A Knowledge-Light Mechanism for Explanation in Case-Based Reasoning

Dónal Doyle

A thesis submitted to the University of Dublin, Trinity College

in fulfillment of the requirements for the degree of

Doctor of Philosophy

October 2005

Declaration

Dónal Doyle

Dated: December 19, 2005

Acknowledgements

Firstly, I would like to express my gratitude to my supervisor Prof. Pádraig Cunningham for his guidance and friendship over the past three years. His constant assistance and help is greatly appreciated and this work could never have been done without him.

Secondly, I would like to thank all the members of the MLG; Nadia Bolshakova, Kenneth Bryan, Peter Byrne, Michael Carney, Patrick Clerkin, Lorcan Coyle, Michael Davy, Sarah Jane Delany, Chris Fairclough, Derek Greene, Marco Grimaldi, Conor Hayes, Deirdre Hogan, John Loughrey, Brian Mac Namee, Neil McDonnell, Conor Nugent, François-Xavier Pasquier, Matthew Sammon, Alexey Tsymbal and Anton Zamolotskikh. These people have become more than work colleagues over the last three years and I am proud to consider them friends. It has been a pleasure to work and socialise with all of them. In particular I would like to thank Lorcan, Marco, Michael Carney, Michael Davy and Sarah Jane for their opinions and assistance towards this project. Conor and John, thanks for your immense help and support (We made it!).

I would like to thank everyone working in the Emergency Department of Kern Medical Center, Bakersfield, California. They provided me with invaluable data and feedback. Most of all I would like to thank Dr. Paul Walsh for his endless hours of work organising the data and providing me with tremendous assistance.

I owe a debt of gratitude to my family (who thought I'd never finish school!) for their incredible support throughout my academic career. Thanks Mam, Dad, John, Mary and Adrian. Finally I'd like to thank all my friends and in particular to Carol for her love and patience.

Dónal Doyle
University of Dublin, Trinity College
October 2005

Abstract

Decision support systems are currently achieving higher classification accuracies by using more complex reasoning mechanisms. Examples of such mechanisms include support vector machines and neural networks. However in spite of these increases in accuracy many decision support systems are not accepted by users. In domains where there is a high cost associated with incorrect classifications, such as medical domains, users are not always willing to accept a decision support system's classification without proper justification.

In every walk of life, from the home to the workplace, people use explanations all the time to justify their opinions. Explanations can have many different forms depending on the context in which they are used. Over the last few decades there has been a vast amount of research by philosophers into the importance and the requirements of suitable explanations.

In spite of the importance of suitable explanations to justify an opinion, many decision support systems fall sort of this requirement. Part of the reason for this is that in many types of decision support systems it is often extremely difficult, if not impossible, to produce explanations. This is particularly the case for black box systems such as support vector machines. However in other systems such as rule-based systems, where the explanation can be in the form of a rule, the explanations can often be complicated and result in confusion for users.

Alternatively Case-based Reasoning (CBR) Systems lend themselves naturally to producing explanations. As the reasoning in CBR systems is performed on the most similar past case(s) to a current problem, these similar cases can be used as an explanation for a classification. As these similar cases are real past problems, they are generally easily understood by users.

It is our believe however, that in CBR systems, that there are more suitable cases to use as an explanation than simply using the most similar cases. It is our belief that these more suitable cases lie between the problem case and the perceived decision boundary. This results in the cases forming an a fortiori argument. We describe a framework that we have developed for selecting such cases.

We also believe that it is often not enough, regardless of the suitability, to just use a case as an explanation. In our framework we included a mechanism for generating explanatory text that can express why the case is suitable, or in some situations aspects of the case that may not be suitable. This explanatory text can further assist users to decide if they agree with the opinion of

CBR system.

Based on the developed framework we implemented a decision support system for use in the domain of Bronchiolitis, a viral infection that effects young children. This system was used and evaluated in the Kern Medical Center, Bakersfield, California during their Bronchiolitis season.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Explanations are an important aspect of every day life. They are used in many different places for many different reasons. Lawyers use them in court in an attempt to strengthen their arguments. Teachers use them to help their students understand complicated concepts. Children use them to argue why they should be allowed to do something, while parents use them in return to argue why they are not allowed.

The primary goal of expert systems is to make expertise available to decision makers. However, many of these systems are not used by end users because they don't justify their decisions. For example, in the medical domain it is extremely difficult for physicians to accept recommendations without any justification. This shortcoming becomes more of a problem due to the fact that it is often difficult to offer 100% accuracy with expert systems.

Decision support systems attempt to address this problem by *aiding* the process of decision making (Finlay 1994). The philosophy behind this is that the system now aids the user in reaching a decision rather than by presenting the solution as an expert might to a novice. One technique for doing this is for decision support systems to generate explanations to support their recommendations. These explanations are generally either a trace of the reasoning process or evidence that supports the recommendation. In this thesis we describe a framework that we have developed for producing more convincing explanations for case-based reasoning systems. We perform evaluations of our explanations. If we can produce convincing explanations then our strategy is useful.

1.1 Case-Based Reasoning

Case-based Reasoning (CBR) is an artificial intelligence methodology for solving problems by using or adapting solutions to old problems (Riesbeck and Schank 1989). Case-based Reasoning systems are made up of a collection of cases called the case-base. These cases are previous experiences, which are generally described as problem-solution pairs. The problem represents the description of a previous situation, and the solution represents the outcome or the action taken on that occasion.

CBR systems have the defining characteristic of deferring their processing until run-time. This is commonly referred to as being a lazy learner. Although leaving processing until run-time may introduce certain penalties, albeit with faster computation power these penalties are diminishing, it has the advantage of being as up to date as possible (Aha 1997). This is of particular benefit in domains where data is scarce as new data can be easily added to the case-base at any time.

1.2 Explanations in Decision Support Systems

Explanations are an important part of our human lives. Every day, people use them to argue their opinions. In a similar manner explanations are important for decision support systems to argue their classifications. It is possible that part of the reason for the lack of usage of deployed Decision Support Systems is that they don't provide adequate explanations. Richards (2003) and Brzillon et al. (1996) argue that verification is not enough, but Decision Support Systems need to justify and be accountable for their classifications.

CBR has a major advantage over other types of decision support systems when it comes to producing explanations. CBR systems can present the most suitable case(s) from the case-base, often the most similar case, as an explanation. As these cases are real previous examples they are understandable by users. Unlike CBR, other systems perform their reasoning on models extracted from training data. In Rule Based Systems the underlying rules can be displayed as an explanation, but these rules can become complicated for users to understand. However, in the case of support vector machines and neural networks it is near impossible to generate explanations based on the reasoning process.

1.3 Contributions of this Thesis

This thesis describes the implementation of a decision support system for use in medical domains. During its development and operation several issues were encountered. The most interesting ones, which constitute the main contributions of this thesis are:

- The implementation of a prototype decision support system for use in a medical domain
- An assessment of the usefulness of case-based explanations
- The implementation of an explanation utility framework
- The implementation of techniques to assess classification confidence
- The development of a representational format for CBR data called CBML

1.3.1 Assessment of Case-Based Explanations

As already mentioned, a main advantage of CBR, compared to other types of Decision Support Systems, is its ability to easily produce understandable explanations. Although this advantage has never been empirically validated it is often mentioned by members of the CBR community. In this thesis our initial work was to perform an evaluation to show that case-based explanations are in fact more convincing than rule-based explanations.

1.3.2 Explanation Utility Framework

Although we showed that case-based explanations are convincing, we believe that their is still room for improvement. Many CBR systems produce explanations by simply displaying the most similar case to a given problem case. We believe that more convincing cases can be selected from the case-base. We developed the explanation utility framework to select cases from the case-base that we believe provide more convincing explanations. Also using this framework, we developed techniques for highlighting aspects of the retrieved cases that support and oppose the systems classifications.

1.3.3 Assessing Confidence

The main goal for decision support systems in providing explanations, is to increase users confidence in the system. For this reason we feel that decision support systems should also convey its own confidence in a classification as part of an explanation. To achieve this we implemented a number of confidence measures to allow a CBR system to assess a level of confidence in its classification.

1.3.4 Bronchiolitis Decision Support System

To test our explanation utility framework and confidence assessment measures we developed a CBR based system for use in the medical domain of Bronchiolitis. Bronchiolitis is a viral infection that affects the lungs of young children. This system produces recommendations and explanations on whether a child presented to an Emergency Department should be admitted or discharged. While this system was being used we performed an evaluation on the quality of the explanations produced.

1.3.5 CBR Representation

In Chapter 6 we describe the Case Based Mark-up Language (CBML), an XML-based language we have developed to facilitate the representation of case bases for use by CBR systems. We detail its benefits in terms of extensibility, ease of reuse and interoperability. The language allows us to make the formal definition of the structure of our cases and similarity measures completely independent of the application code. In this way we allow the structure and definition of our cases to be easily described and modified.

1.4 Publications Related to this Thesis

- Dónal Doyle, Pádraig Cunningham and Paul Walsh.: 2005, An Evaluation of the Usefulness of Explanation in a CBR System for Decision Support in Bronchiolitis Treatment, *Workshop on CBR in the Health Sciences, The Sixth International Conference on Case-Based Reasoning (ICCBR-05)*, pp32-41.
- Sarah Jane Delany, Pádraig Cunningham, Dónal Doyle and Anton Zamolotskikh.: 2005, Generating Estimates of Classification Confidence for a Case-Based Spam Filter, *6th International Conference on Case-Based Reasoning,* eds. H. Muñoz-Avila and F. Ricci, pp177-190, Springer LNAI 3620.
- Conor Nugent and Pádraig Cunningham and Dónal Doyle.: 2005, The Best Way to Instil Confidence is by Being Right; An Evaluation of the Effectiveness of Case-Based Explanations in providing User Confidence, *6th International Conference on Case-Based Reasoning,* eds. H. Muñoz-Avila and F. Ricci, pp368-381, Springer LNAI 3620.
- Conor Nugent, Dónal Doyle and Pádraig Cunningham.: Gaining Insight through Case-Based Explanation. To appear in *CBR in Knowledge Discovery and Data Mining*, eds. S. Pal, D. Aha and K. Moy Gupta, Wiley.
- Dónal Doyle, John Loughrey, Conor Nugent, Lorcan Coyle, Pádraig Cunningham.: 2005, Fionn: A Framework for Developing CBR Systems, in P. Funk and P. Calero (eds), *Expert Update* 8(1), 11-14.
- Dónal Doyle, Pádraig Cunningham, Derek Bridge and Yusof Rahman.: 2004, Explanation Oriented Retrieval, in P. Funk and P. Calero (eds), *Advances in Case-Based Reasoning (Procs. of the Seventh European Conference on Case-Based Reasoning)*, Springer, pp. 157-168.
- Lorcan Coyle, Dónal Doyle and Pádraig Cunningham.: 2004, Representing similarity for CBR in XML, in P. Funk and P. Calero (eds), *Advances in Case-Based Reasoning (Procs. of the Seventh European Conference on Case-Based Reasoning)*, Springer, pp. 119-127.
- Pádraig Cunningham, Dónal Doyle and John Loughrey.: 2003, An Evaluation of the Usefulness of Case-Based Explanation, in K.D. Ashley and D.G. Bridge (eds), *Case-Based Reasoning Research and Development, 5th International Conference on Case-Based Reasoning*, Vol. 2689 of *Lecture Notes in Computer Science*, Springer, ICCBR 2005, Trondheim Norway, pp. 122-130.

- Dónal Doyle, Alexey Tsymbal and Pádraig Cunningham.: 2003, A Review of Explanation and Explanation in Case-Based Reasoning, *Technical Report TCD-CS-2003-41*, Department of Computer Science, Trinity College, Dublin.

1.5 Summary and Structure of this Thesis

In brief: Chapter 2 describes some case-based reasoning techniques that can be applied to our explanation process. Chapter 3 describes the use of explanations and confidence assessment in decision support systems. Chapter 4 describes our explanation utiliy framework and Chapter 5 describes our work in developing confidence measures. Chapter 6 describes the implementation of our work on a case representation format and a decision support system in a medical domain. Chapter 7 contains evaluations of our explanation techniques and the thesis is concluded with Chapter 8.

