Combined Feature Selection and Similarity Modelling in Case-Based Reasoning Using Hierarchical Memetic Algorithm

Ning Xiong, Peter Funk

***Abstract*—This paper proposes a new approach to discover knowledge about key features together with their degrees of importance in the context of case-based reasoning. A hierarchical memetic algorithm is designed for this purpose to search for the best feature subsets and similarity models at the same time. The objective of the memetic search is to optimize the possibility distributions derived for individual cases in the case library under a leave-one-out procedure. The information about the importance of selected features is revealed from the magnitudes of parameters of the learned similarity model. The effectiveness of the proposed approach has been shown by evaluation results on the benchmark data sets from the UCI repository and in comparisons with other machine learning techniques.**

I. INTRODUCTION

Feature selection is attaining increasing importance in many research areas such as system modelling, pattern classification, and machine learning [1-3]. Progress in data storage and database technologies has led to the availability of huge data sets with large numbers of variables/features in countless practical situations. However, not all presented features need to be taken into account in the modelling procedure. Some features may be irrelevant, redundant, or contaminated by heavy noise. Feature selection aims to identify a subset of key attributes from an initial set of candidates by exploiting the information in the given data set. It leads to improved performance of learning, lower input dimensionality, reduced computational cost, as well as better understanding of the systems in consideration.

Existing approaches to feature selection can be divided into two categories, called filter and wrapper respectively. Filter approaches [4-5] try to assess feature goodness as an intrinsic property independent of the modelling algorithm. Usually they attempt to detect possible dependency between a pair of variables based on the given data. But individual features are evaluated in isolation of each other without considering the influence of others. In contrast, wrapper approaches [6-8] use a modelling algorithm to evaluate feature subsets in terms of the modelling accuracy. Hence they may yield less prediction error than filtering approaches. The weakness of the wrapper approach is that, since a group of features is assessed as a whole, we can't distinguish different levels of importance among the features that are selected.

Indeed knowledge about feature importance plays a crucial role in building similarity models for case-based reasoning (CBR) [9]. As noted in [10], a competent similarity model should function as a knowledge container by encoding domain knowledge. So far the mainstream of the works involving similarity models has been focused on feature weighting [11-12]. The method is to assign a numerical weight to every case feature in accordance with its importance, and thereby different features will have different influences (with different weights) in similarity matching between cases. Many interesting approaches have been developed to adjust parameters of similarity models automatically, including incremental learning [13-14], probability-based estimation [15-16], adaptation in terms of case-ranking [17-18], utility approximation [19], as well as accuracy optimization [20-21]. However, none of the aforementioned methods includes feature selection in similarity modelling, which makes it hard ´for CBR systems to tackle large databases with many irrelevant and noisy data.

In this paper we propose a new approach to discovering the knowledge about key features together with their degrees of importance in the context of case-based reasoning. A hierarchical memetic algorithm is designed for this purpose to search for the best feature subsets and similarity models at the same time. The objective of the memetic search is to optimize the possibility distributions derived for individual cases in the case library under a leave-one-out procedure. The information about importance of selected features is revealed from the magnitude of parameters of the learned similarity model. The proposed method is superior to the filter approach by considering the combined effects of features in assessing feature significance. Moreover, our new approach is also computationally more efficient than the wrapper approach in that it does not require the procedure of similarity modelling to evaluate a candidate subset of features.

The paper is organized as follows. Section II gives a general perspective of CBR. In section III we introduce the structure of the similarity model addressed in this paper. Then, in section IV, a memetic algorithm approach is discussed for selecting features and optimizing similarity models simultaneously within a CBR framework. The evaluation results are presented in section V, and finally concluding remarks are given in section VI.

