Interactive Knowledge Validation in CBR for Decision Support in Medicine

Monica H. Ou[1], Geoff A.W. West[1], Mihai Lazarescu[1], and Chris Clay[2]

[1] Department of Computing,
Curtin University of Technology
GPO Box U1987, Perth 6845, Western Australia, Australia
{ou, geoff, lazaresc}@cs.curtin.edu.au
[2] 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 by means of conceptual change. This paper proposes a technique that enables the domain expert who is non-expert in artificial intelligence (AI) to interactively supervise the knowledge validation process in a CBR system. The technique 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 developed technique has been evaluated by a dermatology consultant, and has shown to be useful for discovering ambiguous cases and keeping the database consistent.

1 Introduction

Case-base reasoning (CBR) is an artificial intelligence (AI) technique that attempts to solve new problems by using the solutions applied to similar cases in the past. Reasoning by using past cases 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, such as in PROTOS and CASEY [9]. However, relatively little effort has been put into investigating how new knowledge can be validated. We have developed a web-based CBR system that provides decision support to general practitioners (GPs) for diagnosing patients with dermatological problems. This paper mainly concentrates on developing a tool to automatically assist a dermatology consultant to train and validate the knowledge in the CBR system. 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 dealing with the problem it is often desired to design a CBR system that disallows automatic updates, especially in medical applications where lives could be threatened from incorrect decisions. Normally CBR systems are allowed to learn by themselves in which 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 in which the valid cases are determined by the validation tool 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. Adding contradictory cases into the knowledge base will negatively affect classification accuracy and data interpretation. Therefore, it is vital to constantly check and maintain the quality of the data in the database as new cases get added. The main aspects that need to be addressed are:

1. How to enable non-computing experts to easily and efficiently interact with the CBR system.
2. How to provide a simple but effective mechanism for incrementally validating the knowledge base.
3. How to provide a way of measuring the conceptual variation between the previous and new knowledge.

This paper proposes a new approach for validating the consistency of the new acquired knowledge against the past knowledge using a decision tree classifier [6], Formal Concept Analysis (FCA), and a dissimilarity metric. FCA is a mathematical theory which formulates the conceptual structure and displays relations between concepts in the form of concept lattices which comprise attributes and objects [3, 4]. A *concept* in FCA is seen as a set of objects that share a certain set of attributes. A lattice is a directed acyclic graph in which each node can have more than one parent, and the convention is that the superconcept always appears above all of its subconcepts. In addition, we derive a dissimilarity metric for quantifying the level of conceptual changes between the previous and current knowledge.

2 Related Work

FCA has been used as a method for knowledge representation and retrieval [1, 7], conceptual clustering [3], and as a support technique for CBR [2]. In [8] ripple-down rules (RDR) are combined with FCA to uncover an abstraction hierarchy of concepts and the relationship between them.

We use FCA and a dissimilarity measure approach to assist the knowledge validation process and highlight the conceptual differences between previous and current knowledge. Conceptual graph comparison has been widely studied in the area of information retrieval [5, 11, 12]. The comparison is categorised into two main parts: syntactic and semantic. Syntactic comparison uses a graphical representation, and determines their similarity by the amount of common structures shared [5, 11]. Semantic comparison measures whether the two concepts are similar to each other based on whether one concept representation can generalise the other [11, 12]. However, these techniques are not suitable for users that are non-experts in AI due to their complexity.

3 Interactive Knowledge Validation Approach in CBR

3.1 An Overview of the Knowledge Validation Process

Consider GPs using the CBR system to help in diagnosis. They generate new cases using the decision support system which are stored in the database and marked as “unchecked”. This means the cases need to be checked by the consultant before deciding whether or not to add them to the knowledge base. FCA is used for checking the validity of the new cases to minimise the validation time, as the consultant is not required to manually check the rules generated by the decision tree. This is important because manual rule inspection by the human user quickly becomes unmanageable as the rule database grows in size. In most cases, if the consultant disagrees with the conclusion the correct conclusion is needed to be specified and some features in the case selected to justify the new conclusion or the instances are stored in a repository for later use.