In more detail: Chapter 2 gives a description of the case-based reasoning methodology. It focuses on aspects of the methodology that are most significant for our work on producing case-based explanations.

Chapter 3 discusses explanation generation and confidence assessment in decision support systems. It also describes some systems that have been developed in this area.

Chapter 4 describes our explanation utility framework. This framework is used to select cases that can be used as *a fortiori* arguments. The framework can also be used to produce explanatory text that can be included with a recommendation. This chapter also includes descriptions of three medical problem domains, e-Clinic, Breathalyser and Bronchiolitis, that are used throughout this thesis.

Chapter 5 describes some measures that we have used for assessing classification confidence. This chapter also includes an evaluation of these measures.

Chapter 6 contains an in-depth description of CBML. CBML is used in the development of the machine learning toolkit called Fionn. Fionn was used to develop a decision support system for the Bronchiolitis domain which is also described in this chapter.

Chapter 7 discusses the advantages and shortcomings of our explanation strategies. It contains detailed results, including comparisons against other explanation techniques and discusses the implications of these results.

Finally Chapter 8 concludes the thesis and describes some further work that could be investigated.

Chapter 2

Case-Based Reasoning

From the start, AI research (Turing 1950, Minsky 1961) has often attempted to replicate problem-solving techniques in computers in a similar manner to how humans perform problem-solving. Neural Networks are mathematical models that attempt to mimic the human brains computational, adaptation and learning capabilities. Rule-based Systems use the chaining of generalised rules to imitate the human reasoning process. Fuzzy Logic Systems are models attempting to replicate the brain's ability to represent and reason with imprecise knowledge and data. However, all these approaches rely on knowledge of a domain, resulting in problems being solved from first principles.

It was research by Roger Schank and his students into cognitive science (Schank and Abelson 1977, Schank 1982, Kolodner 1983, Hammond 1988) that recognised that people commonly solve problems by remembering how they solved similar problems in the past. They discovered that solutions to real life problems are rarely truly original solutions, but merely adaptations to previously known solutions to similar problems. It is from this principle that Case-Based Reasoning (CBR) emerged. A major advantage of this approach is that CBR supports incremental learning, since each time a problem is solved it can be retained for future problem solving scenarios.

Another advantage of CBR is that it has proven itself to be a suitable methodology in weak theory domains. In these domains it is often impossible or extremely difficult to use first principles as a basis for solving problems. It is for this reason that in such domains, CBR can potentially result in a lower knowledge engineering cost than using other systems (Cunningham 1998). CBR has become widely used in both academic and industrial applications. Examples of some applications include:

- Classification
- Diagnosis
- Recommender Systems
- Planning

- Decision Support

Recently an emphasis has been placed on CBR's ability to produce more useful explanations to support its solutions to problems. In this chapter we will examine some of the background and general theory of how CBR systems operate.

2.1 CBR Background

The CBR approach is based on two observations of real world problem solving. The first is that similar problems tend to have similar solutions, and secondly that the types of problems encountered tend to reoccur over time. The idea behind CBR is to retrieve and adapt previous cases when solving a problem. This process eliminates the need to model the causal interactions in a particular domain. The process is illustrated in Figure 2.1 where Q_p represents the details of a problem. During the retrieval stage S_p is found to be the most similar previous problem with a solution of S_s. Using S_s an adapted solution, Q_s can be derived as the solution for the problem Q_p. Using a retrieval and adaptation technique in this way, is often a simpler task to implement than solving the problem from first principles.

Using CBR as a reasoning system has a number of advantages over other knowledge based systems including rule-based and model-based systems. According to Leake (1996) there are five main problems identified in real-world AI systems that can be alleviated using CBR systems.

1. *Knowledge Acquisition:* In complicated domains the development of rule-based or model-based systems often involves an intensive knowledge engineering effort. If these domains have previous case solutions readily available (in the form of records, diagnostic logs or catalogues) the knowledge acquisition task becomes relatively easier. In domains that do not have previous case solutions readily available there may be a significant effort needed for knowledge acquisition, but may result in easier maintenance of the system after deployment

2. *Knowledge Maintenance:* CBR is considered a *lazy learning* technique Aha (1997) as it does not perform any reasoning until presented with a query. As a result cases can be easily added to the CBR systems even after system deployment. This ease of maintenance contrasts favourably to models that would need to be recompiled when new cases are added.

3. *Increasing Problem-Solving Efficiency:* CBR systems often retain failed as well as successful solutions. Retaining failed solutions can warn of future potential problems.

4. *Increasing Quality of Solutions:* In complicated domains where the underlying theories are not well understood, it is a difficult task to build a model or rule based systems to accurately represent the domain. However cases are real world solutions to real world problems, so they generally have a higher quality solution. The ability of CBR systems to expand its case-base allows for solutions to become more accurate over time.

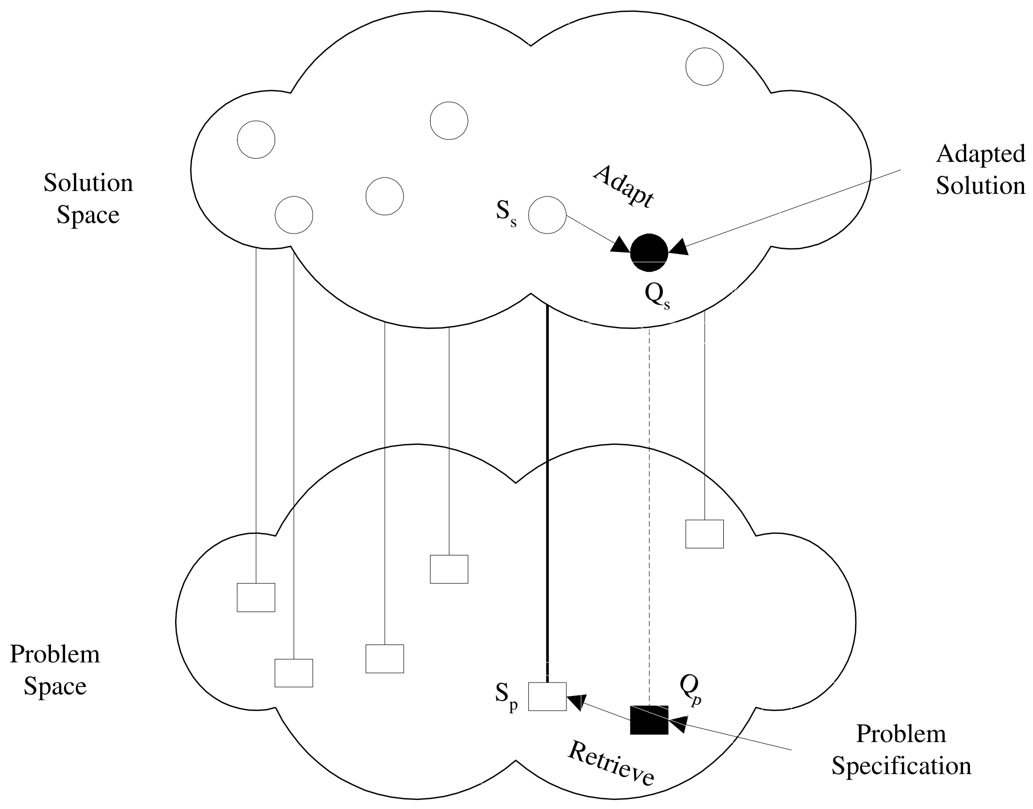


Figure 2.1: Transformation mappings used in CBR

5. *User Acceptance:* No matter how accurate an AI system is, it must still be accepted by the end user. In order for a user to accept a solution, the user must be convinced that the solution was achieved in a reasonable manner. This is a difficult task with many AI systems: Neural Networks, Support Vector Machines and Simulations generally cannot provide explanations to support their decisions, and rule-based systems explain their decisions using rules that the user may not fully understand or accept (Riesbeck 1988). Chapter 3 will provide a more detailed analysis on explanations in decision support systems.

2.2 CBR Methodology

In its most general form the CBR methodology can be described in terms of a four phase cycle that occurs at runtime (Aamodt and Plaza 1994) . Figure 2.2 shows these four stages which are also known as the *four REs*.

1. Retrieve the most relevant previous case(s).

2. Reuse the knowledge and information contained in the retrieved case(s) to propose a solution

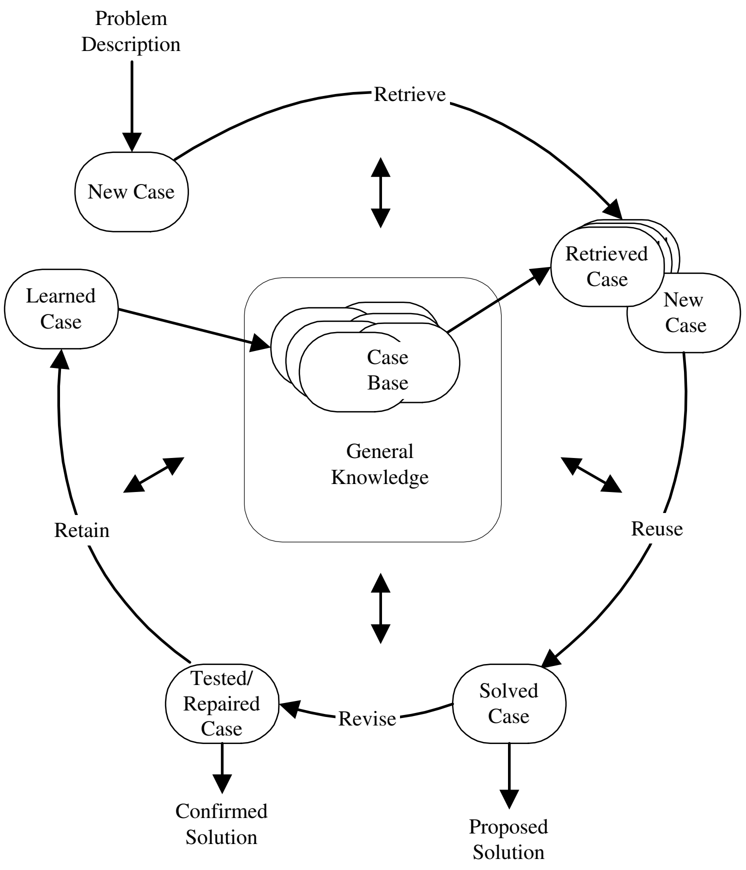


Figure 2.2: CBR Cycle (Aamodt and Plaza 1994)

to the current problem.

3. Revise the proposed solution.

4. Retain the parts of this experience which can be used to solve new problems that may occur in the future.

This model is beneficial in that it separates the design of a CBR system into distinct steps. The remainder of this section will describe each of these steps in more detail.

2.2.1 Retrieval Phase

The retrieval phase in the CBR cycle is generally considered the most important phase, as the solution to the problem will be based upon the retrieved case(s). Before retrieval the case-base is organised into a structure called the *case-memory* that supports retrieval. Retrieval is then triggered when a problem or *target case* is presented to the CBR system. Using a retrieval mechanism the most relevant previous case(s) are selected from the case-memory. As selecting the most relevant case(s) from the case-memory is considered an important part of CBR it will be dealt with in more detail in section 2.5

2.2.2 Reuse Phase

In the Reuse Phase of CBR the retrieved case(s) are used to propose an adapted solution to suit the current problem case. A common way of performing adaptation is to use a rule-based system to adapt the solutions of the retrieved cases. Although this approach may be suitable in some domains, it is often not an adequate solution in knowledge-intensive domains. Adaptation rules may often rely on strong domain knowledge, but since CBR is often used in domains that are not well understood, the adaptation phase may not be efficiently possible. This is a particular problem for planning or configuration domains. In this situation a small difference between a retrieved case and the problem case can require a significant adaptation process to generate a suitable solution. It is for this reason that Watson (1997) refers to the adaptation process as the Achilles' heel of CBR.