II. CASE-BASED REASONING: A GENERAL PARADIGM

Case-Based Reasoning (CBR) attempts to solve new problems via analogy with previously solved ones. The underlying tenet is that similar problems have similar solutions. Based on this principle, the case-based approach

Authors are with the School of Innovation, Design, and Engineering, Mälardalen University, Västerås, SE-72123 Sweden (phone: +46-21-151716; fax: +46-21-103110; e-mail: ning.xiong@mdh.se, peter.funk@mdh.se).

exploits information of previous similar cases in solving a new problem. A general CBR paradigm addressed in this paper is shown in Fig. 1. It starts with similarity matching between a query problem and known cases in the case library. A properly defined similarity function has to be employed at this stage. As the evaluated similarity values reflect the utility or appropriateness of solutions of the known cases, they offer important information to be utilized in the next step of decision fusion to figure out the final solution for the problem in query.

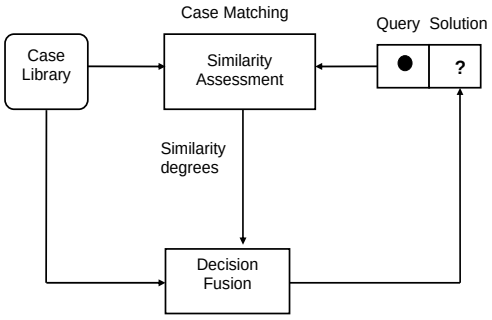

Fig. 1. An overview of case-based reasoning

In decision fusion, we are concerned with possibilities of candidate solutions for a query problem given the cases in the case library. We follow the inference rule that "The more similar the two cases are, the greater possibility there is that their solutions are similar" [22]. Further, solutions of cases are represented by discrete and mutually exclusive labels in the context of this paper. We define the degree of possibility contributed by a single case C_i (from the case library) by

$$P_i(b) = \begin{cases} Sim(Q, C_i) & if \quad Label(C_i) = b \\ 0 & if \quad Label(C_i) \neq b \end{cases} \tag{1}$$

where b represents a candidate label, and $Sim(Q, C_i)$ denotes the degree of similarity between query problem Q and case C_i. It bears mentioning that the possibility in (1) indeed represents a degree of confirmation, which is supported by the observation that case C_i has a label identical to b. More specifically, we will have $P_i(b)=0$ if C_i has a label different from b, whereas it merely means that no support information for label b is derived from case C_i rather than the impossibility of b as the solution to query Q.

Next we consider the overall possibility distribution in light of the whole case library. For calculating the overall possibility $Poss(b)$ for label b, we only need to focus on a subset of cases which have that label. This is owing to the fact that all other cases in the case library contribute no information for the possibility of label b, as indicated in Eq. (1). In principle, $Poss(b)$ should be established as a disjunctive combination of the possibility estimates derived from the individual cases belonging to this case subset. In fuzzy set theory, the disjunctive combination can be done by using the t-conorms. In view of this, the overall possibility distribution $Poss(b)$ is formulated with a general logical operator (t-conorm) as follows:

$$Poss(b) = \bigoplus_{i \in S_b} P_i(b) \tag{2}$$

where $S_b = \{ i \mid Label(C_i) = b \}$ denotes the set of indices of the cases having label b.

Note that a t-conorm is a mapping $\oplus : [0, 1]^2 \mapsto [0, 1]$ which is commutative, associative, monotone increasing with both arguments, as well as satisfies the boundary conditions: $x \oplus 0 = x$ and $x \oplus 1 = 1$. In this paper, we employ the operator: $\oplus(x, y) = x + y - x \cdot y$ as the implementation of the t-conorm in deriving the possibility distribution for candidate labels. Other common forms of t-conorms include the max-operator and bounded summation $\oplus(x, y) = \min[1, x + y]$.

Finally we select the label b^* that has the largest possibility value from the total possibility distribution as the estimated label for query Q, i.e.,

$$b^* = \arg\max_{\forall b} [Poss(b)] \tag{3}$$

It is clear from equations (1) to (3) that similarity degrees play a central role in possibility assessment and thereby exert crucial influence on the final outcomes. Creating competent similarity assessment involves both feature selection and construction of similarity models based on relevant features. Later in this paper we will discuss how feature selection and similarity modelling can be conducted simultaneously via hierarchical memetic algorithms.

