Knowledge-Based Systems 15 (2002) 391–398

Knowledge-Based SYSTEMS

www.elsevier.com/locate/knosys

Formalising optimal feature weight setting in case based diagnosis as linear programming problems

Lu Zhang*, Frans Coenen, Paul Leng

Department of Computer Science, University of Liverpool, Liverpool L69 3BX, UK

Received 25 July 2001; accepted 7 January 2002

Abstract

Many approaches to case based reasoning (CBR) exploit feature weight setting algorithms to reduce the sensitivity to distance functions. In this paper, we demonstrate that optimal feature weight setting in a special kind of CBR problems can be formalised as linear programming problems. Therefore, the optimal weight settings can be calculated in polynomial time instead of searching in exponential weight space using heuristics to get sub-optimal settings. We also demonstrate that our approach can be used to solve classification problems. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Case based reasoning; Feature weight; Linear programming

1. Introduction

Case based reasoning (CBR) is a multi-disciplinary subject that focuses on the reuse of experiences [2]. Typically, a CBR approach retains a fairly large number of previous experiences (which are usually called cases) in a database (which is usually called the case base). When a new problem occurs, it will be represented as a new case and compared to the cases in the case base. Thus, the cases similar to the new cases will be used to suggest to users a solution for the new problem. Usually, the solved new case will also be added into the case base. The underlying assumption for CBR is that similar cases will have similar solutions. Many CBR approaches have been reported for solving problems in various domains, such as planning [11], design [14], software engineering [4,42], image processing [21,40], and diagnosis [8,31,49].

Cases can be represented in various forms. In this paper, we focus on the well-structured form. Other forms of case representation include *directed acyclic graphs* [30] and *examplars* [9,41]. A well-structured case is defined as a vector of features: $x = \{x_1, x_2, \ldots x_q\}$, where q is the number of features. Therefore, an algorithm such as k-nearest neighbour (k-NN) [18,22] can be exploited through calculating the distance between case $x = \{x_1, x_2, \ldots x_q\}$ and case $y = \{y_1, y_2, \ldots y_q\}$ using a distance function. However, a universal distance function may not be suitable for solving problems in different domains. One solution is to investigate multiple distance functions [55,56]. The other is to parameterise the distance function with feature weights [3,53]. For a feature weighting approach, a main research topic is to determine feature weight settings (see e.g. [16,32,34,48,54]).

In particular, CBR approaches can also be applied to diagnosis problems. In a diagnosis problem, the solution to a case is a set of faults, which is usually easy to be compared with the set of faults of another case. For two cases whose solutions have been found, the similarity in each feature, the calculated overall similarity, and the real similarity between the two sets of faults can be obtained. However, when diagnosing a new case, only the similarity in each feature and the calculated overall similarity between the new case and an existing case can be obtained. The calculated overall similarity is used to predict the fault similarity.

Linear programming (LP) [17] has been a rapidly developing mathematical discipline since it started in 1947. Theoretically, LP problems have been proved to be polynomial time solvable [26,27]. Practically, the simplex algorithm [17] and its derivatives and the interior point algorithm [26] and its derivatives are fast enough to be used in real world applications. Therefore, many realistic problems that have been expressed as LP problems have been well solved, and there is also a lot of commercial or free software for LP problems available from vendors and/or on the Internet.

* Corresponding author. Tel.: +44-151-794-3792; fax: +44-151-794-3715.

E-mail addresses: lzhang@csc.liv.ac.uk (L. Zhang), frans@csc.liv.ac.uk (F. Coenen), phl@csc.liv.ac.uk (P. Leng).

In this paper, we demonstrate that optimal feature weight setting in a general form of case based diagnosis can be formalised as LP problems. Therefore, available LP software can be used to solve case based diagnosis problems.

The organisation of the remainder of the paper is as follows. Section 2 gives a brief overview of LP problems, and presents a general form of case based diagnosis. Section 3 demonstrates that calculation of optimal initial weights can be formalised as an LP problem. Section 4 demonstrates that calculation of new case weights can also be formalised as an LP problem. Section 5 provides some preliminary empirical results. Section 6 further discusses some related works of our approach. Section 7 concludes this paper.

2. Preliminaries

2.1. Linear programming problem

In general, an LP problem is to calculate the maximum or minimum value of a linear combination of a set of variables subject to a set of linear equations and/or linear inequalities as constraints. Therefore, an LP problem can be represented in the following form [12]:

maximise or minimise

$$\sum_{j=1}^{n} c_j x_j$$

subject to Constraint$_i$ ($i = 1, 2, \ldots m$)

Constraint$_i$ is of one of the three forms:

$$\sum_{j=1}^{n} a_{ij} x_j \geq b_i, \text{ or} \tag{1}$$

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \text{ or}$$

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i$$

$$x_j \geq 0 (j = 1, 2, \ldots n)$$

In Eq. (1), $x_1, x_2, \ldots x_n$ are the *decision variables*; $c_1, c_2 \ldots c_n$ are the *cost coefficients*; $\sum_{j=1}^{n} c_j x_j$ is the *objective function*; $b_1, b_2, \ldots b_m$ are the *right-hand-side constants*; n is the number of decision variables; and m is the number of *constraints*.