However adaptation techniques have been developed in a number of knowledge intensive domains. These include CBR-TFS a system that proposes useful ingredients in order to manufacture viable tablets (Craw et al. 1998), PARIS a case based action planning system (Bergmann 1993) and MOCAS a fully developed diagnostic system of a CNC (Computer Numeric Control) machine (Pews and Wess 1993)

Kolodner (1991) argues that although people find it difficult to remember suitable past solutions for new problems, they find it easy to adapt suitable past solutions to solve new problems. Based on this argument CBR systems can be used to retrieve suitable cases, while leaving the adaptation to the user. Clavier (Mark et al. 1996) is an example of a successfully developed system in which the users were willing to participate in the adaptation process once the retrieval was performed by the system.

In knowledge-light classification the simplest method of reusing the retrieved case(s) is to use the solution(s) from the most relevant case(s) as the solution to the current problem. A number of techniques can be used to propose a solution to a given problem based on the solutions to the most similar previously occurring cases. These include:

- Solution of the most relevant retrieved case,
- Majority solution of a predetermined number of the most relevant retrieved cases,
- Solution based on a majority voting by a predetermined number of the most relevant retrieved cases where the vote can be proportional to the degree of similarity of each of the retrieved cases to the original problem case.

2.2.3 Revise Phase

It is not always possible to have a CBR system that is 100% accurate. However, one of the advantages with CBR systems is that their solutions can be easily evaluated and repaired if necessary.

The evaluation of the quality of a solution is often performed outside of the CBR system. This is done by using the solution as the solution to the real world problem for which it was proposed. The results from applying the solution are then validated. For example in a fault diagnosis system, if the proposed solution corrects the fault, then the solution was correct, otherwise there is a problem with the solution. Alternatively in a classification task, if the classification by the CBR system was incorrect then there was a problem with the solution.

If the solution proposed by a CBR system is considered incorrect, the solution needs to be repaired. In the situation of a classification task, this is simply a matter of updating the classification of the problem. However, in fault diagnosis the actual cause of the fault must be determined and the solution updated to be the determined fault.

2.2.4 Retain Phase

Once the solution has been evaluated, and if necessary repaired, a decision is made as to what information from the problem solving experience should be retained in order to assist problem solving in the future. The retention policy might be to store the entire contents of the problem case along with the revised solution, or to only store relevant parts of the problem. It should be noted that the revise and retain stage may be performed anywhere from moments to many months after the start of the cycle. This is due to a possible time delay for the effects of applying a solution to take effect. For example in a medical diagnosis task, the success or failure of a particular treatment may not be known for months afterwards (Aamodt and Plaza 1994).

2.3 The Knowledge Contained in CBR Systems

A knowledge container contains structured knowledge that can be used by many tasks. For example prominent knowledge containers in rule based systems are facts and rules. Richter (1995, 1998) has identified four knowledge containers which defines the knowledge used in CBR systems. Richter's knowledge containers are:

- **Vocabulary Knowledge**: Consists of structural and semantic components that describes the system.
- **Case Knowledge**: Consists of the previous problems that can be used to solve future problems.
- **Similarity Knowledge**: Describes how the most relevant case(s) for solving a given problem are selected from the case knowledge.

- **Adaptation Knowledge**: Is the knowledge required to translate a prior solution to fit a given query.

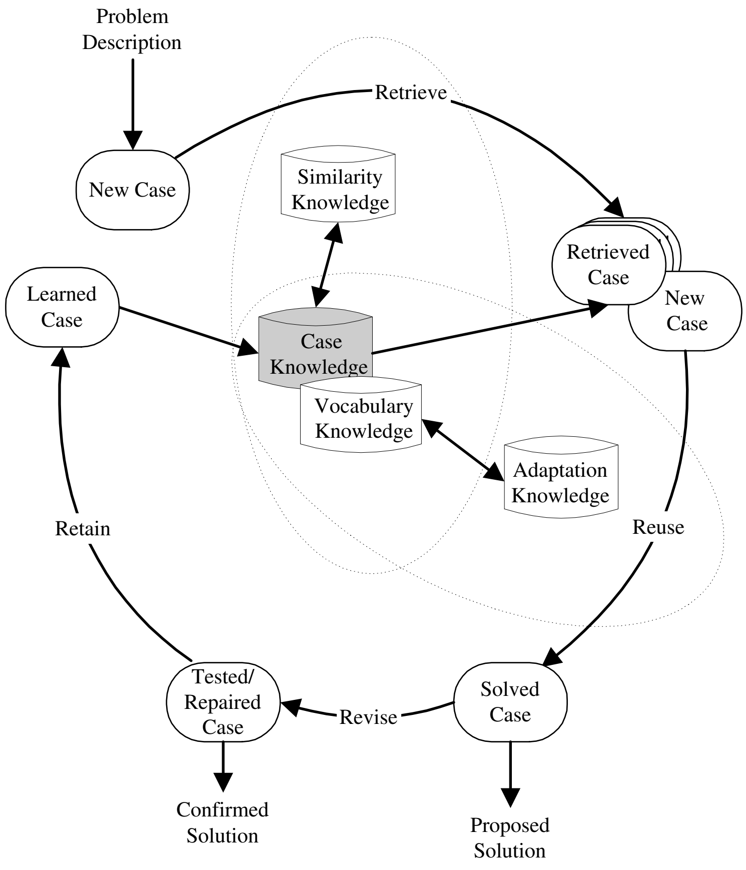


Figure 2.3: CBR Knowledge Containers

While vocabulary, similarity and adaptation knowledge are structured and used at compile time, the knowledge contained in the case-base is not used until run-time. According to Richter this is a major advantage of CBR as the acquisition of cases provides a minimal work load, circumventing the knowledge acquisition bottleneck of other knowledge based systems. Figure 2.3 shows how the knowledge containers interact with the CBR cycle. As retrieval is most pertinent to our research Section 2.5 will provide further detail on the similarity knowledge container. Firstly in order to understand similarity knowledge better, we will briefly cover how case knowledge is structured in the next section.

2.4 Case Knowledge

The case knowledge is stored as unique cases that represent a particular problem. Cases are made up of several features and the corresponding values that occurred in the situation. The first part of the case, the *specification*, consists of a set of features made up of attribute-value pairs. The second part, the *solution* is an optional part of a case. Table 2.1 shows an example of a case from the travel case-base (Lenz 1993). In this example the solution for the case is the *Hotel* field. In this situation the case is simply represented as a list of features (made up of attribute-value pairs). Using a flat layout like this to represent cases is a commonly used technique in CBR. However, in some domains a flat representation scheme cannot adequately describe cases in the domain.

Table 2.1: Example of a flat attribute-value case representation from the travel case base (Lenz 1993)

Case	**Journey 149**
Holiday Type	Recreation
Price	922
Number Of Persons	3
Region	Black Forest
Transportation	Car
Duration	7
Season	August
Accommodation	Three Stars
Hotel	**Berghotel Kandel, Black Forest**

There has been a number of representation schemes developed that can also be used to represent cases. Examples of which include:

Textual Representation Cases can be represented as free text as described in Lenz and Ashley (1998) or as a list of questions and answers as used by the Navy Conversational Decision Aids Environment *NaCoDAE* (Aha et al. 2001).

Object Oriented Representation Cases are made up as a collection of objects each of which contain their own set of features. This approach is useful for complex or structured case knowledge as taxonomic relations and inheritance features can be represented. However in such situations similarity and case retrieval are more complex (Bergmann and Stahl 1998). Examples of object oriented representation schemes include CASUEL (Manago et al. 1994) and Noos (Plaza 1997).

XML-based Representations Over the past few years several XML-inspired representation schemes have been introduced into the CBR community. Some of the earliest work on using XML was the introduction of CBML (CBMLv1) by Hayes et al. (1998). The main motivation behind the development of CBML was to facilitate the storage and distribution of case data over a network and possible inter-operability with non-CBR systems. The initial plan for CBML was to provide similar functionality to that provided by the CASUEL representation scheme, therefore a lot of its design as it initially evolved was based on CASUEL. A more detailed description of CBML can be found in Section 6.1. Other early work on using XML was in the CARET system (Shimazu 1998), which used XML to mark-up cases of natural language text describing support problems and solutions. This representation scheme was also the basis for a scheme used by the sales support system HVAC developed by (Watson and Gardingen 1999). A more recent version of XML based representation schemes is OML (Outline Markup Language) (Bergmann 2002).

Using the most similar case to solve a problem can often leave CBR systems susceptible to noisy cases. This problem is overcome to a certain extent by generating a solution based on a number of similar cases rather than just one. More information on this approach is given in section 2.6. Another approach to handling the noise problem is the removal of problem cases from the case base. Some of the early work in removing noisy training examples in case-based classifiers was performed by Wilson (1972) and Tomek (1976). Recently there has been a renewed focus on case editing (Delany and Cunningham 2004, Brighton and Mellish 2002, McKenna and Smyth 2000). Noise reduction is not just useful for improving the accuracy of a CBR system, but also useful in areas where retrieved cases are used for explanation since unsuitable cases will undermine the credibility of the explanation for the user (Roth-Berghofer 2004).

2.5 Similarity Knowledge

As it is the retrieved cases that are used to solve a problem in a CBR System, it is important that the retrieval phase selects the case(s) that are most suitable for adaptation (Bergmann et al. 2001). Even in situations where adaptation is not performed or left to the user, the retrieval phase should still select the most re-useable case(s). The function that is used to identify the most suitable case is often referred to as the *similarity* function Althoff and Richter (1999).

Generally similarity functions work by attempting to select cases with the most similar problem description. This is based on the assumption that the greater the similarity between the specification of a query case and the specification of another case the more useful that case is for solving the query case. This section will look at how similarity knowledge can be used to calculate the similarity between two cases.

There are three aspects to similarity:

- Local Similarity: Refers to similarity at the feature level,
- Feature Weight: Refers to the importance of the feature,
- Global Similarity: Refers to overall similarity at the case level. The global similarity function is an amalgamation of the local similarity functions and the corresponding feature weights.

The amalgamation function is commonly the weighted sum of the local similarity measures. Equation 2.1 shows a commonly used amalgamation function to calculate the global similarity between a query case Q and a case in the case base X, where q_f and x_f are particular features from a set of possible features F, $\sigma(q_f, x_f)$ represents the local similarity between q_f and x_f and w_f is the weight associated with the feature f

$$Sim(Q, X) = \sum_{f \in F} w_f \ \sigma(q_f, x_f) \tag{2.1}$$

Sometimes specially suited measures need to be developed to calculate the local similarity, $\sigma(q_f, x_f)$, between two features. However there are a number of generic measures that are often used in CBR systems. Equation 2.2 shows a commonly used generic function for calculating local similarity.

$$\sigma(q_f, x_f) = \begin{cases} 1 & f \text{ discrete and } q_f = x_f \\ 0 & f \text{ discrete and } q_f \neq x_f \\ 1 - \frac{|q_f - x_f|}{x_{\max} - x_{\min}} & f \text{ continuous} \end{cases} \tag{2.2}$$

For example if comparing two discrete attributes with the same value, then they would have a local similarity of 1. However, if they contained different values their similarity would be 0. For real valued attributes the similarity will result in a value in the range of [0-1], 1 if their values are both equal.

However the local similarity measures defined in Equation 2.2 are often too simple for practical applications. There are numerous other techniques for defining the local similarity measure. Often the local similarity measure is not feature specific but based on the feature type. The following are a list of some possible feature types:

- Numeric - can be integer or double type with a particular range.
- Boolean - the value can be either true or false.
- Symbolic - the attribute value must be from a specified list of possible values.
- String - the value can be any string.
- Taxonomy - similar to symbolic type except that the possible values are represented with a tree structure.

Although there are no similarity measures that can cover a particular feature type for all situations, there are a number of measures that can be used to cover a lot of situations. Sections 2.5.1 and 2.5.2 investigate two techniques that are often useful for defining local similarity measures for some of the above mentioned feature types.

2.5.1 Array Similarity Measure

Consider the feature *Temperature* in the Tennis dataset[1], this feature is of a symbolic type and can have three possible values; `Hot`, `Mild` and `Cool`. Although the values `Cool` and `Hot` are completely dissimilar and hence a corresponding similarity value of zero (using Equation 2.2) is an appropriate value, it may be beneficial to represent a partial similarity between `Cool` and `Mild`, and likewise between `Mild` and `Hot`. To allow a partial similarity representation in a local similarity measure for symbolic features the array similarity measure can be used. The main advantage of array similarity measures is that they allow a partial similarity between attribute values, even if they are not equal. Table 2.2 shows an example array similarity measure for the Temperature feature. In this example the similarity between the values `Hot` and `Cool` is 0, while the similarity between `Hot` and `Mild` is 0.5.

