Proceedings of the 6th WSEAS International Conference on Robotics, Control and Manufacturing Technology, Hangzhou, China, April 16-18, 2006 (pp56-61)

A Design of Analysis Model using Feature Weighting on CBR Method

YOUNG JUN KIM
Division of Business Administration
Baekseok College of Cultural Studies
393, Anseo-dong, Cheonan, Chungnam 330-705
KOREA

Abstract: - This paper is a principal idea of case-based reasoning to feature weighting. The feature weighting method called CaDFeW (CAse-based Dynamic FEature Weighting) stores classification performance of randomly generated feature weight vectors. Also it retrieve similar feature weighting success story from the feature weighting case base and then designs a better feature weight vector dynamically for the a new input problem while solving the problem. The CaDFeW is wrapper model-based feature weighting method that uses classifier error rate as evaluation procedure. To explain the results of applications, this paper is introduced a new definition of input dependency of feature relevance and measured the new concept in the application domains. The empirically measured results showed that relative performance of a local feature weighting method to a global feature weighting method.

Key-Words: - Machine learning, wrapper method, case-based classifier, feature weighting method.

1 Introduction

There are two types of learning modes: eager learning and lazy learning. Eager learning approaches to induction produce generalizations that explicitly represent the classes under study, often in a language different from that used to represent the cases. Lazy approaches, in contrast, delay this generalization process until classification time; it is performed implicitly when a new case is compared to the stored cases and the class of the nearest one(s) is assigned to it. Lazy learning methods such as case-based reasoning (CBR) [6] have several advantages when compared to eager methods like decision tree and artificial neural networks [7,8]. They are conceptually simple, and yet able to form complex decision boundaries in the problem space even when only relatively little information is available. They can be applied easily to both categorical and numeric output problems. They have no difficulty in handling both categorical and numeric input features. Special cases that cannot be handled by eager approaches can be retained and recognized.

The goal of this study is to develop an effective and efficient local feature weighting method for a lazy learning method, specifically, a CBR method. The effectiveness of the new method is measured by the improvement of the classification accuracy after applying it to a basic CBR system. The application results are compared with some other feature weighting methods. Fig.1 shows the conceptual framework of this study. In the initial stage of this study, we review existing researches related to feature weighting methods. This can be an overall infrastructure for this study. Based on the findings from the in-depth analysis of the previous studies, we attain the possible directions to develop a new method. Prototypical design of the methods are tested and modified iteratively until the method shows sufficient level of effectiveness.

After a few iterations of refinement process, we compile the findings on the relationships between feature weighting and CBR from the empirical experiments based on the test results. This is also a contribution of this study. We propose, CaDFeW method, a case-based dynamic feature weighting method that is an alternative local feature weighting approach for lazy learners. CaDFeW keeps randomly generated feature weight vectors and the train-time experiences, i.e., test results. Then it retrieves these experiences from the case base and generates new feature weight vectors in run-time.

2 Case-Based Reasoning

Case-based reasoning (CBR) is one of such machine learning approaches. Previous cases are used to make a solution for a new problem. From the cases available, a CBR system retrieves the most similar case(s) to the input problem and then adapts the solution of the retrieved case to fit the context of the Input problem. The basic idea of CBR is based on the process of human problem solving. Human beings use previous experiences of problem solving when encountered a new problem to solve. This natural problem solving approach allows the reuse of problem solving experiences and is considered a breakthrough from the knowledge acquisition bottleneck. Since there are numerous possible CBR design options, testing all the possible CBR designs in order to develop a good feature weighting method will be an exhaustive and time-consuming work. In this study, we use a typical CBR system for classification as a base classifier. When encountered a new problem to solve, the CBR system retrieves a set of most relevant cases from the case base and suggests a new solution.

We focus on the typical classification problems that have the following characteristics. First, the problems have discrete output classes. Hence, the performance of CBR system can be investigated by checking the results whether they are correct or not. Second, the problems have relatively many features and have both numeric and categorical features in most cases. In the adaptation stage, some simple voting heuristics are used. When a CBR system for classification problem retrieves more than one case, it may encounter conflicting solutions. Then the CBR system simply selects the most frequent solution. There can be some possible variations of this voting method [6]. In this study, however, we use only equally weighted voting method while varying the number of cases retrieved, i.e., the value of k. Because this study mainly focuses on typical classification problems, other issues such as case indexing, sophisticated adaptation, and repair are not considered.

3 Feature Weighting Methods

3.1 Categorization of feature weighting

Feature weighting efforts attempt to find the optimal feature weight vector that makes the classifier show best classification accuracy. Feature weighting methods search through the feature weight vectors, and find the best one among the unlimited number of candidate weight vectors according to an evaluation criterion. However, this procedure is exhaustive because it tries to find only the best one. It may be too costly and practically prohibitive, even for a small size of feature set. Other methods based on heuristic or random search attempt to reduce computational complexity at the cost of performance. These methods need a stopping criterion to prevent an exhaustive search of weight vectors. Table 1 shows the modified framework and categorization of some representative methods.

