Interactive Knowledge Validation and Query Refinement in CBR [*]

Monica H. Ou, Geoff A.W. West, Mihai Lazarescu
Department of Computing,
Curtin University of Technology
GPO Box U1987, Perth 6845, Western Australia, Australia
{ou, geoff, lazaresc}@cs.curtin.edu.au

Chris Clay
Royal Perth Hospital
Perth, Western Australia, Australia
claycd@iinet.net.au

Abstract

In most case-based reasoning (CBR) systems there has been little research done on validating new knowledge, specifically on how previous knowledge differs from current knowledge as a result of conceptual change. This paper proposes two methods that enable the domain expert, who is non-expert in artificial intelligence (AI), to interactively supervise the knowledge validation process in a CBR system, and to enable dynamic updating of the system, to provide the best diagnostic questions. The first method is based on formal concept analysis which involves a graphical representation and comparison of the concepts, and a summary description highlighting the conceptual differences. We propose a dissimilarity metric for measuring the degree of variation between the previous and current concepts when a new case is added to the knowledge base. The second method involves determining unexpected classification-based association rules to form critical questions as the knowledge base gets updated.

Introduction

Case-base reasoning (CBR) is a popular approach that has been applied to various domains, with most of the research having been focused on the classification aspect of the system. In medical applications, CBR has been used with considerable success for patient diagnosis (Watson 1997). However, relatively little effort has been put into investigating how new knowledge can be validated. We have developed a web-based diagnostic system to assist general practitioners (GPs) to diagnose patients with dermatological problems. This paper describes tools developed to automatically assist a dermatology consultant to train and validate the knowledge in the CBR system, and to enable dynamic updating of the system, to provide the GPs with the best diagnostic questions. Note that the consultants and GPs are non-computing experts.

Knowledge validation continues to be problematic in knowledge-based and case-based systems due to the modelling nature of the task. In medical applications it is generally desirable not to have automatic updates, as automatic updating (without supervision) can lead to increase inconsistency in the cases, resulting in reduced diagnostic performance. Normally CBR systems are allowed to learn by themselves, in which case the user enters the new case, compares it with those in the knowledge base and, once satisfied, adds the case to the database. In our method, the consultant needs to interactively supervise the CBR system, and the valid cases are determined by the validation tools and chosen by the consultant. The inconsistent or ambiguous cases can then be visualised and handled (modified or rejected) by the consultant. The reason for human supervision is to ensure that the decisions and learning are correct, and to prevent contradictory cases from being involved in the classification process. A vital part of this is to constantly check and maintain the quality of the data in the database as new cases get added.

In addition, we propose a method for validating existing knowledge by uncovering and ranking *unexpected rules* to dynamically form questions used for the query process. In this paper, an unexpected rule is defined as a rule that is not obvious in the consultant's perception. Unexpected rules are useful for discovering new knowledge/concepts that the consultant is unaware of, and checking the validity of the rules. The latter reflects the quality of data in the database. The main aspects that need to be addressed are:

1. How to provide non-computing experts with a simple but effective mechanism for incrementally validating the knowledge base.
2. How to provide a way of measuring the conceptual variation between the previous and new knowledge.
3. How to check if the questions in the diagnostic system are correct and comprehensive.

This paper proposes a new approach for validating the consistency of the newly acquired knowledge against the past knowledge using a decision tree classifier (Quinlan 1993), Formal Concept Analysis (FCA) (Cole & Amardeilh 2003; Díaz-Agudo & Gonzalez-Calero 2001; Ganter & Wille 1999; Ganter 2000; Richards 1999; 2000). We propose a dissimilarity metric for quantifying the level of conceptual changes between the previous and current knowledge. Conceptual graph comparison has been widely studied in the area of information retrieval (Montes-y Gómez *et al.* 2001; Yeh, Porter, & Barker 2003; Zhong *et al.* 2002).