Figure 1 presents the process of knowledge validation. It involves using J48 [10], the Weka[3] implementation of the C4.5 decision tree algorithm [6] for inducing the rules, and the Galicia[4] implementation for generating lattices. The attribute-value pairs of the rules are automatically extracted and represented as a context table which shows relations between the features and the diagnoses. The context table gets converted to a lattice for easy visualisation via the hierarchical structure of the lattices. 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. There are three different options available to assist the consultant in performing knowledge validation:

1. A graphical representation of the lattices that enables the consultant to graphically visualise the conceptual differences.
2. A summary description highlighting the conceptual differences.
3. A measure which determines the degree of variation between lattices.

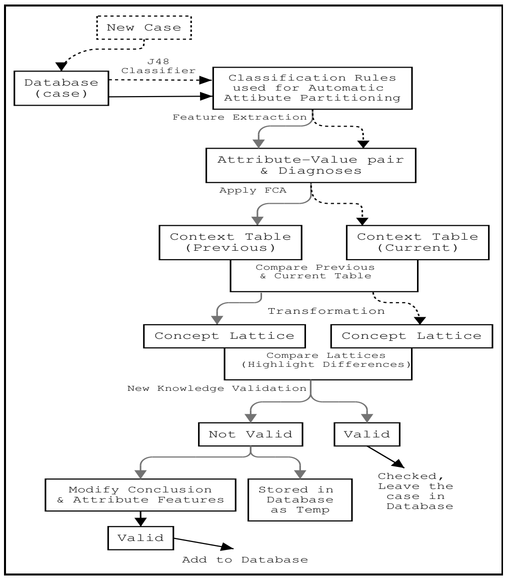
New Case
J48 Classifier
Database (case)
Classification Rules used for Automatic Attibute Partitioning
Feature Extraction
Attribute-Value pair & Diagnoses
Apply FCA
Context Table (Previous)
Context Table (Current)
Compare Previous & Current Table
Transformation
Concept Lattice
Concept Lattice
Compare Lattices (Highlight Differences)
New Knowledge Validation
Not Valid
Valid
Checked, Leave the case in Database
Modify Conclusion & Attribute Features
Stored in Database as Temp
Valid
Add to Database

Fig. 1. Knowledge base validation.

3.2 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

[3] www.cs.waikato.ac.nz/ml/weka [Last accessed: October 22, 2004].

[4] www.iro.umontreal.ca/ galicia/index.html [Last accessed: September 10, 2004].

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 this means the new cases are in fact valid and they will be updated to "checked" by the dermatologist.

3.3 Formal Context and Concept Lattices

FCA enables a lattice to be built automatically from a context table. The table consists of attribute-value pairs that are generated by the decision tree algorithm that automatically partitions the continuous attribute values into discrete values. This is because context tables need to use discrete or binary formats to represent the relations between attributes and objects. The formal context (C) is defined as follows:

Definition: A formal context $C = (D, A, I)$, where D is a set of diagnoses, A is a set of characteristics, and I is a binary relation which indicates diagnosis d has characteristic a.

Tables 1 and 2 show simple examples of formal contexts for our system using only a small number of cases for simplicity. Table 1 is derived from previously checked data which consists of 6 cases, and Table 2 is generated when a new unchecked case is added i.e. consists of 7 cases. The objects on each row represent the diagnosis D, each column represents a set of characteristics A, and the relation I is marked by "X". Note, only attributes that are relevant to a context are shown (i.e. allows discrimination), and that Table 2 shows that new attributes became relevant while others dropped. 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. As can be seen from the tables, some characteristics have been introduced while some have been removed when a new case is added due to the decision tree partitioning mechanism. In Table 1, *age≤29*, *age>29*, *gender=M*, and *gender=F* are considered to be important features for classifying *Eczema*, *Psoriasis*, and *PityriasisRubraPilaris*. However, in Table 2, *age≤29* and *age>29* are no longer important in disambiguating the diagnoses but *lesion_status=1*, *lesion_status=2*, *lesion_status=3*, and *lesion_status=4* are, and the characteristics *gender=M* and *gender=F* remain unchanged.

Table 1. Context table of previous data (consists of 6 cases).

	Characteristics			
Diagnoses (Concept Types)	age≤29	age>29	gender=M	gender=F
Eczema	X		X	
Psoriasis				X
PityriasisRubraPilaris		X	X	

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 that

Table 2. Context table of current data (consists of 7 cases).