There have been many efficient algorithms for LP problems reported in the literature. Algorithms that can solve the problem in Eq. (1) in polynomial time of $m + n$ have been reported in the literature (see e.g. [26,27]). Although the *simplex algorithm* [17] has been proved to be exponential in the worst case [29], there is strong empirical evidence suggesting that it typically takes $O(m + n)$ time to solve the problem in Eq. (1) [12]. In Ref. [37], it is also demonstrated that some interior point algorithms are even faster than the simplex algorithm when dealing with problems of large sizes. However, a thorough theoretical analysis of the exact complexity of interior point algorithms is still on its way. Other material concerning the complexity of algorithms for LP problems can be found in Refs. [5,35].

2.2. A general form of case based diagnosis

A feature weighting approach to the case based diagnosis problem can be summarised as follows. Every case is fully structured as a vector of features. Given a case x in the case base, we can obtain the similarity between x and any other case in any feature. Therefore, we can calculate the overall similarity between x and the other case as a linear combination of the similarity in each feature. Supposing there are q features in a case, the overall similarity between x and the other case i can be calculated as formula (2) [51].

$$\sum_{k=1}^{q} S_{xik} W_{xk} \tag{2}$$

In Eq. (2), S_{xik} is the similarity between case x and case i in feature k and W_{xk} is the weight for case x in feature k. Each weight for one case represents the contribution of the similarity of the corresponding feature to the overall similarity. Informally, we would say that W_{ik} represents the influence of feature k in using case i to diagnose another case. When diagnosing a new case t, we will calculate every overall similarity between any case x in the case base and t. The most similar l cases will be used to suggest the faults for t. After diagnosis, case t may be also added into the case base to diagnose future cases.

We call this form of case based diagnosis a general form; because we assign each feature in each case a weight, rather than assigning each feature a weight for all the cases. This distinction was previously referred as *local weighting* (i.e. different feature weights for different cases) and *global weighting* (i.e. same feature weights for all cases) [23,54]. In this paper, it is not our aim to discuss which form is preferable. We use the general form only because global weighting is a simplification of local weighting for our method.

In the general form, the calculated overall similarities rely heavily on the weights. Therefore, to calculate the initial weights for the training cases and to calculate the weights for the new case to be added into the case base are two main tasks, which will be discussed in Sections 3 and 4.

3. Calculation of initial set of weights

3.1. The problem

There are n training cases in the case base, each having a value in each of q features. For any two cases, the similarity between the two cases in each feature and the real similarity between the two cases can both be obtained. The overall

Table 1
Symbols used in formalising initial weight calculation

Symbols	Meanings and constraints
$S_{ijk}(i,j = 1,2...n, i < j; k = 1,2...q)$	S_{ijk} is the similarity between case i and case j in feature k, and $0 \leq S_{ijk} \leq 1$
$R_{ij}(i,j = 1,2...n, i < j)$	R_{ij} is the real similarity between case i and case j, and $0 \leq R_{ijk} \leq 1$
$W_{ik}(i = 1,2...n; k = 1,2...q)$	W_{ik} is the weight of case i in feature k, $0 \leq W_{ik} \leq 1$, and $\sum_{k=1}^{q} W_{ik} = 1$

similarity between the two cases can be calculated using formula (2). The problem is to determine the optimal value of each initial weight for each case in each feature. The optimal values of the weights should satisfy the following condition. When diagnosing any case in the training set using the other $n - 1$ cases, the similarities calculated using the similarity function in formula (2) should be well in accordance with the real similarities.

3.2. Symbols

We summarise the symbols in our formalisation, their meanings and their constraints in Table 1. Here we only use normalised similarities and weights. Therefore, any similarity is a value between 0 and 1, and so is any weight. The sum of all the weights for any case should be 1, because each weight represents the contribution of the similarity of the corresponding feature to the overall similarity.

3.3. Formalisation

The set of satisfactory initial weights should have the following property. For cases i and j, both $\sum_{k=1}^{q} S_{ijk} W_{ik}$ and $\sum_{k=1}^{q} S_{ijk} W_{jk}$ should be approximately equal to R_{ij}. This property means that when comparing case j with case i, the value calculated from the similarity function using the weights of case i will be similar to the real similarity, and vice versa.