[*]The research reported in this paper has been funded in full by a grant from the AHMAC/SCRIF initiative administered by the NHMRC in Australia.

However, these techniques are not suitable for users who are non-experts in AI due to their complexity.

Decision trees use only the features they need to unambiguously classify cases. There are typically many other features that are in the cases and may be useful for helping a GP reach a diagnosis. To determine if these attributes (in combination with those already used in decision tree) are useful we use classification-based association rule mining such as CMAR[1] to generate additional characteristic rules. CMAR generates a large number of rules depending on the selected support and confidence thresholds. In particular we are interested in unexpected rules, which are determined and ranked by a distance matrix measure for a tree (Wang *et al.* 2003) to dynamically generate critical questions for the diagnostic query.

Knowledge Validation and Query Refinement

The decision support system is used by the GPs to assist in diagnosis. This results in new cases being generated. These are stored in the database and marked as "unchecked". The consultant can choose whether or not to add the cases to the knowledge base. We apply FCA for checking the validity of the new cases as opposed to manually checking the rules generated by the decision tree. This is important because manual rule inspection by the human user quickly becomes unmanageable as the database grows in size. In general, if the consultant disagrees with the diagnosis then the correct diagnosis needs to be specified and if necessarily features need to be modified to justify the new diagnosis. In some cases, if the consultant cannot solve the ambiguity then the instances are stored in a repository for later diagnosis.

The knowledge validation and query refinement process is shown in Figure 1. The knowledge validation stage involves using J48 (Witten & Frank 2000), the Weka[2] implementation of the C4.5 decision tree algorithm (Quinlan 1993) for inducing the rules, and the Galicia[3] implementation for generating lattices. The attribute-value pairs of the rules are automatically extracted and represented as a context table which shows relationships between the features and the diagnoses. The context table is converted to a lattice for easy visualisation. As each new case gets added, the context table gets updated and a new lattice is generated. If adding a checked case will drastically change the lattice, the consultant is alerted and asked to confirm that the new case is truly valid given its effect on the lattice. The query refinement stage identifies unexpected rules. First, the consultant groups attributes in a hierarchy according to their type. Second, based on the hierarchy, we calculate the distances between the attributes using a distance matrix. The distance matrix is built based on the minimum number of edges between one node and another. Finally, rules are ranked according to their distance values. The rules with the highest distance are assumed to be the most unexpected. There are four different options available to assist the consultant in performing knowledge validation:

1. A graphical representation of the lattices that enables the consultant to visualise the conceptual differences.
2. A summary description highlighting the conceptual differences.
3. A measure which determines the degree of variation between lattices.
4. A distance matrix which determines the list of unexpected decision rules.

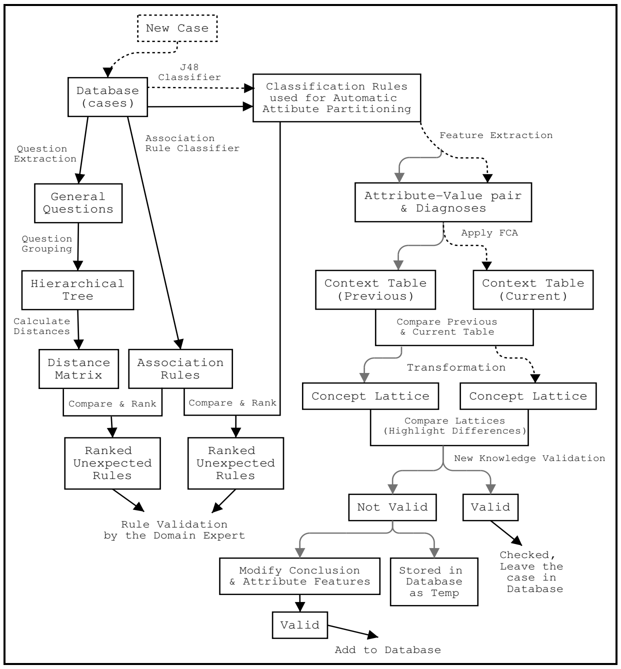

Figure 1: Knowledge validation and query refinement process.