	Characteristics					
Diagnoses (Concept Types)	gender=M	gender=F	lesion_status=1	lesion_status=2	lesion_status=3	lesion_status=4
Eczema	X			X	X	X
Psoriasis		X				
PityriasisRubraPilaris	X		X			

formulates the hierarchical structure of the concept grouping. Figures 2 and 3 are built from Tables 1 and 2, respectively.

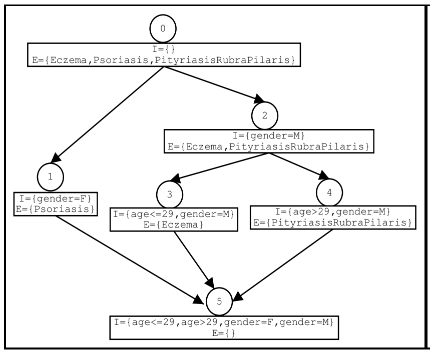
0
I={}
E={Eczema,Psoriasis,PityriasisRubraPilaris}
2
I={gender=M}
E={Eczema,PityriasisRubraPilaris}
1
I={gender=F}
E={Psoriasis}
3
I={age<=29,gender=M}
E={Eczema}
4
I={age>29,gender=M}
E={PityriasisRubraPilaris}
5
I={age<=29,age>29,gender=F,gender=M}
E={}

Fig. 2. Previous graph G_{t-1} (6 cases).

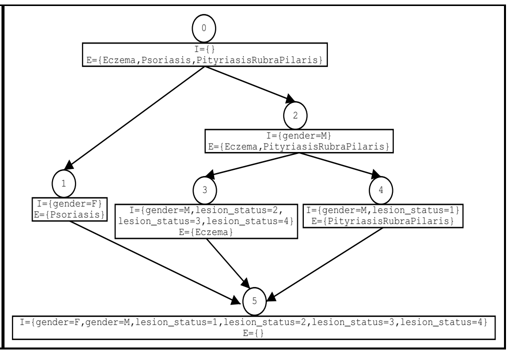
0
I={}
E={Eczema,Psoriasis,PityriasisRubraPilaris}
2
I={gender=M}
E={Eczema,PityriasisRubraPilaris}
1
I={gender=F}
E={Psoriasis}
3
I={gender=M,lesion_status=2,
lesion_status=3,lesion_status=4}
E={Eczema}
4
I={gender=M,lesion_status=1}
E={PityriasisRubraPilaris}
5
I={gender=F,gender=M,lesion_status=1,lesion_status=2,lesion_status=3,lesion_status=4}
E={}

Fig. 3. Current graph G_t (7 cases).

Given Table 1, the system generates a lattice shown in Figure 2. When a new case is added, a new lattice is generated. If we compare the previous and current lattices, we expect the lattices to be the same or with minor variations if changes do occur. If the concept changes significantly, the consultant needs to check if the new case is in fact 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. The validation task is vitally important to prevent contradictory cases from being used in the classification process as these could lead to inaccurate diagnoses.

4 Conceptual Dissimilarity Measure

We derive a dissimilarity metric for determining the level of changes between the two lattices. Emphasis is put on illustrating the changes in the characteristics of the diagnoses. The measure is particular effective when the lattices become too large to be manually visualised by the consultant. Therefore, it is desirable to have a measure that indicates the level of changes between the current and previous lattices, and use it as a quick representation of the changes.

4.1 Highlighting Conceptual Changes between Concept Lattices

We propose 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. *Missing concepts* are determined by those present in G_{t-1} but not in G_t. Whereas, *added concepts* are those not present in G_{t-1} but present in G_t. To highlight the conceptual changes, the system displays the information such as 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 could be time consuming and error-prone. Note in the rest of this paper the terms *lattice* and *graph* are the same.