Table 2.2: An example array similarity measure definition for the Temperature attribute

	Cool	Mild	Hot
Cool	1	0.5	0
Mild	0.5	1	0.5
Hot	0	0.5	1

The array similarity measure is useful for features with a finite number of possible values, but requires the user to precalculate the similarities in advance. For symbolic features that contain a small list of possible values, this is not a major problem. However, as the number of cells in the similarity array is n^2, where n is the number of possible attribute values, there is a significant increase in the workload required to setup this type of measure as n increases.

2.5.2 Difference Similarity Measure

Local similarity measures can also be directly related to the difference, δ, between features. This measure is only suitable where a difference can be defined between attribute values, e.g. with numeric features the difference between two features could be the arithmetic difference between the two feature values. There also exist some simple techniques that are sometimes useful to compute the difference between features of symbolic and taxonomic types. For taxonomic attributes the differences can be calculated as the number of branches between the two attribute values in the taxonomy (Bergmann 1998). On the other hand for symbolic features, if the possible feature values can be logically ordered, the difference could be the relative difference between the two positions in the ordered list of possible values. It is also possible to define difference functions for string

[1]The tennis dataset contains data for whether it is suitable to play tennis or not. Its specification contains 4 different attributes *Outlook*, *Temperature*, *Humidity* and *Wind*. The full dataset can be found in Appendix A

attributes. However, it would be difficult to define one as a generic function as these functions would generally be domain specific.

Once a difference function has been defined, a relationship between difference and similarity is required. One solution to this is the use of a mathematical equation, e.g. an inverse function. Another alternative is to use a graph to relate them. Figure 2.4 shows an example graph for the mapping of difference to similarity. In this example if the difference between two attribute values is 8 then the similarity from the graph is 0.4. Likewise a difference of 4 results in a similarity of 0.6.

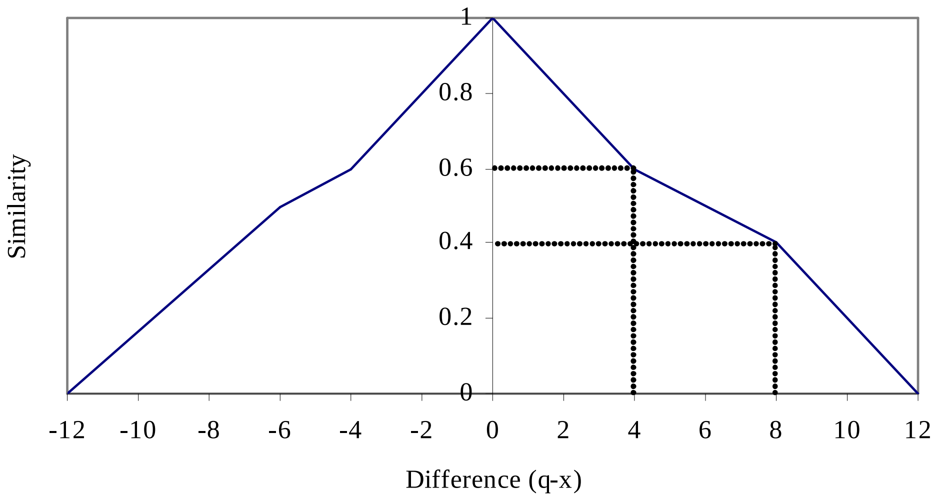
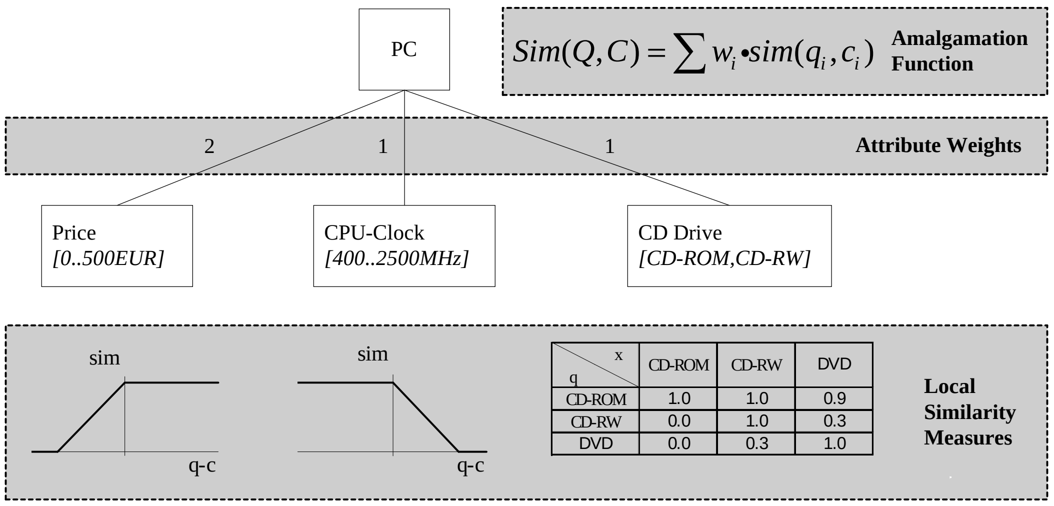


Figure 2.4: Example similarity graph

2.5.3 Utilising Similarity Knowledge

Bergmann et al. (2001) argue that similarity measures should consider knowledge about the utility of cases used to solve problems. The local similarity measures examined so far have been used to select cases that look similar by only considering the syntactical differences between the attributes being compared. By incorporating a notion of utility into similarity a more relevant case for problem solving can be selected.

Stahl (2002) illustrates this approach with a simple example in the domain of PC sales where each case represents the configuration of a particular computer. Figure 2.5 shows an example from this domain where, Q is the query specification and C a target case, with each containing three attributes: *Price*, *CPU-Clock* and *CD Drive*. The local similarity measure used for the *price* is a difference function where the similarity graph represents a 'less is perfect' function. This type of function implies that if a customer is looking for a PC configuration at a particular price, that a PC with the same configuration but cheaper is a perfect solution, however the utility of a PC decreases as the price increases over what the customer was originally intending to pay. Complementing this approach is the 'more is perfect' function which is used for the *CPU-Clock* attribute. In this situation the utility of a PC increases as the *CPU-Clock* increases over the original query.



q \ x	CD-ROM	CD-RW	DVD
CD-ROM	1.0	1.0	0.9
CD-RW	0.0	1.0	0.3
DVD	0.0	0.3	1.0

Figure 2.5: Local similarity measures for the domain of PC sales (from Stahl 2002)

Also shown in Figure 2.5 is an example of an asymmetrical array similarity measure. The reasoning behind this measure is that a `CD-RW` is a perfectly acceptable replacement and a `DVD` is a highly acceptable replacement for a situation where a `CD-ROM` is requested. However, if a `DVD` or `CD-RW` was requested, a `CD-ROM` is an inadequate compromise.

The use of utility-based similarity measures, may enable a customer to find a PC that has a faster *CPU-Clock*, a better type of *CD-Drive* but is actually less expensive than requested in their original query, hence a more useful case being identified than one that might have been retrieved if distance based similarity metrics were used. One problem with utility-based similarity metrics is that it adds to the knowledge acquisition bottleneck that CBR attempts to avoid. This leads to a trade off between the usefulness of the cases being selected during the retrieve phase, and the intensity of knowledge acquisition. Nevertheless a number of current CBR applications already employ some forms of utility in their similarity measures. It should also be noted that similarity measures often attempt to approximate some form of utility (von Neumann and Morgenstern 1944). This can often lead to similarity measures becoming extremely sophisticated and also resulting in a knowledge acquisition bottleneck.

Another main problem with using utility functions for retrieval is that they are most often given by *a-posteriori* criteria, i.e., they can only be used after the problem is solved. However some techniques have been adopted to allow the use of utility measures before a problem is solved Bergmann et al. (2001).

2.5.4 Feature Weight

An important component of the amalgamation function (Equation 2.1) is the attribute weight. The attribute weight is an indication of its relevance. Therefore the weight associated with important attributes needs to be greater than the weight associated with less important attributes. In a similar manner irrelevant or noisy attributes need to have their associated weights reduced so that they have less of an influence compared to relevant attributes. Feature weighting is a well established domain in machine learning and CBR, with many techniques available for automatically learning feature weights (Wettschereck et al. 1997). An alternative to feature weighting is feature selection, whereby features are assigned binary weights. Work on feature selection has been around for a considerable length of time (Fu 1968, Mucciardi and Gose 1971, Quinlan 1993). The main advantage of feature selection is that it reduces the search space, hence resulting in faster retrievals. A number of techniques can be easily used for both feature selection and feature weighting, these include RELIEF (Kira and Rendell 1992a,b, Kononenko 1994) and wrapper based approaches (Kohavi et al. 1997).

2.6 Retrieval

The simplest retrieval technique is the k-Nearest Neighbour (k-NN) approach. Retrieval is performed by sequentially comparing the query case to each case in the case base, and returning the k cases with the highest similarity score. The main advantage of k-NN is its simplicity. It does not require indexing structures for retrieval and cases can be added to the case base without having to recompile the case memory. The major drawback with k-NN is its scalability as Retrieval time increases linearly with the size of the case base.

Lenz and Burkhard (1996) presents three major conditions that should be met by a retrieval system:

Efficiency: Access to relevant cases should avoid an exhaustive search through the entire case memory.

Completeness: Every sufficiently similar case in memory will be found during retrieval.

Flexibility: Users should be allowed to conduct a search with whatever information is available to the user.

Based on these criteria they have developed Case Retrieval Nets to perform retrieval (Lenz 1999). Case retrieval nets are an efficient memory model that uses a bottom up approach to calculate similarity where cases are indexed based on information that they have in common. Case retrieval nets contain the following components:

Case Nodes: Represent the cases in the case-base.

Information Entities: Represents attribute-value pairs.

Relevance Arcs: link case nodes to the information entity that represents a particular attribute-value pairs in that case.

Similarity Arcs: interconnect information entities of the same feature. These arcs contain a weight which is equal to the local similarity between the two attribute-value pairs

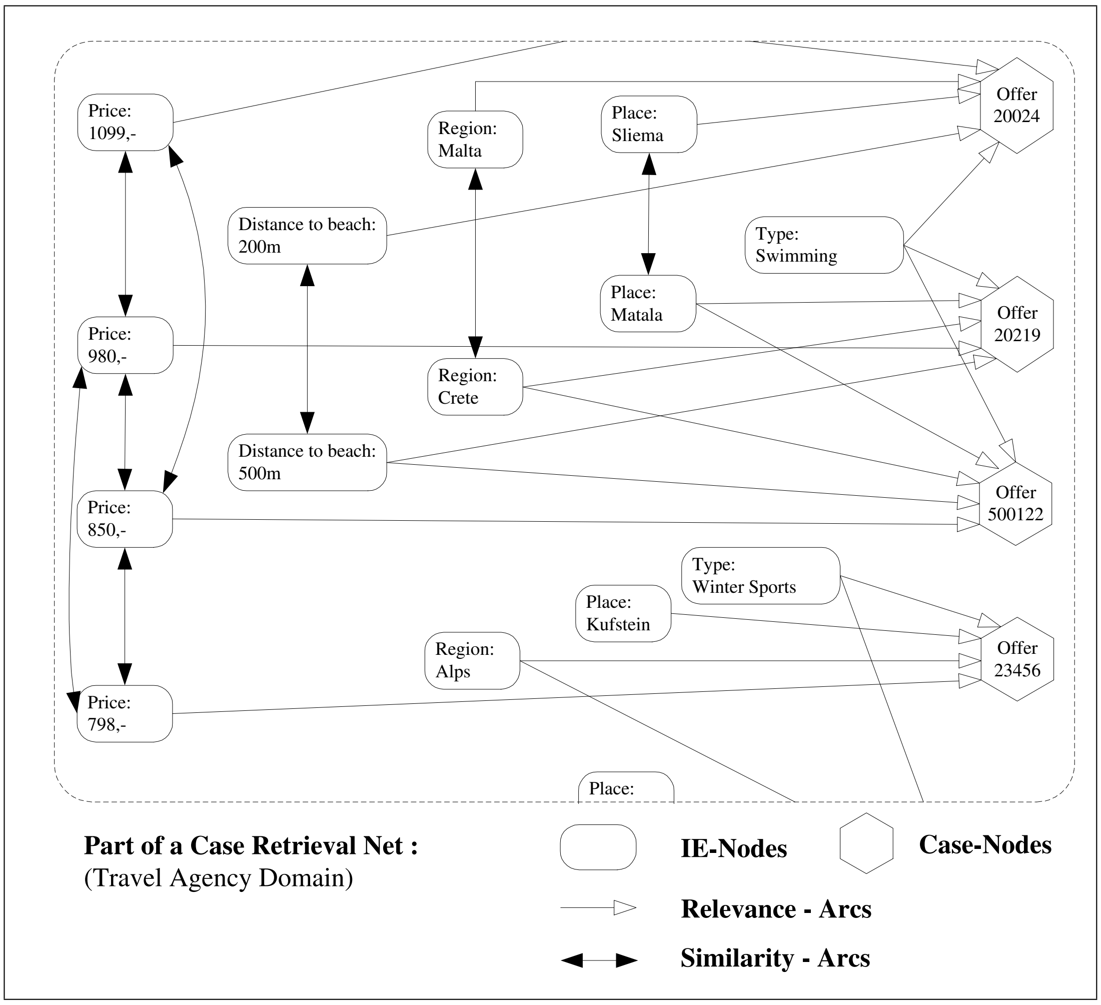


Figure 2.6: The case retrieval net structure (from Lenz and Burkhard 1996)