Dermatology Dataset

The dataset we use in our experiments consists of patient records for the diagnosis of dermatological problems. The dataset is provided by a consultant dermatologist. It contains patient details, symptoms and importantly, the consultant's diagnoses. Each patient is given an identification number, and episode numbers are used for multiple consultations for the same patient. Currently, the data has 17 general attributes and consists of cases describing 32 different diagnoses. Data collection is a continuous process in which new cases get added to the database.

Of interest to the consultant is how each new case will affect the knowledge base. New cases are collected in two ways: 1) the diagnosed cases provided by the GP, and 2) cases provided by the consultant. Before the consultant validates the new cases, they are marked as "unchecked" and combined with cases already in the database (previously "checked") for training and generating lattices. If the lattices do not show any ambiguity, the new cases are valid and they will be updated to "checked" by the dermatologist.

[1] www.csc.liv.ac.uk/∼frans/KDD/Software/CMAR

[2] www.cs.waikato.ac.nz/ml/weka

[3] www.iro.umontreal.ca/∼galicia/index.html

Additionally, we use the “Dermatology” dataset[4] to illustrate the attribute grouping concept for identifying and ranking unexpected rules. The dataset contains 366 cases with 12 historical features.

Formal Context and Concept Lattices

FCA enables a lattice to be built automatically from a context table. Context tables need to use discrete or binary formats to represent the relations between attributes and objects. Therefore, the decision tree algorithm is used to automatically partition continuous attribute values into discrete values.

For the new unchecked case, the context table may or may not be affected to reflect the changes in characteristics used for describing the diagnoses. When a new case is added, some characteristics will be added and others removed, due to the decision tree partitioning mechanism.

Conceptual changes are determined by comparing the relationships between the characteristics and the diagnoses of the current and previous context tables. However, the comparison can often be done more effectively using a lattice representation, shown in Figures 2 and 3.

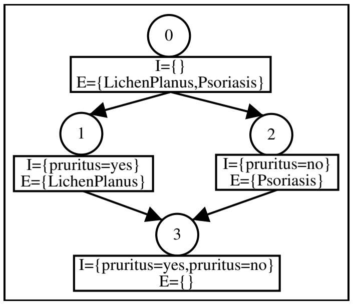

Figure 2: Previous graph G_{t-1} (7 cases).

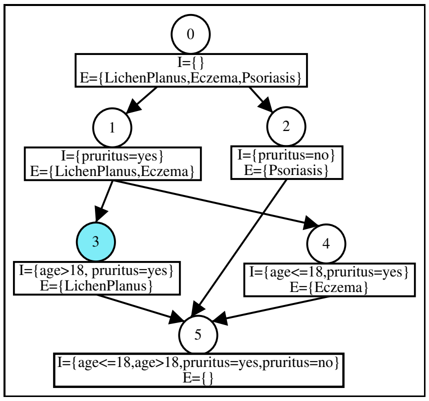

Figure 3: Current graph G_t (8 cases).

When a new case is added to the database, a new lattice is generated. If we compare the previous and current lattices, we expect the lattices to be the same or with only minor variations if changes do occur. If the concept changes significantly, the consultant needs to check if the new case is valid. For each invalid case, the consultant is required to change the diagnosis or the characteristics to satisfy the new case or store it in a repository for later consideration. The validation task is vitally important to prevent contradictory cases from being used in the classification process as these could lead to inaccurate diagnoses.

Conceptual Dissimilarity Measure

We propose a dissimilarity metric for determining the level of change between the two lattices. Emphasis is put on illustrating the changes in the characteristics of the diagnoses. The measure is particular useful when the lattices become too large to be manually visualised by the consultant. This section briefly describes the proposed technique. A full description can be found in Ou *et al.* (2005).