To highlight changes, we use the concepts *extent* and *intent*. A is the extent of the concept (A, B) which comprises all the attributes (characteristics) that are valid for all the objects (diagnoses); B is the intent of the concept (A, B) which covers all the objects that are belonging to that particular concept [4]. We define the number of concepts that have the same intents in G_{t-1} and G_t as $|B(t-1)_{concept}| \cap |B(t)_{concept}|$. The number of missing concepts $n(G_{t-1})$ and added concepts $n(G_t)$ are defined as:

$$n(G_{t-1}) = |B(t-1)_{concept}| \setminus |B(t-1)_{concept}| \cap |B(t)_{concept}| \tag{1}$$

$$n(G_t) = |B(t)_{concept}| \setminus |B(t-1)_{concept}| \cap |B(t)_{concept}| \tag{2}$$

where $B(t-1)$ represents B in the previous graph G_{t-1}, and $B(t)$ represents B in the current graph G_t. However, if the results for both $n(G_{t-1})$ and $n(G_t)$ are equal to zero, this means the two graphs contain exactly the same concepts and implies that the new added cases are consistent with the previous cases in the database. In the worse case, the values of $n(G_{t-1})$ and $n(G_t)$ can be equal to the number of concepts in G_{t-1} and G_t, respectively. This implies a radical change between the two graphs and it means that the new added cases may be contradictory and need to be revised by the consultant. The values of $n(G_{t-1})$ and $n(G_t)$ provide a rough estimate of the changes in concepts between the two lattices. The locations at which the changes occur are highlighted in different colour for quick reference to the nodes in the lattice, and provide a useful indication of the number of concepts being affected. The closer the changes to the root of the lattice the more concepts get affected.

The missing and added nodes in Algorithm 1 are used as an entry point for the matching to prevent unnecessary processing of every node in the lattice. Algorithm 2 selects the lowest child node in the lattice that contains a certain extent element. The reason being the lowest child node provides the most detailed description compared to those that are higher up in the hierarchy. In a lattice, lower nodes convey more meaning than those above them.

For illustration purposes, consider the two simple examples in Figures 2 and 3. In Figure 3, suppose nodes 3 and 4 are the added nodes. If node 3 is selected for processing, then the parent node would be 2. At the parent node, the extent element *Eczema* is selected and stored. Iterate down to the lowest child node that contains *Eczema* which is node 3 with the intent elements of *gender=M, lesion_status=2, lesion_status=3 and lesion_status=4*. Then perform the matching between the two graphs. The results show that node 2 of G_{t-1} matches node 2 of G_t. Based on the stored extent element *Eczema* in G_{t-1}, iterate down to the lowest child node that contains *Eczema* which is node 3 with the intent elements of *age≤29* and *gender=M*. Compare and highlight the differences between the current and previous intent elements. In this case, the characteristics *lesion_status=2*, *lesion_status=3* and *lesion_status=4* have been introduced in G_t to give the diagnosis *Eczema* a more detailed description, but the characteristic *age≤29* has become insignificant. Repeat the same steps for node 4. The results show that in the current graph the characteristic *lesion_status=1* has become important for classifying *PityriasisRubraPilaris*, whereas the characteristic *age>29* in G_{t-1} has become in-

Algorithm 1: Conceptual differences between concept lattices

```
input : missing/added node $i$, $G_{t-1}$, and $G_t$.
foreach $i$ in $G_{t-1}$ do
    Find the parent node $p$ of the node $i$
    foreach $p$ do
        $extentOfParent$ ← getExtent($G_{t-1}$, $p$)
        Iterate to get each extent element $e$ in $extentOfParent$
        while there are more $e$ do
            $node$ ← getLowestChild($p$, $extentOfParent$)
            if $node$ is equal to $null$ then
                $node$ ← node $i$
                Get and store the intent of $node$ as $intentG1$
            else Get and store the intent of $node$ as $intentG1$
            Find matched node $m$ in $G_t$ based on the intent of $p$
            if $m$ is not equal to $null$ then
                $extentOfMatchedParent$ ← getExtent($G_t$, $m$)
                $node$ ← getLowestChild($m$, $extentOfMatchedParent$)
                if $node$ is equal to $null$ then
                    $node$ ← matched node $m$
                    Get and store the intent of $node$ as $intentG2$
                    Highlight intent element differences between $intentG1$ and $intentG2$
                else
                    Get and store the intent of $node$ as $intentG2$
                    Highlight intent element differences between $intentG1$ and $intentG2$
            else
                Get top node $tn$
                $extentOfTopNode$ ← getExtent($G_t$, $tn$)
                $node$ ← getLowestChild($tn$, $extentOfTopNode$)
                if $node$ is equal to $null$ then
                    $node$ ← top node $tn$
                    Get and store the intent of $node$ as $intentG2$
                    Highlight intent element differences between $intentG1$ and $intentG2$
                else
                    Get and store the intent of $node$ as $intentG2$
                    Highlight intent element differences between $intentG1$ and $intentG2$
```