Figure 2.6 illustrates the structure of a case retrieval net using data from the travel case base (Lenz 1993). In this figure each case node is associated with a set of information entities using relevance arcs. For example, the information entities in Figure 2.6 for case node *"Offer 20219"* is represented in Table 2.3.

We can see from Figure 2.6 that several other case nodes share the information entities used by *"Offer 20219"*. For example *"Offer 500122"* also contains the same information entities for Distance to beach, Region and Place as *"Offer 20219"*. Each information entity is also connected

Table 2.3: Offer 20219 from the travel case base

Case	***Offer 20219***
Place	Matala
Region	Crete
Distance to Beach	500m
Price	980
Type	Swimming

to other information entities of the same type by a similarity arc that represents the local similarity between the two nodes.

When a query case is presented for retrieval, the following stages occur:

1. Activation of the information entities that are described by the query case.
2. Propagating the activation according to the similarity arcs through linked information entities.
3. Calculation of the total activation for each case node.

The case node with the highest activation is the most similar case to the target case.

The main advantage of case retrieval nets is that the local similarity calculations are only performed once and that missing values are ignored. However case retrieval nets are not suitable for all case-base applications as there is a significant cost with initialising a case retrieval net. There is a trade-off between the time spent setting up the net and faster retrieval times. The case retrieval net speed up mechanisms are counter-productive if features contain many possible feature values. Therefore the speed-ups should not be used unless there are features in the case-base which will benefit from the net structure.

2.7 Conclusion

Starting with the origins of CBR, in Section 2.1, this chapter described some of the important principles common in CBR systems. After a high level overview of the methodology of CBR in Section 2.2 the four knowledge containers in CBR were described in 2.3. As the case and similarity knowledge containers are the most important containers in our work on producing explanations these containers are described in further detail in Sections 2.4 and 2.5. In Chapter 6 we present a medical decision support system that we implemented using the case retrieval nets described in Section 2.6 of this chapter. This system uses case-based explanations to produce explanations in the hope of increasing user confidence in the system. Next, in Chapter 3 we describe the use of explanations and confidence measures in decision support systems.

Chapter 3

Explanation

Aamodt (1991) defines two different interpretations of explanations in AI:

- Explanation as part of the reasoning process itself.
- Explanation for the benefit of a user.

This thesis is concerned with the latter interpretation of explanation. The purpose of explanations for the benefit of a user is to make the reasoning process, its result or the usage of the result understandable to the user (Sørmo et al. 2005). For example, in rule based explanation systems, the user is often presented with a rule or set of rules as an explanation, while in case-based explanations the user is often presented with actual cases. However, the task of producing good explanations does not stop at simply presenting a set of rules or a case to the user.

The first part of this chapter will provide a review on what is needed to produce good quality explanations; both in everyday life and in decision support systems. After this review some examples of explanation techniques used in decision support systems in general will be presented in Section 3.2 with Section 3.3 focusing more on explanations in CBR systems. Finally Section 3.4 will provide a brief review on how confidence measures can be used to improve explanations.

3.1 Explanations

In our daily lives explanations are used everywhere in numerous different forms. The type of explanation used is often determined by the circumstance. For example it may be suitable for a parent to say to a child "Because I said so!" as an explanation to why the child can't see a particular movie. However in many situations a more informative explanation is required. For example lawyers in a court case often use past cases to support their arguments. From these two examples it is possible to imagine how diverse types of explanations can be. According to Leake (1995a) it is the goals of both the sender and the recipient of an explanation that influence what types of explanations are acceptable.

Although, it is difficult to completely define how an explanation should be formed to cover every circumstance, explanations can be considered to consist of two main elements; what is to be explained, called the *explanandum*, and the explanatory expression, called the *explanans*. For example: You cannot go to the cinema (explanandum); You are too young (explanans); You cannot go to the cinema because you are too young (explanation).

Over the last few decades there have been numerous different approaches to explanations proposed by philosophers. Early approaches include a logical deductive approach suggested by Carl Hempel and Paul Oppenheim (Hempel and Oppenheim 1948, Hempel 1965) to more pragmatic approaches given by Gilbert Harman (1965) and Wesley Salmon (1971). More recent work on explanations includes explanation in natural language by Peter Achinstein (1983) and using cognitive models of explanation by Roger Schank (1986) and David Leake (1995b)

It is possibly the work on scientific explanation by Bas van Fraassen (1980) that presents the minimum criterion necessary for explanations. Van Fraassen takes a strictly empiricist approach claiming that explanations are always an answer to an implicit or explicit contrastive why question. By 'contrastive' he means a question of the form "Why S_0 rather than $S_1 \dots S_n$?". Van Fraassen maintains for an explanation to be acceptable, it must favour the observed state S_0 over the other states. In other words the explanation must increase the probability of S_0 relative to $S_1 \dots S_n$. This approach is of particular use when generating explanations in decision support systems based on a given classification.

Van Fraassen's approach provides the minimum criterion for a suitable explanation. However it is not adequate to just have a suitable explanation. In order to improve on this approach other properties of a good explanation must be considered. The philosopher Stephen Toulmin developed a six-element argument structure that can be used in a broad range of situations (Toulmin 1958, Toulmin et al. 1984). The six elements of Toulmin's argument structure as given by Shankar and Musen (1999) are:

1. *Data*: The particular facts about a situation on which a claim is made.
2. *Warrant*: The knowledge that justifies a claim made using the data.
3. *Backing*: The general body of information or experience that validates the warrant.
4. *Qualifier*: The phrase that shows the confidence with which the claim is supported to be true.
5. *Rebuttal*: The anomaly that shows the claim not to be true.
6. *Claim*: The assertion or conclusion put forward for general acceptance.

Figure 3.1 shows how these six elements are used to perform an argument. This argument structure has been widely used in explanation research. In fact Wick (1992) states that a lot of early research in explanation has, without stated intent evolved to engulf Toulmin's argument structure.

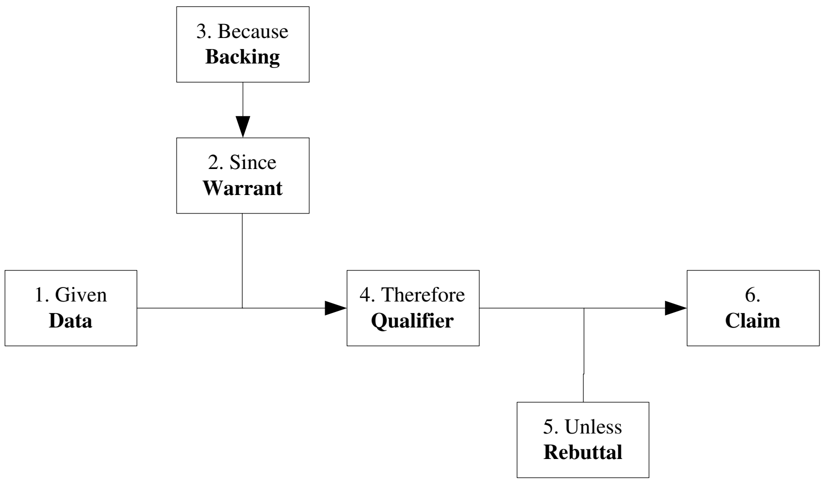


Figure 3.1: Toulmins Argument Structure. The structure reads given Data, therefore Claim, since Warrant, because Backing, therefore Qualifier unless Rebuttal. The elements of the structure and the relationship among them can be used to generate explanations.

Although Toulmin's argument structure provides a good basis for what is required in a decent explanation, other factors must be considered when generating explanations in Decision Support Systems. Swartout and Moore (1993) present five requirements for explanations in decision support systems:

1. Fidelity: The explanation should mirror the knowledge used by the system in its reasoning
2. Understandability: Users should be able to understand the explanation. Components to consider are terminology, user sensitivity, abstraction, summarisation, perspectives, linguistic competence and feedback.
3. Sufficiency: There should be enough knowledge to provide explanations in different contexts.
4. Low construction overhead: Explanations should not be time consuming and difficult to build.
5. Efficiency: Generation of explanations should not degrade runtime performance of the system.

Most of these requirements are straight forward and acceptable. However there is some debate as to the importance of fidelity when generating explanations in decision support systems. Although expert system designers might find explanations that accurately represent the reasoning done by an expert system useful, actual users of the system may not find them useful. In fact the use of explanations with a high fidelity may cause confusion for users. According to Majchrzak and Gasser (1991) more than 50% of decision support systems, which are installed in companies, are not being used. An analysis of many of these early Decision Support Systems showed that a major cause of these systems failing was that the explanations produced were incomprehensible by users or they failed to address the users' goals in an explanation. Wick and Thompson (1992) go a step further by arguing that fidelity should be avoided in explanations. They argue that the generation of explanations should be a completely separate process from the classification process. In support of their argument they point out that explanations provided by human experts often tend to lack fidelity even though they may often be considered to be useful. Swartout (1984) and Chandrasekaran et al. (1989) have also shown that providing effective explanations frequently requires supplementary knowledge in addition to the knowledge required for reasoning.

3.2 Decision Support System Explanations

The use of explanations to increase a users confidence is quite common in decision support systems. Some types of systems are inherently easier to generate explanations for than others. This section will examine some techniques used for generating explanations in decision support systems.

3.2.1 Decision Trees

A decision tree is a hierarchical model for supervised learning whereby the local region is identified in a sequence of recursive splits in a smaller number of steps (Alpaydin 2004). A decision tree can be considered as a tree-like map of the reasoning process and is composed of internal decision nodes and terminal leaves. Figure 3.2 shows an example decision tree from the tennis dataset. Each decision node implements a test function with discrete outcomes labelling the branches. Given an input, at each node, a test is applied. Depending on the outcome of this test a particular branch is taken. This process starts at the root and repeats until a leaf node is hit, at which point the value of the leaf is the output. For example, if the `outlook` is `sunny` and the `humidity` is `high` then the output is not to play tennis.