Highlighting Conceptual Changes between Lattices

An algorithm for determining the conceptual differences between the two lattices: G_{t-1} and G_t, where $t-1$ and t represent lattices derived from previous and current cases, respectively. The algorithm determines which concepts are missing from G_{t-1} and which have been added to G_t. The system displays the characteristics and diagnoses that have been ignored or added to the classification process. Based on this information the consultant does not need to manually analyse the lattice, which would be time consuming and error-prone.

Consider the two simple examples in Figures 2 and 3. In Figure 3, the added node 3 is selected for processing. The algorithm iterates down to the lowest child node that contains *LichenPlanus* with the characteristics of *age>18 and pruritus=yes*, which is in node 3. Then the algorithm performs the matching between the two graphs. The results show that node 1 of G_{t-1} matches node 3 of G_t. The algorithm compares and highlights the differences between the two nodes. In this case, the characteristic *age>18* has been introduced in G_t to give the diagnosis *LichenPlanus* a more detailed description. The algorithm repeats the same steps for other missing/added nodes. Generally, characteristics for describing the diagnosis remains unchanged in both G_{t-1} and G_t, or is removed if it has become insignificant. In some cases, new diagnoses are introduced, as shown in node 4, as the result of adding new cases, and this affects the total dissimilarity value.

Dissimilarity Metric

Conceptual variation between graphs G_{t-1} and G_t when a new case is added is derived as follows in Ou *et al.* (2005).

Let $c(G_j)$ be the number of diagnoses affected after node i is removed/added to G_j, where $i = 1, 2, 3, ..., n$, and $j = t-1, t$; n is the total number of nodes removed/added from G_j; $C(G_j)$ is the total number of diagnoses in G_j, $a(G_{ji})$ is the number of features that exist in node i of G_{t-1} but not in G_t, and vice versa; $A(G_{ji})$ is the total number of features in node i; $h(G_j)$ is the height of G_j; and $l(G_{ji})$ is the level

[4] www.cormactech.com/neunet/download.html

of node i from top of the graph to the current position of the node. The variation measure $v(G_j)$ for each graph is:

$$v(G_j) = \begin{cases} v_1(G_j) & : \quad h(G_j) = 1 \\ v_2(G_j) & : \quad h(G_j) > 1 \end{cases} \qquad (1)$$

where $v(G_j) = \{v_1(G_j) | v_2(G_j)\}$, and $v_1(G_j)$ and $v_2(G_j)$ are defined as:

$$\begin{aligned} v_1(G_j) &= \sum_{i=0}^{n} \frac{c(G_j)}{C(G_j)} . \frac{a(G_{ji})}{A(G_{ji})} . \frac{1}{n+1} \\ v_2(G_j) &= \sum_{i=0}^{n} \frac{c(G_j)}{C(G_j)} . \frac{a(G_{ji})}{A(G_{ji})} . \frac{h(G_j) - l(G_{ji})}{h(G_j) * n} \end{aligned}$$

The total dissimilarity measure $d(G)$ between G_{t-1} and G_t is then defined as:

$$d(G) = v(G_{t-1}) + v(G_t); 0 \leq d(G) \leq 1 \qquad (2)$$

Query Refinement using Unexpected Rules

This section describes the second technique to assist the consultant in uncovering new or unexpected knowledge. This technique allows the consultant to validate the quality of the rules and to ensure the data in the database is consistent. In addition, it makes the query process dynamic in the sense that the questions used in the query as well as the order in which they are presented changes to reflect the knowledge being accumulated over time. The technique uses a distance matrix on the attribute groupings obtained from the domain expert for measuring the unexpectedness of the discovered knowledge, and an association rule mining algorithm for generating rules that are hidden in the dataset and might not be included in the classification rules obtained by C4.5. However, association rule mining usually generates a large set of rules that are obvious to the domain experts. Therefore, it is important to eliminate these rules, and only display those that are unexpected. The rules are ranked from the most unexpected to the least unexpected.