Therefore, the calculation of optimal initial weights can be formalised as an LP problem by introducing some margin variables. For $i,j = 1,2...n(i < j)$, we introduce margin variables L_{ij}, L'_{ij}, G_{ij} and G'_{ij}, each of which has a non-negative value. Then we can set up the following constraint equations in Eqs. (3) and (4):

$$\sum_{k=1}^{q} S_{ijk} W_{ik} + L_{ij} - G_{ij} = R_{ij} \qquad (3)$$

$$\sum_{k=1}^{q} S_{ijk} W_{jk} + L'_{ij} - G'_{ij} = R_{ij} \qquad (4)$$

In Eqs. (3) and (4), each L_{ij} or L'_{ij} stands for the part by which the predicted similarity is less than the real similarity, and each G_{ij} or G'_{ij} stands for the part by which the predicted similarity is greater than the real similarity. The goal is to minimise the sum of the margin variables, which means to minimise the sum of predictive errors in similarity. Therefore, we can use the function in Eq. (5) as the objective function.

$$\text{minimise} \sum_{i=1}^{n} \sum_{j=i+1}^{n} (L_{ij} + L'_{ij} + G_{ij} + G'_{ij}) \qquad (5)$$

In an optimal solution, for any $i,j = 1,2,...n(i < j)$, either L_{ij} is 0 or G_{ij} is 0 and either L'_{ij} is 0 or G'_{ij} is 0, because any predicted similarity can only be either greater or less than the real similarity. A simple proof is as follows. When both L_{ij} and G_{ij} are greater than 0, if $L_{ij} \geq G_{ij}$, setting L_{ij} to $L_{ij} - G_{ij}$ and G_{ij} to 0 will both satisfy the constraints in Eqs. (3) and (4) and make the result in Eq. (5) less; and similar for $L_{ij} < G_{ij}$.

3.4. Complexity

From the above formalisation, there are $n * q$ weight variables and $2n * (n - 1)$ margin variables. The number of the constraints presented in Eqs. (3) and (4) is $n * (n - 1)$ and there are n constraints in the form of $\sum_{k=1}^{q} W_{ik} = 1$. Therefore, the problem of calculating the optimal initial weights for the n cases, each of which is represented as a vector of q attributes, can be formalised as an LP problem with $n * (2n + q - 2)$ variables and n^2 constraints. As LP problems are polynomial time solvable, the problem of calculating the optimal initial weights is also polynomial time solvable.

If we require that each feature has only one weight for all the cases, we need only equations in Eq. (4) or equations in Eq. (5). Therefore, we will have q weight variables, $n * (n - 1)$ margin variables and $(n * (n - 1))/2 + 1$ constraints. Then, the optimal initial weight setting problem becomes an LP problem with $n * (n - 1) + q$ variables and $(n * (n - 1))/2 + 1$ constraints.

3.5. Sub-optimal simplification

In the formalisation, each case is compared with all the other cases. Therefore, we have $O(n^2)$ constraints and $O(n^2)$ margin variables. If we only compare each case with a subset of other cases, we can reduce the complexity of the problem of calculating the initial weights. Obviously, this is only a sub-optimal simplification.

Supposing a case is only compared with another $p(p \leq n - 1)$ cases, we will have $n * q$ weight variables and $2n * p$ margin variables. The number of constraints will be $n * (p + 1)$. Therefore, the problem of calculating the initial weights can be further simplified as an LP problem with $n * (2p + q)$ variables and $n * (p + 1)$ constraints.

Table 2
Symbols used in formalising new case weight calculation

Symbols	Meanings and constraints
$S_{ik}(i = 1,2...n; k = 1,2...q)$	S_{ik} is the similarity between case i and the new case in feature k, and $0 \le S_{ik} \le 1$.
$R_i(i = 1,2...n)$	R_i is the real similarity between case i and the new case, and $0 \le R_i \le 1$.
$W_k(k = 1,2...q)$	W_k is the weight of the new case in feature k, $0 \le W_k \le 1$, and $\sum_{k=1}^{q} W_k = 1$.

4. Calculation of new case weights

4.1. The problem

As different cases have different feature weights, the weights of a new case to be added into the case base should also be calculated. There are already n cases in the case base, each having a value in each of q features. For each feature in each case in the case base, the weight has been determined. There is a new case whose solution has been acquired. Therefore, the similarity between the new case and any case in the case base in each feature and the real similarity between the two cases can also both be obtained. The overall similarity between the two cases can be calculated using formula (2). The problem is to calculate the optimal value of each weight for the new case in each feature. The optimal values of the weights should satisfy the following condition. When diagnosing any existing case using the new case, the similarities calculated using the similarity function in formula (2) should be well in accordance with the real similarities.