The construction of a decision tree is considered a greedy process. At each step, starting at the root, the complete set of training data is analysed for the best attribute to split the data on. Generally the selection of the best attribute to split on is performed using an entropy based algorithm. Some popular algorithms include ID3 (Quinlan 1986), its successor C4.5 (Quinlan 1993) and ASSISTANT (Kononenko et al. 1984, Cestnik et al. 1987). After the best attribute to split on is selected, depending on wether the chosen attribute is numeric or discrete, the training data is split into two or n groups of data. The splitting process then continues recursively with the corresponding subsets until there is no need to split anymore. At this stage a leaf node is created and labelled. In the decision tree for the tennis dataset the attribute `Outlook` is considered the best attribute to use as the root node. In this situation the data is split into `Sunny`, `Overcast` and `Rainy` subsets. All the cases of the `Overcast` subset have a `yes` classification, hence a leaf node

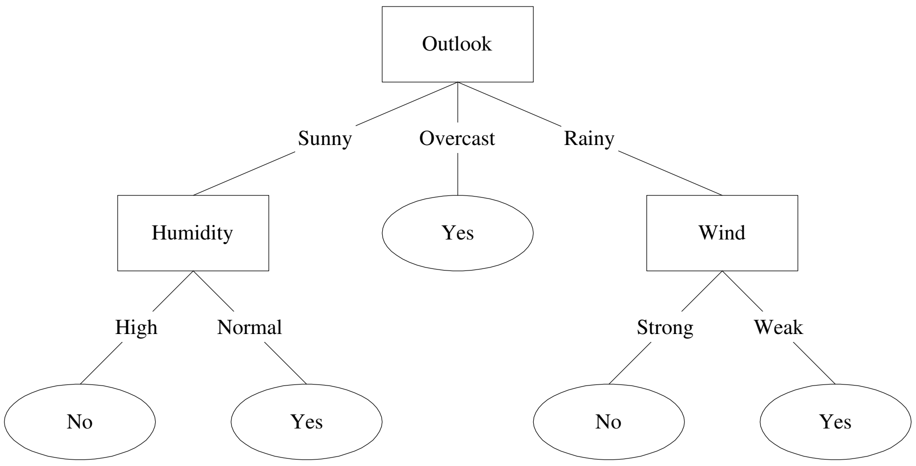


Figure 3.2: An example decision tree whether to play tennis or not from the tennis dataset (Appendix A)

is created and labelled as `Yes`. However, the examples for `Sunny` and `Rainy` need to be further expanded. Mitchell (1997) provides further information on generating decision trees.

A major advantage of decision tress is their interpretability and ease for generating explanations. As each path from the root to a leaf corresponds to one conjunction of tests the paths can be written down as a set of *IF-THEN* rules. One method of producing rules is C4.5 (Quinlan 1993). These rules can then be used as the basis for an explanation. For example consider a hot humid sunny day with a high temperature and weak winds. From Figure 7.3 we can see that this type of day is not suitable for playing tennis. The leaf node for this decision is the bottom left node of the decision tree. Therefore an example rule that can be used as an explanation for this path is:

Rule: IF(Outlook=Sunny) AND (Humidity=High) THEN NO

Explanation: As the outlook is sunny and humidity is high it is not suitable to play tennis today.

Explanations of this form are not always that simple for a user to understand. The decision tree shown in Figure 3.2 is a fairly simple decision tree, hence the path traces are relatively simple for a user to understand. However as the depth of a decision tree increases so to does the complexity of the generated rules. Figure 3.3 shows an example rule generated from a medical domain using a bigger decision tree. This type of rule would result in a longer more complicated explanation than the one produced from the tennis dataset.

The reason why such an explanation is less understandable for a user is the information overload on the user. One solution to reducing the information overload is to prune the decision tree. There are two approaches to pruning a decision tree; pre-pruning (Quinlan 1987) and post-pruning (Quinlan 1993). Alternatively the rules produced can be pruned to make them more understand-

If Overall Work of Breathing is not Moderate
and Change in Work of Breathing is not Worse
and Alertness is not Drowsy
and Change in Work of Breathing is not Improved
and Overall Work of Breathing is not None
and Age is less than 5.9 months
and Respiratory Rate is less than or equal to 46
and Respiratory Rate is greater than or equal to 38
and Heart Rate is less than 151
then Admit the patient

Figure 3.3: Example explanation from a medical domain.

able. Another solution to reduce the complexity of rules presented to a user is the use of viewpoints (Finch 1999). A simplistic approach to viewpoints is to consider them as a filter that can be passed over the knowledge, only allowing certain facts and concepts to be seen through it and holding back the facts and concepts that the user may not understand or are "insignificant" in explaining the produced classification.

3.2.2 MYCIN

MYCIN (Shortliffe 1976) an early expert system for diagnosing meningitis disorders introduced explanation queries that formed the foundation of many other explanation facilities (Wick and Slagle 1989). MYCIN consists of five components (Jackson 1986):

1. A *knowledge base* which consists of factual and judgemental knowledge about the domain.
2. A *dynamic patient database* containing information about a particular case.
3. A *consultation program* which asks questions, draws conclusions, and gives advice about a particular case based on the patient data and the static knowledge.
4. An *explanation program* which answers questions and justifies this advice, using static knowledge and a trace of the program's execution.
5. A *knowledge acquisition program* for adding new rules and changing existing ones.

MYCIN's knowledge base is organised around a set of rules of the general form

if condition$_1$ *holds with certainty* x$_1$ *and* … *and* condition$_m$ *holds with certainty* x$_m$
then draw conclusion$_1$ *with certainty* y$_1$ *and* … *and* conclusion$_n$ *with certainty* y$_n$

MYCIN uses a backward chaining mechanism to search through the set of rules. As MYCIN is running it asks the user particular questions in order to further the search. To support this process MYCIN supplies two forms of explanation:

- why a particular question was put;
- and how a particular conclusion was reached.

The answer to how a conclusion was reached is the set of rules that enabled the conclusion to be inferred. This is similar to the types of explanation from decision trees described in the last section. In general the answer to why a question is being asked is the goal that is trying to be achieved by asking the question. For example when attempting to answer the following why question:

> ‘Why do you ask if the stain of the organism is Gram negative?’

the system will look at the rule that caused this question to be asked, any other conjunctive conditions and the goal that it is trying to achieve from an answer to the question. Therefore an example answer is:

If:	1) The stain of the organism is gram negative, and
	2) The morphology of the organism is rod, and
	3) The aerobicity of the organism is aerobic
Then:	There is strongly suggestive evidence (0.8) that the class of the organism is enterobacteriaceae.

3.2.3 Black Box Explanations

In the drive to increase accuracy some machine learning techniques have evolved to use more complicated algorithms. Ensemble approaches and algorithms such as Support Vector Machines and Neural Networks have reached a level of complexity where they are not readily interpretable. Such approaches are commonly referred to as black box algorithms owing to their lack of transparency with regard to the logic behind the classifications they make. In contrast to explanation generation in rule-based systems this lack of transparency makes it near to impossible to generate explanations based directly on the reasoning process of black box systems. Although these increases in accuracy are welcome, interpretability and explanation are still needed for people to accept classifications by decision support systems (Andrews et al. 1995).

Many approaches to explanation generation for black box systems use the black box systems as an oracle to generate an unlimited amount of training data. Based on this data more inherently interpretable machine-learning systems, such as tree-based or rule-based systems, can be trained in an attempt to describe the underlying black box system (Andrews et al. 1995, Tickle et al. 1998, Zhou and Jiang 2003). The hope is that, with an abundance of training data, the explanation system should offer a good description of the underlying black-box system. However, in reality it is difficult for these tree-based and rule-based systems to fully capture the operations of the black-box without them become overly complex and also losing their transparency. As mentioned in Chapter 2, a common means of producing explanations in CBR systems is to display an actual prior case from the training set as an explanation. The use of actual training data, cases from the case-base, as evidence in support of a particular classification is a powerful and convincing form of explanation. Nugent et al. (2004, 2005) have developed a case-based explanation facility for black-box systems that avoids the complexity of tree-based and rule-based methods discussed previously.

By treating the black box system as an oracle, an artificial case-base can be built up generating artificial cases that are similar to the query case and presenting them for predication to the black box system. This artificial case-base approximates points on the function that is learnt by the black box system. Once the artificial case-base is constructed around the query case a locally weighted linear model can be derived from the artificial cases. Figure 3.4 shows an example function that might be learnt from a black box system with two artificial cases AC_1 and AC_2, plotted on the function, and the local linear approximation of the function at the query case, QC.

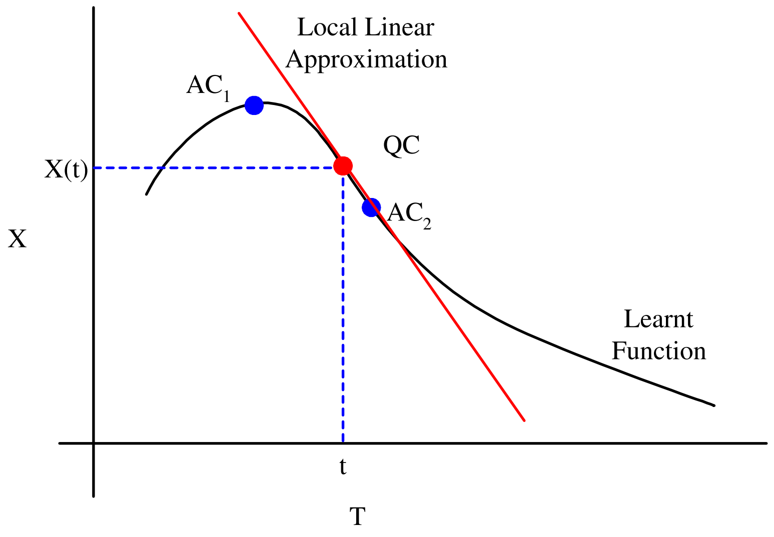


Figure 3.4: Locally weighted linear model applied to black box learnt function

The linear approximation provides a set of coefficients for each attribute. These coefficients can be used to infer how sensitive the prediction variable P(t) is to changes in the feature value, hence the relevance of the feature on a local level, and whether the feature is negatively or positively correlated with the prediction variable around the query case. The learnt feature weights can then be used in Equation 2.1 to select the nearest neighbour, from the original dataset, to the query case. The nearest neighbour can then be used as the explanation for the prediction generated by the black box system. Using the learnt feature weights and the correlation of the features it is also possible to generate supporting explanatory text to present as an explanation along with the selected nearest neighbour.

3.3 Explanation in CBR Systems

Most tutorials on CBR point to the advantages arising from the transparency and interpretability of the CBR approach. This transparency has particular advantages for explanation as pointed out by Leake (1996):

> *"... neural network systems cannot provide explanations of their decisions and rule-based systems must explain their decisions by reference to their rules, which the user may not fully understand or accept. On the other hand, the results of CBR systems are based on actual prior cases that can be presented to the user to provide compelling support for the system's conclusions."*

The view that case-based explanations are more convincing than rule-based explanations is a view that is shared by many CBR researchers (McSherry 2003a). Research by us has provided empirical evidence that supports this hypothesis (Cunningham, Doyle and Loughrey 2003). Section 7.1 provides a detailed analysis of this research.

Work on case-based explanation can be divided into knowledge-light and knowledge-intensive approaches. However, all approaches to explanation in CBR will share an important characteristic. On the spectrum of possibilities between specific and general, the case-based explanation will be at the specific end of the spectrum. When discussing explanation patterns (see (Kass and Leake 1988) for instance) Kolodner (1996) argues that what differentiates CBR from similar ideas in model-based reasoning is the concreteness of the cases. So, whether knowledge-light or knowledge intensive, case-based explanation is case-based.

There is still disagreement among CBR researchers on the implications CBR has for knowledge engineering effort. Some, such as Mark et al. (1996), argue that CBR still entails a "full knowledge acquisition effort". Others would argue that knowledge-intensive CBR is missing the point of CBR, which is the potential CBR has to finesse knowledge engineering effort by manipulating cases that are compiled chunks of knowledge. These alternative views of CBR are reflected in the different approaches to case-based explanation.

A knowledge-intensive approach to case-based explanation will incorporate mechanisms such as rule-based or model-based inference that can be used to generate explanations. Developing knowledge-intensive case-based applications will, in the words of Mark et al. (1996), involve a "full scale knowledge acquisition effort". Amongst the earliest examples of this approach is the work on SWALE and its descendants. These systems incorporate explanation patterns (XPs) that can be used for explanation. Typically, these XPs are pretty specific, e.g. the JANIS-JOPLIN-XP.

Even the more abstract XPs are pretty specific; the MAFIA- REVENGE-XP can be instantiated directly. The key point is that the system designers have incorporated model-based representations that can be used for explanation.