Grouping of Similar Attributes

We have explored two different approaches for determining unexpected and commonsense rules.

1. Domain Expert Grouping: This part of the experiment uses the "Dermatology" dataset. The consultant is required to manually group the attributes in a hierarchy according to their type. Figure 4 shows how the groupings are structured as a tree. Leaf nodes represent all the possible attributes. The dissimilarity between each pair of attributes is determined by the distance between the two leaf nodes. We calculate the unexpectedness of a rule by taking the maximum dissimilarity between any pair of attributes.

   The distance measure between the attributes in the leaf nodes is defined as:

   $$distance(i, j) = d_i + d_j - 2d_{p(i,j)} \qquad (3)$$

   where i and j are nodes representing antecedents in the association rules; d_i and d_j are the depth of nodes i and j, respectively; and $d_{p(i,j)}$ is the depth of the lowest common parent node of nodes i and j from root.

   We apply Equation 3 to the groupings in Figure 4 to derive the distance matrix shown in Table 1. As can be seen from the table, the further apart the leaf nodes the greater the distance between them.

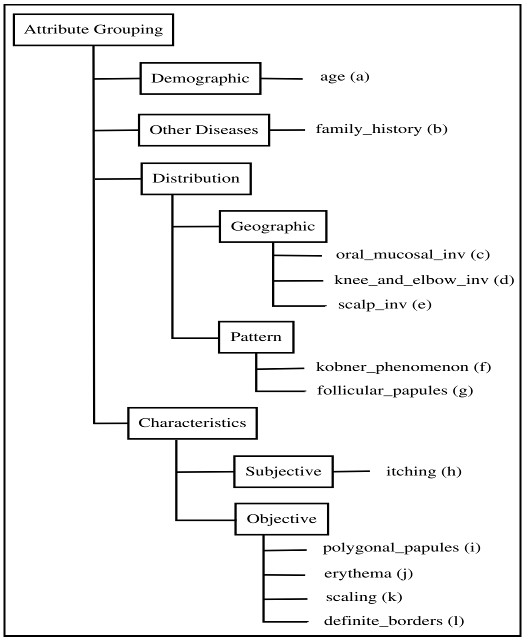

Figure 4: Hierarchical attribute grouping by the consultant.

Table 1: Distance matrix derived from Figure 4.

	a	b	c	d	e	f	g	h	i	j	k	l
a	0	4	5	5	5	5	5	5	5	5	5	5
b	4	0	5	5	5	5	5	5	5	5	5	5
c	5	5	0	2	2	4	4	6	6	6	6	6
d	5	5	2	0	2	4	4	6	6	6	6	6
e	5	5	2	2	0	4	4	6	6	6	6	6
f	5	5	4	4	4	0	2	6	6	6	6	6
g	5	5	4	4	4	2	0	6	6	6	6	6
h	5	5	6	6	6	6	6	0	4	4	4	4
i	5	5	6	6	6	6	6	4	0	2	2	2
j	5	5	6	6	6	6	6	4	2	0	2	2
k	5	5	6	6	6	6	6	4	2	2	0	2
l	5	5	6	6	6	6	6	4	2	2	2	0

2. Decision Tree Classification Grouping: Decision trees only use features they need to unambiguously classify cases. However, there are many other useful features that exist in the cases that have not been extracted by the decision tree classifier. Therefore, we need to use an association rule mining technique to provide a full description of the diseases to assist the query process. This automatic approach dynamically generates a set of commonsense and unexpected classification-based association rules. To identify the commonsense/unexpected rules, the antecedence of the association rules are com-

pared to those generated by the decision tree as association rule is regarded as unexpected if the antecedents do not match the antecedents of the decision tree rules. The association rules can then be ranked from most unexpected to least unexpected. We are particularly interested in association rules that are highly dissimilar to the decision tree rules. The commonsense and unexpected rules provide better questions for the query process and help a GP reach a correct diagnosis.