We will also note here that the above condition may not be the optimal condition for all the $n + 1$ cases, because adding a new case may make the previous determined weights become not optimal and thus the previous weights should also be adjusted to be optimal. To cope with this, we can formalise this problem in the same way as described in Section 3 using all the $n + 1$ cases as the training cases. However, that will increase the computational complexity.

4.2. Symbols

We summarise the symbols in our formalisation, their meanings and their constraints in Table 2. Here we also only use normalised similarities and weights. Therefore, any similarity is a value between 0 and 1, and so is any weight. The sum of all the weights for any case should be 1, because each weight represents the contribution of the similarity of the corresponding feature to the overall similarity.

4.3. Formalisation

The set of new case weights should have the following property. For case i, $\sum_{k=1}^{q} S_{ik}W_k$ should be approximately equal to R_i. This property means that when comparing case i with the new case, the value calculated from the similarity function using the weights of the new case would be similar to the real similarity.

Therefore, the calculation of new case weights can be formalised as an LP problem by introducing some margin variables. For $i = 1,2...n$, we introduce margin variables $L_i(L_i \ge 0)$ and $G_i(G_i \ge 0)$. Then we can set up the following constraint equations in Eq. (6).

$$\sum_{k=1}^{q} S_{ik}W_k + L_i - G_i = R_i \tag{6}$$

The goal is to minimise the sum of the margin variables. Therefore, we can use the function in Eq. (7) as the objective function.

$$\text{minimise} \sum_{i=1}^{n} (L_i + G_i) \tag{7}$$

In an optimal solution, for any $i = 1,2,...n$, either L_j is 0 or G_i is 0.

4.4. Complexity

From the above formalisation, there are q weight variables and $2n$ margin variables. The number of the constraints presented in Eq. (6) is n and there is another constraint in the form of $\sum_{k=1}^{q} W_k = 1$. Therefore, the problem of calculating the weights for the new cases, each of which is represented as a vector of q attributes, can be formalised as an LP problem with $2n + q$ variables and $n + 1$ constraints. As LP problems are polynomial time solvable, the problem of calculating new case weights is also polynomial time solvable.

4.5. Sub-optimal simplification

In the formalisation, the new case is compared with all the existing cases. Therefore, we have $O(n)$ constraints and $O(n)$ margin variables. If we only compare each case with a subset of existing cases, we can reduce the complexity of the problem of calculating the new case weights. Again, this is a sub-optimal simplification.

Supposing a case is only compared with other $p(p \le n)$ cases, we will have q weight variables and $2p$ margin variables. The number of constraints will be $(p + 1)$. Therefore, the problem of calculating the new case weights can be further simplified as an LP problem with $2p + q$ variables and $p + 1$ constraints.

Table 3
Preliminary empirical results

Number of training cases	Training time (s)	Average deviation of estimation		
		Average value estimation (%)	Training case estimation (%)	Previous case estimation (%)
50	3.30	42.68	15.48	11.41
76	19.00	40.90	14.43	11.62
100	64.30	37.96	12.48	10.67

5. Preliminary empirical results

5.1. Experimental method

To evaluate the effectiveness of our approach, we implement a simplified version of our approach on a Pentium III 500 MHz PC with 128M RAM running Windows NT 4.0, and apply it to solve the 'Auto-Mpg' problem provided by Ref. [13]. The problem, which is originally proposed in Ref. [44], is to predict the fuel consumption in miles per gallon in terms of three multi-valued discrete and five continuous features. This problem can be viewed as a diagnosis problem by viewing the fuel consumption as the faults to diagnose. There are totally 398 cases including six cases that have missing feature values. In our experiment, we only use the 392 cases that have no missing feature values.

We randomly select some cases as training cases and the others as test cases. Due to the ease of implementation, we require only one weight for each feature for all cases in the simplified version. Therefore, we only need to solve one LP problem while training and no other LP problems while diagnosing. This makes us to be able to use an off-the-shelf LP software product to calculate the optimal weights instead of implementing our own LP algorithm in our experiment.

While training, we first generate the corresponding LP problem using the training cases. Then we solve the LP problem using IBM OSL [25], and thus get the optimal feature weight setting. Finally, we use the calculated optimal weights to diagnose the test cases. While diagnosing, the fuel consumption of the most similar case is used to predict the case under diagnosis. Actually, we test two methods of diagnosis. One is to diagnose a new case only using the training cases (which is called the training case estimation in Table 3). In this kind of diagnosis, none of the test cases under diagnosis will be added into the case base. The other is to diagnose a new case using the training cases and all the previously diagnosed cases (which is called the previous case estimation in Table 3). In this kind of diagnosis, each diagnosed test case will be added into the case base and thus used for diagnosing the next cases.