XPs are made up of facts and belief-support nodes that link the facts together. The facts include the premise, intermediate facts and conclusions. The intermediate facts are inferred from the facts. The JANIS JOPLIN XP is shown in figure 3.5. Janis Joplin was a young rock star who died as a result of a drugs overdose. The fact that she is a “Rock Star”, leads to intermediate facts that she has “Wealth”, “Drug Using Friends” and “Stress”. In turn “Wealth” and “Drug-Using Friends”, leads to another intermediate fact that she has “Access to Drugs” and so forth. Finally this explanation pattern concludes with “Death” due to a “Drug Overdose”. The idea with SWALE is that this explanation can be invoked to explain target cases that map appropriately to this explanation pattern.

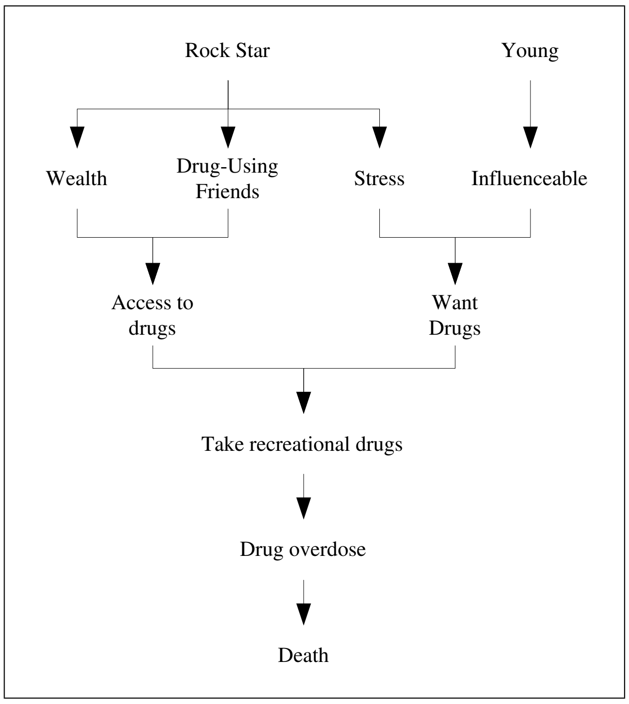


Figure 3.5: An illustration of the JANIS JOPLIN XP from (Kass 1988).

Another more recent example of a knowledge-intensive approach to case-based explanation is the DIRAS system for assessing long-term risks in diabetes (Armengol et al. 2001). The approach in DIRAS is more dynamic than that in the SWALE systems in that the explanation is built at run-time using a process called Lazy Induction of Descriptions.

The majority of commercially successful CBR applications have been knowledge-light systems; usually retrieval-only systems or mixed initiative systems involving interactive adaptation. In CBR systems that use a feature-value based representation, the retrieved cases can be used in explanation as follows:

> "The system predicts that the outcome will be X because that was the outcome in case C1 that differed from the current case only in the value of feature F which was f2 instead of f1.
>
> In addition the outcome in C2 was also X..."

Explanation in these terms (i.e. expressed in the vocabulary of the case features) will not always be adequate but in some situations, such as in medical decision support, it can be quite useful. The main difference between this and the more knowledge-intensive approach is that the explanation is expressed in terms of similarity only. The more knowledge-intensive systems still produce explanations that reference the retrieved case but the explanation is expressed in terms of causal interactions rather than simple similarity.

A good example of knowledge-light explanation in CBR is the Strategist system developed by McSherry (2001) for fault diagnosis in a toy domain. Strategist organises its cases into a decision tree using decision tree induction and it can use that tree to explain its reasoning. Explanation in Strategist is focused on explanation of reasoning (for example the relevance of a question) rather than explanation of diagnoses/predictions. The commercial CBR tool Orenge from Empolis[1] relies on comparison to retrieved cases as a mechanism for explanation. The following subsections will present three examples of knowledge-light systems.

3.3.1 CARES

Colorectal cancer is becoming the leading cause of death from cancer in the Western World. The main treatment for colorectal cancer is surgery. However, up to 50% of people that undergo surgery will eventually die within the following 5 years, the majority from local, regional or distant tumour recurrence. Early identification of recurrence can greatly increase the effectiveness of therapy and survival of patients. The National University of Singapore in conjunction with Singapore General Hospital have developed the CARES (Cancer Recurrence Support) System with the primary objective of predicting the recurrence of colorectal cancer (Ong et al. 1997). In the CARES system doctors are presented with the 10 most similar cases to the current query case, This allows the doctor to detect any possible anomalies in the prediction generated by the system.

3.3.2 ProCon

McSherry (2004) argues that the challenge in generating convincing explanations is not to convince users that the classification is correct, but that it is justified in spite of opposing evidence. McSherry argues that although the technique of displaying the most similar case as an explanation is a major advantage of case-based explanations, it can sometimes present contradictory evidence against the

[1]See the White Paper on Orenge available at www.empolis.com

classification. This occurs when some of the features of the most similar case actually oppose the classification when compared to the original problem. As similarity measures are often symmetrical, rewarding features that are more similar to the problem case regardless of whether they support the classification or not, it is common for this situation to occur in case-based explanations.

McSherry (2004, 2003b) developed ProCon-2 a system that highlights supporting and opposing features present in a case-based explanation. McSherry (1999) argues that a given feature always increase the probability of an outcome class if it is more likely in that outcome class than in any competing class. He refers to such features as *supporters* of the outcome class. In a similar manner he refers to features that are less likely in the outcome class than in any other competing class as *opposers* of the outcome class. Based on the Bayes theorem, McSherry defines supporting and opposing features of an outcome class using the following criteria:

The Support Criterion *A feature E is a supporter of an outcome class H_1 if there is at least one competing outcome class H_2 such that $p(E|H_1) > p(E|H_2)$ but no competing class H_2 such that $p(E|H_1) < p(E|H_2)$*

The Opposition Criterion *A feature E is an opposer of an outcome class H_1 if there is at least one competing outcome class H_2 such that $p(E|H_1) < p(E|H_2)$ but no competing class H_2 such that $p(E|H_1) > p(E|H_2)$*

In a binary classification task these criteria can be simplified to the following criterion:

In a classification task with two possible outcomes H_1 and H_2, a given feature E is a supporter of H_1 if $p(E|H_1) > p(E|H_2)$ and an opposer of H_1 if $p(E|H_1) < p(E|H_2)$

In the Breathalyser Domain[2] there are two possible outcome classes; `over the legal drink driving limit` and `under the legal drink driving limit`. One of the features in this domain is gender. Considering this feature we can see that.

$$p(gender = female|over\ the\ limit) = 0.23$$
$$p(gender = female|under\ the\ limit) = 0.19$$

Using the above criteria we can see that female is a supporter of `over the limit` and an opposer of `under the limit`.

Using the above criteria the task of determining supporters or opposers for nominal or discrete features compared to continuous features is a relatively trivial task. For continuous features the task is complicated by the fact that a decision needs to be made as to where to "draw the line" between values that support and oppose the outcome class. For example, for the feature `units consumed` with a value of x or greater to be supportive of `over the limit` the following inequality must hold true

$$p(units \geq x|over\ the\ limit) > p(units \geq x|under\ the\ limit)$$

[2]The Breathalyser Domain is used for predicting blood alcohol levels based on weight, gender, meal, duration of drinking and units of alcohol consumed. No prior knowledge is assumed on this domain for this section, however Section 4.2.1 provides further details on this domain.

The main problem with this approach is what value of x to use for a realistic threshold to support `over the limit`. For a binary classification, with H_1 and H_2 as the possible outcome classes, the selection of the threshold x is given by that value of x that maximises:

$$MIN(we(units \geq x,\ over\ the\ limit), we(units < x,\ under\ the\ limit))$$

where

$$we(E, H_1) = \frac{p(E|H_1)}{p(E|H_2)}$$

Using the above techniques for the breathalyser domain, supportive and opposing attributes can be selected for each feature. Table 3.1 shows the supportive and opposing features for the outcome `over the limit`. As this domain is a binary classification task, features that support an outcome of `over the limit` are opposers of `under the limit`. Likewise opposers of `over the limit` are automatically supporters of `under the limit`.

Table 3.1: Supportive and Opposing attributes for `over the limit` in the Breathalyser Domain (McSherry 2004).

Feature	Supportive	Opposing
Weight	≥ 73	< 73
Duration	≥ 150	< 150
Gender	Female	Male
Meal	Lunch	Full
		None
		Snack
Units Consumed	≥ 9.1	< 9.1

Although this technique is useful for selecting supportive and opposing features of a classification, the *a priori* global determination of supportive and opposing attributes can have detrimental affects on the explanation produced. The first problem is that the local interactions between features are completely ignored. For example the attribute `female` is always considered to be an opposer of being `under the limit` regardless of the values of the other attributes. Consider a problem case of a `female` that has consumed no alcohol with a predicted outcome of `under the limit`. The explanation in this situation would highlight the fact that the problem case is a `female` opposes the classification of `under the limit` in spite of the fact that no alcohol has been consumed. Such an explanation can have a negative effect on the confidence a user may have in the system.

Another problem with this approach is that what was originally considered to be a supporter of a particular class can become the opposer of that class after the addition of new cases to the case base. This reversal can have a major affect on the users of a system. Imagine the reaction of a user being told for numerous classifications that being female is a supporter of `over the` `limit` and then all of a sudden after some cases have been added that being female is actually an opposer of `over the limit`. This problem is likely to occur when a particular attribute is evenly distributed between different classes. Table 3.2 shows the distribution of cases based on gender in the Breathalyser dataset. Here we can see that the addition of some more under the limit females would cause a reversal of the supporters and opposer for this feature. As a major advantage of CBR systems is their suitability in domains with small amounts of training data, hence the addition of cases to the case-base being a common occurrence, it is perhaps naive to make global assumptions on the data.

Table 3.2: Distribution of Gender in the Breathalyser Dataset

	Over the limit	Under the limit
Female	7	8
Male	23	35
Total	30	43

3.3.3 FormuCaseViz

There have been a number of tools developed for visualising case-bases. Inselberg (1985) proposed and implemented a parallel coordinate plot. This approach was further extended by Falkman (2002) to develop, The Cube, which displays a case-base using three dimensional parallel co-ordinate plots. This approach allows the underlying data and the similarity metrics to be visualised. McArdle and Wilson (2003) present a dynamic visualisation of case-base usage by using a spring based algorithm. Although this technique was developed for supporting the maintenance of case-bases it could also be easily adapted to visualise retrieved cases. The work of Falkman is exploited by Massie et al. to display the case-base and the underlying knowledge from the CBR knowledge containers and the retrieval process itself in the FormuCaseViz System (Massie et al. 2004a,b).

The formulation of a pharmaceutical tablet for a given dose of a new drug is a complicated task. The process involves selecting inert excipients (e.g. Lactose, Maize Starch, etc.) to mix with the drug so that the tablet can be manufactured in a robust form.

In addition to the drug a tablet consists of five distinct components:

Filler: Provides bulk to be large enough to handle and compress.

Binder: Makes it cohesive to hold together.

Lubricant: To eject tablet from die.

Disintegrant: Allows rapid break down after swallowing.

Surfactant: Aids drug wetting and dissolution.

The formulation task is made up of choosing a suitable excipient and amount of excipient for each component so that they can perform their roles and be compatible with each other. Craw et al. (1998) provides further information on this problem domain.

FormuCase, the predecessor of FormuCaseViz, was developed along the normal CBR lines for use in the problem domain of tablet formulation. The case is made up of 5 physical attributes 20 chemical attributes and 10 solution attributes; 5 nominal values identifying the excipients used and 5 numeric values representing the quantity of each excipient. The output from FormuCase is presented in a report form. An evaluation of this system showed a low confidence in the retrieval system due to a reluctance to accept that the retrieved cases were in fact the most similar cases and difficulty in accepting the adaptation because the differences between the problem and retrieved cases were not obvious.