Ranking Rules based on Unexpectedness

Association rules are generated based on the defined support and confidence thresholds. It is important to keep set the confidence threshold high, because the rules need to be consistent. For example, given a set of symptoms (i.e. rule antecedents), the corresponding diagnosis (i.e. consequents) must always be found and valid based on the input.

The association rule mining algorithm generates a large number of rules which makes it almost impossible for the consultant to interpret and extract any useful rules. Therefore, we need to rank the rules according to the distance shown in the matrix. The highly unexpected rules are shown on the top of the list. By presenting the unexpected rules to the consultant, they will be alerted to new knowledge.

Results and Discussion

Discussion and testing with the consultant revealed the effectiveness of the proposed techniques. First, the consultant recommended the grouping of the dissimilarity values into five different categories. The categories provide a qualitative measure of the changes ranked from “No change” to “Radical change”. The value of less than 0.10 is considered “No change”.

The evaluation process analysed how the quality of the existing cases in the database is affected by the update process. It is important to note that the decision tree classifier built from the cases gives 100% correct classification and hence the lattices reflect perfect classification. First, we considered the case where no consultant interaction occurred and allowed the CBR to incrementally add new cases. Figure 5 shows the dissimilarity values as each new case is added. As can be seen, there are significant and minor variations between the previous and current concepts as we incrementally update the database (indicated by the variations in the dissimilarity values). This is expected since the decision tree classifier repartitions the attributes to get the best classification, causing the context tables to change and hence change the lattices.

In general, the dissimilarity values decrease as the number of cases used for training increase. The decreasing trend shows the CBR system increases the consistency, and this means the classifier is becoming more stable and generalising better. This is shown by the linear regression line that indicates a steady decrease as new cases are added. To check the results for consistency, the cases are randomly added to the lattice one by one. After a number of trials, the results are similar to those shown in Figure 5 with decreasing regression lines.

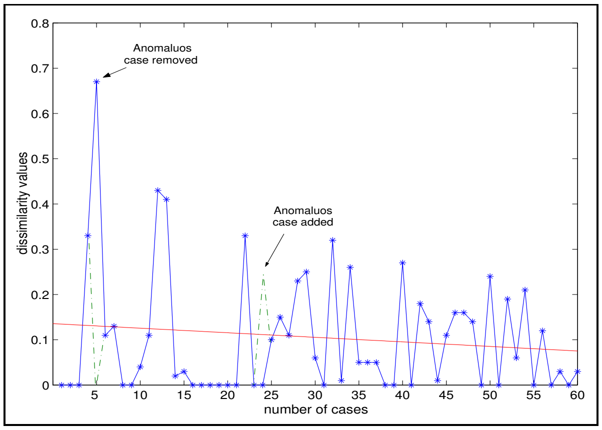

Figure 5: Dissimilarity values vs. number of new cases.

Rerunning the updating process with the consultant present reveals that the high values of dissimilarity in Figure 5 closely match the consultant’s perceptions of the new cases as they are added. For the two large peaks at cases 5 and 12 in Figure 5, the consultant examined the inconsistency, and observed some anomalies in case 4 and modified the features accordingly, resulting in lower ambiguity (represented by the dotted line). Many of the anomalies are due to the GPs’ and consultants’ interpretation of the symptoms. This can be quite subjective and checking of the cases reduces these anomalies[5]. Case 12, however, is considered to be valid and no modification has been made. The increase in the dissimilarity value at case 12 is due to the repartitioning of existing attributes plus some new ones selected for describing the reoccurring diagnosis (i.e. a diagnosis that exists multiple times in the database).

To further illustrate this, we randomly chose case number 24, which has a dissimilarity value of zero, and modified its characteristics to determine whether the value does increase to show that the modified case is no longer valid. The result shows a significant increase in the dissimilarity value (represented by the dashed line) which suggests that the case is no longer valid or consistent with other cases in the database, and this leads to a slight decrease in the accuracy of the decision tree classifier.