We perform the experiment three times each, respectively, using 50, 76 and 100 training cases. For each time, we record the average deviations of the estimated fuel consumption from the real consumption for the two diagnosis methods. As a comparison, we record the average deviation of merely using the average fuel consumption of the training cases as the estimation of the fuel consumption of each test case. We also record the running time reported by IBM OSL as the training time. The results are summarised in Table 3.

5.2. Results

From Table 3, we can find the following trends. First, typically the larger the training case number is, the more accurate the estimations are, which, we think, is due to the following reason. The larger the training case number is, the more the representability of the training cases is. Secondly, applying our approach can dramatically improve the accuracy of estimation, because the average deviations of estimation in our approach are about only one third of that of the average value estimation. Thirdly, using all previous cases in diagnosis is more accurate than merely using training cases, which, we think, implies that the feature weights calculated from the training cases can well fit into the test cases. Finally, the training time for our approach is quite acceptable, and it may be predicted that training larger number of cases should still be feasible.

6. Related work

6.1. Classification

Classification is a research focus, mainly investigated in the domain of data mining. A classification problem can be summarised as follows. Each instance x is structured as a vector of attributes $x = \{x_1, x_2 ... x_q\}$, where q is the number of features. Each existing instance will be assigned to a class, which is noted as $C(x)$. When inputting a new instance $y = \{y_1, y_2 ... y_q\}$, a classifier using some learning algorithm will predict its class.

The aim of classification is to predict the class rather than the faults of a new instance. When predicting the class of a new instance, the new instance can be compared with existing instances, and the class of the most similar instance is the predicted class of the new instance or the k most similar instances are used to determine the predicted class via some voting rules. Investigations about voting rules can be found in Refs. [20,47,50]. If each class is viewed as a fault, a classification problem can be viewed as a special case based diagnosis problem.

Therefore, our approach to case based diagnosis can also be applied to classification. The calculated overall similarity of the two instances can be defined as a linear combination of the similarity in each feature. The real similarity between two instances whose classes have been assigned can be defined as the similarity between the corresponding classes. For example, in Eq. (8), similarity between two instances in the same class is defined as 1, and similarity between two instances in different classes is defined as 0.

$$S_{ij} = \begin{cases} 1, & \text{if instance } i \text{ and instance } j \text{ belong to the same class} \\ 0, & \text{if instance } i \text{ and instance } j \text{ belong to different classes} \end{cases} \quad (8)$$

Discussions about the relationship between case based diagnosis and classification can also be found in Refs. [31,43].

6.2. Distance functions

In many approaches, distance functions are used to measure the similarity between two cases. Actually, there are many distance functions reported in the literature, including Euclidean [18], Minkowsky [10] and Camberra and Chi-square [38] etc. More comprehensive discussions about distance functions can be found in Refs. [55,56]. In this paper, we directly use a similarity function instead of distance functions. Although the similarity function used in this paper is a linear combination of feature similarities, many different distance functions can be represented by our similarity function using properly defined normalisation functions. We demonstrate here as an example that the widely used Euclidean distance function can be represented by our similarity function.

The Euclidean distance function is represented as Eq. (9).

$$D_{ij} = \sqrt{\sum_{k=1}^{q} (A_{ik} - A_{jk})^2} \quad (9)$$

In Eq. (9), D_{ij} is the distance between case i and case j, q is the number of features, and A_{ik} is the value of case i in feature k. Supposing Max_k is the maximum value difference in feature k, we define the normalisation function for overall similarity between case i and case j in Eq. (10) and the normalisation function for similarity between case i and case j in feature k in Eq. (11).

$$S_{ij} = \frac{\sum_{k=1}^{q} \text{Max}_k^2 - D_{ij}^2}{\sum_{k=1}^{q} \text{Max}_k^2} \quad (10)$$

$$S_{ijk} = \frac{\text{Max}_k^2 - (A_{ik} - A_{jk})^2}{\text{Max}_k^2} \quad (11)$$

From Eq. (10), we can have $D_{ij}^2 = \sum_{k=1}^{q} \text{Max}_k^2 - S_{ij} \sum_{k=1}^{q} \text{Max}_k^2$, and from Eq. (11), we can have $(A_{ik} - A_{jk})^2 = \text{Max}_k^2 - S_{ijk}\text{Max}_k^2$. By substituting D_{ij} and $(A_{ik} - A_{jk})^2$ in Eq. (9) with the above results, we can get the equation in Eq. (12), and therefore, we can get the similarity function in Eq. (13). Therefore, using the distance function in Eq. (9) is actually using special weight settings in our similarity function, in which formula (11) is used to calculate similarity in each feature and formula (13) is used to calculate the overall similarity.