III. The Structure of Similarity Models

This section introduces the structure of similarity models that are employed in this paper for similarity modelling and identifying feature importance. The point of departure is the assumption that the similarity of a known case with respect to a new problem is dependent upon the difference between the problem and the case's condition part, both described with relevant features. The less distinction between both, the more similarity is to be expected. This motivates us to consider differences in values on relevant features when addressing similarity assessment.

We suppose that n relevant features have been identified for 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 feature values in this case and s_i is the corresponding label for the case. Similarly we use an n-tuple $(y_1, y_2, \cdots, y_n)$ to represent a query problem Q with y_j referring to the value of the jth feature in the problem description. All feature values in both the library cases and the query problem are normalized for further analysis.

The task now is to compare the condition part of case C_i and the query problem Q to obtain a similarity assessment.

The key role herein is taken by the difference of values $d_{ij} = y_j - x_{ij}$ in single features in the sense that every difference as such contributes more or less to a degradation of the similarity. In the following, we start with local matching functions taking (relevant) feature differences as arguments, and then we present a method that combines local matching values into a global similarity degree.

A. Compatibility Measure on Single Features

At the first step we perform matching on single features to see how the values in the condition part of the case are compatible with their counterparts in the query problem. The degree of compatibility is assessed by a compatibility measure m_{ij} for each feature. 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 modelling of the compatibility measure m_{ij} by a triangular function as:

$$m_{ij}(d_{ij}) = \begin{cases} (d_{ij} + w_j)/w_j, & if \quad d_{ij} \in (-w_j, 0] \\ 0, & if \quad d_{ij} \notin (-w_j, w_j) \\ 1 - d_{ij}/w_j, & if \quad d_{ij} \in (0, w_j) \end{cases} \qquad (4)$$

where w_j is the function parameter that must be specified appropriately in advance.

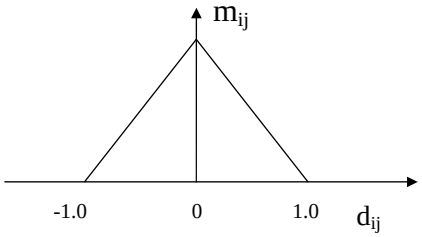

Fig. 2a. The compatibility measure with its parameter set to unity

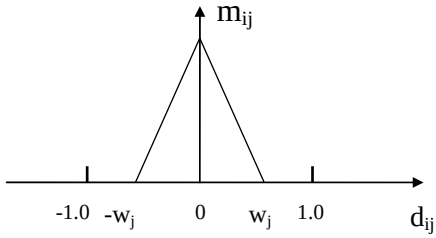

Fig. 2b. The compatibility measure with its parameter less than unity

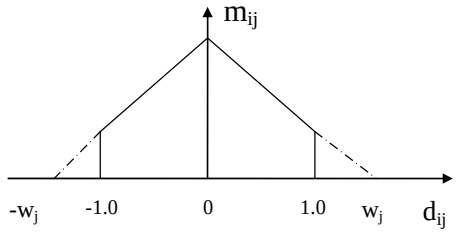

Fig. 2c. The compatibility measure with its parameter greater than unity

It is worth noting that the parameter w_j in equation (4) reflects the information about the importance of the associated feature. This is illustrated in Figs. 2a, 2b, and 2c respectively, where a variation of compatibility measures is created by three distinct parameter values. Fig. 2a shows a compatibility measure characterized by w_j =1, which means that the value of compatibility becomes zero only when the largest feature difference occurs. Reducing the value for w_j to less than unity, we arrive at the matching function in Fig. 2b which can be considered as related to some more critical feature. The function 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 feature can be modelled by setting the parameter 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.

