sim(P, C_i) = G_1 \text{ and } G_2 \text{ and } G_3 \text{ and } \ldots \text{ and } G_n \qquad (6)$$

-

The logical *and* is used in the above to make connections between the individual criteria. An established tool for implementing this connective with multi-valued logic is the t-norm [9]. The algebraic product is employed herein as a concrete form for the t-norm, and thus we obtain the similarity assessment as an overall satisfaction of the criteria by

$$Sim(P, C_i) = m_{i1}(d_{i1}) \times m_{i2}(d_{i2}) \times \cdots \times m_{in}(d_{in}) \tag{7}$$

Defining the similarity metric as a t-norm as specified by (7) can be justified by considering CBR for system modeling and prediction in an industrial environment. The outcome of an underlying process is determined therein by a set of feature variables as inputs. It follows that the accuracy of using a known output from a library case to predict the outcome in a new situation is fully dependent on the differences in all inputs. A highly usable case must possess compatible input values in each dimension in order to be selected, and a difference in one dimension too large would suffice to invalidate a candidate case in predicting process outcomes.

An advantage of the similarity metrics structured above is that they are quite easy to be customized to approximate an arbitrary degree of utility. The way to do this is to adapt the parameters $w_{1,j}$ and $w_{2,j}$, in accordance with local compatibility measures, given a set of feature differences, and so that the global similarity score appears to be very close to the desired utility value. As a simple extreme, if the known utility is zero, we need to set only one parameter ($w_{1,j}$ or $w_{2,j}$ depending on the sign of the feature difference) small enough to reach a zero compatibility in the corresponding attribute. In the other extreme, should a highest utility estimate be preferred, excellent compatibility in all the attributes must be enforced and this can be achieved by assigning sufficiently large values to the related parameters in each local compatibility measure.

Finally we would like to add that, although the utility of a case demands all of its features to be compatible, the semantics of compatibility varies from attribute to attribute as indicated by the parameters $w_{1,j}$ and $w_{2,j}$ in (5). Individual attributes are enabled to impose different influences on the assessed similarity because the compatibility measures utilized in (7) are subject to variable semantic meanings.

-

4. Utility Oriented Similarity Modeling

With the structure of similarity having been proposed in the preceding section, we now turn to discussing the identification of the parameters to build a concrete similarity metric. Our aim is to elicit similarity assessments that can be regarded as reliable and precise estimates of utility. This means that we wish for the relation $Sim(P, C_i) = Utility(P, s_i)$ for any target problem P from the domain and any case C_i from the case base, with s_i denoting the solution to C_i. For reducing the discrepancy between assessed similarity and domain-decided utility, we need the involvement of a learning mechanism to revise the similarity metric. This is illustrated in Fig. 2 in which a modeling algorithm is tasked to adapt the parameters of the similarity metric using approximation errors as the feedback.

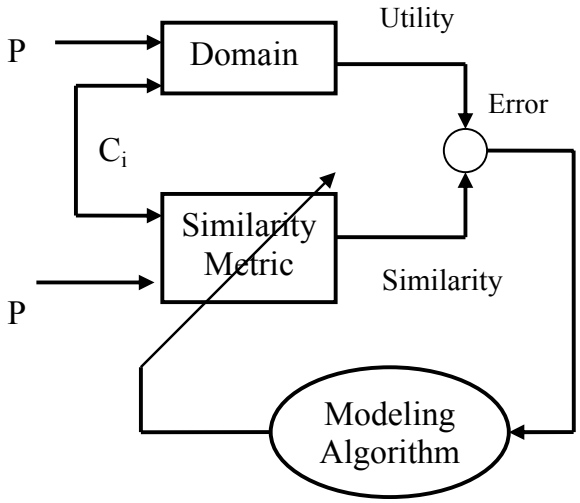

Fig. 3. Revising the similarity metric to approximate utility

However a difficulty arises when implementing the general perspective depicted in Fig. 2 for similarity modeling. The reason is that the utility of case C_i with respect to problem P is determined by the application domain but generally is unknown a priori, due to the unavailability of adequate domain knowledge. As a result no correct feedback can be acquired from examining

approximation errors, which disables the proper function of the employed modeling algorithm.

