Spring 2006
Vol 8, No 3
ISSN 1465-4091
£12.50

The Specialist Group on Artificial Intelligence

EXPERT UPDATE

SGAI

ECCAI

BCS®
THE BRITISH COMPUTER SOCIETY

Selected Papers from the 9th UK CBR Workshop

EXPERT UPDATE
(ISSN 1465-4091)

What is Expert Update?
Expert Update (www.comp.rgu.ac.uk/staff/nw/expertUpdate.htm) is the bulletin/magazine of the SGAI, the British Computer Society's Specialist Group on AI (BCS-SGAI: www.bcs-sgai.org). The purpose of Expert Update is to foster the aims and objectives of the group by publishing news, conference reports, book reviews, conference announcements, calls for papers and articles on subjects of interest to the members. Expert Update is generally published 3 times per year by BCS-SGAI. The group's official postal address is: SGAI, The BCS, Davidson Building, 5 Southampton Street, London, WC2E 7HA.

How do I subscribe?
It is free to all SGAI members. Please visit www.bcs-sgai.org/sgai/sgai.htm for details on joining the group.

How do I subscribe?
It is free to all SGAI members. Please visit www.bcs.sgai.org/sgai/sgai.htm for details on joining the group.

How do I contribute?
Submissions are welcome and must be made in electronic format sent to the editor or sub-editor.

Who owns Copyright?

EDITORIAL

Welcome to Expert Update's Spring 2006 issue.

There have been a few changes to the BCS-SGAI committee this year. The main being the departure of Prof Ann Macintosh, who for many years has been the group's vice-chairperson. Prof Macintosh will however remain an honorary life time member of the group.

The BCS-SGAI are once again hosting its annual AI conference in Cambridge with Dr Andrew Tuson as conference chair. Submissions are encouraged on all aspects of AI theory and practice. The deadline for submissions is 5th June. See back cover for details.

Moving on to the contents of the current Expert Update issue, we have seven selected papers from the 9th UK Case-Based Reasoning workshop (UKCBR'04) organised and chaired by Dr Brian Lees, Univeristy of Paisley. Contributions in this issue report on work undertaken by CBR groups in Spain, Ireland, Germany, France and the UK. The papers cover a wide range of current research trends: from textual indexing, authoring tools, recommender systems, similarity-based classification to applications in the medical domain.

Nirmalie Wiratunga
Editor Expert Update
sgai-newsletter@bcs.org.uk

Introspective Knowledge Acquisition in Case Retrieval Networks for Textual CBR

Sutanu Chakraborti, Stuart Watt and Nirmalie Wiratunga

School of Computing,
The Robert Gordon University
Aberdeen AB25 1HG, Scotland, UK
{sc,sw,nw}@comp.rgu.ac.uk

Abstract. This paper identifies issues in knowledge acquisition for textual Case based Reasoning. Though previous literature has highlighted the need for a principled knowledge acquisition using knowledge layers of increasing complexity, practical experience suggests that this involves considerable manual effort from domain experts, often resulting in long lead times in system deployment. We propose that it may be preferable to start off with a crude representation of cases and similarity measures and use introspective learning techniques to incrementally refine this knowledge. We discuss two such approaches in the context of Case Retrieval Networks (CRNs) applied to text classification. The first is based on discovering hidden patterns in text and the second exploits class knowledge. Experimental results show that acquired knowledge improves retrieval effectiveness of CRNs when applied to classification tasks.

Keywords: Textual Case based Reasoning**,** Case Retrieval Networks, Latent Semantic Indexing, Knowledge Acquisition, Neural Networks

1 Introduction

Case based Reasoning (CBR) is known to reduce the knowledge acquisition bottleneck by modelling past problem solving experiences directly as cases. However considerable manual intervention is often needed to ensure competence of the case base and appropriateness of other knowledge containers [10] (vocabulary, similarity and adaptation knowledge) in relation to the underlying domain. In this paper we address knowledge acquisition issues in the context of textual Case based Reasoning where a CBR system needs to directly operate on textual data (rather than structured cases). A textual case is composed of terms or keywords. The set of distinct terms and key-phrases in the document collection is treated as the feature set.

Previous research by Lenz [7] identifies seven knowledge layers in textual CBR. In practice, however, there is considerable resistance in investing manpower to allow for such a systematic knowledge acquisition process. This also results in a huge lead-time in system deployment. A more plausible approach is to start off with a crude representation and perform successive refinements using machine learning strategies by exploiting implicit co-occurrence patterns and class knowledge. We present and compare two approaches that demonstrate the effectiveness of such "knowledge-light" strategies in classification tasks. Our discussion is centred on a Case Retrieval Network (CRN) formalism, which has been demonstrated to be effective and efficient in retrieval over large and high dimensional case bases, typical with textual data. We focus on acquiring two types of knowledge: knowledge about how entities like words/phrases in a domain are related to each other (similarity knowledge); and knowledge about relatedness of entities to cases (relevance knowledge).

2 Case Retrieval Networks

CRNs have been proposed as a representation formalism for CBR in [5]. In CRNs, an Information Entity (IE) is the lowest granularity of knowledge representation (like an attribute value pair). A case consists of a set of IEs. IEs are related to each other by a similarity function σ. IEs are related to cases by a relevance function ρ. A propagation function is defined for each node, which aggregates the effects of spreading activation used for retrieval of cases closest to the query. CRNs can easily be extended to textual CBR by treating a term/key-phrase as an IE, and each document as a case [6]. Lenz et al. [8] have successfully deployed CRNs over large case bases containing as many as 200,000 cases.

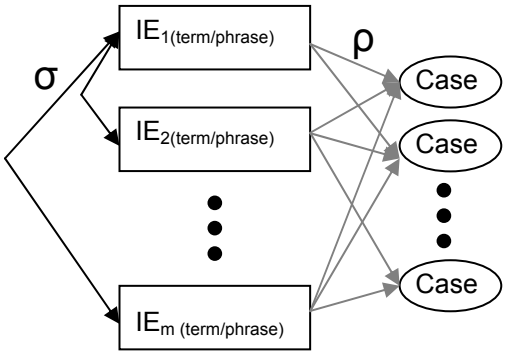

Fig. 1. A Basic Case Retrieval Network

Fig. 1 is a graphical representation of a BCRN. Given any query item, the corresponding IEs (shown as rectangles) are activated and a spreading activation is initiated across the similarity and relevance arcs. The propagation function at each IE node aggregates all activations it receives via the similarity arcs. Each incoming activation has strength proportional to the similarity between the IEs that are connected by that arc. In the next phase, the propagation function at each case node aggregates all activations received via the relevance arcs. The output of the system is a set of cases ordered by their cumulative activations that indicate the degree of relevance to the query.

3 Acquiring Similarity and Relevance Knowledge for Textual CBR

Similarity is fundamental to the underlying domain and exists irrespective of the cases. In contrast relevance acts as a bridge between the world of IEs and the world of cases. Similarity between IEs is used as a measure of local similarity; in contrast, relevance arcs have the usual connotation of degree of presence of a term (an IE) in a case and hence play a role in computation of global similarities.

Similarity between terms can be inferred from their co-occurrence patterns in texts, which in turn is given by relevance values. We use a toy example in Fig. 2 with six IEs and six cases. The left matrix shows the relevance arc values: a value 1 is assigned if an IE is present in a case and 0 otherwise. The right matrix records all pairwise co-occurrences of IEs. It can be verified that the right matrix can be derived from the left matrix by multiplying the transpose of the left matrix with its self. If we assume that terms similar to each other are more likely to co-occur with each other than terms that are not, it appears that the second matrix is a good estimate of term similarities.

We observe, however, that the flipping of interpretation from relevances to similarities does not generate any additional knowledge. Thus a CRN whose similarity arcs are constructed from the similarities between IEs obtained above would be just as effective as a CRN using relevance knowledge instead. We view this as a manifestation of the "Knowledge Conservation Principle (KCP)" which

suggests that the overall system performance is not affected by transferring knowledge from one knowledge container to another. The KCP has interesting implications in terms of design principles to

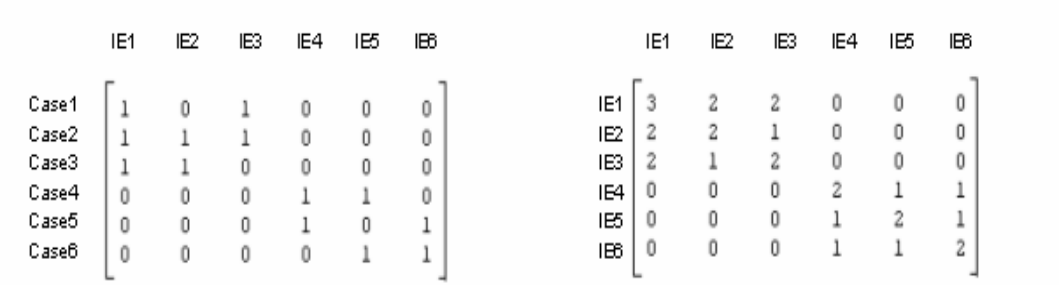

Fig. 2. Relevance and Similarity

be adopted for textual CBR systems. It is intuitive to separate knowledge encapsulated in relevance and similarity arcs. Relevance arcs can capture associations present in (or inferred from) the texts from which cases originate. In contrast similarity knowledge can be obtained from background domain knowledge elicited from experts; or external knowledge sources.

4 Two Knowledge-Light approaches

In this section we describe two approaches to knowledge-light acquisition of relevance weights. The first approach uses Latent Semantic Indexing to arrive at relevance weights by exploiting co-occurrence patterns of terms across textual cases, and the second uses a VSM initialization for the relevance weights and uses class knowledge to refine these associations.

4.1 Latent Semantic Indexing

Latent Semantic Indexing (LSI) [1] is an improvement over the naïve VSM model in that it attempts to uncover associations between terms based on their co-occurrences. As an example, in the computer hardware domain LSI can construct a concept space in which the terms "laptop" and "palmtop" move closer to each other since they tend to co-occur in many documents. LSI also recovers from problems like synonymy and polysemy [1]. LSI constructs conceptual indices that move semantically related IEs closer to each other. The inference that two IEs are semantically related is purely based on their co-occurrences. Also, an IE can have non-zero relevance to a case even if it does not appear in the case, provided it is semantically related to IEs present in the case.

The key idea behind LSI is that it is mathematically possible to extract a smaller number of (significant) dimensions that are more robust indicators of meaning than individual terms (IEs). LSI makes use of a linear algebraic decomposition namely Singular Value Decomposition (SVD) to arrive at the latent structure. We do not discuss the finer details of SVD here, but the interested reader is directed to [1]. Using SVD, the original term-document matrix A is factored into three matrices U, Σ and V where U and V contain the left and right singular vectors of A and the matrix Σ is a diagonal matrix containing the singular values.

The decomposition is:

$$A = U \Sigma V^T$$

If the matrix A is of dimension $m \times n$ (m unique IEs across n cases) and only top k singular values are considered, then we can construct an approximation to the original term-document matrix A_k which is obtained as the product of U_k (an $m \times k$ matrix), Σ_k (a $k \times k$ matrix) and V^T_k (a $k \times n$ matrix). Intuitively,

since $k << m$, minor differences in case vocabulary will be ignored. IEs which occur in similar cases will occur close to each other in the k dimensional factor space even if they never co-occur in the same case. If we use this approximation A_k instead of A to define relevance strengths, there may be cases which share no IEs with the query and yet are retrieved because of similarity of query IEs with case IEs in the latent semantic space.

4.2 ECRN: Class-Knowledge Based Refinement of VSM

In this approach we learn relevance weights using class knowledge for a CRN in situations where each case belongs to a class. The CRN is embedded in an architecture as shown in Fig 3 which we refer to as the Extended Case Retrieval Network (ECRN). In addition to IE and case nodes, ECRN has a third layer pertaining to class labels of the cases. Henceforth we refer to these layers as IE, case and class layers respectively. Feature selection (described in [3]) is used to arrive at features used as IEs.

ECRN integrates learning behaviour associated with Multi-layer Feed-Forward Neural Networks (MLFNs)[2] into CRNs by the following three phases of operation:

Initialization Phase: In this phase, relevance weights are assigned a value 1 if the IE is present in the case and 0, otherwise. The weights connecting the case-layer to the class-layer are assigned binary values based on whether the case belongs to that class or not.

Training Phase: In this phase, weights are modified to improve classification accuracy of the system. The training cases are fed into the IE layer one at a time and the classification output of the system is compared against the expected outcome. The desired output is a binary value 0 or 1 depending on the class to which the input case actually belongs. The error is fed back to alter the set of weights between IE and case layers. This is done by using a variant of the back-propagation algorithm used widely in training MLFNs [2]. Now the revised set of weights is used to classify the set of training documents again. If the mean squared error of the output layer is less than the previous iteration, the current set of weights is retained; otherwise we revert back to the old set of weights. The iterative process terminates once there is no improvement in accuracy.

Operation phase: After training, we evaluate classification performance over test data. For this the new case to be classified is preprocessed – the IEs pertaining to the case are activated and the activation is propagated through the case nodes and the class nodes. It may be interesting to note that the classification phase can be seen as a two step process: in the first step, the nodes pertaining to cases that are close to (nearest neighbours of) the incoming document are activated – in the second step, the activated cases vote for the class that they belong to (via the connections to the output layer) and the results are aggregated in the output nodes: the class with the strongest activation is returned as the result.

In a retrieval scenario, the case rankings based on relevance to the query are significant, but the classification output is not needed. In such a case, we can remove the final class layer and associations from the case layer to the class layer after ECRN training is over. Since ECRN's training is focused by the aim of bringing conceptually cases closer to each other, the resulting relevance weights should have a positive impact on the retrieval performance.

It may be noted that although we have used a back-propagation algorithm to learn relevance weights, there are some significant differences between a traditional MLFN approach and the ECRN approach. Firstly, unlike ECRN, in "black-box" MLFNs, the hidden nodes and weights do not carry any domain specific interpretation. Furthermore, ECRN uses naïve VSM to initialize the network; this results in significant reduction in training times, compared to MLFNs where weights are randomly initialized.

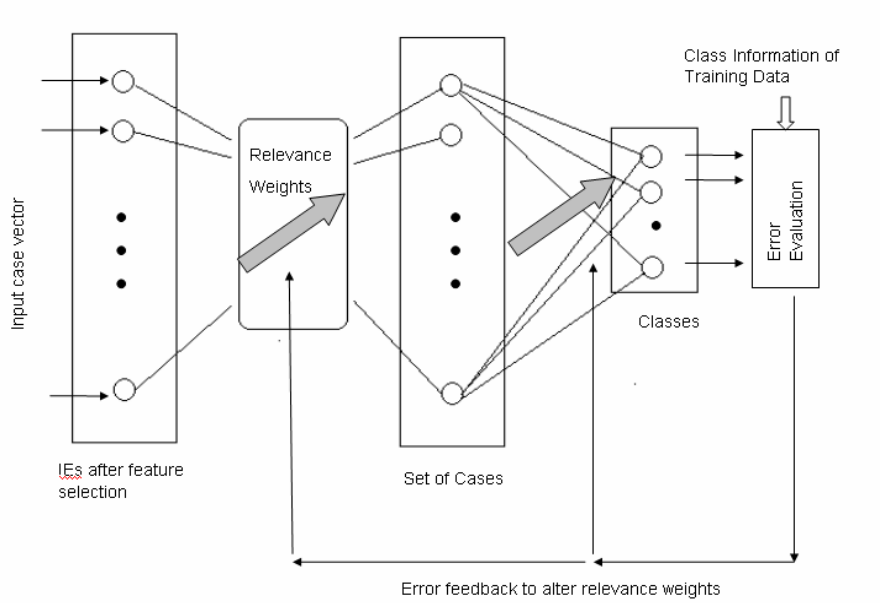

Fig. 3 A Schematic of Extended Case Retrieval Network (ECRN)

5 Evaluation

We evaluate the two approaches using the following corpora: (a) **LingSpam**: This dataset has been used to study the problem of Spam. It contains 2893 email messages of which 83% are non-spam messages belonging to a linguistic mailing list and the rest are spam [9] (b) **20NewsGroups:** This dataset is a corpus of about 20,000 Usenet news postings into 20 different newsgroups. 1000 messages from each of the twenty newsgroups were chosen at random and partitioned by newsgroup name [10]. For our experiments, we use one sub-corpus, which we call the **SpcMed** in which the messages from the Space and Medical Sciences newsgroups are combined to form a binary classification.

We created equal sized disjoint training and test sets, where each set contains 20% of documents randomly selected from the original corpus, preserving class distribution in the original corpus. For repeated trials, 15 such train-test splits are formed. All accuracy results reported below are figures obtained after averaging over 15 trials. In our experiments, textual cases are formed by pre-processing documents by removing stop words and special characters. Remaining words are reduced to their stem by using Porter's algorithm [12].

In order to compare the effectiveness of knowledge light acquisition with LSI and ECRN, we compare accuracy with one Nearest Neighbour (1NN) on unseen data. 1NN accuracy is defined as the proportion of instances in the test dataset that have the same class label as their nearest neighbour in the training dataset and gives an idea of the generalization achieved by the classifier.

While LSI performance is critically dependant on the dimensionality of the underlying representation, ECRN performance is a single value reflecting the effectiveness and generalization ability of the classifier function learnt by the network. In the discussion below, the best performance of LSI is compared with the ECRN performance.

The graphs in Fig. 4 show that LSI has a favourable impact on improving 1NN accuracy over the test dataset. Lower dimensions correspond to representations where cases sharing conceptually related IEs (inferred on the basis of their co-occurrence patterns with other IEs) are drawn closer to each other. An intuitive explanation for LSI effectiveness in class separation is that the co-occurrence patterns of

IEs create a clustering of IEs that has a direct correspondence to case clusters. ECRN improves the INN performance on test data from 68.17% to 84.45% in the SpcMed dataset and from 82.42% to 97.83% in the LingSpam dataset. The graphs in Fig. 4 show that LSI is able to improve with dimensionality reduction over the 1NN accuracy on test data. 1NN performance on test data obtained by ECRN is comparable to the best performance of LSI – 86.6% and 97.6% in the SpcMed and LingSpam datasets respectively.

It is interesting to explore ways of combining LSI and ECRN. In one experiment, we investigated the number of significant dimensions in the LSI decomposition of the relevance weights learnt by ECRN. Interestingly, there were only two significant dimensions in the LSI decomposition. This is substantially fewer than the number of significant dimensions (around 60) in the relevance weights before training. This shows an interesting correspondence between LSI and ECRN: ECRN achieves dimensionality reduction when it is actually designed to achieve class separation. LSI while intended to do dimensionality reduction, also achieves class separation.

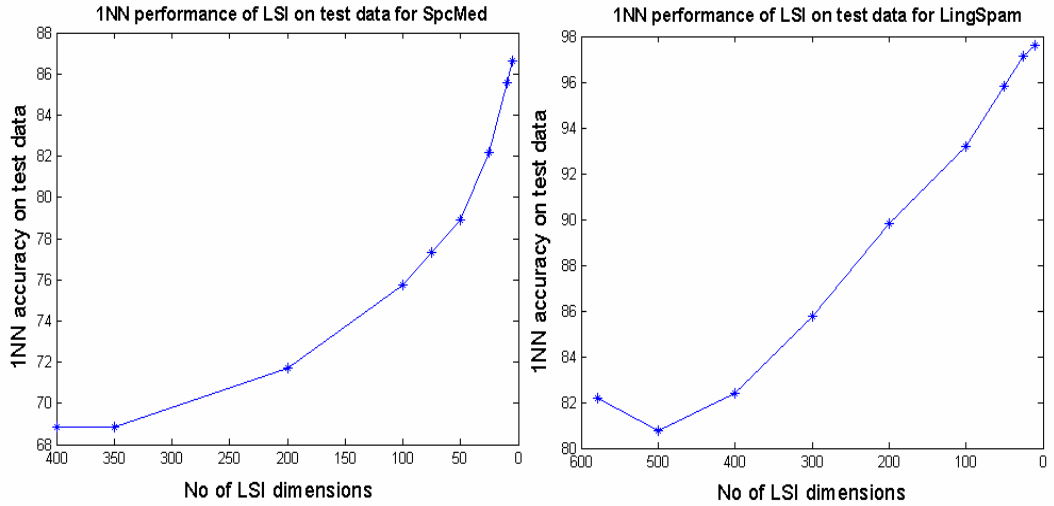

Fig. 4. One Nearest Neighbour Performance of LSI on Test Data

6 Related Work

In [6] LSI is used for initializing relevance weights and WordNet for arriving at similarity weights. Since the knowledge sources for the similarity and the relevance weights are distinct, the Knowledge Conservation Principle is implicitly obeyed. The focus, unlike the current paper, is on a retrieval rather than classification. Zelikovitz et al. [4] report LSI applied to text classification in the presence of background knowledge. From the KCP perspective, this is interesting: while relevance knowledge can be obtained directly from the document collection, the background knowledge can be used for mining term similarities. Wiratunga et al. [3] present a technique for mining association rules from textual data. It is interesting to note that association rule mining is used for feature generalization – both techniques described in this paper also have feature generalization aspects, in terms of reducing sparseness of the naïve VSM representation.