Algorithm 2: getLowestChild

```
input : Node $n$ and extent $ext$.
return lowest node.
$Node\ lowest = null$
Get children node of $n$
foreach children of node $n$ do
    Get the children that contain extent element $ext$
    if lowest child with extent element $ext$ is found then
        return $lowest$
```

significant. The characteristic for describing the diagnosis *Psoriasis* remains unchanged in both G_{t-1} and G_t.

4.2 Dissimilarity Metric

The measure of conceptual variation between graphs G_{t-1} and G_t when a new case is added is based on the following factors:

1. The number of diagnoses that have been affected.
2. The level in which the concept variations occur in the lattice.
3. The ratio of the characteristics of each diagnosis that have been affected.

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 the graphs which is calculated using Equation 1 or 2 depending on 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; $md(G_j)$ is the maximum depth of G_j; and $d(G_{ji})$ is the depth 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:

$$v(G_j) = \begin{cases} v_1(G_j) & : \quad md(G_j) = 1 \text{ and } d(G_{ji}) = 0 \\ v_2(G_j) & : \quad md(G_j) = 1 \text{ and } d(G_{ji}) \neq 0 \\ v_3(G_j) & : \quad md(G_j) > 1 \end{cases} \tag{3}$$

where $v(G_j) = \{v_1(G_j)|v_2(G_j)|v_3(G_j)\}$, and $v_1(G_j)$, $v_2(G_j)$ and $v_3(G_j)$ are defined as:

$$v_1(G_j) = \sum_{i=0}^{n} \frac{c(G_j)}{C(G_j)} . \frac{a(G_{ji})}{A(G_{ji})} . \frac{md(G_j) - d(G_{ji})}{md(G_j) * (n+1)}$$

$$v_2(G_j) = \sum_{i=0}^{n} \frac{c(G_j)}{C(G_j)} . \frac{a(G_{ji})}{A(G_{ji})} . \frac{1}{n+1}$$

$$v_3(G_j) = \sum_{i=0}^{n} \frac{c(G_j)}{C(G_j)} . \frac{a(G_{ji})}{A(G_{ji})} . \frac{md(G_j) - d(G_{ji})}{md(G_j) * n}$$

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

$$d(G) = v(G_{t-1}) + v(G_t) \tag{4}$$

The dissimilarity measure between the two graphs falls into the range of $[0, 1]$. The lower the value of $d(G)$ the higher the consistency since there is less variation between the two graphs. If $d(G) = 0$, the new cases are consistent with the previous cases. In the worse case, the value of $d(G)$ is close to or equal to 1, implying the new added cases contradict the previous cases. This might be because there are not enough cases to disambiguate some diagnoses, or the diagnoses are classified using different sets of characteristics.

We use the examples in Figures 2 and 3 to illustrate how the dissimilarity measure is calculated. Based on G_{t-1}, $v(G_{t-1}) = 0.111$ with $c(G_{t-1}) = 2$; $C(G_{t-1}) = 3$; $a(G_{t-1(1)})/A(G_{t-1(1)}) = 0.5$ and $a(G_{t-1(2)})/A(G_{t-1(2)}) = 0.5$; $md(G_{t-1})$ = 3; and $d(G_{t-1(1)}) = 2$ and $d(G_{t-1(2)}) = 2$. Graph G_t has $v(G_t) = 0.139$ with $c(G_t) = 2$; $C(G_t) = 3$; $a(G_{t(1)})/A(G_{t(1)}) = 0.75$ and $a(G_{t(2)})/A(G_{t(2)}) = 0.5$; $md(G_t) = 3$; and $d(G_{t(1)}) = 2$ and $d(G_{t(2)}) = 2$. Thus, applying Equation 4 we get $d(G) = 0.25$ which implies *significant change* according to the classifications that are defined in Table 3.