To circumvent the difficulty stated above we suggest narrowing down the scope of *P* in Fig. 2 to the case base. The advantage of doing so is that, since the solution of *P* is now known from the case library, it is a straight forward matter to derive the utility of other cases by comparing their solutions with that of *P*. In addition, presuming the case library as covering adequate typical cases, it is also justified to focus on the case library rather than the whole application domain in the similarity modeling. A feasible means of revising the similarity metric based on known cases is illustrated in Fig. 3, where the problem *P* in Fig. 2 has been replaced by a library case C_j and the utility between cases is derived as will be detailed in the following subsection.

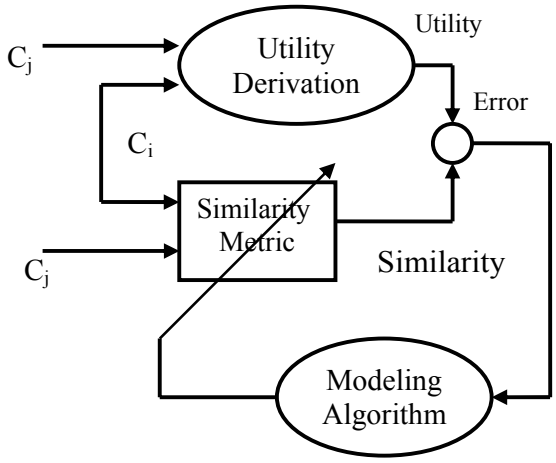

Fig. 4. Revising the similarity metric using known cases

4.1 Deriving Utility Values between Cases

Utility values between library cases can be derived directly by only matching their known solutions. Principally, owing to the reflexivity property of similarity, all case solutions stored in the case library are assumed to be optimal. Our point is that, given two cases C_i and C_j, the utility of C_i with respect to C_j is determined by the relation between their optimal solutions, s_i

and s_j respectively. The closer the similarity of solution s_i is to solution s_j, the more adaptable is solution s_i for problem solving in case C_j and the less expensive it is to adapt. In view of this, we define the utility between cases as equivalent to the similarity between their solutions. Thus we can write:

$$Utility(C_j, C_i) = Sim(s_j, s_i) \tag{8}$$

The criterion of similarity between solutions is usually domain dependent, and thus we cannot further concretize equation (8) without considering problem context and specifics. Nevertheless, for some common CBR applications such as classification and numerical prediction, the following measurements, which we consider to be reasonable functions under their respective circumstances, can be recommended.

1. In classification problems with symbolic classes without orders, the similarity between classes can be defined by a binary function as

$$Sim(s_j, s_i) = \begin{cases} 1 & if \quad s_j = s_i \\ 0 & if \quad s_j \neq s_i \end{cases} \tag{9}$$

2. In classification problems with symbolic classes having ordinal values, the similarity between classes should reflect the relative distance in the order:

$$Sim(s_j, s_i) = \begin{cases} 1 & if \quad s_j = s_i \\ 1 - \dfrac{h(s_j, s_i)}{K} & if \quad s_j \neq s_i \end{cases} \tag{10}$$

where K is the total number of classes and $h(s_j, s_i)$ denotes the number of classes between s_j and s_i in the order.

3. For problems of numerical predictions, the similarity between outputs could be defined only measuring syntactical differences such that

$$Sim(s_j, s_i) = 1 - \frac{\left| s_j - s_i \right|}{\max\limits_k \{s_k\} - \min\limits_k \{s_k\}} \tag{11}$$

-

Once the derivation of utility between known cases is established, we can collect many sampled utility values by working on pairs of cases from the case library. Next we will show that these sampled values can be utilized as training patterns to build a competent similarity metric.

4.2 Similarity Learning Based on Utility Samples

The utility derivation between known cases enables us to acquire many sample utility values given pairs of cases in the case library. These derived samples can then be treated as training examples for the similarity metric to mimic. The task of the learning algorithm in Fig. 5 is to adapt the parameters in the similarity metric to minimize the differences between the utility values specified by the training patterns and the similarity degrees assessed by the similarity model.