The idea of using algorithms like Back-propagation over a neural network initialized with prior domain knowledge is used in KBANN [10]. However in KBANN, the prior knowledge is a domain theory consisting of non-recursive propositional Horn Clauses; in contrast our approach uses a

knowledge-light naïve VSM initialization. This helps us preserve the CRN topology while allowing for relevance weight learning.

7 Conclusions

In this paper we identify issues in knowledge acquisition for textual CBR. Generally, a considerable amount of manual intervention is needed to arrive at complex knowledge layers. This paper suggests that it may be more practical to use keyword-based representations and rely on machine learning techniques to refine this knowledge. We suggested some general design principles towards this end. Two knowledge light approaches were presented and empirically compared.

References

1. Berry, M.W., Dumais,S.T., O'Brien, G.W : Using linear algebra for intelligent information retrieval, Technical Report UT-CS-94-270, 1994.
2. Lin , Lee: Neural Fuzzy Systems : A Neuro-Fuzzy Synergism to Intelligent Systems, Prentice Hall Inc., May 1996.
3. Wiratunga, N., Koychev I., Massie, S. : Feature Selection and Generalization for Retrieval of Textual Cases, Proceedings of ECCBR 2004 pp. 806- 820
4. Zelikovitz, S., Hirsh H.: Using LSI for Text Classification in the Presence of Background Text, CIKM 2001, pp. 113- 118
5. Lenz, M., Burkhard, H.: Case Retrieval Nets: Basic Ideas and Extensions. KI 1996: 227-239
6. Chakraborti, S., Ambati, S., Balaraman, V., Khemani, D.: Integrating Knowledge Sources and Acquiring Vocabulary for Textual CBR. In Proceedings of the 8th UK CBR Workshop, pages 74-84, 2003
7. Lenz, M. : Knowledge Sources for Textual CBR Applications, Textual Case based Reasoning: Papers from the 1998 Workshop Technical Report WS-98-12 AAAI Press pp. 24-29
8. Lenz, M., Auriol, E., Manago, M. : Diagnosis and Decision Support (Chapter 3) In Case Based Reasoning Technology, Lecture Notes in Artificial Intelligence 1400, Mario Lenz, Brigittie Bartsch-Sporl, Hans-Dieter Burkhard, Steffan Wess (Eds.) 1998, pp. 51 – 90
9. Sakkis, G., Androutsopoulos, I., Paliouras, G., Karkaletsis, V.,Spyropoulos C., Stamatopoulos, P.: A memory-based approach to anti-spam filtering for mailing lists. Information Retrieval, 6(2003) 49-73
10. Mitchell, T. Machine Learning. Mc Graw Hill International (1997)
11. Richter, M.M., Introduction. In Case based Reasoning Technology: From Foundations to Applications, LNAI 1400. Springer, 1998
12. Porter, M.F., 1980, An algorithm for suffix stripping, Program, 14(3) :130-137

Case-based Analysis of the Scenarios of Multiagent Interaction

Boris Galitsky[1], Sergei O Kuznetsov[2], and Don Peterson[3]

[1]School of Computer Science and Information Systems
Birkbeck College, University of London
Malet Street, London WC1E 7HX, UK
galitsky@dcs.bbk.ac.uk

[2]All-Russian Institute for Scientific and Technical Information (VINITI),
Usievicha 20, Moscow 125190, Russia
serge@viniti.ru

[3] Institute of Education
University of London
20 Bedford Way London WC1H 0AL UK
d.peterson@ioe.ac.uk

Abstract. A case-based reasoning framework for handling scenarios of interaction between conflicting human agents is suggested. The cases are represented as graphs with labeled vertices (for communicative actions) and edges (for temporal and causal relationship between these actions and their parameters). We develop a special technique to relate scenarios of inter-human interactions to a class. Developed scenario representation and case-based reasoning technique is applied to such domain of multiagent conflict as textual customer complaints.

1 Introduction: Reasoning with Conflict Scenarios

Cases of interaction between human agents are an important subject of study in Artificial Intelligence. Extensive body of literature addresses the problem of logical simulation and case-based reasoning (CBR) of agents' behavior, taking into account their beliefs, desires and intentions [1]. A substantial advancement has been achieved in building the scenarios of multiagent interaction, given the properties of agent including their attitudes. Current models of inter-human interactions are mostly based on logical deduction [10,11] or simulation [2]; means of automated comparative analysis of cases of inter-human interactions are still lacking.

In the logical deduction approach, the sequence of mental states of agents is deduced from their initial mental states and initial attitudes. Deductive reasoning about actions and the logic with agents' attitudes as modalities are the most popular means

to yield sequences of mental states of agents [1,11]. In the simulation approach, the system imitates the decision-making of agents, choosing the best action for each agent at each step, taking into account its current intentions, beliefs and desires, as well as those of others. Having defined the preference relation on the set of resultant states, each agent selects an action that is expected to lead to the most wanted state [2].

The above approaches lack the capability of learning from previous experience [5] how inter-human conflicts have been handled in the past, where such experience cannot be explicitly formalized via rules. A general framework to reuse the experience accumulated in previous cases of multiagent interaction has not been developed. For effective building and predicting of interaction between agents, it is helpful to augment deductive reasoning and/or simulation with case-based reasoning (CBR) [6]. It would reduce the number of possible agents' actions at each step taking into account how these agents acted in previous cases.

A number of studies have addressed the applications of CBR technique to BDI Model; such hybrid approach was used in information retrieval [9,12], planning [14] and others. The limitations of the stand-alone BDI model which include the lack of learnability and explicit multiagent aspects of behaviour, are well known. Combination of multiagent system theory and machine learning has been tackling this problem for a decade [4, 11, 12, 13] to improve our understanding of learning principles in natural and computer systems [11].

In this paper we build the representation machinery for conflict scenarios and propose a CBR technique that is oriented to multiagent conflict. The task is to relate a case (a scenario of inter-human interactions) to a class; the simplest case we are considering here is the pair of classes for *valid* and *invalid* cases. We refer to a case as valid if it is consistent, properly communicated, can be trusted, and/or similar to the set of cases which have been assigned as valid by experts (an invalid case has the respective definition). Each class assumes a special handling of scenarios (e.g. compensation, return, replacement of a product in the former class, and further communication with an upset customer in the latter case).

Multiagent conflict is a special case of scenarios where the agents have inconsistent goals and a negotiation procedure is required to achieve a compromise [10]. Following the logical structure of how negotiations are represented in text, it is possible to judge on a consistency of this scenario [3,15]. Scenarios require the involvement of complex data structures; we use labeled bipartite graphs with multiedges for representing interaction of two parties of a conflict.

2 The Domain of Conflict Scenarios

In this Section we present our model of a conflict scenario which is oriented to the use in our CBR setting. Here we develop the knowledge representation methodology that is based on approximation of a natural language description of a conflict [3].

It has been shown that an adequate description of mental world can be performed using mental entities and merging all other physical action into a constant predicate for an arbitrary physical action and its resultant physical state [2,3]. Furthermore, we

express a totality of sequential mental states for a scenario via a set of communicative actions that would unambiguously lead to these mental states. Hence we approximate a scenario as a sequence of communicative actions, ordered in time, with a causal relation between certain communicative actions. Our approximation follows the style of situation calculus, scenarios are simplified to a degree which allows effective matching as graphs.

Only a sequence of communicative actions remains as a most important component to express similarities between scenarios. Each vertex is a communicative action; directed edges depict the sequence of actions. The thicker edges link communicative actions which have the same argument. The arc denotes a causal link between the arguments of communicative actions, for example *service is not as advertised* $\Rightarrow$ *there are particular failures in a service contract,*
ask ~> confirm.

Let us consider an example of a scenario and its graph (Figure 1).

*I **asked** why the service was not as advertised*
*They **explained** that I did not understand the advertised features properly*
*I **disagreed** and **confirmed** the particular failures in a service contract*
*They **agreed** with my points and **suggested** compensation*
*I **accepted** it and **requested** to send it to my home address together with explanations on how it happened.*
*They **promised** to send it to me.*
*In a month time I **reminded** them to mail it to me*
*After two months I **asked** what happened with my compensation…*

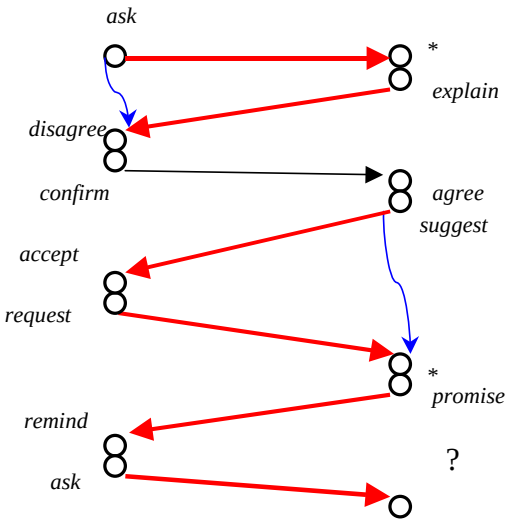

Fig. 1: The graph for approximated scenario

In our model of a conflict scenario, there is a pair of communicative actions per interaction: *receiving* communicative action and *sending* communicative action (with its node below).

3. Evaluation and user interface

In this section we present the results of preliminary evaluation of our CBR setting. Firstly, we evaluate our dataset consisting from 10 scenarios, 5 valid and 5 invalid. For each of ten scenarios, we set its class as unknown and verify if it can be related to its class properly, building common sub-scenarios with four representatives of its class and five foreign scenarios. Only 2 scenarios out of 10 failed to get proper assignment.

Secondly, our evaluation involved an improvement of existing software for processing customer complaints, *ComplaintEngine*, available for download at http://www.dcs.bbk.ac.uk/~galitsky/ComplaintEngineSuite.html. Our database primarily originates from the data on financial sector, obtained from the website of publicly available textual complaints PlanetFeedback.com.

Prior to the current study, *ComplaintEngine* used combination of deterministic machine learning and situation calculus reasoning machinery. When we augmented it with the CBR reasoning presented here, noticeable improvement of complaint recognition accuracy has been achieved. Judging on the restricted dataset of 80 banking complaints (40 complaints make the training set and 40 complaints have to be classified), the performance of *ComplaintEngine* was improved by 7% to achieve the resultant recognition accuracy of 83%. This increase of performance was mainly due to the dynamic update of the case database, minimizing the number of inconsistencies under prediction. Also, interactive update of the database of cases, a wider set of features (communicative actions and emotions [3]), hierarchical structure of stored cases have contributed in the overall increase of the accuracy of recognition (valid or invalid). We believe that this improvement fit well the general CBR methodology [6-8].

For a further evaluation, we used the data for fourteen banks, 20 complaints for training and 20 complaints for further evaluation. Firstly, the consistency of each training dataset is evaluated when complaint validity for each complaint is assumed *unknown* and classification is performed. This way we evaluate the completeness and consistency of the training dataset itself.

The resultant recognition accuracy is 72.5%. It is below the one (83%) obtained during the above mentioned preliminary evaluation due to a richer dataset of cases (. Being quite low in accordance to pattern recognition standards in such domains as speech and visual object recognition, this accuracy is believed to be satisfactory for the decision-support settings where the number of complaints which have to be re-assessed manually is relatively low. Obtained classification accuracy cannot be compared with 50% that one would get by a random prediction because our prediction setting requires *a highest accuracy providing an explanation for the case assignment*.

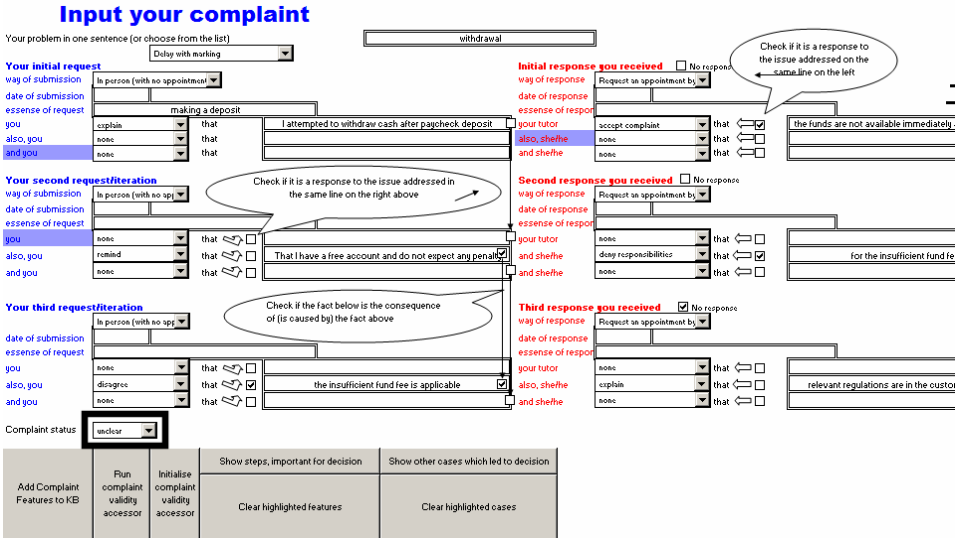

Fig.2. The screen-shot of the Interactive Complaint Form as a front end to the system *ComplaintEngine*

We proceed to the outline of the functionality of *Complaint Engine* [3]. The user interface to specify a complaint scenario is shown at Figure 2. Communicative actions are selected from the list of twenty or more, depending on the industry sector of a complaint. The parameters of communicative actions are specified as text in the Interactive Form; however they are not present in the formal graph-based scenario representation.

Communicative actions selected by a user in the list boxes constitute the vertices of the graph. Check boxes on the right of the list boxes are used to specify whether the incoming edge is thick (checked) or thin (unchecked). Check boxes linked with the vertical line are used to specify the causal link.

Having performed the justification of complaint validity, *ComplaintEngine* sets the list box for complaint status at “unjustified”. *ComplaintEngine* provides the explanation of its decision, highlighting the cases which are similar to a given complaint (e.g. unjustified), and which are different from it (respectively, justified). Moreover, *ComplaintEngine* indicates the communicative actions (cycles) that are common for given and other unjustified complaints to further back up its decision.

In a real-life scenario, a complainant has a choice to use the Interactive Form or to input complaint as a text so that the linguistic processor processes the complaint automatically and fills the Form for her. Communicative actions are selected from the list of twenty or more, depending on the industry sector of a complaint. The parameters of communicative actions are specified as text in the Interactive Form. Communicative actions selected by a user in the list boxes constitute the vertices of the graph (Figure 1). Check boxes on the right of the list boxes are used to specify whether the incoming edge is thick (checked) or thin (unchecked). Check boxes linked with the vertical line are used to specify the causal links (between the arguments of communicative actions).

Using the Interactive Form encourages the complainant to enforce a logical structure on a complaint, provide a sound argumentation. After complaint is partially or fully specified, the user evaluates its consistency. *Jasmine* then evaluates whether the current complaint (its mental component) is consistent or not, it may issue a warnings and advices concerning the improvement of the logical structure of this complaint.

When the complainant is satisfied with the response of *Jasmine*, she submits the completed form. The other option is if a user observes that it is not possible to file a reasonable complaint, it may be dismissed at this early stage by the author.

When submitted complaint is processed by a customer service representative, a wider set of tools is applied. When validity of a current complaint is estimated, Complaint Assistant provides a set of similar cases, and indicated particular communicative actions of a complainant or his opponent (from the standpoint of this complainant) which have led to a particular decision (compare with [8]).

4. Conclusions

In this paper we have demonstrated that CBR machinery is a convenient and effective tool to operate with formal scenarios. As we have observed our approach to CBR in the domain of multiagent interaction follows the general methodology [6,7]:

- It is important to store old similar cases in the case database to approach new ones;
- Scenarios are interpreted by comparison and contrasting with previously accumulated ones with (manually or automatically) assigned validity;
- Learning happens as a part of the process of integrating new cases into the current case database;
- If no consistent match can be achieved (similar to one class and not similar to other classes), *Jasmine* adapts the best solution by dynamically eliminating the cases from the case dataset which lead to such inconsistency.
- If a current problem cannot be solved by the existing case, manual assignment to a class and/or introduction of new status is required. The respective elements of the case database needs to be repaired (modified) to adjust to this new case.

Also, the ontology-based CBR architecture [16] was found to be adequate for processing customer complaints in our setting. In the domain where the cases can be only partially formalized (understood) because of high logical complexity and poor formalisation, we are fully taking advantage of a generic CBR framework.

We believe the current work is one of the first targeting practical applications of CBR in such domain as inter-human multiagent conflicts (particularly, complaints; compare with [9]). Our evaluation shows that CBR implemented via *Jasmine* is an adequate technique to handle sophisticated objects (both in terms of knowledge representation and reasoning) such as customer complaints. It has been achieved by integrating the reasoning components of deduction, induction, abduction and analogy [3,4] into the CBR framework of *Jasmine*.

References

1. Bratman, M.E. Intention, plans and practical reason. Harvard University Press (1987).
2. Galitsky, B. A Library of Behaviors: Implementing Commonsense Reasoning about Mental World. *8th Intl Conf on Knowledge-Based Intelligent Info Syst* (2004).
3. Galitsky, B. and Tumarkina, I.: Justification of Customer Complaints using Emotional States and Mental Actions *FLAIRS*, Miami, FL (2004).
4. Ganter, B. and Kuznetsov, S.O. Hypotheses and Version Spaces, *Proc. 10th Int. Conf. on Conceptual Structures*, ICCS'03, A. de Moor, W. Lex, and B.Ganter, Eds., Lecture Notes in Artificial Intelligence, vol. 2746 (2003), pp. 83-95.
5. Mitchell, T. Machine Learning, McGraw-Hill (1997).
6. Kolodner, J. Case-Based Reasoning. Morgan Kaufmann (1993).
7. Hoffmann, A., Sowmya. A. A New Approach for the Incremental Development of Adaptation Functions, *Advances in Case-Based Reasoning*, Springer, pp. 260 – 272 (2000).
8. McSherry, D.M.G. Explanation in Case-Based Reasoning: an Evidential Approach, Proceedings of 8th UK Workshop on CBR, pp 47-55 (2003).
9. Olivia,C., Chang, C.F., Enguix, C.F. and Ghose A.K. Case-Based BDI Agents: an Effective Approach for Intelligent Search on the World Wide Web . Intelligent Agents in Cyberspace. Papers from AAAI *Spring Symposium* (1999).
10. Muller, H.J. Dieng, R. (eds) Computational Conflicts : Conflict Modeling for Distributed Intelligent Systems Springer-Verlag New York (2000).
11. Guerra-Hernandez, A., Fallah-Seghrouchni,A. E. and Soldano, H. Learning in BDI Multi-agent Systems. CLIMA IV - Computational Logic in Multi-Agent Systems Fort Lauderdale, FL, USA (2004).
12. Stone, P., Veloso, M. Multiagent Systems: A Survey from a Machine Learning Perspective. Autonomous Robotics, 8(3) pp. 345-383(2000).
13. Weiss, G., Sen, S. Adaptation and Learning in Multiagent Systems. Lecture Notes in Artificial Intelligence, Vol. 1042. Springer-Verlag, (1996)
14. Laza, R. and Corchado, J. M. CBR-BDI Agents in Planning. *Symposium on Informatics and Telecommunications* (SIT'02), Sevilla, Spain, September 25-27, pp. 181-192 (2002).
15. Jasmine- a hybrid inductive logic programming machine learning system (2006) www.dcs.bbk.ac.uk/~galitsky/Jasmine.
16. Diaz-Agudo, B. and Gonzalez-Calero, P.A. Knowledge Intensive CBR through Ontologies, Expert Update Vol. 6, N. 1 (2003).

Authoring Tools in jCOLIBRI

Pedro A. González-Calero, Belén Díaz-Agudo, Juan A. Recio-García,
Juan José Bello-Tomás

Departamento de Sistemas Informáticos y Programación
Universidad Complutense de Madrid, Spain

pedro@sip.ucm.es

Abstract. In this paper we present the authoring tools that support the instantiation of jCOLIBRI, an object-oriented framework in Java for building Case-Based Reasoning systems. jCOLIBRI is a software artifact that promotes software reuse for building CBR systems, integrating the application of well proven Software Engineering techniques with a knowledge level description that separates the problem solving method, that defines the reasoning process, from the domain model, that describes the domain knowledge.Authoring tools in jCOLIBRI support, through a graphical interface, the configuration of a particular CBR system, built trough task decomposition and resolution from a library of readily available problem solving methods. These tools alleviate the effort required to understand and become proficient when reusing sophisticated software artifacts.

Keywords. Case-Based Reasoning, Object Oriented Frameworks, Authoring Tools

1 Introduction

Developing a CBR system is a complex task where many decisions have to be taken. The system designer has to choose how the cases will be represented, the case organization structure, which methods will solve the CBR tasks and which knowledge (besides the specific cases) will be used by these methods. This process would greatly benefit from the reuse of previously developed CBR systems.

jCOLIBRI [3] is a software artifact that promotes software reuse for building CBR systems, and tries to integrate the application of well proven Software Engineering techniques with the KADS [9] key idea of separating the problem solving method, that defines the reasoning process, from the domain model, that describes the domain knowledge. jCOLIBRI is an evolution of the COLIBRI architecture [4], that consisted of a library of problem solving methods (PSMs) for solving the tasks of a knowledge-intensive CBR system along with an ontology, CBROnto [5], with common CBR terminology. COLIBRI was prototyped in LISP using LOOM as knowledge representation technology. This prototype served as proof of concept but was far from being usable outside of our own research group. jCOLIBRI is a technological evolution of COLIBRI that incorporates in a distributed architecture a description

logics (DLs) engine, GUI clients for assembling a CBR system from reusable components and an object-oriented framework in Java.