$$\sum_{k=1}^{q} \text{Max}_k^2 - S_{ij} \sum_{k=1}^{q} \text{Max}_k^2 = \sum_{k=1}^{q} \left(\text{Max}_k^2 - S_{ijk}\text{Max}_k^2\right) \quad (12)$$

$$S_{ij} = \sum_{k=1}^{q} S_{ijk} \frac{\text{Max}_k^2}{\sum_{l=1}^{q} \text{Max}_l^2} \quad (13)$$

6.3. Feature weight setting algorithms

Good surveys of previous works on feature weighting can be found in Refs. [3,53]. We only summarise the main results in them. In most methods, each feature will usually be assigned only one weight. Therefore, the feature weight setting problem is usually viewed as a search problem in the weight space, which is exponential in the number of features (i.e. $O(y^q)$), where y is the number of different weight settings a feature can have and q is the number of features. Most approaches use the hill climbing technique or its variants to do the search, such as Refs. [1,15,28,34,36,45,54]. Some other approaches directly calculate weights using the training cases and/or some other knowledge, such as Refs. [16,48]. None of them can definitely get the optimal weight settings. In Ref. [32], an optimal weight setting method in the sense of minimising predictive errors is reported, but this method is restricted to two features and is optimal only for predicting the most similar case. Other efforts on optimal weight settings can be found in Refs. [39,46], which are based on other optimal criteria.

An alternative approach to feature weighting is to assign a case weight to each case besides feature weights. The underlying assumption is that some cases are more important than others are. Efforts on case weighting can be found in Refs. [6,7,45,52]. In our approach, if we remove the constraints on the sum of the weights of one case (see Tables 1 and 2), or let the constraints to be $\sum_{k=1}^{q} W_{ik} \leq 1$ and $\sum_{k=1}^{q} W_k \leq 1$, the effect of letting different cases have different importance will also be achieved. The larger the sum of the weights of the case is, the more important the case is. Therefore, our approach can also address the issues previously addressed by case weighting approaches.

6.4. Linear discriminant functions

In the literature, there have been reports on using linear discriminant functions to solve classification problems (see e.g. [24,33]). The basic idea of linear discriminant function approaches is to use the linear combination of the values in the features of an instance to estimate the class of the instance. A main task of a linear discriminant function approach is to calculate the combinatory coefficients in the linear discriminant function using the training data, and therefore, the LP technique can be used for the calculation [19]. In this paper, we use linear combination of feature similarities to calculate the similarity between two cases. The main difference between our approach and the of linear discriminant function approaches is that the feature weights in our approach are influence coefficients in the similarity function rather than the coefficients in the discriminant function.

7. Concluding remarks

In this paper, we have proved that the two kinds of optimal weight setting in a general form of case based diagnosis can be formalised as LP problems. As LP problems are polynomial time solvable, optimal weight settings can be calculated in polynomial time rather than searching the exponential weight space using heuristics to get sub-optimal settings. Optimal weight settings calculated in our method are in the sense of minimising predictive errors, which is quite in accordance with common sense. Our preliminary empirical results also show that our approach can not only calculate the theoretically optimal feature setting, but also be used to solve real world problems quite accurately.

We demonstrate that we have solved the optimal feature weight setting problem in a general form of case based diagnosis, and therefore, some alternative approaches focusing on various distance functions and case weight setting can actually be represented by our approach. On the basis of our approach, the global weighting can be treated as a simplification of the local weighting, and therefore, the two approaches can be directly compared with each other on the same ground. We also demonstrate that classification problems can be viewed as a subset of case based diagnosis problems, and therefore, our approach is also applicable to classification.

Future works include further testing the performance of our approach using more real world data and investigating LP algorithms that are particularly efficient for weight setting in case based diagnosis.

References

[1] D.W. Aha, Tolerating noise, irrelevant, and novel attributes in instance-based learning algorithms, International Journal of Man–Machine Studies 36 (1992) 267–287.

[2] D.W. Aha, The omnipresence of case-based reasoning in science and application, Knowledge-Based Systems 11 (5–6) (1998) 261–273.

[3] D.W. Aha, Feature weighting for lazy learning algorithms, in: H. Liu, H. Motoda (Eds.), Feature Extraction, Construction and Selection: a Data Mining Perspective, Kluwer, Norwell, MA, 1998.

[4] K.-D. Althoff, A. Birk, C.G. von Wangenheim, C. Tautz, Experimental software engineering, Case-Based Reasoning Technology: From Foundation to Applications, Mario Lenz, Brigitte Bartsch-Sporl, Hans-Dieter Burkhard, Stefan Wess (Eds.), Lecture Notes in Artificial Intelligence (1998) 1400.