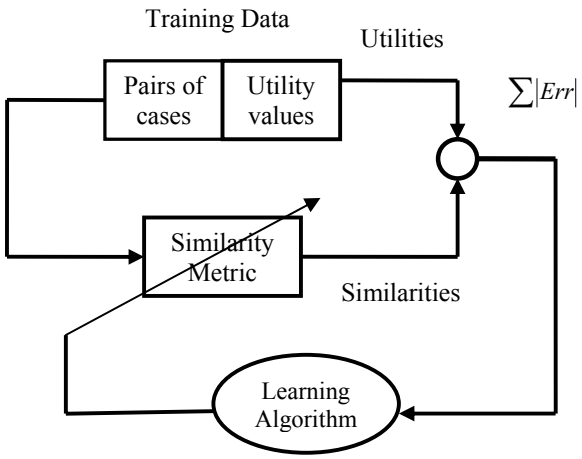

Fig. 5. Learning of similarity based on training patterns

As many pairs of cases are included in the training data set, we must consider the total sum of the modeling errors in order to improve the overall accuracy for utility approximation. The total error function is given by

$$TE(\mathrm{w}_{1,1}, w_{2,1}, \cdots, w_{1,n}, w_{2,n}) = \sum_{(i,j)\in SI} \left| utility(C_j, C_i) - Sim(C_j, C_i) \right| \tag{12}$$

and it supplies feedback information to the learning procedure in Fig. 5. By *SI* in (12) we denote the set of pairs of case indexes corresponding to the pairs of cases included in the training data set. Should we only focus on classification and diagnosis, this general error function in (12) can be specialized into

$$TE(\mathrm{w}_{1,1}, w_{2,1}, \cdots, w_{1,n}, w_{2,n}) = \sum_{(i,j)\in SI} \begin{cases} 1 - Sim(C_j, C_i) & if \quad s_j = s_i \\ Sim(C_j, C_i) & if \quad s_j \neq s_i \end{cases} \tag{13}$$

and $$TE(\mathrm{w}_{1,1}, w_{2,1}, \cdots, w_{1,n}, w_{2,n}) = \sum_{(i,j)\in SI} \begin{cases} 1 - Sim(C_j, C_i), & if \quad s_j = s_i \\ \left| \left(1 - \dfrac{h(s_j, s_i)}{K}\right) - Sim(C_j, C_i) \right| & if \quad s_j \neq s_i \end{cases} \tag{14}$$

which are suitable for treating non-ordered classes and classes with ordinal values respectively.

Any effective method may be chosen as the learning algorithm in the framework. Certain mathematical optimization methods such as simplex or conjugate gradient algorithms are feasible options in cases with a small number of attributes. However, to scale up to high dimensional environments, we prefer more powerful and robust learning algorithms for successful searching in more complex spaces. We consider that the genetic algorithm outlined in the following offers a strong alternative for this purpose.

4.3 Learning of Similarity Metrics by GA

A Genetic Algorithm (GA) is an effective tool for searching and learning based on the principle of natural selection and evolution [10]. Compared with other traditional optimization techniques, GAs demonstrate their strength in the global exploration of complex, ill-structured problem spaces and require no differential information of the objective functions. Moreover, the property of convergence ensured by GAs is of considerable benefit for the similarity learning in our framework.

The GA customized in this paper is mainly based upon a standard version introduced in [10] but uses strings of real numbers rather than binary strings in

the population. Our algorithm for optimizing a set of parameters to minimize the total error function in (12) consists of the following steps:

Step 0 (Initialization): Generate an initial population containing N_{pop} strings of real-valued numbers with N_{pop} being the population size. In this stage each element in a string is assigned a positive real number stochastically, using an appropriate distribution function.

Step 1 (Initial evaluation): Apply every similarity metric created initially to all the pairs of cases in the case base to get assessed similarity degrees which can then be compared with the desired utility values to yield an overall error quantity according to (12).

Step 2 (Selection): Select $0.5N_{pop}$ pairs of strings from the current population. The selection probability $prob(S_r)$ for a string S_r in a population ψ is specified as:

$$prob(S_r) = \frac{TE_{\max}(\psi) - TE(S_r)}{\sum_{S_r \in \psi} \{TE_{\max}(\psi) - TE(S_r)\}} \tag{15}$$

where
$$TE_{\max}(\psi) = \max\left\{TE(S_r) \middle| S_r \in \psi\right\} \tag{16}$$

Step 3 (Crossover): For every pair of parents (strings of real numbers) to be combined, choose one or more breakpoints at random. At the breakpoints the parent elements are passed on to the offspring alternatively. This means that the offspring get elements from one of the parents until a breakpoint is encountered, at which they switch and take elements from the other parent. This operation is subject to a crossover probability approximately 0.867.