Table 1. Categorization of feature weighting methods in a 3-dimensional framework

Weight scope	Evaluation function	Generation procedure		
		Heuristic	Complete	Random
Global	Distance	Relief	+	-
	Information	DTI	+	-
	Dependency	+	-	-
	Consistency	-	+	+
	Classifier error	+	+	GA
Local	Classifier error	RC	-	**This study (CaDFeW)**

+ There are several methods but they are not presented here.
- There are no known methods.

Evaluation procedures can be categorized differently by whether they use feedback from the performance task. Methods that do not use feedback are called filters, whereas methods that use the classifier itself as the evaluation procedure are wrappers [5]. The weight scope means the generality of the weights in the case space. The weight scope that most feature weighting methods produce is global: their weights apply across the entire case space. In contrast, local feature weighting methods allow feature weights to vary in different parts of the case space. In local feature weighting methods, weights can vary by class [4], feature value [9], and/or individual case or subset of cases [1].

3.2 Case-based dynamic feature weighting

We start by describing the CaDFeW (CAse-based Dynamic FEature Weighting) method in this subsection and then discuss the motivations behind it in the next subsection. The method for weighting features of a CBR system by another CBR, i.e., CaDFeW is simple and can be stated as follows*: By remembering the train-time performance of randomly generated feature weight vectors, a CBR system designs its input-specific feature weight vector dynamically in the run-time.*

CaDFeW stores pre-processed results and then dynamically generates a desirable feature weight vector when it is running. The behavior of CaDFeW is similar to that of RC. Both methods generate weights input-sensitively and they are wrapper methods. CaDFeW is a wrapper method. Therefore it shares the general properties of the wrapper methods discussed in John et al. [5]. However, CaDFeW is relatively faster than other global wrapper methods and a local method such as RC. In comparison with some feature weighting methods, CaDFeW is flexible. Firstly, we can use multiple weighting experiences to design a new feature weight vector for a new input case to enhance the classification accuracy of CBR. However, if there is no significant improvement and computational resource is a concern, we can choose a negotiation point. Secondly, the current version of CaDFeW provides continuous weights rather than binary weights. However, since binary weight is the special case of continuous weight, we can easily adjust CaDFeW to provide binary weights. The power of CaDFeW's feature weighting is currently being under investigation. However, some empirical test results presented later show us the potential of the method. The simplicity and the naturalness of the algorithm are the most attractive advantages.

3.3 Combing a global feature selection method and CaDFeW

To make a feature weighting method efficient, we need to reduce the number of features before assigning weight values. Combining feature weighting and feature selection can be one of these alternatives. We designed an open and combined method called LowGoS (LOcal Weighting after GlObal Selection). We can eliminate globally useless or irrelevant features in a certain criteria by executing a global feature selection method before applying a local feature weighting method. The global feature selection (GFS) method removes possibly very irrelevant features from the initial input feature set. The underlying motivation of applying this combined method is to improve both classification accuracy and reduce search space significantly. However, there are some unanswered questions. First, there is possibly unlimited number of alternative GFSs. Therefore we have to select the most appropriate method for both CBR and CaDFeW. Second, most GFS use conservative, i.e., wrong-selection averse, strategy. Therefore we have to determine the desirable level of conservativeness.

In this study, instead of using only existing GFS methods, we designed a simple wrapper-based heuristic method called UniFeS (UNIvariated analysis-driven FEature Selection). UniFeS first examines the predictive power of each feature using a wrapper approach. Only one feature, with all of the other attributes being ignored, is used as an input attribute of CBR to classify a given problem. The same procedure is executed for all features logically and possibly in parallel. Classification accuracy of a feature is assumed to be an index of predictive power of the feature.

After examining the predictive power of all features, relative value of features among all the features are compared. However, we use classification accuracy as the measure of potential predictive power instead of deterministic measures such as mean. A set of relatively less powerful predictors is discarded in the final stage of LowGoS, i.e., CaDFeW. Then, CaDFeW starts with the remaining features.

4 Empirical Evaluations

We describe a set of empirical tests to investigate the usefulness of CaDFeW method. The method will be used for a set of CBR systems in several different domains. Problem structure of each application is reformulated to have common properties of classification tasks. Since the main goal of the feature weighting is to improve classification accuracy of CBR system, we measure the classification accuracy of the CBR systems by using different versions of CaDFeW feature weighting method and LowGoS. The CBR system without feature weighting effort is considered as a baseline. We use alternative feature weighting methods such as DTFS and UniFeS for the purpose of comparative analysis.