[5] E. Andersen, J. Gondziio, C. Meszaro, X. Xu, Implementation of interior point methods for large scale linear programming, in: Tamas Terlaky (Ed.), Interior Point Methods in Mathematical Programming, Kluwer, Boston, MA, 1996.

[6] C. Atkeson, Using local models to control movement, in: D.S. Touretzky (Ed.), Advances in Neural Information Processing Systems 2, Morgan Kaufmann, San Mateo, CA, 1989.

[7] C. Atkeson, A. Moore, S. Schaal, Locally weighted learning, Artificial Intelligence Review 11 (1997) 11–73.

[8] E. Auriol, R.M. Crowder, R. McKendrick, R. Rowe, Integrating case-based reasoning and hypermedia documentation: an application for the diagnosis of a welding robot at Odense steel Shipyard, Engineering Applications of Artificial Intelligence 12 (1999) 691–703.

[9] R. Bareiss. PROTOS: a unified approach to concept representation, classification and learning. PhD Dissertation, University of Texas at Austin, Department of Computer Science, Technical Report AI88-AI83, 1988.

[10] B.G. Batchelor, Pattern Recognition: Ideas in Practice, Plenum Press, New York, 1978.

[11] R. Bergmann, H. Munoz-Avila, M. Veloso, E. Melis, CBR applied to planning, Case-Based Reasoning Technology: From Foundation to Applications, Mario Lenz, Brigitte Bartsch-Sporl, Hans-Dieter Burkhard, Stefan Wess (Eds.), Lecture Notes in Artificial Intelligence (1998) 1400.

[12] D. Bertsimas, J.N. Tsitsiklis, Introduction to Linear Optimization, Athena Scientific, Belmont, MA, 1997.

[13] C.L. Blake, C.J. Merz. UCI Repository of Machine Learning Databases (http://www.ics.uci.edu/~mlearn/MLRepository.html). University of California, Department of Information and Computer Science, Irvine, CA, 1998.

[14] K. Borner, CBR for design, Case-Based Reasoning Technology: From Foundation to Applications, Mario Lenz, Brigitte Bartsch-Sporl, Hans-Dieter Burkhard, Stefan Wess (Eds.), Lecture Notes in Artificial Intelligence (1998) 1400.

[15] C. Cardie. Using decision trees to improve case-based learning, Proceedings of the Tenth ICML, pp. 25–32, Amherst, MA, 1993.

[16] R.H. Creecy, B.M. Masand, S.J. Smith, D.L. Waltz, Trading mips and memory for knowledge engineering, Communications of the ACM 35 (1992) 48–64.

[17] G.B. Dantzig, Linear Programming and Extensions, Princeton University Press, Princeton, NJ, 1963.

[18] B.V. Dasarathy, Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques, IEEE Computer Society Press, Los Alamitos, CA, 1991.

[19] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Recognition, second ed, Wiley, New York, 2001 pp. 215–281.

[20] S.A. Dudani, The distance-weighted *k*-nearest-neighbor rule, IEEE Transactions on Systems, Man and Cybernetics 6 (4) (1976) 325–327.

[21] V. Ficet-Cauchard, C. Porquet, M. Revenu, CBR for the management and reuse of image-processing expertise: a conversational system, Engineering Applications of Artificial Intelligence 12 (1999) 733–747.

[22] E. Fix, J.L. Hodges. Discriminatory analysis, nonparametric discrimination, consistency properties, Technical Report 4, United States Air Force, School of Aviation Medicine, 1951.

[23] K. Fukunaga, T. Flick, An optimal global nearest neighor metric, IEEE Transactions on Pattern Analysis and Machine Intelligence 6 (1984) 314–318.

[24] A.J. Grove, N. Littlestone, D. Schuurmans, General Convergence Results for Linear Discriminant Updates, Proceedings of the COLT 97, ACM Press, New York, 1997 pp. 171–183.

[25] IBM. Optimization Solutions and Library (Version 3), (http://www6.software.ibm.com/sos/osl/optimization.htm), 2001.

[26] N. Karmarkar, A new polynomial-time algorithm for linear programming, Combinatorica 4 (4) (1984) 373–395.

[27] L.G. Khachian, A polynomial algorithm in linear programming, Soviet Mathematics Doklady 20 (1979) 191–194.

[28] K. Kira, L.A. Rendall, The Feature Selection Problem: Traditional Methods and a New Algorithm, Proceedings of the Tenth IJCAI, AAAI Press, San Jose, CA, 1992 pp. 129–134.

[29] V. Klee, G.J. Minty, How good is the simplex algorithm? in: O. Shisha (Ed.), Inequalities—III, Academic Press, New York, NY, 1972 pp. 159–175.

[30] J. Kolodner, Case-Based Reasoning, Morgan Kaufmann, San Mateo, CA, 1993.