Step 4 (Mutation): Each element of the child strings generated by the crossover operation undergoes a disturbance with its magnitude determined by a Gaussian distribution function. Every mutated element less than zero must be set back to its lower limit to retain its physical significance.

Step 5 (Selective breeding): Employ the error function (12) to evaluate each child string in the offspring set. After this, select the best N_{pop} individuals from the current population and the offspring set to form the next generation.

Step 6 (Termination test): If a pre-specified generation number has not then been reached, go to Step 2, otherwise terminate the search procedure and return

the best individual in the population as the solution for the similarity metric to be modeled.

5. Experimental Evaluations

We have applied our proposed approach to the problems of classification and diagnosis. In this section some experimental results obtained from a real world problem of wine data classification will be presented. The wine data set is downloadable from the address: ftp.ics.uci.edu/pub/machine-learning-databases. It consists of 178 samples of three classes, the condition part of every sample being depicted by 13 continuous attributes. We are ignorant of the order of the three wine classes

In this wine problem, a feasible presumption is that the compatibility in each attribute is symmetric and consequently the global similarity metric is also symmetric. Owing to the symmetry property each compatibility measure for single attributes can be characterized by only one parameter, i. e., $w_{1,j} = w_{2,j} = w_j$ for all j. If we further take into account all pairs of library cases as training patterns, the total error function in this scenario can be rewritten as:

$$TE(w_1, w_2, \cdots, w_{13}) = \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} \begin{cases} 1 - Sim(C_j, C_i) & if \quad s_j = s_i \\ Sim(C_j, C_i) & if \quad s_j \neq s_i \end{cases} \tag{17}$$

where M is the size of the case base. Experiments in similarity modeling were performed for the proposed similarity metric in (7). Corresponding to the calculation of the product in (7), the threshold α for case retrieval was intuitively defined in our tests as $\alpha = \beta^{13}$ with $\beta \in [0,\ 1)$.

5.1 Learning Ability on the Case Library

To examine the learning ability of our method, all the 178 instances in the data set were treated as a case library. GA-based similarity modeling was then performed using utility patterns derived from all case pairs. The quality of the similarity metric learned was examined by applying it to classification of individual cases as targets. In table 1 we show the "leave-one-out" accuracy of the learnt metric on the case library against different values of β for the

threshold of case retrieval. It is seen clearly from the table that the classification accuracy remains rather stable under threshold variations. This is because precise similarity assessments are enforced globally on all library cases such that the decision fusion procedure remains robust against such changes in the threshold.

Table 1: Classification accuracy on the whole data set as the case base

The value of β	Leave-one-out accuracy
0.0	93.2584%
0.1	93.2584%
0.2	93.2584%
0.3	93.2584%
0.4	93.2584%
0.5	93.2584%
0.6	93.2584%
0.7	93.2584%
0.8	93.2584%
0.9	92.6966%

5.2 Generalization Ability on Test Cases

To test the generalization ability of our similarity modeling method on problems that are not included in the case library, experiments were also made with division of the whole data set into two subsets: one as the case base used for learning and the other as the test data set assumed to contain target problems. The similarity metric learnt with the case base was applied to classification of problems in the test set. Preliminary studies as shown in the following reveal that our modeling algorithm conveys generalized results, it can survive with even very small case libraries without the risk of over-fitting.

In Table 2 we illustrate the classification accuracy of the learnt similarity metrics on the test data in a 10-fold cross-validation, for which the entire wine data was partitioned into 10 parts and only one part was used as the case base for learning and the remaining nine parts were taken as the test data in each of the ten trials. Surprisingly, despite the extremely small sized case bases with 17-18 instances in each, acceptable accuracy was achieved

by the learned similarity metrics when applied to the test data. This can be attributed to the pair-wise comparison of cases in the case library, which produces multiplication of training patterns for the utility oriented similarity modeling.