B. Global Similarity in View of Multi-Criteria Satisfaction

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

The jth feature in the case is compatible with the jth feature in the query problem

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

Since the similarity between a case and the query problem is contingent upon the compatibility in each of the relevant features, satisfying all the criteria G_j (j=1…n) is required for the case to be evaluated as similar. This leads to expressing the global similarity criterion as equivalent to a logical statement as follows:

$$Sim(Q, C_i) = G_1 \text{ and } G_2 \text{ and } G_3 \text{ and } \ldots \text{ and } G_n \qquad (5)$$

The t-norm in fuzzy set theory can be applied here to implement the logical *and* connections between the individual criteria. In this paper we use the MIN operator as a concrete form for the t-norm, thus we obtain the similarity assessment as an overall satisfaction of the criteria by

$$Sim(Q, C_i) = \min_{j=1}^{n} [m_{ij}(d_{ij})] \qquad (6)$$

Finally we would like to add that, although the similarity of a case demands its compatibility on all relevant features with the query problem, the semantics of compatibility varies from feature to feature as indicated by the parameter w_j in (4). Different features are allowed to impose different influences on the assessed similarity because the

compatibility measures utilized in (6) are subject to variable semantic meanings.

IV. SIMILARITY LEARNING VIA HIERARCHICAL MEMETIC ALGORITHM

Here we mean by similarity learning a process consisting of two tasks. The first task is selection of a group of relevant features out of candidates, while the second task is creation of a concrete similarity model (identifying parameters of the compatibility functions) using the selected features as inputs. Owing to the inherent interplay between the above two tasks, it is not desired to perform them sequentially. Instead we develop a hierarchical memetic algorithm to enable the conduction of these two tasks simultaneously. The objective of the memetic algorithm is to optimize the combined effect of selected features and parameters of the compatibility functions from a global perspective.

Memetic algorithms (MAs) [23] are population-based metaheuristic search methods inspired by the principle of natural evolution and Dawkin's notion of memes capable of local adaptation. MAs work very similarly as GAs [24] but embed local search to allow for self-refinements of individuals. According to the idea of Lamarckian learning [25], local search can be done on all or part of the population to reach a local optimum or improve the current solution. Next we discuss the issues of coding scheme, fitness evaluation, local search mechanism, as well as genetic operators (crossover and mutation), which present key components for our memetic-based similarity learning.

Coding Scheme. We need to define a chromosome to encode individual solutions including both selected features and parameters of compatibility functions on features. Since the information about feature subset and the parameters of compatibility functions are on different levels, we suggest a hierarchical formulation of chromosomes for the MA. A chromosome as such consists of two different kinds of genes: control genes and parametric genes. A control gene at the higher level takes a binary bit and corresponds to a feature candidate, with bit "1" signifying that the corresponding feature is relevant and selected whereas bit "0" meaning that the feature is irrelevant and discarded. On the other hand, parametric genes at the lower level are actually real-valued numbers which represent the parameters of the local compatibility functions.

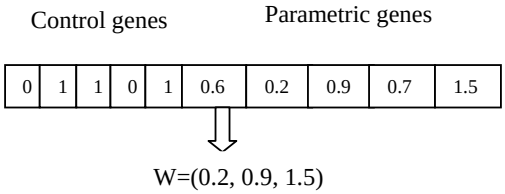

Fig. 3. Hierarchical chromosome representation

Important is to note that the parametric genes are controlled by the corresponding control genes. When a control gene is assigned with bit "1", the feature is selected and thus the associated parametric gene at the lower level is activated. Otherwise, if the control gene has bit "0", the related parametric gene is deactivated and becomes useless as the corresponding feature is not adopted in similarity modelling. As illustration, an example of hierarchical chromosomes is shown in Fig. 3, where the first and fourth parametric genes are deactivated by their control genes, hence only three parameter values for the compatibility functions are derived from this chromosome.

Fitness Evaluation. The ''leave-one-out'' procedure is used here to evaluate the selected features and the associated similarity model. Every case from the case base is treated as a query and its possibility distribution is derived based on the rest cases in the case library. By $Poss^i_{HC}$ we denote the possibility distribution derived for case C_i based on the other cases and using the selected features and similarity model encoded by hierarchical chromosome *HC*. An ideal and perfect possibility distribution should yield the highest possibility value (unity) on the true label, s_i, of the case while zero degrees on all other labels. With thoughts as this, we define the accuracy of the possibility distribution $Poss^i_{HC}$ as follows

$$Acc\left(Poss^i_{HC}\right) = 0.5 Poss^i_{HC}(s_i) + 0.5\left(1 - \max_{s \neq s_i} Poss^i_{HC}(s)\right) \quad (7)$$

Further, the fitness of chromosome *HC* is assessed as the mean accuracy of the possibility distributions of all cases in the case library. Thus, we get the fitness function as written as

$$Fit(HC) = \frac{1}{K} \sum_{i=1 \cdots K} Acc\left(Poss^i_{HC}\right) \quad (8)$$

where *K* is the number of cases in the case library.