[31] M. Lenz, E. Auriol, M. Manago, Diagnosis and decision support, Case-Based Reasoning Technology: From Foundation to Applications, Mario Lenz, Brigitte Bartsch-Sporl, Hans-Dieter Burkhard, Stefan Wess (Eds.), Lecture Notes in Artificial Intelligence (1998) 1400.

[32] C.X. Ling, H. Wang, Towards optimal weights setting for the 1-nearest neighbor learning algorithm, Artificial Intelligence Review 11 (1997) 255–272.

[33] N. Littlestone, Learning quickly when irrelevant attributes abound: a new linear-threshold algorithm, Machine Learning 2 (4) (1988) 285–318.

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

[35] I. Lustig, R. Marsten, D. Shanno, Interior point methods: computational state of the art, ORSA Journal on Computing 6 (1994) 1–14.

[36] O. Maron, A.W. Moore, The racing algorithm: model selection for lazy learners, Artificial Intelligence Review 11 (1997) 192–225.

[37] R. Marsten, R. Subramanian, M. Saltzman, I. Lustig, D. Shanno, Interior point methods for linear programming: just call Newton, Lagrange, and Fiacco and McCormick! Interfaces 20 (4) (1990) 105–116.

[38] R.S. Michalski, R.E. Stepp, E. Diday, A recent advance in data analysis: clustering objects into classes characterized by conjunctive concepts, in: Kanal, Rosenfeld (Eds.), Progress in Pattern Recognition, vol. 1, North-Holland, New York, 1981, pp. 35–56.

[39] M. Mohri, H. Tanaka, An optimal weighting criterion of case indexing for both numeric and symbolic attributes, in: Aha (Ed.), Case-Based Reasoning: Papers from the 1994 Workshop, AAAI Press, Menlo Park, CA, 1994.

[40] P. Perner, An architecture for a CBR image segmentation system, Engineering Applications of Artificial Intelligence 12 (1999) 749–759.

[41] B. Porter, R. Bareiss, R. Holte, Concept learning and heuristic classification in weak theory domains, Artificial Intelligence 45 (1–2) (1990) 229–263.

[42] H. Praehofer, J. Kerschbaummayr, Case-based reasoning techniques to support reusability in a requirement engineering and system design tool, Engineering Applications of Artificial Intelligence 12 (1999) 717–731.

[43] F. Puppe, Systematic Introduction to Expert Systems, Springer, Berlin, 1993.

[44] R. Quinlan, Combining Instance-Based and Model-Based Learning, Proceedings on the Tenth International Conference of Machine Learning, Morgan Kaufmann, University of Massachusetts, Amherst, 1993 pp. 236–243.

[45] S.L. Salzberg, A nearest hyperrectangle learning method, Machine Learning 6 (1991) 251–276.

[46] K. Satoh, S. Okamoto, Toward PAC-learning of weights from qualitative distance information, in: Aha (Ed.), Case-Based Reasoning: Papers from the 1994 Workshop, AAAI Press, Menlo Park, CA, 1994.

[47] D.F. Specht, Enhancements to probabilistic neural networks, Proceedings of International Joint Conference on Neural Networks (IJCNN'92), vol. 1 (1992), pp. 761–768.

[48] C. Stanfill, D. Waltz, Toward memory-based reasoning, Communications of the ACM 29 (1986) 1213–1228.

[49] A. Varma, N. Roddy, ICARUS: design and deployment of a case-based reasoning system for locomotive diagnosis, Engineering Applications of Artificial Intelligence 12 (1999) 681–690.

[50] P.D. Wasserman, Advanced Methods in Neural Computing, Van Nostrand Reinhold, New York, NY, 1993 pp. 147–176.

[51] I. Watson, F. Marir, Case-based reasoning: a review, The Knowledge Engineering Review 9 (4) (1994).

[52] D. Wettschereck. Weighted *k*NN versus majority *k*NN: a recommendation, Technical Report 943, Sankt Augustin, Germany, German National Research Center for Computer Science, Artificial Intelligence Research Division, 1995.

[53] D. Wettschereck, D.W. Aha, T. Mohri, A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms, Artificial Intelligence Review 11 (1997) 273–314.

[54] D.R. Wilson, T.R. Martinez. Instance-based learning with genetically derived attribute weights, Proceedings of the International Conference on Artificial Intelligence, Expert Systems, and Neural Networks, pp. 11–14, 1996.

[55] D.R. Wilson, T.R. Martinez, Improved heterogeneous distance functions, Journal of Artificial Intelligence Research (JAIR) 6 (1) (1997) 1–34.

[56] D.R. Wilson, T.R. Martinez, Reduction techniques for instance-based learning algorithms, Machine Learning 38 (2000) 257–286.