Table 2: Classification accuracy on the test data in a 10-fold cross-validation

Number of Trials	**Best accuracy β∈{0, 0.1, …0.9}**	**Worst Accuracy β∈{0, 0.1, … 0.9}**
1	88.1988%	87.5776 %
2	85.6250%	79.3750%
3	72.5000%	69.3750%
4	81.2500%	81.2500%
5	91.2500%	88.1250%
6	91.2500 %	89.3750%
7	91.2500 %	90.6250%
8	95.0000%	94.3750%
9	86.2500%	86.2500%
10	90.6832%	84.4720%
Average	87.3257%	85.0800%

5.3 Regarding Conventional Similarity Function

Traditionally, the global similarity function has taken the form of a weighted sum of local matching values as defined by:

$$Sim(P, C_i) = \sum_{j=1}^{n} W_j (1 - |d_{ij}|) \tag{18}$$

where W_j are the feature weights with $W_j \in [0, 1]$ and $\sum_{j=1}^{n} W_j = 1$. Could this similarity function be customized to approximate case utility as has been done with the proposed similarity metric?

Our answer is negative. This is because the weighted average in (18) guarantees that its outcomes remain between the minimum and the maximum of the local matching degrees and therefore we cannot reach anything beyond that range no matter whatever weights are assigned to features. Especially in the

case of $|d_{ij}| = 0.5$ for all j, the global similarity degree is fixed at 0.5 regardless of the weights assigned. In [18] we showed that updating the weights in (18) to imitate the desired utility in the wine problem led to both large modeling errors and low classification accuracy.

Another way is to adapt the feature weights for direct improvement of classification accuracy. Weights were identified by GA to maximize the leave-one-out accuracy for the entire wine data and a high performance of 98.3146% was achieved under the threshold $\alpha = 0.75$. However, such accuracy quickly decreased to 75.2809% and 43.2584% when α was set to 0.7 and 0.6 respectively. This indicates the high sensitivity of the system performance in relation to the threshold deviations.

6. Related Works

As has been stated in the introduction, feature weighting appears to be a common means of building similarity metrics in CBR. Since a set of weights assigned is at least partially responsible for the problem solving performance, methods have been proposed which attempt to update the values of weights using system performance as feedback [8, 11, 12, 13]. However such methods, with the exclusive goal of performance/accuracy optimization, do not account for the utility of cases directly and as a consequence the improvement of similarity assessments cannot be enforced down to the level for individual target-case pairs. Besides, weighting through performance feedback could make results dependent on the adaptation procedure employed, which challenges the potential meaning of derived weights for features.

Investigations into similarity modeling were also made using information about the rank of retrieved cases as the learning signal [4, 6, 15, 16]. The works in [4] and [6] aimed at adapting personal feature weights to reflect customer preferences in E-commerce. Case order feedback was utilized in [15] and [16] for attaining degrees of similarity of the retrieved cases consistent with a desired order. Unfortunately, the learning of similarity based on case order may be subject to two disadvantages. Firstly, the learning signal only contains comparative statements and thus lacks the quantitative information required for precise utility estimations. Secondly, the rank information on the retrieved

cases is partial by giving no attention to other cases not retrieved. It follows that the learning algorithm cannot recognize and respond to such circumstance if a highly relevant case is not retrieved due to an ill-defined similarity metric.

Reinforcement learning [3, 14] presents an any-time algorithm to update feature weights according to the feedback from a case retrieval. The idea is to increase the weights on matching attributes while decrease the weights on non-matching attributes if the retrieval is successful, otherwise the opposite is done in the weight updating. This can be considered as an incremental procedure given successive test problems and we conjecture that the convergence of learning results would require a relatively long process.

7. Conclusion

This paper aims to promote more flexible and accurate similarity assessments capturing the utility of cases. The contribution of the paper is two-fold. Firstly, a general framework for utility-oriented similarity modeling is developed. The basic idea is to exploit the case library to derive adequate samples of utility from pairs of cases. The task of similarity modeling then turns to the customization of the parameters in a similarity metric in order to minimize the discrepancy between assessed similarity values and utility scores desired. Secondly a new similarity structure is proposed. We have explained that this new structure of similarity approximates utility more competently than traditional similarity functions. Moreover, the learned similarity model can be regarded as a kind of knowledge container in the sense that it encodes information about the importance of individual attributes in its local compatibility measures.