Here the number of selected features is not taken into account in the fitness function. Since different importance of selected features will be reflected from identified similarity parameters, it is not necessary to enforce reducing input dimension during a search process. We leave this task later for users to decide suitable trade-offs between accuracy and complexity based upon the ranking of selected features.

Local Search. We perform local search to a part of chromosomes in the population. Each individual chromosome has a probability for undergoing this. When search is applied to an individual, exploration is made randomly in its neighbouring area to find better or local optimal solutions. So far we have implemented local refinements on control genes. The search begins with randomly picking a control gene (binary bit) to reverse. If the original gene is "1", it is changed to "0" meaning that the corresponding feature is no longer selected. Otherwise an original gene "0" will be flipped to "1" signifying that an additional feature has been included. Once a control gene is changed, we get a new chromosome which is subsequently evaluated by the fitness function in (8). If the new chromosome is assessed to be better, it replaces the original one, otherwise the old solution remains unchanged. Next we randomly pick a second control gene for trial, and this procedure is repeated for a fixed number of times to

complete the local search for an individual chromosome in the population.

Crossover. For every pair of hierarchical chromosomes to be combined, randomly choose gene positions. Each gene position is chosen with the probability of 0.5. Interchange the gene values at the chosen positions between the two parent chromosomes to generate child chromosomes.

Mutation. Because of the distinct nature of control and parametric genes, different mutation schemes are needed. Since the parametric genes are actually real-numbers, a small mutation with high probability is more meaningful. Therefore it is so designed that each parametric gene undergoes a perturbation. The magnitude of this perturbation is determined by a Gaussian density function. For the binary control genes, mutation is simply to inverse a gene, replacing '1' with '0' and vice versa. Every control gene undergoes a mutation with a quite low probability.

Finally the memetic algorithm designed for combined feature selection and similarity modelling is summarized as follows:

Step 0 (Initialisation): Randomly generate an initial population containing N_{pop} hierarchical chromosomes each of which consists of both control and parametric genes.

Step 1 (Fitness evaluation): Decode each hierarchical chromosome in the population into the associated similarity model. Use that similarity model to estimate the possibility distribution of every case in the case base in a ''leave-one-out'' procedure. The mean accuracy of the possibility distributions of all cases, as defined in (8), is regarded as the fitness score for the chromosome under evaluation.

Step 2 (Local search): Apply the local search mechanism to a part of the population. Every individual in the population has a probability to be selected for self-improvement via local search.

Step 2 (Termination test): If the stop-condition is not satisfied, then go to Step 4, else terminate the search procedure and return the best individual in the population.

Step 4 (Offspring generation): Apply the genetic operators to generate N_{POP} offspring to form the new generation, based on selection, crossover, and mutation. Go to step 1.

V. EXPERIMENTAL EVALUATIONS

The proposed MA based approach is well applicable to learning similarity for a CBR system based on the case library. In order to evaluate the effectiveness of this approach, we show here the experiments results on two benchmark data sets: wine data and sonar data, downloaded from the UCI Machine Learning Repository [26]. The wine and sonar data sets have 13 and 60 features respectively, both of them consist of cases categorized into discrete classes or labels. The experiments were made on both data sets to find out relevant features as system inputs and to identify optimal parameters in the similarity model concurrently. The magnitudes of the identified parameters in the similarity model also reflect the importance of the selected features.