One of the biggest problems with frameworks is learning how to use them. In order to alleviate framework instantiation effort, jCOLIBRI features a semiautomatic configuration tool that guides the instantiation process through a graphical interface. In this paper we focus mainly in this set of graphical authoring tools. The rest of this paper runs as follows. The next section describes the key ideas of jCOLIBRI. Section 3 describes the framework instantiation process using the authoring tools and Section 4 reviews related work and concludes.

2 jCOLIBRI

jCOLIBRI is an object-oriented framework in Java for building Case-Based Reasoning applications. A framework is a reusable, "semi-complete" application that can be specialized to produce custom applications [7]. Application frameworks are targeted at a given application domain providing the design for a family of applications within that domain.

jCOLIBRI is built around a task/method ontology, a knowledge level description [8] that guides the framework design, determines possible extensions and supports the framework instantiation process. Task and methods are described in terms of domain-independent CBR terminology which is mapped into the classes of the framework.

Although various authors have applied knowledge level analysis to CBR systems, the most relevant work is the CBR task structure developed in [1]. At the highest level of generality, they describe the general CBR cycle in terms of four tasks (4 Rs): *Retrieve* the most similar case/s, *Reuse* its/their knowledge to solve the problem, *Revise* the proposed solution and *Retain* the experience. Each one of the four CBR tasks involves a number of more specific sub-tasks. There are methods to solve tasks either by decomposing a task in subtasks or by solving it directly. The task structure identifies a number of alternative methods for a task, and each one of the methods sets up different subtasks in its turn. This kind of task-method-subtask analysis is carried on to a level of detail where the tasks are primitive with respect to the available knowledge (i.e. there are resolution methods).

2.1 Framework Design

The design of the jCOLIBRI framework comprises a hierarchy of Java classes plus a number of XML files. The framework is organized around the following elements:

Tasks and Methods XML files describe the tasks supported by the framework along with the methods for solving those tasks.

Case Base Different connectors are defined to support several types of case persistency, from the file system to a data base. Additionally, a number of possible in-memory Case Base organization are supported, thus configuring a two-layer architecture for case bases where any combination of persistency, in-memory organization can be settled.

Cases A number of interfaces and classes are included in the framework to provide an abstract representation of cases that support any type of actual case structure.

Problem solving methods The actual code that supports the methods included in the framework.

Tasks are the key element of the system since they drive the CBR process execution and represent the method goals. Tasks are just identified by its name and description included in an XML file. Tasks can be added to the framework at any time, although including a new task is useless unless an associated method exists.

Method descriptions follow an elaborated description that includes the following elements: *Name* of the class that implements the method, *Description* of the method, *Context Input Precondition* describing the applicability requirements for the method, *Type* of method (jCOLIBRI manages two types of methods: execution methods are those that directly solve the task, for which has been assigned to, while decomposition ones divide the task into other tasks), *Parameters* of the method, *Competences* (list of tasks this method is able to solve), *Subtasks* obtained when executing a decomposition method, and *Context Output Postcondition* that describes the Output data information obtained from this method execution.

For example, the CBRMethod implements the *CBR Task* by decomposing it in the 4 Rs subtasks. Notice how this description is coupled with task descriptions through the *Competence* and *SubTask* elements, and with the classes of the framework through the *Name* element, i.e., tasks with those names must have been included in the XML tasks file and a class with that name must exist in the framework.

3 JCOLIBRI Configuration Tools

jCOLIBRI is designed to easily support the construction of CBR systems taking advantage of the task/method division paradigm described in previous sections. Building a CBR system is a configuration process where the system developer selects the tasks the system must fulfill and for every task assigns the method that will do the job. Different types of CBR systems can be built using jCOLIBRI, from retrieval only to full featured 4 Rs systems.

In order to alleviate framework instantiation effort, jCOLIBRI features a number of GUI tools that support the management of task and methods as well as the construction of the particular combination of task/methods that defines a CBR system.

3.1 Task and Method Management Tools

Tasks and methods are the basic concepts in the jCOLIBRI framework. Ideally, the system designer would find every task and method needed for the system at hand, so that she would program just the representation for cases. However, in a more realistic situation a number of new methods may be needed and, less probably, some new task. Although advanced users might edit the corresponding XML files directly, these changes would become tedious and error prone. jCOLIBRI provides graphical configuration tools that allow developers to use the framework without editing XML files.

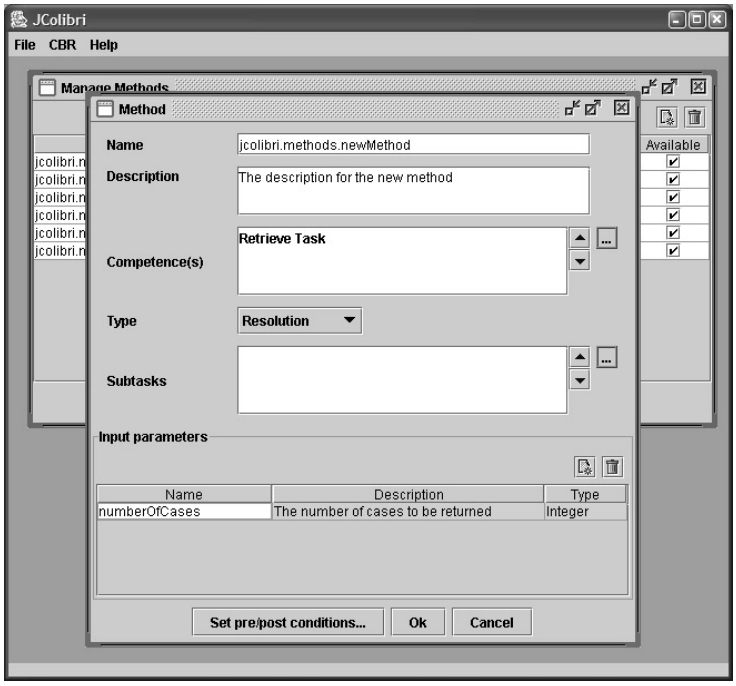

Figure 1: Adding a new method

The "Manage Task" option (from the main CBR menu) allows designers to retrieve the actual available tasks, and manipulate them without editing the XML file. Once the designer has chosen the system components by means of tasks declarations, it is time for checking the available methods. The "Manage Methods" option (from the main CBR menu) shows the methods defined in the system. In the same way, the designer can add or delete declarations of methods. The fields involved in the creation of a new method declaration can be discerned in Figure 1. The "Name" field identifies the class that will contain the method implementation. Both capability and subtask fields must be filled with previously declared tasks to avoid designer errors. It is possible to point out the execution conditions of a method related with its application context, as well as the configuration parameters that will be filled by the designer when he chooses a method for performing a task.

When the designer finishes a new method, it will be represented in the XML file and, at this point, it will be available for future use. Anyway, it must be remembered that the method is only defined but not implemented. As a user guide, the system will generate a java class scheme that the developer must implement.

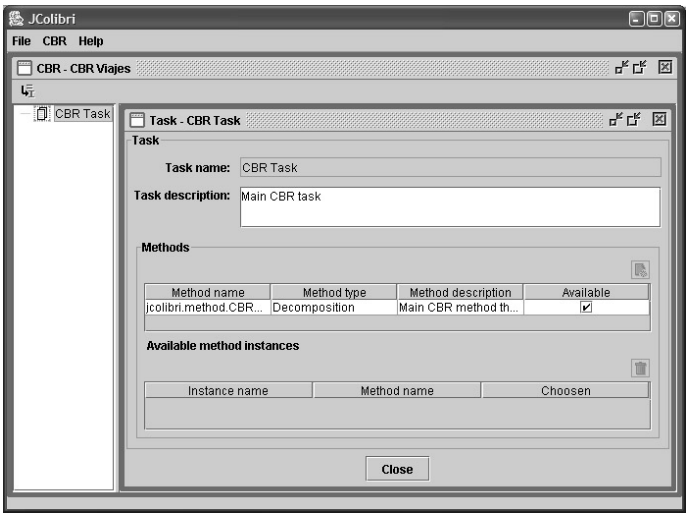

Figure 2: JCOLIBRI configuration interface.

3.2 System Configuration Tool

Once the system has all the required tasks and methods that the designer believes appropriate, the next step is the setup of the execution cycle to obtain a correct solution. This configuration process consists of the election of goal sequences by means of available tasks, and match them with suitable methods.

The "New CBR system" option (from the main CBR menu) opens the setup tool that facilitate the configuration process to create a new CBR system. This interface is dynamically built to reflect the actual contents of the task/method ontology, relying on the XML files describing task and method constraints and profiting from reflection facilities implemented in Java. Figure 2 shows the jCOLIBRI configuration interface. To the left is shown the task panel with the task tree. This tree shows the decomposition relations between tasks. To the right appears the task configuration panel where available methods for the given task in the given situation are provided. The configuration of a CBR system using this interface consists of the following processes:

- Defining the case structure, the source for cases and the case base organization.
- While the system is not complete, select one of the tasks without a method assigned, select and configure a method for that task. At startup the task tree has only one element, *CBRTask*, which is solved by a decomposition method that results in additional tasks. Task/method constraints are being tracked during the configuration process so that only applicable methods in the given context are offered to the system designer.
- Once the system is configured, the configuration code is generated so that a running CBR system is available. The configuration tool also provides a default interface for running the configured CBR system although in a real settlement an application specific interface should be developed.

4 Conclusions

Althoug some other efforts exist within the CBR community for developing domain independent development tools, such as CBR*Tools[6], Orenge [10] or CAT-CBR[2], jCOLIBRI is unique for being built around a task/method ontology that facilitates the understanding of an intrinsically sophisticated software artifact. A key aspect of this work is the availability of a semiautomatic configuration tool that facilitates the instantiation process.

The framework implementation is evolving as new methods are included. We are already porting previously developed CBR systems into the framework and, once tested, will make it publicly available to the CBR community. Apart from methods oriented to knowledge-intensive CBR in the line of our previous work, we are also including a growing list of standard CBR methods, needed to make jCOLIBRI an interesting framework for a bigger portion of the CBR community.

Our (ambitious) goal is to provide a reference framework for CBR development that would grow with contributions from the community. This reference would serve for pedagogical purposes and as bottom line implementation for prototyping CBR systems and comparing different CBR approaches to a given problem. In this sense, the evolution and further testing of the GUI configuration tools should promote the acceptance of the framework. For further information, jCOLIBRI and its user manual can be obtained visiting sourceforge: jCOLIBRI-cbr.sourceforge.net.

References

[1] A. Aamodt and E. Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 7(i), 1994.

[2] C. Abásolo, E. Plaza, and J.-L. Arcos. Components for case-based reasoning systems. *Lecture Notes in Computer Science*, 2504, 2002.

[3] J. J. Bello-Tomás, P. A. González-Calero, and B. Díaz-Agudo. JCOLIBRI: an object-oriented framework for building cbr systems. In *Procs 7th European Conference on Case Based Reasoning (ECCBR'04). Springer LNAI 3155.* 2004.

[4] B. Díaz-Agudo and P. A. González-Calero. An architecture for knowledge intensive CBR systems. In E. Blanzieri and L. Portinale, editors, *Advances in Case-Based Reasoning – (EWCBR'00)*. Springer-Verlag, Berlin Heidelberg New York, 2000.

[5] B. Díaz-Agudo and P. A. González-Calero. CBROnto: a task/method ontology for CBR. In S. Haller and G. Simmons, editors, *Procs. of the 15th International FLAIRS'02 Conference*. AAAI Press, 2002.

[6] M. Jaczynski. *Modèle et plate-forme à objets pour l'indexation des cas par situations comportementales: application à l'assistance à la navigation sur le Web*. PhD thesis, L'Université de Nice-Sophia Antipolis, 1998.

[7] R. Johnson and B. Foote. Designing reusable classes. *J. Object-Oriented Programming*, 1(5):22–35, 1988.

[8] A. Newel. The knowledge level. *Artificial Intelligence*, 18:87–127, 1982.

[9] A. T. Schreiber, B. J. Wielinga, and e. J. A. Breuker. *KADS: A Principled Approach to Knowledge-Based System Development. Volume 11 of Knowledge-Based Systems Book Series.* Academic Press, London, 1993.

[10] J. Schumacher. empolis orenge – an open platform for knowledge management applications. In *1st German Workshop on Experience Management*, 2002.

An Intelligent Peer-to-Peer Multi-Agent System for Collaborative Management of Bibliographic Databases

Hager Karoui, Rushed Kanawati, and Laure Petrucci

LIPN, CNRS UMR 7030, Université Paris XIII
99, avenue Jean-Baptiste Clément
F-93430 Villetaneuse, FRANCE
{hager.karoui, rushed.kanawati, laure.petrucci}@lipn.univ-paris13.fr

Abstract. This paper describes the design of a peer-to-peer system for collaborative management of distributed bibliographical databases. The goal of this system is twofold: firstly, it aims at providing help for users to manage their local bibliographical databases. Secondly, it offers the possibility to exchange bibliographical data among like-minded user groups in an implicit and intelligent manner. Each user is assisted by a personal agent that provides help such as: filling in bibliographical records, verifying the correctness of information entered and more importantly, recommendation of relevant bibliographical references. To do this, the personal agent needs to collaborate with its peers in order to get relevant recommendations. Each agent applies a case-based reasoning approach in order to provide peers with requested recommendations. The paper focuses mainly on describing the recommendation computation approach.

Keywords. Peer-to-Peer systems, Recommender systems, Case-based Reasoning, Bibliographical data sharing.

1 Introduction

Maintaining an up-to-date annotated bibliographical database is a central activity of research teams. However the multiplication of document sources (e.g. workshops, conferences, journals, etc.) as well as the on-line availability of most documents have contributed in making the task more complex and more time-consuming. Actually, in addition to the classical information overload problem, researchers have now direct access to papers that are seldom coupled with the complete bibliographical data. It is frequent now to start a new search session in order to find the exact reference of an interesting paper that was found previously. Luckily, researchers usually work in like-minded teams. It is highly possible that information obtained or known to one or more users can be useful to another. Moreover, colleagues may have useful hints about the quality of papers and what to read if interested in a given topic. It is obvious that

sharing bibliographical knowledge could not only enrich each member knowledge but also reduce time and effort required to manage personal databases. However, lessons learned for groupware design studies have shown that for collaborative tools to be successful, they should meet a number of requirements [4,5]. The effort required for using a collaborative tool should be as much as the same of that needed for using an individual tool that provides the same service. In addition, users will likely be willing to use a collaborative tool if they are well rewarded. In our example, a user that spends time in feeding a shared bibliographical base without getting any or few valuable recommendations will abandon the search.

Based on these remarks, we propose a new peer-to-peer multi-agent system for collaborative management of distributed bibliographical databases. Each user is assisted by a personal assistant software agent. An assistant agent observes the user interactions with a local personal bibliographical database. It tracks the user current hot topics and finds out information missing in the local database (e.g. the location of a cited conference, the number of pages of a given paper, etc.). Agents communicate with each other in order to find missing information but also to recommend their associated users with references, related to their hot topics, that have been found relevant by other colleagues. Agents can also verify the correctness of local references by comparing these with peers' records. The only additional effort required by the users is to accept or to reject provided recommendations. Since recommendations are made in the context of each user's hot topics, we expect that users will be willing to examine these recommendations.

A more detailed description of the system architecture and services is given in section 2. The paper's main focus is the recommendation computation technique. This is done by applying a case-based reasoning (CBR) approach. The CBR methodology has been used successfully by a number of recommender systems [3]. It allows for learning in an incremental manner by dynamically associating similar users. The reasoning cycle applied is detailed in section 3. Related work is discussed in section 4 and finally we conclude and give directions for future work in section 5.

2 System Description

Figure 1 illustrates the overall system architecture where each user maintains a personal bibliographical database locally. The user manages the local database using a management module that provides the classical management functions such as adding, deleting, editing and searching. It also provides functionalities for selecting a list of references to add to an ongoing work (e.g. building a report's bibliography) and exporting lists into different formats (e.g. BibTeX, html, pdf, etc.). In order to ease the exportation/transformation operations, the references are stored in an XML database. Each reference r is described by a record that contains the following information:

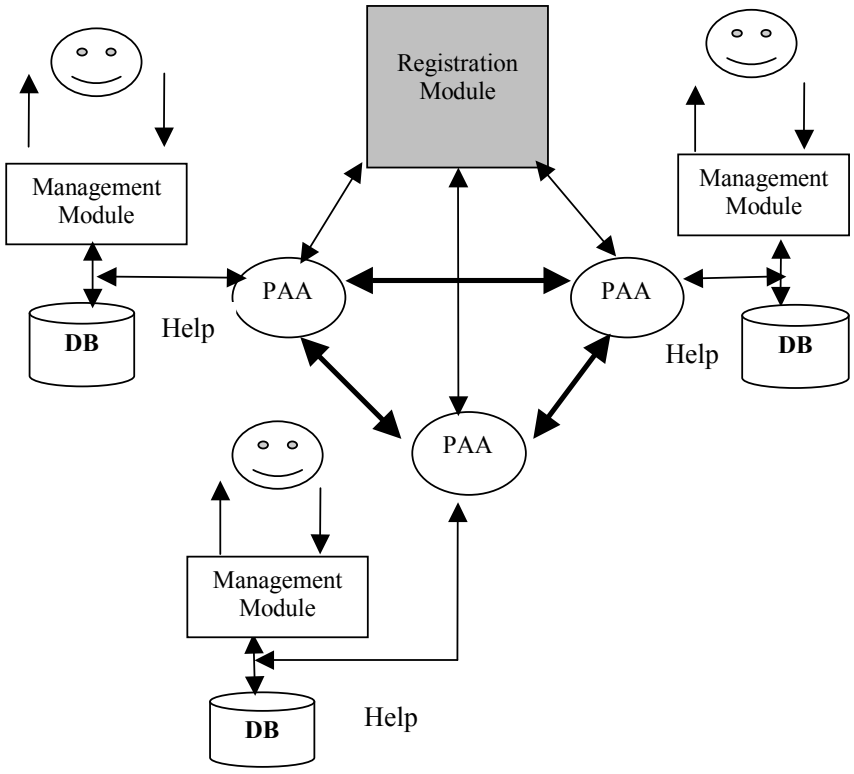

Fig.1. The system architecture

- Bibliographical data (denoted r.biblio): the classical data composing a bibliographical reference such as the type (e.g. Article, In Proceedings, Report, etc.), list of authors, title, etc.
- Keywords (denoted r.keywords): a list of keywords describing the reference.
- Topics (denoted r.topics): a list of topics the reference is related to. The same topic map is used by all users. In this work, we use a tree topic hierarchy. It is reasonable to say that the same research team uses the same topic trees to index the bibliographical references (e.g. using the ACM topic hierarchy in computer science research teams). However, we stress that the same hierarchy will be used differently by different users. Hence, the same reference can be related to different topics by different users. For example one may index all CBR-related papers to the same topic, e.g. CBR, while another user may index the same papers differently: some related to memory organization in CBR systems and others for CBR case maintenance. A third may index the same references as all related to lazy learning.

- Evaluation (denoted r.eval): an evaluation made by each user indicating the quality of the reference from the user's point of view. Evaluation is made on a 4-degree scale starting from very interesting going down to rubbish (if any)!
- Annotation (denoted r.annotation): a free-text field.

Each user is assisted by a personal agent that helps in managing her/his own database. Different services are provided by the local assistant:

- Editing references: when the user is editing a reference, the local assistant can help her/him with filling some data fields (e.g. when the user types an acronym of conference name, the assistant provides her/him with the complete name and can fill the other related information such as the conference date, location, etc.)
- References correctness: the correctness verification service is started when references are added or edited in the bibliographical database. After the data fields (e.g. title of conference, name of author, …) concerning a reference are filled and validated, the system checks the correctness of the input by starting the corresponding agent. A first check of the data correctness is done by matching the local database of the user. If the data is correctly written then the task is confirmed. Otherwise, if errors are found, the agent suggests a correction, or, if some fields are empty the agent suggests relevant data if available. If the data is not found in the local base, the agent collaborates with its counterpart agents in order to find them.
- Recommendation: this service aims at sharing bibliographical knowledge between the users.

The goal is to take advantage of past experiences of a single user or even a group of users for recommending more relevant references. A CBR approach is used to compute the different recommendations in a cooperative way, i.e. the different assistant agents must collaborate with each other to obtain various and relevant recommendations from several agents. We want the local assistant to suggest various and interesting recommendations to its user according to her/his current activity. The user can choose to either accept or refuse the recommendation proposed.