In summary, the novel features offered by the presented framework of similarity modeling include:

- The enforcement of direct minimization of utility estimation errors on individual cases, leading to more accurate similarity assessments. In many application scenarios, precise similarity values play a crucial role for distinguishing cases quantitatively and also for decision fusion from relevant cases.
- The possibility of deriving sufficient utility samples for similarity modeling. The way to achieve this is pair-wise comparison of cases from the case base

such that a multiplication of utility samples can be created from cases available. This is a very attractive property, reducing the risk of over-fitting, particularly when the case library is very small.

- The implementation of global similarity metrics without weights. The point is that information about the importance of features is already contained in the local compatibility measures and as a result feature weighting is no longer necessary. Our analysis also reveals that the similarity model structured in this paper performs utility approximation more competently than traditional schemes using weights.

References

[1] A. Aamodt and E. Plaza, Case-based reasoning: foundational issues, methodological variations, and system approaches, *Artificial Intelligence Com.* 7 (1994), 39-59.

[2] R. Bergmann, M. Richter, S. Schmitt, A. Stahl and I. Vollrath, Utility-oriented matching: A new research direction for case-based reasoning, in: Proceedings of *The German Conference on Professional Knowledge Management*, 2001, pp. 264-274.

[3] A. Bonzano, P. Cunningham and B. Smith, Using introspective learning to improve retrieval in CBR: A case study in air traffic control, in: Proceedings of *The 2nd International Conference on Case-based Reasoning*, Providence RI, USA, 1997, pp. 291-302.

[4] K. Branting, Acquiring customer preferences from return-set selections, in: Proceedings of *The 4th International Conference on Case-Based Reasoning*, 2001, pp. 59-73.

[5] N. Cercone, A. An and C. Chan, Rule-induction and case-based reasoning: Hybrid architectures appear advantageous, *IEEE Trans. Knowledge and Data Engineering* 11 (1999), 166-174.

[6] L. Coyle and P. Cunningham, Improving recommendation ranking by learning personal feature weights, in: Proceedings of *The 7th European Conference on Case-Based Reasoning*, 2004, pp. 560-572.

[7] R. H. Creecy, B. M. Masand, S. J. Smith and D. J. Waltz, Trading MIPS and memory for knowledge engineering, *Communications of the ACM* 35 (1992), 48-64.

[8] W. Dubitzky and F. Azuaje, A genetic algorithm and growing cell structure approach to learning case retrieval structures, in: *Soft Computing in Case Based Reasoning*, S. K. Pal, T. S. Dillon and D. S. Yeung, eds, Springer, 2001, pp. 115-146.

[9] D. Dubois and H. Prade, A review of fuzzy sets aggregation connectives, *Information Sciences* 36 (1985), 85-121.

[10] D. E. Goldberg, *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley, New York, 1989.

[11] J. Jarmulak, S. Craw and R. Rowe, Genetic algorithms to optimize CBR retrieval, in: Proceedings of *The European Workshop on Case-Based Reasoning* (EWCBR 2000), 2000, pp. 136-147.

[12] R. Kohavi, P. Langley and Y. Yun, The utility of feature weighting in nearest neighbor algorithms, in: Proceedings of *The European Conference on Machine Learning* (ECML-97), 1997.

[13] D. Lowe, Similarity metric learning for a variable-kernel classifier, *Neural Computation* 7 (1995), 72-85.

[14] F. Ricci and P. Avesani, Learning a local similarity metric for case-based reasoning, in: Proceedings of *The International Conference on Case-Based Reasoning* (ICCBR-95), Sesimbra, Portugal, Oct. 23-26, 1995.

[15] A. Stahl, Learning feature weights from case order feedback, in: Proceedings of *The 4th International Conference on Case-Based Reasoning*, 2001, pp. 502-516.

[16] A. Stahl and T. Gabel, Using evolution programs to learn local similarity measures, in: Proceedings of *The 5th International Conference on Case-Based Reasoning*, 2003, pp. 537-551.

[17] D. Wettschereck and D. Aha, Weighting features, in: Proceedings of *The 1st International Conference on Case-based Reasoning*, 1995, pp. 347-358.

[18] N. Xiong and P. Funk, Learning similarity measures for improved utility assessment, in: Proceedings of *The Annual Swedish Artificial Intelligence and Learning Systems Event*, Västerås, 12-14 Apr. 2005, pp. 177-191.