We did 10-fold cross-validation in examining the capability of the proposed approach on both of the data sets. A given data set was divided into 10 parts of equal size. In each of the 10 trials, nine parts were used as the case base while the rest part was used as the test set comprising query problems. The search with MA was conducted on the case bases to find the best combination of feature subset and similarity model by maximizing the fitness function defined in (8). The learned similarity model was then utilized by the CBR system to classify the problems in the test set. The results of the cross-validations on the two data sets are illustrated in Table I, where the accuracy on test data and the numbers of selected features are presented.

TABLE I: RESULTS OF 10-FOLD CROSS-VALIDATIONS

Data sets	Accuracy on test data	Selected features
Wine	0.955	9.8
Sonar	0.851	31.9

Next we compare our results with those of some other machine learning techniques on the same data sets. Such comparisons are demonstrated in Tables II and III, for Wine and Sonar data respectively. The classification accuracy of the other works was also obtained in a 10-fold cross-validation procedure but using all the presented features as inputs.

TABLE II: COMPARISON ON WINE DATA

Methods	Accuracy on test data	Input number
This paper	0.955	9.8
Ref. [27]	0.901	13
Ref. [28]	0.937	13
Ref. [29]	0.916	13
Ref. [30]	0.938	13
Ref. [31]	0.944	13
SOP-3 [32]	0.935	13
MOP-3 [32]	0.970	13

TABLE III: COMPARISON ON SONAR DATA

Methods	Accuracy on test data	Input number
This paper	0.851	31.9
Ref. [27]	0.746	60
Ref. [28]	0.715	60
Ref. [29]	0.841	60
Ref. [30]	0.822	60
Ref. [31]	0.754	60
SOP-3 [32]	0.757	60
MOP-3 [32]	0.825	60

It can be seen from the tables that, on the Wine data, we obtained a reduction of input dimension with 25% in comparison to using all presented features as inputs as done by the other works. On the Sonar data, the input reduction was even more dramatic with almost 50%, which implies a considerably large decrease of the complexity of the system. Regarding accuracy, our method achieved the best result on the Sonar data among all the works though roughly half of the features were discarded therein. On the Wine data, the accuracy from this paper ranks the second best among all the

methods and appears very close to the best result from MOP-3 [32]. All these give clear evidence of the capability of our proposed method for dimension reduction and performance improvement, in particular when the number of features in the original data sets is very large.

VI. Conclusion

This paper proposes a novel approach based on a Hierarchical Memetic Algorithm (HMA) for combined feature selection and similarity modelling in a case-based reasoning context. The key of the designed HMA lies in the hierarchical formulation of chromosome that consists of control genes and parametric genes. The control genes correspond to a hypothesis for whether a given feature is relevant or not, while the parametric genes are associated with real-valued parameters in the similarity model. Further, a control gene at the higher level overrides the corresponding parametric gene at the lower level. The benefit of this hierarchical structure is that it enables selection of relevant features and searching for similarity parameters by the memetic algorithm in a concurrent manner. The objective of the memetic algorithm is to optimize the possibility distributions estimated for cases in the case library under a leave-one-out procedure.

The benefits of this proposed approach are two-fold. First, it introduces a new aspect of feature selection for similarity learning, which is a topic under-researched in CBR research. To our knowledge, this paper presents a first attempt to integrate feature selection and similarity modelling for a CBR system. Secondly, it contributes a new and effective approach for feature selection based on a case-based paradigm. The merit of our approach is that it improves the complexity of wrapper by not requiring repeated modelling given a feature subset. On the other hand, it also reveals the importance of selected features by the identified parameter values, such that feature ranking would be possible with our approach.

References

[1] A. Jain and D. Zongker, "Feature selection: Evaluation, application, and small sample performance," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 19, no. 2, pp. 153-158, 1997.

[2] M. Dash and H. Liu, "Feature selection for classification," International Journal of Intelligent Data Analysis, vol. 1, no. 3, pp. 131-156, 1997.

[3] M. Dash and H. Liu, "Consistency-based search in feature selection," Artificial Intelligence, vol. 151, no 1/2, 155-176, 2003.