3 Recommendation Computation

3.1 Informal description

In order to provide users with relevant recommendations, each assistant agent applies the following computation cycle: first it computes the list of the most hot topics that interest the user. For each hot topic the agent sends a recommendation request to a set of peer agents that are likely to have references related to the topic. A recommendation request message contains a topic identifier and a list of keywords that describe the set of references saved under that topic in the local database. The agent may receive from each contacted agent, two lists of references: recommended references and unrecommended ones. The later list is formed of references that match the request but are badly evaluated by the associated user. The assistant agent merges and filters the received results in order to remove duplicated references as well as

references that are already stored in the local database (hence known to the user). The highly ranked references are then recommended to the user. The latter can accept to add all or some of these references to the indicated topic or to other topics. Users can also reject some or all provided recommendations. When receiving a recommendation request an agent searches the local database for references that match the received query. The search process starts from the topic indicated in the request. Recall that all users share the same topic hierarchy. When there is an insufficient number of references the agent continues the search in sub-topics following a depth-first search strategy. If not enough references are found the agent continues the search in super-topics. The rationale is to prefer references related to sub-topics that are likely to be more focused to those related to super-topics that are likely to be more general. The search process ends when enough relevant references are found or if the similarity between the requested topic and the searched topic falls below a given threshold. The relevance of a reference is computed by a similarity measurement that takes into account the similarity between topics (requested and searched one) as well as the match between the reference keyword description and the request keyword list.

3.2 Hot topics computation

As stated before all users share the same topic hierarchy that has a tree structure. All topics are not equally interesting for each user. Moreover, for each user the set of interesting topics changes over time. It is crucial for any recommender system to provide recommendations in the area that interests the user most. Recommendations providing should not be intrusive. Furthermore, if users are really interested in the recommendation area they will be more willing to evaluate these recommendations. In order to compute a user's hot topics list we apply a simple algorithm that measures the temperature of each topic. Topics that have a temperature above a given threshold σ will be labeled as hot topics. Initially, all topics have a zero temperature. Each action executed by the user involving a topic t will increase the temperature of that topic by a specified amount. Typical actions that modify a topic temperature are: adding, editing or searching for a reference related to the topic. Different actions add different values to the current topic temperature. A cooling function is also applied in order to decrease the temperature of topics not used by the user. As the topic hierarchy can be used differently by different users, we need to determine the most specific topics (e.g. deepest topics) the user's activity is centered on. To do so, a temperature propagation function is applied. Starting from the leafs of the the topic tree, each topic propagates its current temperature to the parent topic. Topics with a temperature above a system value σ_r will cease to propagate their temperature and will be added to the list of hot topics. The n hottest topics that have been added to the hot topics lists after visiting the tree in a bottom-up way will be returned. The heuristics is to return the most specific topics which concentrate a given level of the user's focus.

3.3 Committee formation

An important issue in any peer-to-peer system concerns the peer committee formation. The idea is to provide each agent with the ability to select a subset of available peers that are likely to provide the most relevant results (in our case recommendations). The goal is not only to enhance the overall system performances by reducing both the network and the agents load. The aim is also to enhance the quality of recommendations provided by avoiding noisy results. Different approaches are proposed in the literature. Some are based on the notion of agent reputation [1]. Others propose to apply automatic learning techniques in order to enable each agent to determine if it needs to increase the committee of peers and if it is the case which peer agent to invite [9]. While we totally agree about the importance of that issue, we have decided to employ a naive approach consisting of broadcasting recommendation requests to all available peers as an initial approach. We suppose that all agents could be assigned the same trust degree.

3.4 Delivering recommendations

A recommendation request message is composed of a triple: $R = <A, T, L>$ where A is the sender agent identifier, T is a target topic and L is a list of keywords that is computed from the set of keywords lists describing references related, directly or indirectly to the topic T. A reference is indirectly related to a topic T if it is related to a topic T' more specific than T.

When receiving a request, an agent starts to search its local database for references that match the pair (T,L). Informally, the keyword list contained in a request will be treated as a query, the designated target topic T indicates the starting point of the document search in the local database. The agent will retrieve from the local database references that match the received query. Reference/query matching is evaluated by a simple similarity function $Sim_{Ref}(R,r)$ that measures the similarity between a request R and a reference r. A reference r matches a request R if their similarity is above a given specified threshold σ_r. The similarity function is a weighted aggregation of two basic similarity functions: topic similarity (Sim_{Topics}) and keyword similarity ($Sim_{Keywords}$). Formally, we we have :

$$Sim_{Ref}(R,r) = \alpha\, Sim_{Topics}(R.T, r.topics) + \beta\, Sim_{Keywords}(R.L, r.keywords) .$$

where α and β are the basic similarities weights. Obviously, we have $\alpha + \beta = 1$. The keyword similarity function used is a simple function measuring the number of common words between two lists. Formally:

$$Sim_{Keywords}(A, B) = |(A \cap B) / (A \cup B)|.$$

The topic similarity measure uses the topics underlying hierarchical structure. The heuristic applied is the following: the similarity between two topics depends on the length of the path that links the two topics and on the depth of the topics in the

hierarchy. Recall that in a tree there exists only one path between any two nodes. Moreover, a match with specific nodes closer to leaf nodes results in a higher similarity than nodes matching at higher levels of the tree. Formally we have:

$$Sim_{Topics}(T_1, T_2) = 1 - (path(T_1, MSCA(T_1, T_2)) + path(T_2, MSCA(T_1, T_2))) / (path(T_1, root) + path(T_2, root)).$$

where path(a,b) returns the path length between nodes a and b, root is the topic's tree root and MSCA(a, b) returns the most specific common ancestor of nodes a and b in the topic tree.

Figure 2 shows a simple example of topic tree where the root is *Computing Methodologies*. We consider here four topic levels. Each topic has a list of associated references.

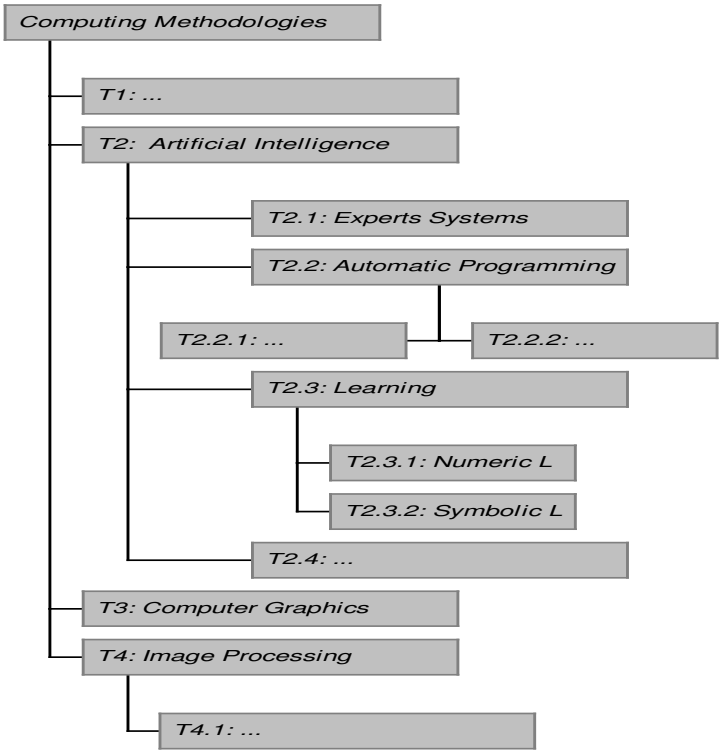

Fig.2. Example of topic tree

Based on the topic tree of Figure 2, we give some examples of topic similarity computation:

$$Sim_{Topics}(T_{2.3.1}, T_{2.3.2}) = 1-((1+1) / (3+3)) = 2/3$$
$$Sim_{Topics}(T_{2.3.1}, T_{2.2}) = 1-((2+1) / (3+2)) = 2/5$$
$$Sim_{Topics}(T_{2.2}, T_{2.3}) = 1-((1+1) / (2+2)) = 1/2$$

The topic similarity values are computed by applying the Sim_{Topics} function. We note that the value of the Sim_{Topics} ($T_{2.3.1}$, $T_{2.3.2}$) is higher than the value of Sim_{Topics} ($T_{2.2}$, $T_{2.3}$) because they are closer to leaf nodes. In addition, the topics ($T_{2.3.1}$, $T_{2.3.2}$) are more similar than ($T_{2.3.1}$, $T_{2.2}$).

Using these similarity functions, the local agent will try to return the *m* most relevant references that match the received request. Starting from the target topic (*R.T*) the agent will search for related references that are similar above a threshold σ_r. If not enough references are found, it examines references related to more specific topics, then it moves to more general topics. The retrieval process ends when *m* relevant references are found or when no more topic is left.

Relevant references that are positively evaluated by the local user form the list of recommended references while those badly evaluated form the list of unrecommended references. Each returned reference is associated with a score representing its similarity with the initial request. The agent that has sent the recommendation request will receive a couple of recommended and unrecommended references from each agent. It merges and filters the received lists, it eliminates firstly duplicated references and those which are already known to the user (those actually stored in the local database). Then it applies a sorting method in order to rank recommended and unrecommended references regardless of their similarity values. In a next step another value presenting the importance of the sender agent will be taken into account. The *k* highly ranked recommended references will be proposed to the user.

4 Related Work

One interesting work directly related to our approach is the Bibster system [2]. The main goal of Bibster is to allow a group of people to search for bibliographical references in each other personal database. A peer-to-peer architecture is used. However, only exact searching is supported. Our system represents an extension to the Bibster system, where similarity-based searching is used. Moreover, the search is made by software agents instead of being initiated by the users themselves.

Collaborative bookmarking systems address a similar problem [7]. However in peer-to-peer collaborative bookmarking systems [6] we lack an unified hierarchy of topics making the matching evaluation harder to compute. Another similar work is the I-SPY information searching engine [12]. The goal of the I-SPY system is to allow a group of like-minded people to share their search results in an implicit way. The system is built in a centralised manner where a hit matrix records for each submitted query the documents selected by the user. When a new query is submitted, results that have been selected by other users in response to similar past queries are provided by the system. In our system the set of topics can be viewed as a set of pre-specified queries. The overlap between user queries is much more likely to happen than in the case of open vocabulary queries.

In [8], McGinty and Smyth describe a collaborative case base reasoning CCBR architecture, which allows problem solving experiences to be shared among multiple agents by *trading* cases. This approach was applied in personalised route planning and it promises a solution by allowing a given user agent to borrow cases from similar agents that are familiar with the target territory. There is a tradeoff between agent similarity and their case base coverage. That is, a remote agent is useful to a target agent if it has a different coverage, but to be considered similar, they must share a set of common problems. Plaza & all investigate in [10] possible modes of cooperation among homogeneous agents with learning capabilities. They present two modes of cooperation among agents: Distributed Case based Reasoning (DistCBR) and Collective Case based Reasoning (ColCBR). In the DistCBR cooperation, an originating agent delegates authority to another peer agent to solve the problem. In contrast, ColCBR maintains the authority of the originating agent, since it decides which CBR method to apply and merely uses the experience accumulated by other peer agents. They prove that the result of cooperation is always better than no cooperation at all. However, these protocols are domain dependent and are the result of a knowledge modelling process. In [11], Plaza and Ontanon present several collaboration strategies for agents that learn using CBR. Agents use a market mecanism (bartering) to improve the performances both of individual and the whole multi-agent system. Two policies are presented: Committee policy and Bounded Counsel policy. In the first collaboration policy, the member agents of a MAC system are viewed as a committee. An agent that has to solve a problem sends it to all the other agents of the committee. The final solution is the class with the maximum number of votes. The next policy is the Bounded Counsel where an agent tries to solve a problem by himself and if it fails to find a "good" solution, then it can ask counsel to other agents in the MAC system. They conclude that the Committee policy is better than the Isolated and Bounded Counsel, however, this precision has a higher cost since a problem is solved by every agent. Because of the bias of the examples of an agent which decreases the accuracy of the system, they propose the collaboration strategy based on bartering cases to improve the performance of the MAC system by diminishing their individual case base bias. The mechanism of case bartering is based on bartering agreement where the result of the interchange diminishes the agents's individual case bases bias.

5 Conclusion and Future Work

This paper describes work in progress that aims at providing a group of like-minded people with an intelligent and an implicit way to share their bibliographical knowledge. Currently, we are working on implementing the first version of this system. The CBR approach described in this paper is limited to the use of similarity guided searching. However we are working on defining a deeper CBR approach that would enable agents to speed up their response time by exploiting similarities between current and past recommendation requests. Another important problem to handle is the committee formation problem: learn how to form the optimal committee for each recommendation request? This is still an open question. Another research direction that we will consider in the future is the application of collaborative CBR schemes in order to enhance the individual recommendation quality of each assistant agent.

References

1. M. Ammar, M. Gupta and P. Judge. A reputation system for peer-to-peer networks. In 13 th international workshop of networks and operating system support for digital and video, Montreal, 2003
2. J.Broekstra, M. Ehrig, P. Haase, F. Harmelen, M. Menken, P. Mika, B. Schnizler, and R. Siebes. Bibster: a semantics-based bibliographic peer-to-peer system. In Proceedings of SemPGRID'04, 2nd Workshop on Semantics in Peer-to-Peer and Grid Computing, pages 3-22, New York, USA, may 2004
3. Robin Burcke. Hybrid recommender systems with case-based reasoning. In 7th European Conference on Advances in Case-based Reasoning ECCBR 04, LNAI 3155, pages 91-105, Madrid, 2004
4. J. Grundin. Why CSCW applications fail: Problems in the design and evaluation of organisational interfaces. In Proceedings of the first ACM Conference on Computer Supported Cooperative Work, 1998
5. R. Kanawati. Groupware: Architectural and control issues. PhD thesis, Institut National Polytechnique de Grenoble, 1997
6. M. Malek and R. Kanawati. Cowing: A collaborative bookmark management system. In Proceedings of CIA'02: International workshop on cooperative information agents, LNAI 2182, pages 34-39, 2001
7. M. Malek and R. Kanawati. Informing the design of shared bookmark systems. In Proceedings of RIAO'2000: Content-based Multimedia Information Access, 28 April 2000, Paris, 2000
8. L. McGinty and B. Smyth. Collaborative case-based reasoning: Applications in personalised route planing. In ICCBR, pages 362-376, 2001
9. S. Ontanon and E. Plaza. Learning to form dynamic committees. In Proceedings of the second international joint conference on Autonomous agents and multiagent systems, pages 504-511, Melbourne, Australia, 2003. ACM Press, NEW YORK, USA
10. E. Plaza, J. L. Arcos, and F. Martin. Cooperation modes among case-based reasoning agents, 1996
11. E. Plaza and S. Ontanon. Cooperative Multiagent Learning. Springer, London, March 2003
12. B. Smyth and E. Balfe. Case-based collaborative web search. In 7th European Conference on Advances in Case-based Reasoning ECCBR 04, LNAI 3155, pages 489-503 Madrid, 2004

Explanation-Oriented Critiquing

James Reilly, Kevin McCarthy, Lorraine McGinty and Barry Smyth

Adaptive Information Cluster, Smart Media Institute,
School of Computer Science and Informatics, University College Dublin (UCD), Ireland

james.d.reilly@ucd.ie

Abstract. Interactive recommender applications are an important technology for online retailers who want to increase their sales by providing potential buyers with suitable recommendations. Critique-based navigation, where a user applies a directional critique to a specific feature of a recommendation, is often used to guide users through such complex product spaces. Dynamic critiquing is a novel extension of this approach, where users are presented with compound critiques that reflect the product options that are available in a given cycle of a recommendation session. We believe that these compound critiques have considerable explanatory capability. In this paper we are interested in how these kind of explanations can be used to help the user to better understand, and effectively navigate, complex product spaces in interactive recommender systems by understanding the trade-offs that might exist between product features. We present sample screen-shots from a purpose-built prototype application to illustrate the potential benefits of our approach.

1. Introduction

The advent of Electronic Commerce has opened up a whole new world to both online shoppers and retailers. One of the many benefits to both parties is that the range of product opportunities that exist is no longer limited by the amount of available real-estate. However, strange as it may seem, the vastness of the product-space also poses an enormous problem to both; online shoppers often find it difficult to find what they want, and this results in missed sales opportunities for the retailer. Several approaches have been proposed to help users navigate through the complex product space of an electronic shop [9]. While *keyword search* and *category-based browsing* are widely used, both online shoppers and retailers have reported that they are substandard [7]. Instead many well-known online retailers, such as Amazon.com, use recommender technology to help the user to narrow-down the range of possible options [13].

Although recommender systems are a popular solution to the narrowing problem, one criticism is commonplace. Typically, recommendations are retrieved on the basis of their *match score*; that is, how closely they match the user's evolving query. Rarely is either the score or how it was computed shown to the user, and even when it is this provides little comprehension value to the user. It is this lack of transparency that prevents users from perceiving the recommendations as credible [6]. Instead users prefer the feature-based style of navigation, where the relationship between their stated requirements and the provided product descriptions is obvious. Critique-based navigation, proposed by Burke *et al.* [3], is an example of a navigation approach often used by recommender systems. The approach is interactive and incremental and does not require the user to

have a specified need at the start. However it is susceptible to two distinct problems; first, this approach can lead to retrieval failures depending on the order in which the critiques are executed. Secondly, since only one feature critique can be executed in a given cycle, lengthy recommendation sessions tend to result.

In previously published work, we have described our *Dynamic Critiquing* approach, which concentrates on presenting the user with a selection of appropriate *compound* (i.e., multi-feature) critiques. Importantly, these compound critiques are representative of the product opportunities that exist at a given point in a recommendation session, as they are generated in real-time. We have already demonstrated the potential of our dynamic critiquing approach when it comes to improving recommendation efficiency. In this paper, we turn our attention to the explanatory benefits of the dynamic critiquing approach. Explanations have an important role to play in helping users to understand the suggestions made by recommender systems. Much of the research conducted to date has focused on ways to justify a particular recommendation to the end-user. Here, we take a different stance by proposing that compound critiques serve as rich explanations, as well as functioning as an intuitive and efficient navigation mechanism. Specifically, we suggest that they help the user to understand the recommendation opportunities that exist beyond the current suggestion, on the assumption that this current suggestion does not satisfy all of the user's implicit requirements.

We present a prototype recommendation application that demonstrates how these kind of explanations can be used to help the user to effectively navigate complex product spaces in interactive CBR systems. The following sections describe how our recent work, in the area of *Dynamic Critiquing* [12], has potential explanatory benefits that better support the user's navigation task, as well as significant recommendation efficiency benefits. First, we discuss some of the related work in the broader area of Case-Based Explanation (CBE), and distinguish how our contribution differs from other approaches relating to product-space navigation.

2. Related work in the Area of Case-Based Explanation

Early work in the area of CBE has focused on the use of explanation-based techniques in order to drive the CBR process model. In this sense explanation structures are generated to fulfill various functions within a CBR system: the explanations are constructed **by** the system, **for** the system (see for example, [1, 2, 8]). More relevant here is the use of explanations for the benefit of the user by developing systems that are capable of explaining the reasoning steps and conclusions. The following sections discuss some of the most recent work that has been carried out in the areas of (1) diagnosis and classification tasks, and (2) product recommendation tasks.

2.1 CBE for Diagnosis and Classification

Diagnosis and classification systems may generate explanations in order to justify a predicted outcome and satisfy the user of its validity. There is considerable optimism among the CBR community regarding the potential value of case-based or *precedent-based* approaches to explanation, when compared to their more traditional rule-based counterparts. The argument has been made that past cases provide a more natural and convincing form of explanation. For example, Cunningham *et al.* [4] report how real

users find similar cases to be more convincing than rule-based explanations in a classification task.

However, using the most similar case in order to justify or explain a particular classification outcome to the user is perhaps the simplest form of case-based explanation and many issues remain a source of active research within the community. Recently, the work of Doyle *et al.* [5] demonstrates that in some classification tasks presenting the nearest-neighbour case may not be the best way to explain or justify a particular classification outcome. This occurs when the nearest-neighbour happens to be farther from the decision surface than the target case. The work demonstrates how superior explanation cases can be selected by using an explanation utility metric that formalises this idea.

Presenting the user with an *explanation* case is just the beginning of the explanation process and on its own may not be sufficient to convince the user that a particular diagnosis or classification outcome is justified. For instance, as McSherry points out, attempting to justify a predicted outcome by presenting the user with the particular explanation case (whether it is chosen because it was a nearest-neighbour to the target or because of some alternative strategy), ignores the possibility that some of the features of this explanation case may conflict with some of the target features [10]. This may mislead the user if they mistakenly view these opposing features to be evidence in favour of the predicted outcome [10]. To combat this problem, McSherry proposes an evidential approach to precedent-based explanation [10]. The user is presented with evidence that both supports and opposes a particular outcome in order to explain the pros and cons of case-based conclusions.

2.2 CBE for Product Recommendation

Generally a product recommender may use explanations to explain (1) the reasons WHY a particular product was recommended, or indeed why there is no product that can be recommended [11], and (2) WHAT opportunities remain; that is, *"where can I get to from here"*, when presented with an unsuitable recommendation. Since the vast majority of related work in the area of CBE and product recommendation has also concentrated on the idea of using cases as a source of explanation, many of the research efforts discussed above apply here also. Of course, product recommenders may not always be able to satisfy all of a user's requests so it is important that the system tries to explain the cause of the retrieval failure. A good example of this is presented in [11], which describes a mixed-initiative approach to recovery from retrieval failure by helping the user to eliminate certain constraints from their initial queries in a conversational product recommender. An important aspect of this work is that explanations are generated in order to explain retrieval failures by highlighting subsets of query features that cannot be satisfied (e.g. *"there are no cameras with price less than €300 and resolution greater than 4 mega-pixels"*). This is particularly relevant and complementary to the work described in this paper if one regards a recommendation cycle where the user has not found their target product to be a type of retrieval failure.