4.1 Credit evaluation application

Credit evaluation is to decide whether we should approve customer application for a loan based on the collected customer information. It is one of the very typical analytical modeling application domains. Therefore there are many precious studies and industrial efforts.

We implemented some designs of CaDFeW and tested them to a benchmark data set. The data set used for this experiment is taken from the UC Irvine machine learning database repository [3]. It is the same data set as Quinlan [8] and Domingos used. This data set has 690 credit evaluation records with some missing values. The data set has 6 numeric and 9 categorical features. The meanings of each feature are not described. We compared the basic CBR and the CaDFeW CBR. CaDFeW CBR B30w3, one of several versions of CaDFeW, uses only three most similar and successful weighting experiences and no failed experiences. It stores three weighting experiences for each case in the case base. In adaptation, voting among top three similar cases with no weighting scheme is employed. DTI was performed using the SAS E-Miner data mining system. DTI-EM (DTI built by E-Miner) is set to have similar parameters to those of C4.5.

Table 2. Classification accuracies of classifiers: credit evaluation

	Average accuracy(%)	Standard deviation(%)	Notes
Basic CBR	84.51	1.91	k=3
CaDFeW CBR	**88.01**	**1.27**	**p=3**
DTl-EM	86.84	1.46	C4.5-like setting
Logistic regression	84.71	1.82	SAS E-Miner

Ten different runs were carried out (tenfold cross-validation). In each, the case base was composed of 40% of the data set, chosen at random, and the other 30% were used as test set. As table 2 shows, the basic CBR obtained about 84.51% classification accuracy on the average. A CaDFeW CBR system showed 88.01% classification accuracy that is better than those of both the basic CBR and DTI-EM. Quinlan and Domingos also used DTS and Domingos used C4.5. Based on a paired *t*-test, we concluded that CaDFeW CBR B30w3 showed significantly better classification accuracy than the basic CBR. However, difference between CaDFeW CBR B30w3 and DTI-EM was not sufficient to tell.

4.2 Housing value estimation application

Boston housing value estimation problem is another benchmark data set from the UCI machine learning database [3]. This data set was originally taken from the StatLib library that is maintained at Carnegie Mellon University. This data set concerns housing values in suburbs of Boston. There are 506 cases, 13 continuous attributes and no missing attribute value.

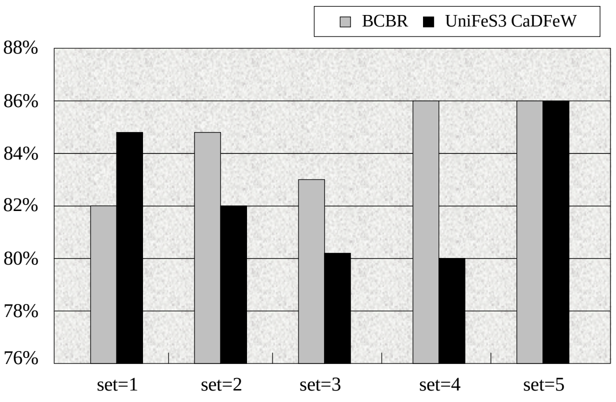

Fig.1. Basic CBR and UniFeS CBR: classification accuracy comparison: housing value estimation

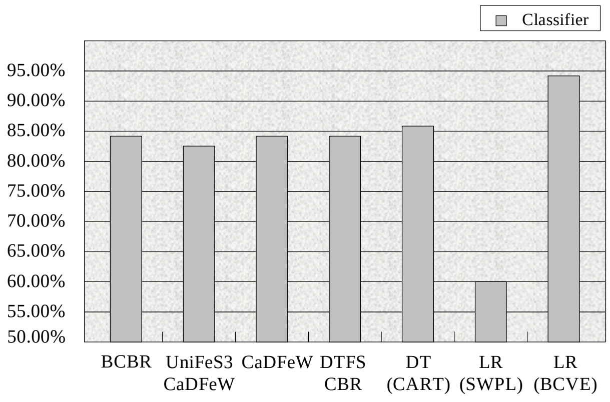

Fig.2. Average classification accuracies of various classifiers: housing value estimation

Fig.1 shows the classification accuracies between UniFeS3 CBR and basic CBR (BCBR) in comparison. Both systems are set to k=3 and UniFeS3 CBR use only top three high performance features selected through the UniFeS process. UniFeS3 CBR does not show better classification accuracies than BCBR in general. However, the point that UniFeS reduced the number of features to 3 has an important meaning.

Dimension reduction like this can provide a positive impact on, at least, enhancing the efficiency of CaDFeW process execution. Fig.2 shows the summarized results of various classifiers. LowGoS (UniFeS3 CaDFeW) and CaDFeW CBR show no improvement.