[4] K. Praczyk, H. Kiendl, and T. Slawinski, "Finding relevant process characteristics with a method for data-based complexity reduction," in Lecture Notes in Computer Science 1625, Springer Verlag, 1999, pp. 548-555.

[5] M. R. Emami and I. B. Türksen, "Development of a systematic methodology of fuzzy logic modelling," IEEE Trans. Fuzzy Systems, vol. 6, pp. 346-361, 1998.

[6] R. Kohavi and G. H. John, "Wrapper for feature subset selection," Artificial Intelligence, vol. 97, no. 1/2, pp. 273-324, 1997.

[7] K. Z. Mao, "Feature subset selection for support vector machines through discriminative function pruning analysis," IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 34, no. 1, pp. 60-67, 2004.

[8] C. N. Hsu, H. Huang, and S. Dietrich, "The ANNIGMA-wrapper approach to fast feature selection for neural nets," IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 32, no. 2, pp. 207-212, 2004.

[9] A. Aamodt, and E. Plaza, "Case-based reasoning: foundational issues, methodological variations, and system approaches," Artificial Intelligence Com., vol. 7, pp. 39-59, 1994.

[10] M. M. Richter, "The knowledge contained in similarity measures," invited talk at the International Conference on Case-Based Reasoning, 1995.

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

[12] D. Wettschereck, and D. Aha, "Weighting features," in *Proc. 1st International Conference on Case-based Reasoning*, 1995, pp. 347-358.

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

[14] F. Ricci, and P. Avesani, "Learning a local similarity metric for case-based reasoning," in *Proc. International Conference on Case-Based Reasoning* (ICCBR-95), Sesimbra, Portugal, 1995.

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

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

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

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

[19] N. Xiong, and P. Funk, "Building similarity metrics reflecting utility in case-based reasoning," *Journal of Intelligent and Fuzzy Systems*, vol. 17, pp. 407-416, 2006.

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

[21] H. Ahn, K. Kim, and I. Han, "Global optimization of feature weights and the number of neighbors that combine in a case-based reasoning system," Expert Systems, vol. 23, pp. 290-301, 2006.

[22] D. Dubois and H. Prade, "Fuzzy set modelling in case-based reasoning," International Journal of Intelligent Systems, vol. 13, pp. 345-373, 1998.

[23] N. Krasnogor and J. Smith, "A tutorial for competent memetic algorithms: model, taxonomy, and design issues," IEEE Trans. Evolutionary Computation, vol. 9, no. 5, pp. 474-488, 2005.

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

[25] Y. S. Ong and A. J. Keane, "Meta-Lamarckian in memetic algorithm," IEEE Trans. Evolutionary Computation, vol. 8, no. 2, pp. 99-110, 2004.

[26] D. Newman, S. Hettich, C. Blake, and C. Merz, "UCI repository of machine learning databases", URL:<http://www.ics.uci.edu/~mlearn/ML.Repository.html>.

[27] J. R. Quinlan, "C4.5: Programs for Machine Learning," Morgan Kauffman, San Mateo, CA, 1993.

[28] S. Y. Ho, H. M. Chen, and S. J. Ho, "Design of accurate classifiers with a compact fuzzy rule base using an evolutionary scatter partition of feature space," IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 34, no. 2, pp. 1031-1043, 2004.

[29] Y. C. Hu, "Finding useful fuzzy concepts for pattern classification using genetic algorithm," Information Sciences, vol. 175, no. 1, pp. 1-19, 2005.

[30] Z. Lei and R-H. Li, "Designing of classifiers based on immune principles and fuzzy rules," Information Sciences, vol. 178, pp. 1836-1847, 2008.

[31] T. Elomaa and J. Rousu, "General and efficient multisplitting of numerical attributes," Machine Learning, vol. 36, no. 3, pp. 201-244, 1999.

[32] H. Ishibuchi and Y. Nojima, "Analysis of interpretability-accuracy tradeoff of fuzzy systems by multiobjective fuzzy genetics-based machine learning," International Journal of Approximate Reasoning, vol. 44, pp. 4-31, 2007.