In our approach [12], instead of trying to explain the retrieval failure we attempt to explain the retrieval opportunities that remain (e.g. *"there are 10 cameras less than €300 but their resolution is between 1 and 4 mega-pixels"* or *"there are 20 cameras with between 4 and 6 mega-pixels but their price is more than €300"*). To put this another

way: instead of explaining what types of products do not exist, we explain what types of products do exist. We argue that this type of *positive* explanation is likely to be more useful as it is more intuitive for users to respond to explanations that tell them where they can go rather than where they can't go.

3. Explanation-Guided Navigation

In the following sections we describe how the standard approach to critique-based navigation operates, discuss how it is subject to retrieval failures, and suggest how the user may better understand remaining product opportunities through explanation.

3.1 The Standard Approach to Critique-Based Navigation

The standard critique-based navigation policy [3] is very simple. The key idea is that by critiquing presented examples (i.e., cases), the search can be re-directed to home-in on appropriate products for the interacting user. Very briefly, each recommendation session is initiated by a vague user query and this results in the retrieval of the most similar case (*the recommended case*), and a set of fixed directional feature critiques. The user has opportunity to accept this case, thereby ending the recommendation session, or to critique it in line with their requirements. For example, a digital camera recommender might present the user with a particular camera recommendation and the user may request to see something *"like this but cheaper"*; here *cheaper* is a critique over the price feature. Individual critiques act as a filter over the remaining cases, such that the case chosen for the next cycle is the one that is compatible with the critique and maximally similar to the previously recommended case.

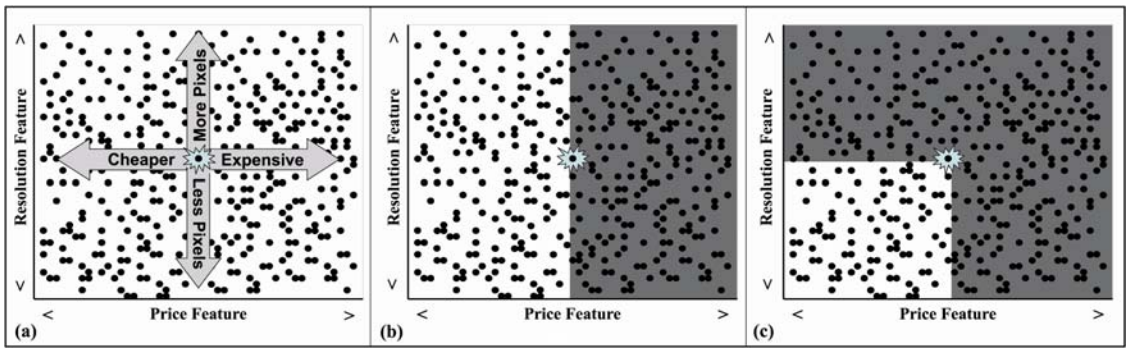

Figure 1. Product space; (a) two features and their available critiques, (b) cases available after "*cheaper*" unit critique chosen, (c) cases available after "*cheaper & less pixels*" compound critique chosen (non grey area).

Figure 1 illustrates how critiquing serves as a useful navigation mechanism. When the user is presented with a particular recommendation, he can immediately re-direct the search focus by applying a navigation action (i.e., feature-critique) in line with his requirements. Figure 1(a) shows two features, Resolution and Price, and their available critiques. The user can now choose to critique either of these features in two directions, respectively. Figure 1(b) shows the product space after the user has decided to look at cheaper products by choosing the *cheaper* critique. This critique has re-directed the

retrieval focus to a different part of the product space and closer to what the user is looking for.

3.2 The Problem: Limited Visibility

From a user's perspective it is often difficult to know how best to critique a proposed case in order to make useful progress through the product space. For example, consider a recommender system for Digital Cameras and suppose the user is presented with a *€300* model with a *2 mega-pixel* resolution, *64MB* of card-memory and *3x* optical zoom. Let's suppose that the user is actually looking for a greater resolution and more memory, but at a lower price. Ordinarily he would start by critiquing one of the features, *price, resolution or memory*. There may be no cameras which satisfy all three of these constraints and by requesting a camera that is *cheaper* he may find that he is recommended a new camera that is cheaper but has lower resolution and less memory, which may not be satisfactory. The point is, that normally this type of recommender system provides the user with little visibility; that is, it does not help the user to understand what product options exist beyond the current case. This can mean that the user spends time following false leads and backtracking as he tries different sequences of critiques.

We believe that explanations can play a useful role in this regard by providing the user with an indication of the type of products that remain available. For instance, in the above example it might be useful for the system to highlight that there are 30 products with a lower price and greater resolution, but that these products also have less memory. Or that there are 15 products with greater resolution and more memory but they are also more expensive.

3.3 Compound Critiques as Explanations

Previously [12], we have described how to generate a set of compound critiques during each recommendation cycle, and a method for presenting the promising compound critiques to the user for consideration. Each compound critique covers a number of features (typically 2 or 3) and they are generated in real time based on the cases that remain for a particular cycle. A data mining algorithm is used to extract common patterns of feature relationships from the remaining cases. These common patterns are then converted into compound critiques, the best of which are selected for presentation. Further details of how we dynamically generate and select compound critiques to present in each recommendation cycle can be found in [12]. We have also demonstrated the potential of our dynamic critiquing approach when it comes to improving recommendation efficiency.

The core hypothesis is that compound critiques help the user to better understand the *recommendation opportunities* that exist beyond the current cycle by helping them to appreciate common interactions between features. We believe that in many recommender domains, where the user is likely to have incomplete knowledge about the finer details of the feature-space, that compound critiques will help to effectively map out this space. For this reason we believe that users will actually find it easier to work with compound critiques, and their associated explanations, than unit critiques and this may, for example, help the user to make fewer critiquing errors. For instance, with standard critiquing in the digital camera domain a user might naively select the [*Price*<] unit critique in the

mistaken belief that this may deliver a cheaper camera that satisfies all of their other requirements. However, reducing price in this way may lead to a reduction in resolution that the user might not find acceptable and, as a result, they will have to backtrack. This problem is less likely to occur if the compound critique {[*Price*<],[*Resolution*<]} is presented because the user will come to understand the implications of a price-drop prior to selecting any critique. In 1(c) we can see that when a user picks a "*Cheaper & Less Pixels*" compound critique, the remaining product space is focused considerably.

4. A Demonstration Prototype

The importance of system transparency for fostering depth of user understanding has been shown by empirical studies [14, 15]. We propose that by presenting the user with a selection of compound critiques that best describe the product opportunities that remain, the user can gain a deeper understanding of the recommendation process. Thus, the compound critiques themselves serve as the explanatory mechanism that facilitates this understanding. To illustrate our idea, Figures 2-3 present a series of screenshots from a prototype application of our dynamic critiquing approach in the context of an online digital camera store. The screenshots present a sequence of recommendation cycles and in each we see the currently recommended case, its features and their unit critiques, plus a set of 3 compound critiques translated into natural language.

Figure 2: Initially the user is presented with a high-end Canon camera but selects the first compound critique to indicate that they are looking for something cheaper and are willing to come down on memory and resolution.

After the user has provided some initial information they are presented with a high-end Canon camera for €995 with 512MB of memory and 6.2 mega-pixels, as shown in Figure 2. The user can critique any of the individual features, such as *manufacturer*, *memory*, *resolution* etc., by selecting the appropriate critique icon on either side of the feature value fields that are displayed for the current camera. In addition, just below these features, three compound critiques are displayed. The compound critiques indicate to the

user what other types of cameras are available and help the user appreciate the relationships that exist between digital camera features. For example, in Figure 2, the first compound critique, *"Less Memory, Lower Resolution and Cheaper"*, tells the user that if he wants a cheaper model than the currently recommended camera then he should also expect to compromise on the *resolution* and *memory* features.

Figure 3: This camera is cheaper with less memory and a lower resolution. It is still too expensive though. The compound critique explains to the user that there are reputable manufacturers remaining, and that the lower resolutions offered are still acceptable given the significant price drop that is available.

Figure 3 shows the next recommendation cycle in which the user is presented with a different camera and a different set of compound critiques. In this cycle, the compound critiques are perhaps not as intuitive and understandable, but that is not to say they are any less relevant. By clicking on the 'explain' option for a compound critique the user can request a more detailed explanation. This explanation is presented in the pane to the right of the feature values, and defaults to be an explanation of the first compound critique. For example in Figure 3, the user asks for a further explanation of the third compound critique (*"Different Manufacturer, Lower Resolution and Cheaper"*). The resulting explanation tells the user that there are 87 remaining cameras that satisfy this critique—that is, there are 87 cameras that are cheaper, with a lower resolution, and made by a different manufacturer, than the currently recommended Sony camera. In addition, the explanation provides information about the ranges of values for these critiqued features. For instance, the user is told that these 87 cameras are made by manufacturers such as Canon, Fuji, Olympus, Kodak and Nikon, that they have resolutions from 1.4 to 4.8 mega-pixels, and that their price ranges from €125 to €399.

5. Conclusions

Explanations have an important role to play in helping users to understand the suggestions made by recommender systems. Much of the research conducted to date has

focused on ways to justify a particular recommendation to the end-user. In this paper we have taken a different stance by highlighting the importance of helping the user to understand the recommendation opportunities that exist beyond the current suggestion, on the assumption that this current suggestion does not satisfy all of the user's implicit requirements. Specifically, we discuss how compound critiques may serve as rich explanations that assist the user to better navigate the product space. Our work has been implemented in a live demonstration system and off-line evaluations have demonstrated that compound critiquing has a potentially valuable role to play in explanation.

Finally it is worth pointing out, of course, that our approach to explanation is but one of many different approaches to explanation. We have focused on the need to help users to understand what options remain available, if the current recommendations should not meet their requirements. As we have seen in Section 2, other approaches to explanation have a different focus, such as justifying a particular recommendation or explaining its pros and cons. In the future, it is likely that we will come to see many of these explanation strategies playing their own particular roles in the next generation of interactive recommender systems.

References

[1] A. Aamodt. Explanation-Driven Case-Based Reasoning. In S. Wess, K.D. Althoff, and M. Richter, editors, *Proceedings of the European Workshop on Case-Based Reasoning (EWCBR-94)*. Springer, 1994.

[2] R. Barletta and W. Mark. Explanation-Based Indexing of Cases. In *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88)*. AAAI, 1988. Minneapolis, MN, US.

[3] R. Burke, K. Hammond, and B.C. Young. The FindMe Approach to Assisted Browsing. *Journal of IEEE Expert*, 12(4):32–40, 1997.

[4] P. Cunningham, D. Doyle, and J. Loughrey. An Evaluation of the Usefulness of Case-Based Explanation. In K. Ashley and D. Bridge, editors, *Case-Based Reasoning Research and Development. LNAI, Vol. 2689.*, pages 191–199. Springer-Verlag, 2003. Berlin.

[5] D. Doyle, P. Cunningham, D. Bridge, and Y. Rahman. Explanation Oriented Retrieval. In *Proceedings of the European Conference on Case-Based Reasoning (ECCBR-04)*. Springer, 2004. Spain.

[6] B.J. Fogg. *Persuasive Technology: Using Computers to Change What We Think and Do.* Morgan Kaufmann, 2002.

[7] P. Hagen, H. Manning, and Y. Paul. Must Search Stink? *The Forester Report*, June 2000.

[8] L. Ihrig and S. Kambhampati. An Explanation-Based Approach to Improve Retrieval in Case-Based Planning. In M. Ghallab and A. Milani, editors, *Proceedings of the European Workshop on Planning (EWSP-95)*, pages 395–406. IOS Press, 1995. Amsterdam.

[9] A. Kobsa, J. Koenemann, and W. Pohl. Personalized Hypermedia Presentation Techniques for Improving Online Customer Relationships. *The Knowledge Enginnering Review*, 16(2):111–155, 2001.

[10] D. McSherry. Explanation in Case-Based Reasoning: An Evidential Approach. In B. Lees, editor, *Proceedings of the 8th UK Workshop on Case-Based Reasoning*, 2003. Cambridge, UK.

[11] D. McSherry. Incremental Relaxation of Unsuccessful Queries. In P. A. Gonzalez Calero and P. Funk, editors, *Proceedings of the European Conference on Case-Based Reasoning (ECCBR-04)*. Springer, 2004. Madrid, Spain.

[12] J. Reilly, K. McCarthy, L. McGinty, and B. Smyth. Dynamic Critiquing. In P. A. Gonzalez Calero and P. Funk, editors, *Proceedings of the European Conference on Case-Based Reasoning (ECCBR-04)*. Springer, 2004. Madrid, Spain.

[13] J.B. Schafer, J.A. Konstan, and E. Riedl. E-Commerce Recommendation Applications. *Data Mining and Knowledge Discovery*, 5(1/2):115–153.

[14] R. Sinha and K. Swearingen. On the Role of Transparency in Recommender Systems. In *Proceedings of the ACM CHI 02 Conference on Human Factors in Computing Systems.*, pages 830–831. Conference Companion, 2002.

[15] J. Zimmerman and K. Kurapati. Exposing Profiles to Build Trust in a Recommender. In *Proceedings of the ACM CHI 02 Conference on Human Factors in Computing Systems.*, pages 608–609. Conference Companion, 2002.

A Case-based Reasoning Approach to Handling Multiple Disorder Diagnostic Problems

Wenqi Shi and John Barnden

School of Computer Science, The University of Birmingham, Edgbaston, Birmingham B15 2TT, UK

w.shi@cs.bham.ac.uk

Abstract Multiple disorder diagnostic problem is a daily problem in medical diagnosis and treatment. Case-based diagnosis handling multiple disorders is often not sufficient in real world application, although Case-based reasoning has been applied successfully in some medical domains. In this paper, we present an approach which integrates two case-based reasoners for diagnosing multiple faults. This multiple case-based reasoning approach has been evaluated by two medical casebases taken from real world application and demonstrated to be promising.

Keyword Multiple Case-based Reasoning (Multiple CBR), Multiple Disorder

1. Introduction

The medical diagnosis problem absorbed much of the attention of Artificial Intelligence (AI) researchers since the medical domain is a weak or intractable domain and Artificial Intelligence has the potential to help diagnosis. In daily medical diagnosis and treatment, multiple disorders are a frequently occurring problem. Previous work on using AI techniques to diagnose multiple disorders has weaknesses such as knowledge acquisition difficulties, poor user acceptance, and maintenance problems.

Case-based Reasoning (CBR) employs previous experience to help current problem solving without necessarily understanding the underlying principles of the application domain [1]. Compared to other methods, Case-based Reasoning has the advantage of more flexible knowledge, automatic knowledge acquisition, easy system integration, and better user acceptance [2, 12]. It has been applied to and has presented very promising results on the medical diagnosis domain. However the majority of work applying Case-based Reasoning in Medicine has focused on single disorder diagnosis due to the Single Disorder Assumption [5] (only a single disorder or a fault is assumed to produce all observed findings), and little work has been done on multiple disorders [2]. Furthermore, traditional Case-based Reasoning method which works well in single disorder cases has a poor performance on multiple disorder cases. According to Baumeister's experiments, traditional Case-based Reasoning could solve only 4% cases in a real world casebase, with mean accuracy 0.73 on a scale of 0 to 1 [3].

In our paper, we present an approach which apply multiple case-based reasoning to fit multiple disorder problems. We evaluated our work with two real medical data sets, through which our method has been demonstrated to be promising.

2. Multiple Disorder Diagnostic Problem

Our context is a medical documentation and consultation system. In our application domain of sonography, the examination considers several partially disjunctive subdomains, eg. liver or kidney, which results in multiple disorders. To support diagnosis, we want to retrieve experiences such as

explanations for a query case based on the presented similarity to former cases. We also want to retrieve additional information about therapy, complications, prognosis or the treating physician as contact person for special questions. We use case-based reasoning to retrieve experiences and help diagnosis.

However case-based diagnosis handling multiple disorders differs from the one handling single disorder. For instance, for a single disorder casebase dealing with 100 disorders, the chance of reusing a case is roughly one to one hundred, while due to the combinatorial rules, the chance of reusing a case with 3 independent diagnoses from 100 alternatives is just one to one million [8]. In real world setting, one of our medical casebases contains about 7 disorders per case on average, and has 221 disorders in total, another casebase has a mean $M_d = 4.32 \pm 2.79$ of diagnoses per case and has 137 diagnoses in total. Thus the chance of reusing an entire previous case is very small. Our research investigated how to use multiple case-based reasoner to handle these multiple disorder situations.

3. Multiple Case-based Reasoning

We introduced Compositional adaptation in [15] and in later experiments, we found that Compositional CBR using Compositional adaptation performs better in multiple disorder only casebase. Here we say multiple disorder casebase with in a nallow sense, which means all the cases in this casebase has multiple disorder diagnosis output.

However, things are getting worse when multiple disorder cases are collected with single disorder cases by the doctors in their daily routine. To deal with the real world situation in which single disorder cases occur together with multiple disorder ones, we separate the existing mixed casebase into Single Disorder only CaseBase (SDCB) and Multiple Disorder CaseBase (MDCB), and integrate compositional case-based reasoning and traditioanl case-based reasoing.

Before we introduce our multiple case-based reasoning, we explain our similarity measurement as follows: For a query case c and another existed case c', we apply Manhattan distance for continuous or scaled parameters

$$md(x,y) = \frac{1}{k}\sum_{i=1}^{k} W_i \left| \frac{x - x_{\max}}{x_{\max} - x_{\min}} - \frac{y - x_{\min}}{x_{\max} - x_{\min}} \right|$$

and Value Difference Metric (VDM) for discrete parameters [14]

$$vdm_a(x,y) = \frac{1}{|\Omega_D|} \bullet \sum_{D \in \Omega_D} \left| \frac{N(a = x|d)}{N(a = x)} - \frac{N(a = y|d)}{N(a = y)} \right|$$

where x and y are values of parameter a in case c and c' respectively.

Given a query case c_q, we use a ReasonerSelector component to find out which case-based reasoner should be applied for this case. According to the result from the ReasonerSelector component, traditional CBR or compositional CBR will be applied. Compositional case-based reasoning retrieves a group of the most similar cases in multiple disorder casebase, then use compositional adaptation to get portions of the solutions of the most similar cases to get the final solution, while traditional case-based reasoning retrieves the most similar case and adopts the solution of that retrieved case. After that, we combine this query case with the candidate solution and restore it either into the single or the multiple casebase according to the number of diagnoses in candidate solution.

In the ReasonerSelector component, we retrieve the most similar case from both SDCB and MDCB respectively, and mark the corresponding similarity value as S_s and S_m. If S_s is greater than S_m,

then we assume that the final solution for this query case has more potential to be a single disorder, thus we use traditional CBR to retrieve the most similar case and adopt the solution for that most similar case. If S_s is smaller than S_m, then the solution comes up after compositional adaptation is applied. If S_s is equal with S_m, we prefer to use the single disorder casebase to get a single disorder solution, according to the Parsimony goal (also referred as "Occam's razor", the goal of minimizing the complexity of explanation [16].

3.1 Traditional Case-based Reasoning

Traditional Case-based Reasoning means retrieve the most similar case from the casebase, and adopts its solution as the diagnosis for the current enquiry case. In our multiple case-based reasoning, we apply traditional case-based reasoning to single disorder casebase.

3.2 Compositional Case-based Reasoning

The main idea of Compositional Adaptation is to combine multiple case solutions to produce a new composite solution. In our application, we retrieve a group of the most similar cases instead of the one most similar case from case base, and then we use a compositional adaptation approach to adapt multiple cases instead of adapting only one single case to get the solution. It means that after retrieving the *k* most similar cases, we calculate the frequency of the disorders appearing in the solutions of the k most similar cases and choose hypothesized disorders with high appearance [15].

4. Evaluation

In the usual task of assigning an example to a single category, the accuracy is just the percentage of cases which are correctly classified. But to quantitatively measure the accuracy of multiple disorder diagnosis, the simple accuracy measurement doesn't fit. Standard accuracy has also be demonstrated to be not very suitable for multiple disorder problem [10], which is defined as $(T^+ + T^-)/N$, where T^+ (True Positives) is the number of disorders in the correct diagnosis that are also in the system diagnosis, T^- (True Negatives) is the number of disorders not in the correct diagnosis and not in the system diagnosis, and N is the total number of disorders.

In our experiments, we adopt the *Intersection Accuracy* measure (*IA*) [10], which is defined as follows:

$$IA(D_c, D_c') = \frac{\left(\frac{|D_c \cap D_c'|}{|D_c|} + \frac{|D_c \cap D_c'|}{|D_c'|} \right)}{2} \tag{2}$$

where D_c is the set of correct diagnoses of the new case c and D_c' represents suggested diagnoses by our system.