5 Input Case Dependency of Feature Relevance

We study the reason why CaDFeW and LowGoS showed different performance in different applications. Since there is enough number of efficient global feature weighting methods, if local or case-specific feature weighting methods do not perform well in some applications then we have to select a global feature weighting method.

Table 3. Measured input dependencies in the application domains

	Input dependency (k=1)		Input dep. (k>1)	Relative performance of LowGoS (%)*	
	Ave.	Std. dev.		DTFS**	UniFeS***
Customer retention	0.89	0.01	0.89	104.32	101.50
Credit evaluation	0.30	0.05	0.29		
Housing value est.	0.27	0.04	0.25	97.87	100.48

** Relative performance of LowGoS = Classification accuracy of LowGoS / Classification accuracy of the baseline global weighting method*
*** Using CART decision tree as global feature selection method*
**** Using UniFeS as global feature selection method*

Domingos tried to measure context dependency of feature relevance. However, since the true degree of context dependency of feature relevance is unknown and the concept of context dependency is hard to measure, he measured the concepts with an empirical study. He measured how far his local feature weighting method strays from selecting the same features for all instances. From a new operational definition, we try to measure the input dependency of feature relevance in application domains to applying CBR. The basic definition of Input case dependency of feature relevance (IDFR) here is made as: *IDFR is the degree of difference among the appropriate weight teeters for different cases.* Table 3 shows the result of input dependency measurement. We used 5 randomly generated weight vectors, 50 test cases and executed 10 times. Housing value estimation has low input dependency and customer retention has very high input dependency.

In Table 3, We can see that the relative classification accuracy of LowGoS is correlated to the input dependency of each domain. Therefore, we can understand why there was no significant improvement of the relative performance in the housing value estimation domain. The baseline global feature weighting methods used here are UniFeS and DTFS.

6 Conclusion

There are some researches on flexible, context-sensitive, and local feature weighting. However, few researches tried to use wrapper model for local feature weighting except for Domingos. We proposed a new local wrapper method for feature weighting named CaDFeW, a new wrapper global feature selection method called UniFeS and a strategy which integrates the local, i.e., CaDFeW, and global, i.e., UniFeS, feature weighting approaches called LowGoS. CaDFeW is very simple and relatively efficient among wrapper model-based feature weighting methods. Our methods overcame the limitations of RC in two key aspects as pointed out below.

- CaDFeW supports more than single nearest neighbor - We can enhance the classification performance of the classifier CBR.
- LowGoS combines local and global wrapper methods - UniFeS supports execution process of the CaDFeW by reducing dimensionality.

Although the results of some applications did not showed sufficient evidences for the usefulness of the new method, we expect that it can be improved and will work effectively in most situations. That is because the core idea of the method is remembering the real experiences of the classifier CBR itself. The core contribution of this study can be stated as:

(1) We extended existing categorizations of feature weighting methods by including the scope dimension, and developed a new 3-dimensional framework and then developed a brand new combination of feature weighting method. (2) We developed a new measurement called input dependency of feature relevance that will be used to determine which type of weights, i.e., local weights or global weights, is appropriate for a particular application. (3) We showed the usefulness of local feature weighting in

various parameter setting based on the empirical test results.

References:

[1] Aha, D. W., "Feature Weighting for Lazy Learning Algorithms," Liu, H. and H. Motoda(Eds.), *Feature Extraction, Construction and Selection: A Data Mining Perspective*, Norwell MA: Kluwer, 1998.

[2] Blake, C., E. Keogh, and C. J. Merz, UCI Repository of machine learning databases, Irvine, CA: U. of California, Department of Information and Computer Science, 1998.

[3] Friedman, J. H., "On Bias, Varian, 0/1-Loss, and the Curse-of-Dimensionality," *Data Mining and Knowledge Discovery*, Vol.1, Kluwer, 1997. pp.55-77.

[4] Howe, N. and C. Cardie, "Examining Locally Varying Weights for Nearest Neighbor Algorithms," *Case-Based Reasoning Research and Development: 2nd Int. Conference on Case-Based Reasoning*, 1997. pp.445-466.

[5] John, G. H., R. Kohavi and K. Pfleger, "Irrelevant Features and the Subset Selection Problem," *Proceedings of the 11th Int. Conference on Machine Learning*, 1994. pp.121-129.

[6] Kolodner, J., *Case-Based Reasoning*, Morgan Kaufman Publishers, 2003.

[7] Nelson, M. M., and W T. Illingworth, *A Practical Guide to Neural Nets*, Addison-Wesley, 2001.

[8] Quinlan, R., "Simplifying Decision Trees," *Int. Journal of Man-Machine Studies*, Vol.27, 1987. pp.221-234.

[9] Stanfill, C. and D. Waltz, Toward Memory-Based Reasoning, *Communications of ACM*, Vol.29, 1986. pp.1213-1228.