4.3 Results

Discussion with the consultant revealed the effectiveness of the proposed techniques. First, the consultant recommended the grouping of the dissimilarity values into 5 different categories shown in Table 3. 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 as "No change", since the variations between the two graphs are not significant enough to have a major effect on the results.

The evaluation process involved analysing 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 4 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 for consistency of the results, 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 4 with decreasing regression lines.

Table 3. Dissimilarity values categorization.

Rank	Category	Dissimilarity Values
1	No change	0.00 - 0.10
2	Slight change	0.10 - 0.15
3	Significant change	0.15 - 0.50
4	Major change	0.50 - 0.90
5	Radical change	0.90 - 1.00

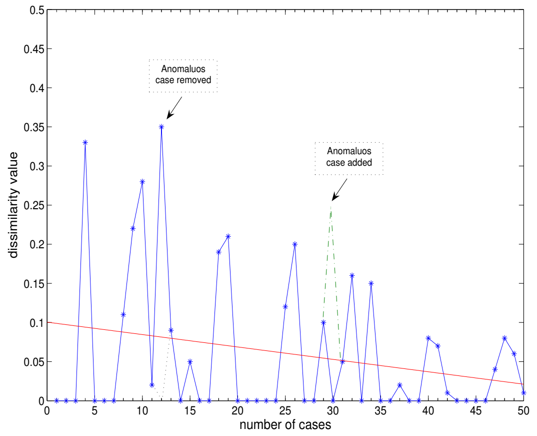

Fig. 4. Dissimilarity values vs. number of new cases.

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

To further illustrate this, we randomly chose case number 30 having the dissimilarity value of zero, and modified its characteristics to determine whether the value does increase to show the fact 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.

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

5 Conclusions

This paper presents a technique that enables the domain expert to interactively supervise and validate the new knowledge. The technique involves using concept lattices to graphically present the conceptual differences, a summary description highlighting the conceptual variations, and a dissimilarity metric to quantify the level of variations between the current and previous cases in the database. The trials involved having the consultant interactively revise the anomalous cases by modifying the characteristics and the diagnoses that are likely to cause the ambiguity. The results show that dissimilarities match a number of actual processes. First, when the lattices change significantly but the decision tree classifier is giving 100% classification as each new diagnosis is added. Second, when there is a problem in the data for a new case requiring consultant interaction.

Acknowledgements

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.

References

1. P. Cole, R. Eklund and F. Amardeilh. Browsing Semi-structured Texts on the Web using Formal Concept Analysis. *Web Intelligence*, 2003.
2. B. Díaz-Agudo and P. A. Gonzalez-Calero. Formal Concept Analysis as a Support Technique for CBR. *Knowledge-Based System*, 7(1):39–59, March 2001.
3. B. Ganter. Computing with Conceptual Structures. In *Proc. of the 8th International Conference on Conceptual Structure*, Darmstadt, 2000. Springer.
4. B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, Heidelberg, 1999.
5. M. Montes-y Gómez, A. Gelbukh, A. López-López, and R. Baeza-Yates. Flexible Comparison of Conceptual Graphs. In *Proceedings of the 12th International Conference and Workshop on Database and Expert Systems Applications*, pages 102–111, Munich, Germany, 2001.
6. J. Ross. Quinlan. *C4.5 Programs for Machine Learning*. Morgan Kaufmann Publishers, USA, 1993.
7. D. Richards. Visualizing Knowledge Based Systems. In *Proceedings of the 3rd Workshop on Software Visualization*, pages 1–8, Sydney, Australia, 1999.
8. D. Richards. 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*, Beijing, China, 2000.
9. I. Watson. *Applying Case-Based Reasoning: Techniques for Enterprise Systems*. Morgan Kaufmann Publishers, USA, 1997.
10. I. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann Publishers, USA, 2000.
11. P. Z. Yeh, B. Porter, and K. Barker. Using Transformations to Improve Semantic Matching. In *Proceedings of the International Conference on Knowledge Capture*, pages 180–189, Sanibel Island, FL, USA, 2003.
12. J. Zhong, H. Zhu, J. Li, and Y. Yu. Conceptual Graph Matching for Semantic Search. In *Proceedings of the 10th International Conference on Conceptual Structures: Integration and Interfaces*, pages 92–106, Borovets, Bulgaria, 2002.