In addition, we have conducted a series of experiments to validate the ranking of unexpected rules. First, a list of unranked association rules is presented to the consultant. Preliminary results show that our method is able to extract both commonsense and unexpected rules. However, the method also extracted rules which the consultant did not consider interesting. Our future work will focus on the refinement of the commonsense/unexpected rules. One avenue of investigation is to use weighted distances for the consultant-specified attribute groupings. An alternative method to extract and rank association rules is to use the decision tree groupings. Initial results indicated that the method is effective for the rule ranking.

[5] Note this can also deal with other factors such as errors in data entry.

Conclusions

This paper makes two contributions. First, we described a method for the interactive knowledge validation in a teledermatology CBR system. The method uses concept lattices to determine the conceptual differences. These are presented in the form of 1) a graphical representation highlighting the differences, 2) a summary description of conceptual variations, and 3) a dissimilarity measure. Second, we presented a technique to extract and rank association rules according to their unexpectedness. The system was tested with the help of a consultant. The developed techniques have been evaluated by a dermatology consultant, and have shown to be useful for discovering ambiguous cases and keeping the database consistent. Furthermore, we obtained promising results from the initial testing of the algorithm used to extract and rank unexpected rules.

References

Cole, R. Eklund, P., and Amardeilh, F. 2003. Browsing Semi-structured Texts on the Web using Formal Concept Analysis. *Web Intelligence*.

Díaz-Agudo, B., and Gonzalez-Calero, P. A. 2001. Formal Concept Analysis as a Support Technique for CBR. *Knowledge-Based System* 7(1):39–59.

Ganter, B., and Wille, R. 1999. *Formal Concept Analysis: Mathematical Foundations*. Heidelberg: Springer.

Ganter, B. 2000. Computing with Conceptual Structures. In *Proc. of the 8th International Conference on Conceptual Structure*. Darmstadt: Springer.

Montes-y Gómez, M.; Gelbukh, A.; López-López, A.; and Baeza-Yates, R. 2001. Flexible Comparison of Conceptual Graphs. In *Proceedings of the 12th International Conference and Workshop on Database and Expert Systems Applications*, 102–111.

Ou, M. H.; West, G. A. W.; Lazarescu, M.; and Clay, C. 2005. Interactive Knowledge Validation in CBR for Decision Support in Medicine. In *Proceedings of the 10th Conference on Artificial Intelligence in Medicine (AIME 05) (To appear)*.

Quinlan, J. R. 1993. *C4.5 Programs for Machine Learning*. USA: Morgan Kaufmann Publishers.

Richards, D. 1999. Visualizing Knowledge Based Systems. In *Proceedings of the 3rd Workshop on Software Visualization*, 1–8.

Richards, D. 2000. The Visualisation of Multiple Views to Support Knowledge Reuse. In *Proceedings of the Intelligent Information Processing (IIP'2000) In Conjunction with the 16th IFIP World Computer Congress WCC2000.*

Wang, J. T. L.; Shan, H.; Shasha, D.; and Piel, W. H. 2003. TreeRank: A Similarity Measure for Nearest Neighbor Searching in Phylogenetic Databases. In *Proceedings of the 15th International Conference on Scientific and Statistical Database Management*, 171–181.

Watson, I. 1997. *Applying Case-Based Reasoning: Techniques for Enterprise Systems*. USA: Morgan Kaufmann Publishers.

Witten, I., and Frank, E. 2000. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. USA: Morgan Kaufmann Publishers.

Yeh, P. Z.; Porter, B.; and Barker, K. 2003. Using Transformations to Improve Semantic Matching. In *Proceedings of the International Conference on Knowledge Capture*, 180–189.

Zhong, J.; Zhu, H.; Li, J.; and Yu, Y. 2002. Conceptual Graph Matching for Semantic Search. In *Proceedings of the 10th International Conference on Conceptual Structures: Integration and Interfaces*, 92–106.