We use Leave-One-Out Cross Validation which in our situation tests the case base *n* times, each time leaving one case out of the case base and trying to find a solution for this case based on the remaining cases.

4.1 Evaluation on Casebase 1

Our casebase 1 is from recently developed system SONOCONSULT. This case base consisted of 744 cases, among which there are 65 single disorder cases and 679 multiple disorder cases. The case base

contains an overall number of 221 diagnoses and 556 symptoms, with a mean $M_D = 6.71 \pm 04.4$ of diagnoses per case and a mean $M_F = 48.93 \pm 17.9$ of relevant findings per case.

In order to measure the performance of our method, we compare it with a recently developed system SONOCONSULT. SONOCONSULT constructs diagnostic profiles from the case base, infers basic set-covering knowledge from those diagnostic profiles and uses set-covering inference and case-based reasoning to build hypotheses for the current multiple disorder problem [3].

In our experiments, we use exactly the same case base and the same Intersection Accuracy measurement as SONOCUNSULT, thus this comparison would be fair enough. Since we are using the SONOCONSULT case base which includes 744 cases [11] and leave-one-out cross validation, we ran our program 744 times (n=744) and each time we generate one suggested diagnosis from the k most similar cases (empirically we got best performance when k =7).

After testing our method on this case base, we could solve 547 cases which is nearly 74% of the whole case base, with *Mean Accuracy* around 0.7004. In the following table, we present some of SONOCONSULT's results as well which demonstrates that our method improves the efficiency for solving multiple disorder problems compared to the Set-covering Strategy SONOCONSULT used, and its performance is much better than traditional CBR method measured by Baumeister [3].

Used Knowledge/Method	Solved Cases (Percentage)	Mean Accuracy
Traditional CBR	33 (4%)	0.73
Set-covering Strategy	443 (60%)	0.70
Multiple CBR	536 (72%)	0.70
Partition Class	624 (84%)	0.73

4.2 Evaluation on Casebase 2

Our casebase 2 consisted of 1370 cases, among which there are 65 single disorder cases and 679 multiple disorder cases. The case base contains an overall number of 138 diagnoses and 285 symptoms, with a mean $M_D = 4.32 \pm 2.79$ of diagnoses per case and a mean $M_F = 76.89 \pm 20.59$ of relevant findings per case.

After testing our method on this case base, we could solve 1072 cases which is about 78% of the whole case base, with Mean Accuracy around 0.67. In the following table, we present this result and the traditional CBR's performance.

Used Knowledge/Method	Solved Cases (Percentage)	Mean Accuracy
Traditional CBR	237 (17%)	0.57
Multiple CBR	1072 (78%)	0.67

We can see from the 4.1 (evaluation on the casebase 1), Multiple CBR performs much better than the traditional CBR, slightly better than the set-covering approach. This is probably due to the facts: Traditional CBR get the most similar case and adopt the solution, but in multiple disorder situation, there are huge combination possibility of both symptoms and disorders, thus even the most similar case from the multiple disorder casebase is usually not good enough for solving the query case, ie. Concerning the diagnostic results. While set-covering approach returns candidate cases in terms of cases with all their solutions and no sophisticated adaptation step is applied.

The knowledge-intensive method using partition class knowledge performs best. However the multiple CBR method and the set-covering approach do not need background knowledge, and so can be applied in arbitrary situations when the partitioning knowledge is not available, while the partition class strategy needs additional background knowledge. From the 4.2 (evaluation on the casebase 2),

Multiple CBR is tested on the second casebase, and this case base is nearly twice bigger than the first one. The results also show that Multiple CBR solved much more cases than traditional CBR.

5. Discussion

Furthermore, there are several points worth noting about our approach. Firstly, the casebased reasoning method itself corresponds to the diagnosing process that physicians always use. They recall former similar diagnostic case and compare the symptoms with those current patient have, make some adjustment based on the previous solution to adapt the current symptoms the patients have. The cognitive similarity between Case-based Reasoning and diagnosis makes it easy to get user acceptance.Secondly our method involves two case-based reasoners working together for handling different casebases. This is different from most case-based reasoning systems using simplex reasoners. It evokes an idea of using multiple case-based reasoner, each of which may be suitable for different casebase or dataset. Thirdly, our system deals with the problem of multiple disorders which hasn't been identified by most knowledge-based diagnostic systems [2]. This is due to the single disorder assumption, which assumes to use only one disorder to explain all the findings presented[16]. Fourthly, our approach uses flexible knowledge, and allows the automatic generation of the knowledge base from an existing database, which not only makes the CBR system easy to integrate into existing clinical information systems, but also, to some extent, avoids knowledge acquisition problem.

6. Previous Non-CBR Work on Multiple Disorders

INTERNIST is a system intending to diagnose the entire scope of general internal medicine. It matches symptoms and diseases based on forward and backward conditional probabilities [4]. The major problem is that INTERNIST does not deal with interacting disorders properly because if the findings can be explained by a disorder, then these findings will be deleted immediately no matter how these findings could also lead to diagnosis of another disorders.

Jang [5] analyzed another system CADUCEUS], saying that "The CADUCEUS program addresses the deficiencies in INTERNIST by using a combined hierarchical-causal network. … A main problem with CADUCEUS is that it depends on differential diagnoses that are hierarchically structured in advance. In ill-structured domains like medical diagnosis, however, the task of organizing differentials cleanly is a difficult problem in itself".

HYDI decomposes knowledge from the causal models into diagnosis units to prevent re-computation for similar problem to improve system efficiency [5]. But HYDI is only tested on heart disease since the diagnosis units in its association base largely rely on the causal models which have been built in Heart Failure Program (HF) on heart disease. Only when all the causal models for other disorders are available, could HYDI's method be applied to diagnose other disorders.

Wessles tried to use belief networks to diagnose multiple disorders [6]. The System HEPARII was extended to multiple disorder diagnosis by changing the structure of Bayesian network models [7]. In their works, they have to build either a belief network or a Bayesian network before diagnosing. These are all based on medical literature and conversations with some medical domain experts, which highlight a knowledge acquisition problem.

7. Conclusions and Future Work

In this paper, we introduce a multiple case-based reasoning approach to deal with multiple disorder problems. We apply compositional case-based reasoning to construct diagnostic solution from multiple disorder casebase and employ traditional case-based reasoning to access single disorder casebase. Using real medical data, this method has been demonstrated to be promising.

There are many opportunities for future work. Firstly, our method has a potential source of error, the

decision of which case-based reasoner should be choosen. We will investigate it in more detail in the future. Secondly, we believe that employing learning methodology to explore interactions between disorders will help to filter the candidate disorder or to add potential disorder during case adaption. Thirdly, experiments in other domains are desirable. Our work has the potential to be used to diagnose multiple faults in other diagnostic problem areas, such as diagnosis problems concerning machine faults.

Acknowledgement

Special thanks to Joachim Baumeister, Martin Atzmueller, Prof. Frank Puppe and Prof. H. P. Buscher who made the casebases available for us.

Reference:

[1] Janet Kolodner. Case-based Reasoning. Morgan Kaufmann Publishers, 1993.

[2] Lothar Gierl, Mathias Bull, and Rainer Schmidt. CBR in Medicine. In Mario Lenz, Brigitte Bartsh-Sporl, Dans-Dieter Burkhard, and Stdfan Wess, editors, *Case-based Reasoning Technology: From Foundations to Applications*, pages 273-297. Springer-Verlag, 1998.

[3] Joachim Baumeister, Martin Atzmueller, and Frank Puppe. Inductive learning for case-based diagnosis with multiple faults. In S, Craw and A, Preece, editors, *Advances in Case-based Reasoning (ECCBR2002),* pages 28-42. Springer Verlag, 2002.

[4] R.A.Miller, H. E. Pople, and J. D. Myers. Internist-1: an experimental computer-based diagnostic consultant for general internal medicine. *New England Journal of Medicine*, 307(8): 468-476, 1982

[5] Yeona Jang. HYDI: *A Hybrid System with Feedback for Diagnosing Multiple Disorders*. PdD thesis, Massachusetts Institute of Technology, 1993.

[6] Linda Gaag and Maria Wessels. *Efficient multiple-disorder diagnosis by strategic focusing*. In A. Gemmerman, editor, Probabilistic Reasoning and Bayesian Belief Networks, pages 187-204, London, 1995. UCL Press.

[7] Agieszka Onisko, Marek J. Druzdzel, and Hanna Wasyluk. Extension of the heparrΠ model to multiple-disorder diagnosis. In M. Kolpotek, M. Michalewicz, and S. T. Wierzchon, editors, *Intelligent Information Systems*, pages 303-313. Physica-Verlag, 2000.

[8] Martin Atzmuller, Joachim Baumeister, and Frank Puppe. Evaluation of two strategies for case-based diagnosis handling multiple faults. In *Proceedings of the 2nd German Workshop on Experience Management (GWEM 2003)*, Luzern, Switzerland, 2003

[9] Eleanor Rosch and Carolyn B. Mervis. Family resemblances: Studies in the internal structure for categories. *Cognitive Psychology*, 7: 573-605, 1975

[10] Thompson Cynthia A and Raymond J. Mooney. Inductive learning for abductive diagnosis*. In Proc. Of the AAAI-94*, volume 1, pages 664-669, 1994.

[11] H. P. Buscher, CH. Engler, A. Fuehrer, S. Kirschke and F. Puppe. HepatoConsult: A Knowledge-Based Second Opinion and Documentation System, *Artificial Intelligence in Medicine*, 24 (3), 2002

[12] Rainer Schmidt, Stefania Motani, Riccardo Bellazzi, Luigi Paorinale, and Lothar Gierl. Case-based reasoning for medical knowledge-based systems. *International Journal of Medical Informatics*, 64: 355-367, 2001

[13] H.E.Pople Jr. Heuristic methods for imposing structure on ill-structured problems: The structuring of medical diagnositics. In P. Szolovits (eds), *Artificial Intelligence in Medicine*, volume 51 of AAAS Selected Symposium Series, 119-190, Westview Press, Boulder, Colorado, 1982

[14] Wilson, D. R. and Martinez, T. R.. Improved heterogeneous distance functions. *Journal of Artificial Intelligence Research, 1997*

[15] Wenqi Shi, John Barnden, A Compositional Case Adaptation Approach To Multiple Disorder, In B. Lees (eds), Proceedings of 8TH UKCBR Workshop, Cambridge, 15th Dec, 2003.

[16] Peng, Y. and Reggia, J. A. , Abductive Inference Models for Diagnostic Problem-Solving. Springer-Verlag, 1990.

Virtual Attributes from Imperfect Domain Theories

Timo Steffens

Institute of Cognitive Science, Osnabrueck, Germany
tsteffen@uos.de

Abstract. This paper proposes to enhance similarity-based classification by virtual attributes from imperfect domain theories. We analyze how properties of the domain theory, such as partialness and vagueness, influence classification accuracy. Experiments in a simple domain suggest that partial knowledge is more useful than vague knowledge. However, for data sets from the UCI Machine Learning repository, we show that vague domain knowledge which in isolation performs at chance level can substantially increase classification accuracy when being incorporated into similarity-based classification.

Keywords. Similarity measures, imperfect domain-knowledge, virtual attributes

1. Introduction

One of the most prominent challenges in machine learning is to identify appropriate features for representing instances, since learning performance depends heavily on the features used. Particularly, the performance of similarity-based classification degrades with the number of irrelevant features [10]. It is also known that adding relevant abstract features can improve classification accuracy [17]. This paper focuses on identifying abstract features that can be used to extend similarity measures in similarity-based classification.

The main contribution of this paper is to show that additional features can be derived from domain theories that are imperfect. This will alleviate the knowledge acquisition bottleneck, as it reduces the overhead associated with obtaining expert knowledge. Although similarity-based classification is only used in domains where no perfect domain theories exist, usually there is imperfect domain knowledge and isolated chunks of knowledge [1,3,6,16]. For example, in [1] open and weak domain theories were integrated into a case-based reasoning system. Similarly, matching knowledge was used to improve the performance of the well-known PROTOS system [16]. Furthermore, it was shown that the combination of CBR and a domain theory outperforms both CBR and the theory itself [6]. In contrast to those weak theories, strong domain theories were used to filter irrelevant features [3].

We present a new approach that exploits imperfect domain knowledge in similarity-based classification by inferring additional abstract features. Furthermore, we analyze the impact of the knowledge's vagueness and partialness. The next section specifies the representation of cases, the similarity measure, and domain theories. Section 3 gives an overview over how additional features can improve classification accuracy. Section 4 reports experiments with two domains from the UCI Machine Learning Repository [5]. Finally, the last section concludes and outlines future work.

2. Representation of the CBR Modules

This section explains the representation details of cases, the similarity measure and the domain theory. We also define partialness, vagueness and inconsistency of a domain theory.

2.1 Cases and the Similarity Measure

A case C is made up of a set of attribute-values pairs for the attributes $A_1, A_2, \ldots, A_n$. While the original attributes can be either discrete or numerical, the additional virtual attributes in this paper are binary. Following the well-known local-global principle, we compute the similarity between two cases C_1 and C_2 as the weighted average aggregation of the attributes' similarities:

$$sim(C_1, C_2) = \sum_{i=1}^{n} \omega_i * s_i \qquad (1)$$

where $s_i = s_i(A_{i1}, A_{i2})$ are the local similarity values, and the ω_i are the corresponding weights.

2.2 Domain Knowledge

In this paper, we examine only domain knowledge that can be represented as a domain theory of the following form: A domain theory is a set of inference rules that relate concepts to each other. These rules specify which concepts exist in the domain and describe how abstract concepts can be inferred from more primitive ones [3]. We assume that the case representation language is compatible with the language of the domain theory, either by sharing primitives or by using a bridging language. The formal definition of the domain theory is skipped here, since it is in large parts analogous to Prolog rules, including logical connectors, equality, and comparison operators.

According to [14], the concepts in a domain theory can be divided into three types: observables are attributes that are directly represented in the cases. Classification goals are the attributes that are to be inferred or approximated. All other attributes are called intermediate attributes.

Intermediate attributes are the focus in this paper, because they are natural candidates for virtual attributes, that is, they can be added to the similarity measure in order to enhance the classification accuracy.

2.3 Properties of Domain Theories

Domains in which CBR is applied lack a perfect domain theory. Hence, the domain theories (or parts thereof) that we work with have at least one of the following properties (cf. fig. 1):

- Partialness: This is the case if some parts of the domain are not modelled, for example a) if conditions are used but not defined, or b) the relation of intermediates or observables to the classification goal is not known, or c) the classification goal does not exist in the rulebase at all. Note that these situations correspond to gaps at the "top" or "bottom" of the domain theory [14].
- Vagueness: Values can only be given within a certain confidence interval.
- Inconsistency: There are two or more rules (or even alternative theories) which make different classifications and it is not known which one is correct. CBR is often used to overcome this problem, because the cases provide knowledge which classification is correct for certain cases.

In this paper, we focus on partial and vague domain theories.

3. Virtual Attributes

Virtual attributes [17] are attributes that are not directly represented in the cases but can be inferred from the already existing attributes. They are useful if the monotonicity-principle is violated. If $sim(C_1, C_2) > sim(C_1, C_3)$ is necessary to reflect class membership, then there must at least be one pair of local similarities, so that $s_i(A_{i1}, A_{i2}) > s_i(A_{i1}, A_{i3})$. If such a pair does not exist, the similarity measure must make use of interdependencies between attributes. For example, the similarity may not depend on two attributes A_1, A_2 themselves, but on their difference $A_1 - A_2$. Virtual attributes can express such interdependencies (e.g. *deposit(C) = income(C) - spending(C)*) and can also encapsulate non-linear relations.

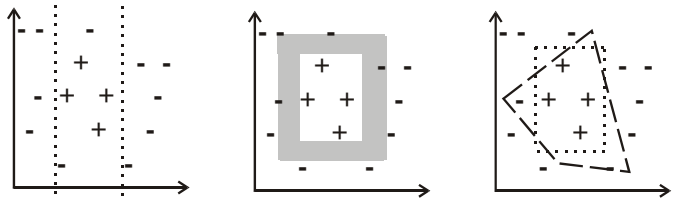

Fig. 1. Properties of domain theories. The theories describe parts of the target concept, of which there are positive (+) and negative (-) instances. Left: Partial knowledge, only parts of the concept boundaries are known. Middle: Vague knowledge, concept boundaries are believed to be somewhere within the shaded areas. Right: Inconsistent knowledge, different rules make differing predictions

We propose to use intermediate concepts of domain theories as virtual attributes. Virtual attributes can easily be added to the set of attributes of each instance. It is crucial to note the difference: Intermediate attributes are contained in domain theories, virtual attributes are those that are added to the similarity measure. Neither of the two notions implicates the other.

Every virtual attribute forms an additional dimensions of the instance space (see fig. 2 (right)). This is most intuitive for numerical attributes. An example is the concept *expectedWealthTillRetirement(C) = (65-age(C)) * income(C)*. Unfortunately, these dimensions can change assumptions about instance distributions and are most likely not orthogonal to the other dimensions, since they are inferrable from other attributes.

In this paper we focus on binary virtual attributes. Although formally they are treated as additional dimensions, they can be visualized as separating lines within the original instance space (see fig. 2 (left)). They divide the instance space into two regions. For example, *taxFree(C)* $\leftarrow$ *income(C) < 330* may divide some instance space into salaries that are or are not subject to paying taxes in Germany. We will show that the most useful virtual attributes describe target concept boundaries.

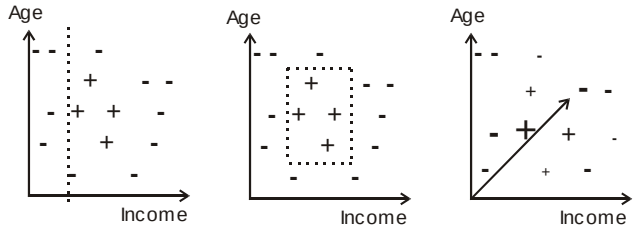

Fig. 2. Types of virtual attributes. Left: A binary virtual attributes divides the instance space into instances satisfying or not satisfying it. Middle: A conjunction of binary attributes. Right: The most general type of virtual attributes is to add a dimension to the instance space

Intermediate attributes that are fully defined (i.e. that do not have gaps at the bottom of the domain theory) can be computed from the values of observables and other intermediates. In order to use an intermediate as a virtual attribute, it is added to the local similarities of the similarity measure. $s_i = 1$, iff both instances satisfy the intermediate concept *i* or both do not satisfy it, and $s_i = 0$ otherwise. We do not treat virtual attributes as additional dimensions in this paper, so in the following additional attributes are assumed to be discrete.

Let us look at how binary virtual attributes influence classification. Assume for sake of illustration that the instance space is formed by the attributes *temp* and *press* denoting the temperature and pressure of a manufacturing oven. Let us assume furthermore that the (to be approximated) target concept is *hardened(C)* $\leftarrow$ *temp(C) > 100* $\wedge$ *temp(C) < 150* $\wedge$ *press(C) > 2* $\wedge$ *press(C) < 3*.

The error distribution of an unweighted k-NN-classifier for the target concept is depicted in fig. 3 (left). Not surprisingly, the misclassifications occur at the boundaries of the target concept.

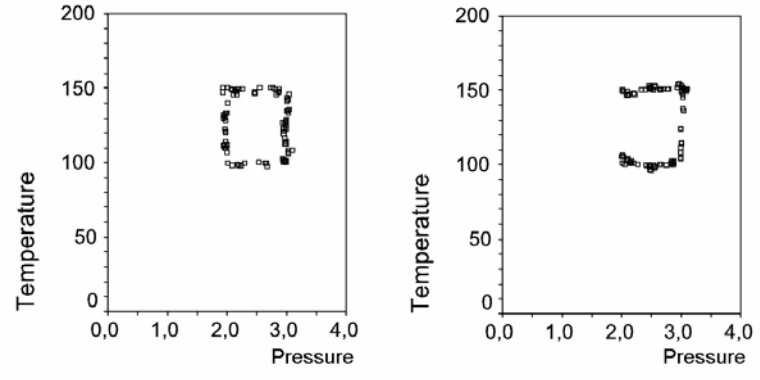

Fig. 3. Distribution of classification errors for the target concept *hardened(C)* ← *temp(C)* > *100* ∧ *temp(C)* < *150* ∧ *press(C)* > *2* ∧ *press(C)* < *3* without virtual attributes (left) and with the virtual attribute *V(C)* ← *press(C)* ≤ *2* (right)

Now let us analyze the effect of different amounts and different qualities of domain knowledge on the classification. In order to control the dependent variables like partialness and vagueness of the domain knowledge, we created a simple test domain. There were two continuous attributes *X* and *Y*, uniformly distributed over the interval [0, 100]. The target concept was *T(C)* ← *X(C)* > *30* ∧ *X(C)* < *70* ∧ *Y(C)* > *30* ∧ *Y(C)* < *70*. There were 100 randomly generated cases in the case-base and 200 test cases were used. Each experiment was repeated 1000 times with random cases in the case-base and random test cases. We used a square centered in the instance space as the target concept, because it is one of the few concepts for which the optimal weight setting for k-NN-classification can be calculated analytically. The optimal weight setting for the target concept is to use equal weights [13]. Thus, the accuracy of 1-NN with equal weights is the optimal accuracy that can be achieved without adding additional attributes.

3.1 Partialness of the Domain Theory

In this experiment, we operationalize the partialness of the domain knowledge as number of known target concept boundaries. The more boundaries are known, the less partial it is.

Adding virtual attributes that correctly specify a boundary of the target concept makes the misclassifications at those boundaries disappear (see fig. 3 (right)). Thus, by adding virtual attributes that describe a boundary correctly, the classification accuracy is increased (see fig. 4).

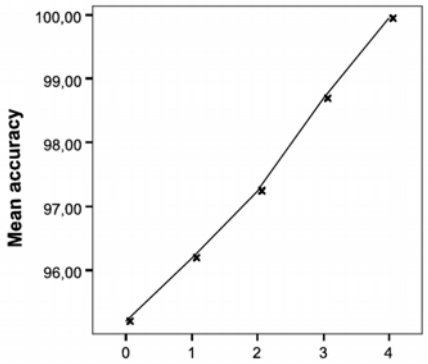

Fig. 4. Percentage of correctly classified cases with different numbers of target concept boundaries described by virtual attributes.

Obviously, even partial knowledge (e.g. adding only one virtual attribute) can improve classification accuracy. However, in this experiment we assumed that the virtual attributes were correct. In the next experiment we analyzed the influence of the correctness of virtual attributes.

3.2 Correctness

Vague knowledge can be informally described as knowing that an attribute should be more or less at a certain value. We operationalize correctness of a virtual attribute as its distance from the correct value. We created virtual attributes of the form $V(C) \leftarrow X(C) < c$, where c was varied from 0 to 100 at steps of 5. Remember that the correct X-value (which was used in the domain theory to generate the cases) was 30. The accuracy of classification using these virtual attributes is depicted in fig. 5.

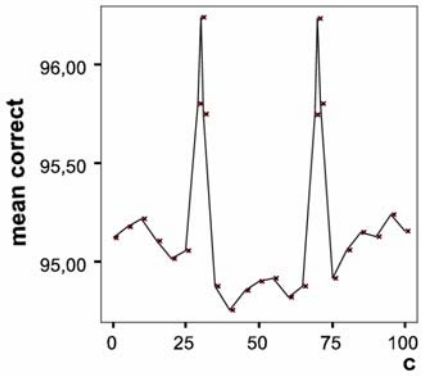

Fig. 5. Classification accuracy for similarity measures using a virtual attribute of the form $V(C) \leftarrow X(C) < c$. c is plotted along the horizontal axis

The results are somewhat disappointing. The accuracy drops rapidly if the virtual attribute is inaccurate. Fortunately, the accuracy with inaccurate virtual attributes is not significantly lower than using no virtual intermediates (the accuracy of a similarity measure with no virtual attribute is equivalent to setting $c = 0$ or $c = 100$). The second peak at $X = 70$ which is the other boundary on the X-attribute is due to the fact that similarity-based classification is directionless: only the position of the concept boundary has to be known, the side on which positive and negative instances are located is encoded in the cases.

These experiments with a simple domain suggest that partial knowledge is more useful than vague knowledge. Adding partial knowledge is likely to increase the classification accuracy, whereas vague knowledge is only useful if there is good evidence that the knowledge is correct which is at odds with the property of vagueness.

In the next section we will evaluate the influence of virtual attributes in several domains from the UCI Machine Learning Repository [5].

4. Experiments on Real-World Data Sets

In previous work [19] we showed that not all intermediate concepts will increase classification accuracy when used as virtual attributes. Hence, mechanisms to select or weight virtual attributes are necessary. We suggested that analyzing the structure of the domain theory could help to select intermediate attributes. Heuristics like abstractness, number of preconditions, and causality were proposed [19]. However, for some real-word data sets these heuristics performed badly. For example, some intermediate concepts had the exact same structure in the domain theory, but had very different impact on the classification accuracy. One concept decreased accuracy, the other increased it. Obviously, heuristics using the domain theory's logical structure are not sufficient to identify good virtual candidates. In this

section we investigate whether weighting virtual attributes is more appropriate than selecting them. In the experiments we apply several existing weighting approaches which will be described in section 4.3.

4.1 The Domains

The domain of the previous section allowed us to vary the vagueness and partialness of the domain theory. However, since the domain was handcrafted and simple, we ran additional experiments with real-world data in two domains from the UCI Machine Learning Repository. We used only data sets that provided imperfect domain theories. Note that some data sets in the repository come along with perfect domain models, as the instances were created by those models. However, we used only data sets whose domain theories are imperfect.

- Japanese Credit Screening (JCS): This domain comes with a domain theory that was created by interviewing domain experts. Accordingly, the theory is imperfect and classifies only 81% of the cases correctly.
- Promoter Gene Sequences (PGS): This domain theory reflects the knowledge of domain experts in the field of promoter genes. It is highly inaccurate and performs at chance level when used in isolation [20]. We included this domain to serve as a worst case scenario, since the domain knowledge is most inaccurate.

4.2 The Virtual Attributes

The domain theories of JCS and PGS have been created by domain experts for real world applications. Hence, they do not separate positive from negative instances in a perfect way. The accuracy of the JCS domain theory is 81%, the accuracy of the PGS domain theory is only 50%. The structure of both theories is depicted in fig. 6.

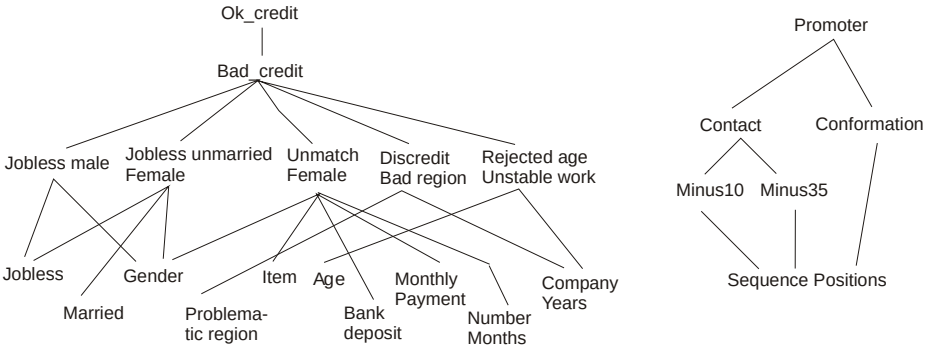

Fig. 6. The domain theory of the JCS domain (left) and of the PGS domain (right)

Most of the concepts process several observables. For example, *rejected_age_unstable_work* processes the observables *age* and *number_years*[1]:

```
rejected_age_unstable_work(S) :-
    age_test(S, N1),
    59 < N1,
    number_years_test(S, N2),
    N2 < 3.
```

[1] This attribute denotes the number of years that the applicant worked at the same company.

Although the concepts are highly imperfect, our experimental results (described in section 4.4) show that these concepts can improve classification accuracy when used as virtual attributes in similarity measures.

4.3 Weighting Methods

According to the classification of weighting methods proposed in [21], we selected four methods with performance bias, and six with preset bias (i.e. statistical and information-theoretic methods). The weights of all attributes (that is, original attributes and virtual attributes) are subject to weight-learning.

Performance bias: Weighting methods with a performance bias classify instances in a hill-climbing fashion. They update weights based on the outcome of the classification process. The performance bias is known to perform well if there are many irrelevant features [21]. Since the intermediate attributes of the domain theories can be assumed to be relevant, we expected performance bias methods to perform badly.

- EACH [18] increases the weight of matching features and decreases the weight of mismatching features by a hand-coded value.
- IB4 [2] is a parameter-free extension of EACH. It makes use of the concept distribution and is thus sensitive to skewed concept distributions. However, it assume that the values of irrelevant features are uniformly distributed.
- RELIEF [11] is a feature selection (rather than feature weighting) algorithm. It calculates weights based on the instance's most similar neighbors of each class and then filters attributes whose weights are below a hand-coded threshold. We used extensions for non-binary target classes and k-NN with $k > 1$ as proposed in [12].
- ISAC [4] increases weights of matching attributes and decreases weights of mismatching attributes by a value that is calculated from the ratio of correct and wrong classifications obtained at prior uses of the instance. The more often the instance was retrieved for correct classifications, the higher the update value.

Preset bias: The bias of the following weighting methods is based on probabilistic or information-theoretic concepts. They process each training instance exactly once.

- CCF [7] binarizes attributes and weights them according to the classes' probability given a feature.
- PCF [7] is an extension of CCF which takes the distribution of the feature's values over classes into account. It calculates different weights for different classes.
- MI [8] calculates the reduction of entropy in the class distribution by an attribute and uses it as attribute weight.
- CD [15] creates a correlation matrix of the discretized attributes and the classes. The weight of an attribute will be higher the better the prediction from attribute value to class.
- VD [15] extends CD in that it does not only consider the best prediction for a class, but considers the predictions of all attributes.
- CVD [15] combines CD and VD.

4.4 Results

For brevity, we will refer to the similarity measure which uses only observables as the *non-extended* measure. The similarity measure which uses virtual attributes will be called *extended*. For most of the weighting methods, the extended similarity measure performs better than the non-extended one. In table 1 we underline the accuracy of the extended similarity measure if it outperformed the non-extended similarity measure when using the same weighting method. In the PGS domain, for seven of the ten weighting methods the extended similarity measure performs better than the non-extended similarity measure. Even more so, in the JCS domain for eight of the ten weighting methods the accuracy of the extended similarity measure outperforms the non-extended one.

In its optimal setting, with an accuracy of 98.1% our approach performs also better than the results from the literature reported for the PGS domain. The accuracy of KBANN in [20] is 96.23%, which to

our knowledge was the highest accuracy reported so far. We found no classification accuracy results for JCS in the literature[2].

Table 1. Classification accuracies of the non-extended similarity measures (without virtual attributes) and the extended measures (with virtual attributes). The columns report the accuracies for the unweighted classification and for several weighting methods.

Domain	Unw.	EACH	RELIEF	IB4	ISAC	CCF	PCF	MI	CD	VD	CVD
JCS (w/o)	74.2	74.2	78.2	74.2	72.6	72.6	72.6	74.2	74.2	72.6	71.8
JCS (w/)	74.2	72.6	79.0	72.6	79.0	73.4	75.0	75.0	77.4	75.0	75.0
PGS (w/o)	86.8	89.6	96.2	88.7	50.0	85.9	87.7	68.9	88.7	77.4	83.0
PGS (w/)	85.9	93.4	96.2	90.6	96.2	91.5	86.8	98.1	88.7	97.2	87.7

Obviously, these improvements are not restricted to a certain class of weighting methods. Methods with performance bias (most notably ISAC), information-theoretic bias (i.e. MI), and with a statistical correlation bias (e.g. VD) benefit from processing virtual attributes.

Even in the PGS domain, the improvements are substantial. This is surprising, since the domain knowledge is the worst possible and classifies at chance level when used for rule-based classification. This is a promising result as it shows that adding intermediate concepts may even increase accuracy if the domain theory is very inaccurate. We hypothesize that this is due the fact that even vague rules-of-thumb provide some structure in the instance space which will be exploited by the similarity measure.

5. Conclusion and Future Work

The main contribution of this paper is to show that imperfect domain knowledge from domain theories can benefit similarity-based classification. This facilitates knowledge elicitation from domain experts as it removes the requirements of completeness and accurateness. Our experiments in a simple domain suggest that partial knowledge is more useful than vague knowledge. However, we showed in the domains from the Machine Learning Repository that even highly inaccurate (i.e. in our sense vague) domain knowledge can be exploited to significantly improve classification accuracy. Future work will include experiments in further domains and transforming intermediate attributes by feature generation [9].

Acknowledgements

Thanks to Collin Rogowski for a thorough proof-reading.

References

1. Aamodt, A.: Explanation-Driven Case-Based Reasoning. Proceedings of the European Workshop on Case-Based Reasoning. (1993) 274—288
2. Aha, D.: Tolerating Noisy, Irrelevant and Novel Attributes in Instance-based Learning Algorithms. In International Journal of Man-Machine Studies, 36 (1992) 267—287
3. Bergmann, R., Pews, G., Wilke, W.: Explanation-Based Similarity: A Unifying Approach for Integrating Domain Knowledge into Case-Based Reasoning for Diagnosis and Planning Tasks. Proceedings of the European Workshop on Case-Based Reasoning (1993) 182—196
4. Bonzano, A., Cunningham, P., Smyth, B.: Using Introspective Learning to Improve Retrieval in CBR: A Case Study in Air Traffic Control. In D. Leake, E. Plaza (Eds.), Proceedings of the second ICCBR conference, pages 291--302, Berlin, Springer, 1997

[2] The domain often referred to as 'credit screening' with 690 instances is actually the credit card application domain.

5. Blake, C. L. Merz, C. J.: UCI Repository of Machine Learning Databases http://www.ics.uci.edu/~mlearn/MLRepository.html. Irvine, CA: University of California, Department of Information and Computer Science, 1998
6. Cain, T., Pazzani, M. J., Silverstein, G.: Using Domain Knowledge to Influence Similarity Judgements. Proceedings of the Case-Based Reasoning Workshop. (1991) 191—198
7. Creecy, R. M., Masand, B. M., Smith, S. J., Waltz, D. L.: Trading MIPS and Memory for Knowledge Engineering: Classifying Census Returns on the Connection Machine. In Communications of the ACM, 35(8):48--63, 1992
8. Daelemans, W., van den Bosch, A.: Generalization Performance of Backpropagation Learning on a Syllabification Task. In Proceedings of the Third Twente Workshop on Language Technology: Connectionism and Natural Language Processing, pages 27—37, Enschede, The Netherlands: Unpublished, 1992
9. Fawcett, T. E., Utgoff, P. E.: Automatic Feature Generation for Problem Solving Systems. In D. Sleeman, P. Edwards (Eds.), Proceedings of the 9th International Conference on Machine Learning, 144--153, 1992
10. Griffiths, A. Bridge, D.: A Yardstick for the Evaluation of Case-based Classifiers. In Ian Watson (Ed.), Proceedings of Second UK Workshop on Case-Based Reasoning, 1996
11. Kira, K., Rendell, L.A.: A Practical Approach to Feature Selection. In Sleeman, D., Edwards, J. (eds.) Proceedings of International Conference on Machine Learning, pages 249-256, Morgan Kaufmann, 1992
12. Kononenko, I.: Estimating Attributes: Analysis and Extensions of RELIEF. In F. Bergadano, L. de Raedt (Eds.), Proceedings of the European Conference on Machine Learning, pages 171-182, Berlin: Springer, 1994
13. Ling, C. X, Wang, H.: Computing Optimal Attribute Weight Settings for Nearest Neighbor Algorithms. Artificial Intelligence Review 11 (1997) 255—272
14. Mooney, R. J., Ourston, D.: Constructive Induction in Theory Refinement. In L. Birnbaum, G. Collins (Eds.), Proceedings of the Eighth International Machine Learning Workshop, pages 178—182, San Mateo: Morgan Kaufmann, 1991
15. Núñez, H., Sànchez-Marrè, M., Cortés, U., Comas, J., Rodríguez-Roda, I., Poch, M.: Feature Weighting Techniques for Prediction Tasks in Environmental Processes. In Proceedings of the 3rd Workshop on Binding Environmental Sciences and Artificial Intelligence (BESAI 2002), 2002
16. Porter, B. W., Bareiss, R., Holte, R. C.: Concept Learning and Heuristic Classification in Weak-theory Domains. Artificial Intelligence, 45 (1990) 229—263
17. Richter, M.: Fallbasiertes Schliessen. Informatik Spektrum 26:3 (Juni 2003) 180—190
18. Salzberg, S.: A Nearest Hyperrectangle Learning Method. In Machine Learning, 6 (1991) 251—276
19. Steffens, T.: Similarity Measures Based on Imperfect Domain Theories. In S. Staab and E. Onainda (Eds.), Proceedings of STAIRS 2004, collocated with ECAI 2004, pages 193—198. Frontiers in Artificial Intelligence and Applications, IOS Press, 2004
20. Towell, G. G., Shavlik, J. W., Noordenier, M. O.: Refinement of Approximate Domain Theories by Knowledge-based Neural Network. In Proceedings of the Eighth National Conference on AI, pages 861—866, 1990
21. Wettschereck, D., Aha, D. W., Mohri, T.: A Review and Empirical Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms. In Artificial Intelligence Review, 11 (1997) 173—314

Prototypicality Measures and Adaptation Rules for Diagnosis of Dysmorphic Syndromes

Tina Waligora, Rainer Schmidt
Institute for Medical Informatics and Biometry, University of Rostock, Germany
[tina.waligora / rainer.schmidt] @medizin.uni-rostock.de

Abstract. Since diagnosis of dysmorphic syndromes is a domain with incomplete knowledge and where even experts have seen only few syndromes themselves during their lifetime, documentation of cases and the use of case-oriented techniques are popular. In dysmorphic systems, diagnosis usually is performed as a classification task, where a prototypicality measure is applied to determine the most probable syndrome. These measures differ from the usual Case-Based Reasoning similarity measures, because here cases and syndromes are not represented as attribute value pairs but as long lists of symptoms, and because query cases are not compared with cases but with prototypes. In contrast to these dysmorphic systems our approach additionally applies adaptation rules. These rules do not only consider single symptoms but combinations of them, which indicate high or low probabilities of specific syndromes.

Keywords: Dysmorphic syndromes, prototypicality measures, adaptation rules

1 Introduction

When a child is born with dysmorphic features or with multiple congenital malformations or if mental retardation is observed at a later stage, finding the correct diagnosis is extremely important. Knowledge of the nature and the etiology of the disease enables the paediatrician to predict the patient's future course. So, an initial goal for medical specialists is to diagnose a patient to a recognised syndrome. Genetic counselling and a course of treatments may then be established.

A dysmorphic syndrome describes a morphological disorder and it is characterised by a combination of various symptoms, which form a pattern of morphologic defects. An example is Down Syndrome which can be described in terms of characteristic clinical and radiographic manifestations such as mental retardation, sloping forehead, a flat nose, short broad hands and generally dwarfed physique [1].

The main problems of diagnosing dysmorphic syndromes are as follows [2]:

- existence of more than 200 syndromes,
- many cases remain undiagnosed with respect to known syndromes,
- usually many symptoms are used to describe a case (between 40 and 130),
- every dysmorphic syndrome is characterised by nearly as many symptoms.

Furthermore, knowledge about dysmorphic disorders is continuously modified, new cases are observed that cannot be diagnosed (a special journal publishes nothing but reports of observed interesting cases [3]), and sometimes even new syndromes are discovered. Usually, even experts of paediatric genetics only see a small count of dysmorphic syndromes during their lifetime.

We have developed a diagnostic system that uses a large case base. Starting point to build the case base was a large case collection of the paediatric genetics of the University of Munich, which consists of nearly 2,000 cases and 229 prototypes. A prototype (prototypical case) represents a dysmorphic syndrome by its typical symptoms. Most of the dysmorphic syndromes are already known and have been defined in literature. And nearly one third of our case base was determined by semiautomatic knowledge acquisition, where an expert selected cases that should belong to same syndrome and subsequently a prototype, characterised by the most frequent symptoms of his cases, was generated. To this database we have added cases from "clinical dysmorphology" [3] and syndromes from the London dysmorphic database [4], which contains only rare dysmorphic syndromes.

1.1 Diagnostic systems for dysmorphic syndromes

Systems to support diagnosis of dysmorphic syndromes have already been developed in the early eighties. The simple ones perform just information retrieval for rare syndromes, namely the London dysmorphic database [3], where syndromes a described by symptoms, and the Australian POSSUM, where syndromes are visualised [5]. Diagnosis by classification is done in a system developed by Wiener and Anneren [6]. They use more than 200 syndromes and apply Bayesian probability to determine the most probable syndromes. Another diagnostic system, which uses data from the London dysmorphic database was developed by Evans [7]. Though he claims to apply Case-Based Reasoning, in fact it is again a straightforward classification, this time performed by Tversky's measure of dissimilarity [8]. The most interesting aspect of his approach is the use of weights according to their intensity of expressing retardation or malformation independent of the specific syndrome. However, Evans admits that even features, that are usually unimportant or occur in very many syndromes sometimes play a vital role for discrimination between specific syndromes.

In our system the user can choose between two measures of dissimilarities between concepts, namely of Tversky [8] and the other one of Rosch and Mervis [9]. However, the novelty of our approach is that we do not only perform classification but subsequently apply adaptation rules. These rules do not only consider single symptoms but specific combinations of them, which indicate high or low probabilities of specific syndromes.

1.2 Case-Based Reasoning systems for diagnosis

Since the idea of Case-Based Reasoning (CBR) is to use solutions from previously solved problems (or cases) [10], CBR seems to be appropriate for diagnosis of dysmorphic syndromes. CBR consists of two main tasks [11], namely retrieval, that means searching for similar cases, and adaptation, that means adapting solutions of similar cases to the query case. For retrieval usually explicit similarity measure or, especially for large case bases, faster retrieval algorithms like Nearest Neighbour Matching [12] are applied. For adaptation only few general techniques exist [13], usually domain specific adaptation rules have to be acquired.

Most medical CBR systems deal with therapeutic support, some deal with interpretation of data (e.g. images or time-oriented parameters of organ functions), while only few systems provide diagnostic support. One reason is that doctors are much more interested in therapeutic than diagnostic support. Another reason is that it is commonly assumed that doctors perform a sort of differential diagnosis and rules seem to be appropriate to compute this process.

In [14] we have presented some older, diagnostic CBR systems, e.g. a system called CASEY [15] that attempts to diagnose heart failures by applying operators, which have been generalised from cases, and MEDIC [16] that uses schemata in the domain of pulmonology. A more recent system [17] has been developed for sonography. It considers multiple faults. Unfortunately, the system does not contain a representative set of cases to cover most fault combinations contains only slightly more cases than possible fault combinations. So it is difficult to find similar cases and the evaluation results remain poor.

1.3 Prototypicality measures

In CBR usually cases are represented as attribute-value pairs. In medicine, especially in diagnostic applications, this is not always the case, instead often a list of symptoms describes a patient's disease. Sometimes these lists can be very long, and often their lengths are not fixed but vary with the patient. For dysmorphic syndromes usually between 40 and 130 symptoms are used to characterise a patient.

Furthermore, for dysmorphic syndromes it is unreasonable to search for single similar patients (and of course none of the systems mentioned above does so) but for more general prototypes that contain the typical features of a syndrome. Prototypes are a generalisation from single cases. They fill the knowledge gap between the specificity of single cases and abstract knowledge in form of cases. Though the use of prototypes had been early introduced in the CBR community [18, 19], still remain under utilised. However, since doctors reason with typical cases anyway, in medical CBR systems prototypes are a rather common knowledge form (e.g. for antibiotics therapy advice in ICONS [20], for diabetes [21], and for eating disorders [22]).

So, to determine the most similar prototype for a given query patient instead of a similarity measure a prototypicality measure is required. One speciality is that for prototypes the list of symptoms is usually much shorter than for single cases.

The result should not be just the one and only most similar prototype, but a list of them – sorted according to their similarity. So, the usual CBR methods like indexing or nearest neighbour search are inappropriate. Instead, rather old measures for dissimilarities between concepts [8, 9] are applied and explained in the next section.

2 Diagnosis of dysmorphic syndromes

Our system performs four steps (fig.1). At first the user has to select symptoms that describe the new patient. This selection is strenuous and time consuming, because we consider more than 800 symptoms. However, diagnosis of dysmorphic syndromes is not a task requiring great speed, but it usually requires thorough reasoning and is followed by a long-term therapy.

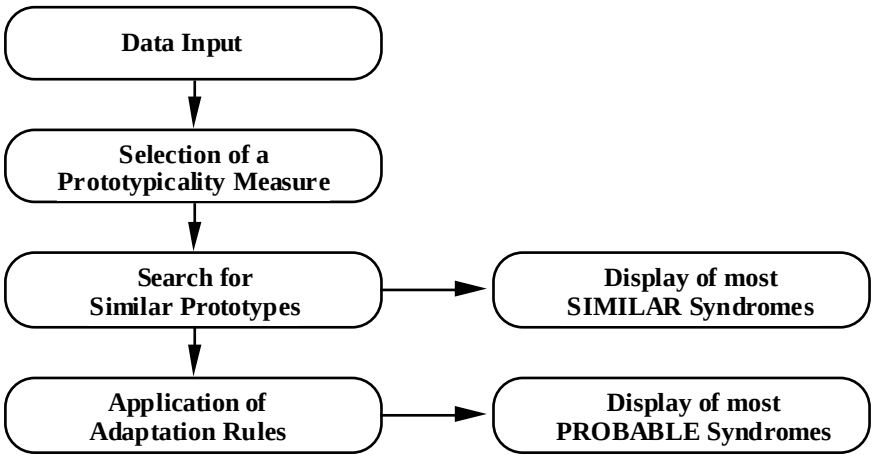

Fig.1. Steps to diagnose dysmorphic syndromes

Since our system is still in the evaluation phase, the user can select a prototypicality measure. In routine use, this step shall be dropped and instead the measure with best evaluation results shall be used automatically. At present there are three choices.

As humans look upon cases as more typical for a query case with increasing numbers of common features [9], distances between prototypes and cases usually mainly consider the shared features.

The first, rather simple measure just counts the number of matching symptoms of the query patient (X) and a prototype (Y) and normalises the result by dividing it by the number of symptoms characterising the syndrome. This normalisation is done, because the lengths of the lists of symptoms of the various prototypes vary to a great degree. It is performed by the two other measures too.

$$D(X,Y) = \frac{f(X + Y)}{f(Y)}$$

The second measure was developed by Tversky [8]. It is a measure of dissimilarity for concepts. In contrast to the first measure, additionally two numbers are subtracted from the number of matching symptoms. Firstly, the number of symptoms that are observed for the patient but are not used to characterise the prototype (X-Y), and secondly the number of symptoms used for the prototype but are not observed for the patient (Y-X) is subtracted.

$$D_{Tversky}(X,Y) = \frac{f(X + Y) - f(X - Y) - f(Y - X)}{f(Y)}$$

The third prototypicality measure was proposed by Rosch and Mervis [9]. It differs from Tversky's measure only in one point: the factor X-Y is not considered:

$$D_{Rosch,\ Mervis}(X,Y) = \frac{f(X+Y) - f(Y-X)}{f(Y)}$$

In the third step to diagnosis dysmorphic syndromes, the chosen measure is sequentially applied on all prototypes (syndromes). Since the syndrome with maximal similarity is not always the right diagnosis, the 20 syndromes with best similarities are presented ranked according similarity (fig. 2).

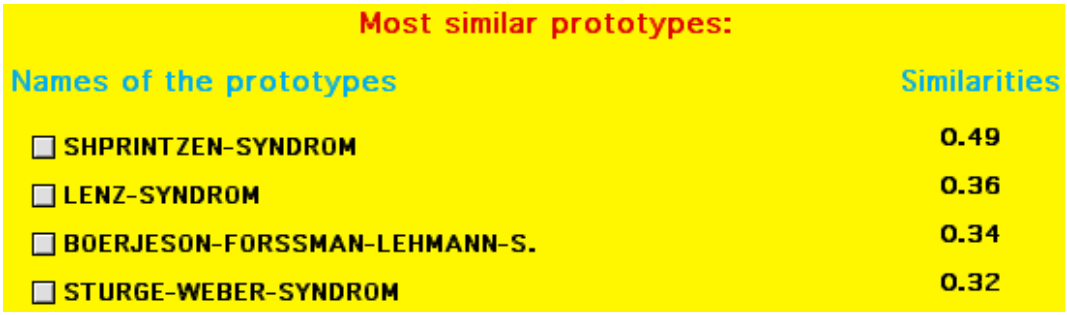

Fig. 2. Top part of the listed prototypes

2.1 Application of adaptation rules

In the fourth and final step, the user can optionally choose to apply adaptation rules on the syndromes. These rules state that specific combinations of symptoms favour or disfavour specific dysmorphic syndromes. Unfortunately, the acquisition of these adaptation rules is very difficult, because they cannot be found in textbooks but have to be defined by experts of paediatric genetics. So far, we have got only 10 of them and it is not possible that a syndrome can be favoured by one adaptation rule and disfavoured by another one at the same time. When we, hopefully, acquire more rules such a situation should in principle be possible but would indicate some sort of inconsistency of the rule set.

The question is how shall adaptation rules alter the results. Our first idea was that the similarity values should be changed. A syndrome that is favoured by an adaptation rule might get a higher similarity. But we had no idea how much an adaptation rule shall increase a similarity. Of course no medical expert can help here and a general value for favoured or disfavoured syndromes by adaptation rules would be arbitrary.

So, instead the result after applying adaptation rules is a menu that contains up to three lists (fig. 3).

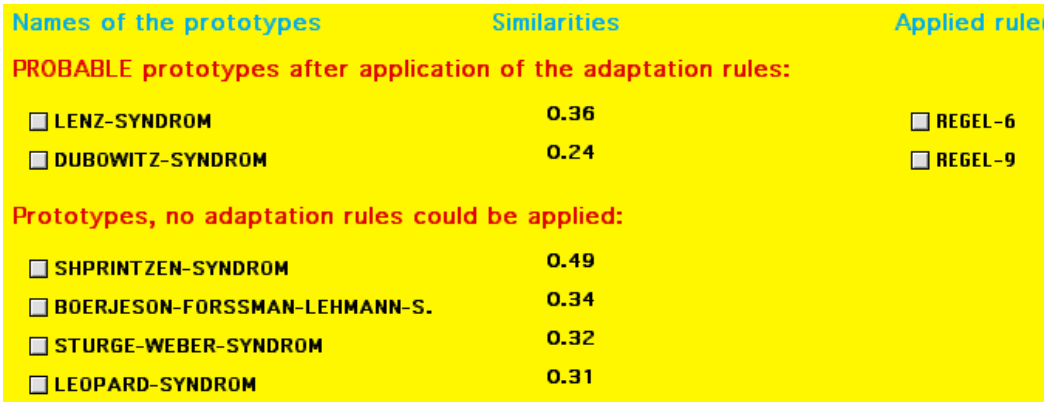

Fig.3. Top part of the listed prototypes after application of adaptation rules

On top the favoured syndromes are depicted, then those neither favoured nor disfavoured, and at the bottom the disfavoured ones. Additionally, the user can get information about the specific rules that have been applied on a particular syndrome (e.g. fig. 4).

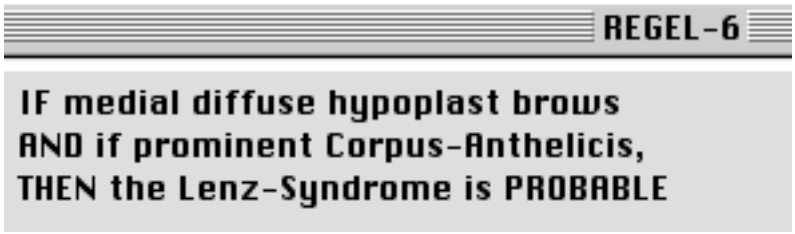

Fig.4. Presented information about the applied rule

In the example presented by the figures 2, 3, and 4, the correct diagnosis is Lenz-syndrome. The computation of the prototypicality measure of Rosch and Mervis Lenz-syndrome was the most similar but one syndrome (here Tversky's measure provides a similar result, only the differences between the similarities are smaller). After application of adaptation rules, the ranking is not obvious. Two syndromes have been favoured, the more similar one is the right one. However, Dubowitz-syndrome is favoured too (by a completely different rule), because a specific combination of symptoms makes it probable, while other observed symptoms indicate lower similarity.

3 Results

Cases are difficult to diagnose when patients suffer from a very rare dymorphic syndrome for which neither detailed information can be found in literature nor many cases are stored in our case base. This makes evaluation difficult. If test cases are randomly chosen, frequently observed syndromes will be frequently selected and the results will probably be fine, because these syndromes are well-known. However, the main idea of the system is to support diagnosis of rare syndromes. So, we have chosen our test cases randomly but under the condition that every syndrome can be chosen only once.

For 100 cases we have compared the results obtained by both prototypicality measures (table 1).

Table 1. Comparison of prototypicality measures

Right Syndrome	Rosch and Mervis	Tversky
on Top	29	40
among top 3	57	57
among top 10	76	69

Obviously, the measure of Tversky provides better results, especially when the right syndrome should be on top of the list of probable syndromes. When it should be only among the first three of this list, both measures provide equal results.

Since the acquisition of adaptation rules is very difficult and time consuming the number of acquired rules is rather limited, namely 10 rules. Furthermore, again holds: the better a syndrome is known, the easier adaptation rules can be generated. So, the improvement mainly depends on the question how many syndromes involved by adaptation rules are among the test set. In our experiment this was the case only with 5 syndromes. Since some had been already diagnosed correctly without adaptation, there was just a small improvement (table 2).

Table 2. Results after the application of adaptation rules

Right Syndrome	Rosch and Mervis	Tversky
on Top	32	42
among top 3	59	59
among top 10	77	71

4 Summary

In this paper, we have presented our system to support diagnosis of dysmorphic syndromes. Our system, which is still under development, is not only based on single cases but mainly on meaningful prototypes. The diagnosis consists of two steps: first similar prototypes are calculated, and subsequently adaptation rules are applied.

For calculating similar prototypes, we have presented two sequential prototypicality measures. Our first test results do not show any significant difference between them.

For adaptation, our system still lacks sufficient number of adaptation rules. These rules are very difficult to acquire warranting a significant effort. The more a dysmorphic syndrome is studied, the easier it is for experts to express an adaptation rule. This is especially unfortunate, because the main idea of our diagnostic support program is to provide support for rare dysmorphic syndromes.

Acknowledgement

The project is funded by the German Research Society (DFG).

References

1. Taybi, H., Lachman, R.S.: Radiology of Syndromes, Metabolic Disorders, and Skeletal Dysplasia. Year Book Medical Publishers, Chicago (1990)
2. Gierl, L., Stengel-Rutkowski, S.: Integrating Consultation and Semi-automatic Knowledge Acquisition in a Prototype-based Architecture: Experiences with Dysmorphic Syndromes. *Artificial Intelligence in Medicine 6* (1994) 29-49
3. Clinical Dysmorphology. www.clindysmorphol.com
4. Winter R.M., Baraitser M., Douglas J.M.: A computerised data base for the diagnosis of rare dysmorphic syndromes. Journal of medical genetics 21 (2) (1984) 121-123
5. Stromme P.: The diagnosis of syndromes by use of a dysmorphology database. Acta Paeditr Scand 80 (1) (1991) 106-109
6. Weiner F., Anneren G.: PC-based system for classifying dysmorphic syndromes in children. Computer Methods and Programs in Biomedicine 28 (1989) 111-117
7. Evans C.D.: A case-based assistant for diagnosis and analysis of dysmorphic syndromes. International Journal of Medical Informatics 20 (1995) 121-131
8. Tversky, A.: Features of Similarity. *Psychological Review 84* (4) (1977) 327-352
9. Rosch E., Mervis CB: Family Resemblance: Studies in the Internal Structures of Categories. *Cognitive Psychology 7* (1975) 573-605
10. Kolodner J: Case-Based Reasoning. Morgan Kaufmann, San Mateo (1993)
11. Aamodt A, Plaza E: Case-Based Reasoning: Foundation issues, methodological variation, and system approaches. *AICOM* 7 (1994) 39-59
12. Broder A: Strategies for efficient incremental nearest neighbor search. Pattern Recognition 23 (1990) 171-178
13. Wilke W, Smyth B, Cunningham P: Using configuration techniques for adaptation. In: Lenz M, Bartsch-Spörl B, Burkhard H-D, Wess S (eds.): Case-Based Reasoning technology, from foundations to applications. Springer Berlin (1998) 139-168
14. Gierl L, Bull M, Schmidt R: CBR in medicine. In: Lenz M et al.: Case-Based Reasoning Technology, Springer-Verlag, Berlin Heidelberg New York (1998) 273-297
15. Koton P: Reasoning about evidence in causal explanation. In: Kolodner J (ed.): First Workshop on Case-Based Reasoning, Morgan Kaufmann, San Mateo (1988) 260-270
16. Turner R: Organizing and using schematic knowledge for medical diiagnosis. In: Kolodner J (ed.): First Workshop on Case-Based Reasoning, Morgan Kaufmann, San Mateo (1988) 435-446
17. Baumeister J, Atzmüller M, Puppe F: Inductive learning for case-based diagnosis with multiple faults. In: Craw S. Preece A (eds.): Advances in case-based reasoning, Springer-Verlag, Berlin Heidelberg New York (2002) 28-42
18. Schank RC: Dynamic Memory: a theory of learning in computer and people. Cambridge University Press, New York (1982)

19. Bareiss R: Exemplar-based knowledge acquisition. Academic Press, San Diego (1989)

20. Schmidt R, Gierl L: Case-based Reasoning for antibiotics therapy advice: an investigation of retrieval algorithms and prototypes. *Artificial Intelligence in Medicine* 23 (2001) 171-186

21. Bellazzi R, Montani S, Portinale: Retrieval in a prototype-based case library: a case study in diabetes therapy revision. In Smyth B, Cunningham P (eds.): Proceedings of European Workshop on Case-Based Reasoning, Springer, Berlin (1998) 64-75

22. Bichindaritz I: From cases to classes: focusing on abstraction in case-based reasoning. In: Burkhard H-D, Lenz M: (eds.): Proceedings of German Workshop on Case-Based Reasoning, University Press, Berlin (1996) 62-69

The Group has links with many outside bodies. It is a specialist group of the British Computer Society and a member of ECCAI, the European Co-ordinating Committee for Artificial Intelligence.

Since its inception the group has enjoyed a good working relationship with government departments involved in the AI field (beginning with the Alvey Programme in the 1980s). A succession of Department of Trade and Industry (DTI) representatives, have been co-opted as committee members. The Group acted as co-organiser of the annual DTI Manufacturing Intelligence awards and has included sessions presenting the results of the DTI Intelligent Systems Integration Programme (ISIP) in its annual conferences.

The group also has a good relationship with the Institution of Electrical Engineers (IEE), with which it has co-sponsored colloquia over many years, and with NCAF, the Natural Computing Applications Forum. We also host the annual UK-CBR (Case-Based Reasoning) workshops at our annual conferences. This year, the group will be co-hosting IJCAI-05 in Edinburgh.

Benefits of Membership

- Preferential rates for the Group's prestigious international conference on Artificial Intelligence, which has run annually since 1981.
- Discounted rates at other SGAI-sponsored events.
- Discounted rates for ECCAI organised events. The Group has been a member of the European Co-ordinating Committee for Artificial Intelligence since 1992.
- Discounts on international journals, and occasional special offers on books.
- Advance information on the SGAI Evening Lectures, which are held on a regular basis in central London.
- Free subscription to the *Expert Update* journal, containing reviews, technical articles, conference reports, comment from industry gurus and product news.
- The SGAI website at www.bcs-sgai.org and the AI-SGES list server to facilitate communication on all aspects of AI.
- A substantial proportion of the Group's membership is from industry. Providing a valuable forum where both academic and industrial AI communities can meet.

How to Join BCS-SGAI?

To join BCS-SGAI you do not need to be a member of the BCS. For further information please visit our website at www.bcs-sgai.org.

Subscription Rates

Annual subscription rates for Individual and Corporate Members are:

INDIVIDUALS

Standard Members (UK addresses)	£31.00
Standard Members (Overseas addresses)	£41.00
BCS Members (UK addresses)	£22.00
BCS Members (Overseas addresses)	£31.00
Students (UK addresses)	£11.00
Students (Overseas addresses)	£21.00
Proof of student status is required	
Retired (UK addresses)	£11.00
Retired (Overseas addresses)	£21.00

CORPORATE

UK addresses	£150.00
Overseas addresses	£190.00

Add £5 to all these rates if not paying by standing order.

26th BCS-SGAI International Conference on
Innovative Techniques and Applications of Artificial Intelligence
11-13th December 2006, Cambridge, UK

http://www.bcs-sgai.org/ai2006

AI-2006

AI-2006 is the twenty-sixth Annual International Conference of the British Computer Society's Specialist Group on Artificial Intelligence (SGAI). The scope of the conference comprises the whole range of AI technologies and application areas. Its principal aims are to review recent technical advances in AI technologies and to show how these advances have been applied to solve business problems.

CONFERENCE VENUE

Peterhouse College, founded in 1284, and its hall, built between 1286 and 1290, was the first collegiate building in Cambridge. Peterhouse Gardens achieved fame between 1830 and 1930 as the smallest deer park in England. Located in the centre of Cambridge, the college combines historic buildings with modern conference facilities, within easy reach of the shopping and entertainment of Cambridge.

Deadline for Paper/Poster Submission: Monday 5th June 2006

CONFERENCE OUTLINE

TECHNICAL STREAM – Areas of interest include (but are not restricted to): knowledge based systems; knowledge engineering; semantic web; constraint satisfaction; intelligent agents; machine learning; model based reasoning; natural language understanding; speech-enabled systems; case based reasoning; neural networks; genetic algorithms; data mining and knowledge discovery in databases; robotics and pervasive computing.

APPLICATION STREAM – Papers in recent years have covered all application domains, including commerce, manufacturing and defence. Papers are selected to highlight critical areas of success (and failure) and to present the benefits and lessons of value to other developers.

TUTORIALS & WORKSHOPS – This year's workshop day on 11th Dec. will include a range of workshops and tutorials.

UKCBR11 – The 11th UK CBR Workshop, will be collocated with AI-2006. It concerns all aspects of case-based reasoning and practical applications.

POSTER SESSION – There is also a poster session for presenting work in progress.

PRIZES – There are sponsored prizes (a trophy plus £500) for the best paper submitted in each stream. There is also a trophy and a cash prize for the best-presented poster.

THE BCS MACHINE INTELLIGENCE PRIZE – Trophy plus £1,000 cash prize for the best live demonstration of progress towards machine intelligence. Deadline October 1st, 2006.

REGISTRATION
Early registration is advised.

Accommodation is available on a first-come, first-served basis.

CONTACT
Mark Firman, Conference Administrator, AI-2006, sgai-conference@bcs.org.uk