FormuCaseViz was developed to address these problems. In FormuCaseViz the problem and solution are displayed in parallel coordinate plots. These have the advantage of being able to display multi-dimensional data in two dimensions. The graphical display of FormuCaseViz is of three panels each containing a two-dimensional parallel coordinate plot; one for the chemical properties, one for the physical properties and one for the solution. Figure 3.6 shows an example problem and solution. The problem case is shown by the continuous black line, while the coloured dashed lines represents the nearest neighbours. The attributes are ordered based on correlation to reduce line crossing on the graph. An evaluation of FormuCaseViz by two domain experts resulted in a higher level of confidence in the solutions and the selection of similar cases than in the original text based version. One evaluator commented that *The graphical display is excellent and shows up similarities and differences in a very clear way.*

3.4 Explaining System Confidence

One of the main goals of employing user orientated explanations in decision support systems is to promote user confidence. All the explanation generation techniques mentioned in the previous sections of this chapter have the goal of increasing users' confidence not only for a current classification but also in the underlying reasoning system as a whole. However, if a system is to fully maximise a users' confidence in a classification, then the system should also have an idea of its own confidence, or lack of confidence, in the classification. In fact, in Toulmins argument structure the qualifier element shows the confidence with which the claim is true (Toulmin 1958, Toulmin et al. 1984).

Cheetham and Price have recently emphasised the importance of being able to attach confidence values to predictions in CBR (Cheetham 2000, Cheetham and Price 2004). This has been a research issue since the earliest days of Decision Support Systems research: it is part of the body of research

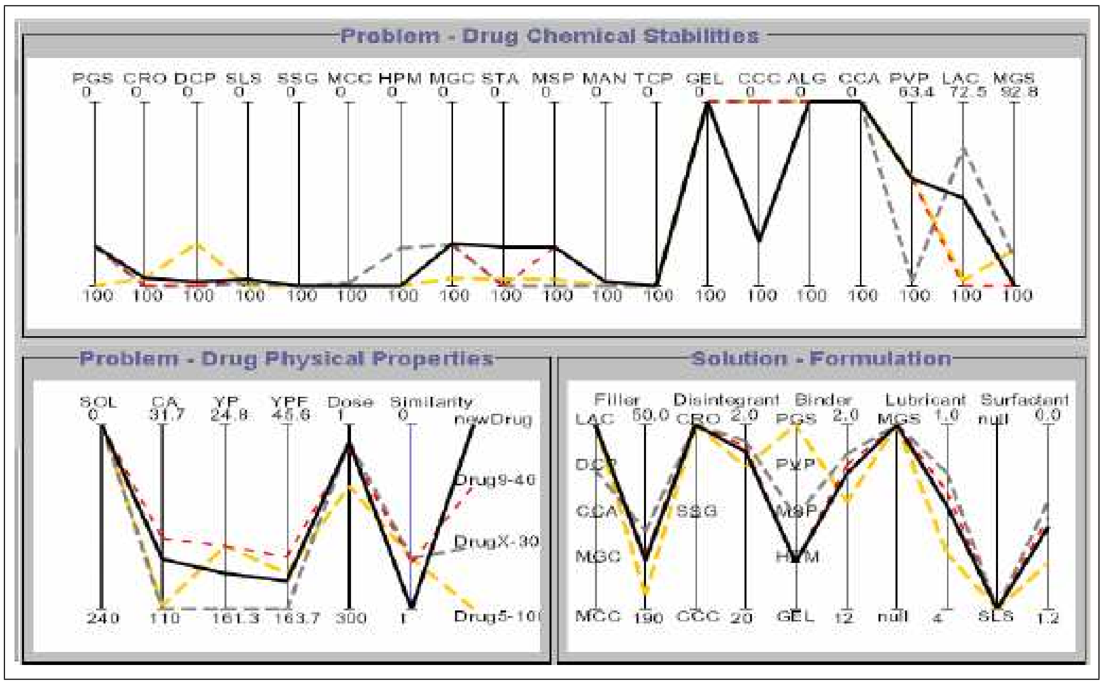

Problem - Drug Chemical Stabilities
PGS CRO DCP SLS SSG MCC HPM MGC STA MSP MAN TCP GEL CCC ALG CCA PVP LAC MGS
Problem - Drug Physical Properties
SOL CA YP YPF Dose Similarity
newDrug
Drug9-40
DrugX-30
Drug5-10
Solution - Formulation
Filler Disintegrant Binder Lubricant Surfactant

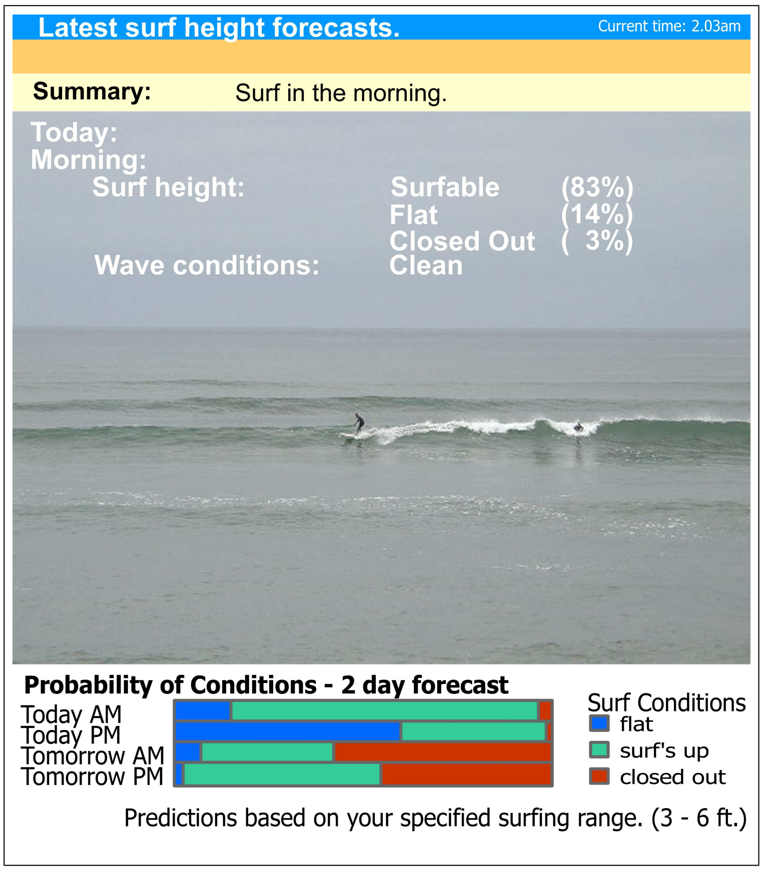
Figure 3.6: Output screen of FormuCaseViz with a problem and proposed solution. Taken from Massie et al. (2004b)

on meta-level knowledge (Lenat et al. 1983, Davis and Buchanan 1985), the view being that it is important for a system to 'know what it knows'. TEIRESIAS is a system in this spirit, it was designed to simply admit its ignorance instead of venturing risky advice (Davis 1982).

More recently, the system SIROCCO from McLaren and Ashley (2001) uses meta-rules to determine the system's confidence. Their system operates in an engineering ethics domain, in which incorrect suggestions could be considered sensitive and damaging. In this system, if any one of the meta-rules are fired then the system considers itself inadequate for the task. Their evaluation of SIROCCO shows that allowing the system to produce 'don't know' results, significantly reduced the number of incorrectly classified cases, with a small trade off, whereby the number of correctly classified cases was only slightly reduced.

Cheetham and Price (2004) describe 12 measures of confidence that can be applicable for a k-NN classifier. Some of these indicators increase with confidence and some decrease. Since no single indicator is capable of producing a robust measure of confidence they explore the use of a decision tree, that is allowed to use all the measures, as a mechanism for aggregating all the available metrics. The authors show that, even using a decision tree to learn a good confidence measure from historic data, it is difficult to avoid the situation where predictions labelled as confident prove to be incorrect. They also emphasise that the confidence estimation mechanism will need to be updated over time as the nature of the problems being solved can change.

Both of these techniques look at binary estimations of confidence or lack of confidence. However sometimes it is useful to have a numeric confidence interval for predictions instead of binary confidence. This is of particular importance in meteorological and financial domains. For example a 20% probability of flooding might be enough to place emergency services on alert, while a 50% probability might be required for preventive measures such as sand bagging to be put into place. Using an ensembles of mixture density networks Carney et al. (2005) have developed a system for predicting surf conditions. Instead of performing a classification the system makes a probabilistic prediction for each of the three possible outcome classes, `Flat`, `Surfable` and `Closed Out`. The predictions are based on the range of surfable wave heights supplied by a user. Figure 3.7 shows an example surf prediction for a surfer with a 3-6ft range of surfable wave heights. In this situation the system is predicting surfable conditions for this surfer with an 83% confidence level. Also included is a 2 day probability forecast for the different classes.


Latest surf height forecasts. Current time: 2.03am

Summary: Surf in the morning.

Today:
Morning:
Surf height: Surfable (83%)
Flat (14%)
Closed Out (3%)
Wave conditions: Clean

Probability of Conditions - 2 day forecast
Today AM
Today PM
Tomorrow AM
Tomorrow PM

Surf Conditions
🟦 flat
🟩 surf's up
🟥 closed out

Predictions based on your specified surfing range. (3 - 6 ft.)

Figure 3.7: Two day probabilistic surf prediction for a surfer based on a surfable wave height range of 3-6ft (Carney et al. 2005).

Regardless of whether a systems confidence is represented as a binary or a probability measure they have a major potential to promote a users confidence. All of the measures can be incorporated into the final explanation that is displayed to a user. For example, in the work of McLaren and Ashley (2001) if the system is not confident an explanation to this extent is a lot better than an incorrect prediction. Alternatively the approach of Cheetham and Price (2004) can be used to convey to the user if a system has a high confidence in a prediction. However generating confidence measures can be difficult. In binary confidence measures, predictions with a high confidence should rarely, if ever, be incorrect. When probability measures are used the percentage accuracy for a particular probability level should correspond to that probability level. For example in the surf domain surf conditions should be flat 14% of the time that it gives a 14% probability of flat conditions.

3.4.1 Active Learning

A research area where classifier confidence plays an important role is Active Learning (Davy 2005). In many domains there is an abundance of unlabelled data. Very often this abundance of unlabelled data results in a considerable effort required to create a training set. This is particularly the case in situations where it is costly to label the data. Active learning allows for a considerable reduction of the dataset that is needed by a learner. This is achieved by allowing the learner to have some control over the cases that are added to the training set. The chosen cases are then presented to an oracle for labelling. In most circumstances the oracle is a domain expert. The cases can either be constructed by the learner or selected from the original unlabelled data.

Active learners are initially supplied with a small training dataset. The cases selected for labelling are the cases that the learner considers to be the most informative, thus speeding up the learning process with less training data.

There are numerous techniques for selecting the most informative cases for labelling. Such techniques include:

Confidence Based: One of the most common approaches for the selection of informative cases is based on the uncertainty of cases in the unlabelled data. The uncertainty of a case is considered to be the lack of confidence the classifier has in its ability to correctly classify the case. The main problem with this approach is the ability to measure the confidence level. Lewis and Gale (1994) suggest that probability estimates from classifiers can be used as a measure of confidence. However work by us, (Delany, Cunningham, Doyle and Zamolotskikh 2005), show that the numeric scores form classifiers such as Naïve Bayes are not well correlated with classification score.

Random Sampling: Although this is the simplest technique it requires the underlying data to be uniformly distributed (Lewis and Gale 1994).

Query By Committee: Learning is performed based on the level of disagreement in the predictions by different committee members (Seung et al. 1992, Freund et al. 1997).

Lookahead Algorithm: Many selection policies are based on selecting cases from regions of uncertainty. The lookahead policy chooses the next example in an attempt to maximise the expected utility of the resulting classifier. Lookahead policies are are of main benefit in domains where a class may occupy more than one region in the problem space.

3.5 Conclusions

In this chapter we opened in Section 3.1 with a review of components of good explanations. This section explored qualities of good explanations along with requirements of generating explanations in decision support systems. Section 3.2 went on to describe approaches to explanation generation in decision trees, the MYCIN system and in black box systems. Section 3.3 moved on from looking at explanation generation in decision support systems in general to focus in on explanation generation in CBR systems. Here a distinction was made between knowledge-light and knowledge-intensive approaches to explanation in CBR, with some examples of knowledge-light explanation systems that have been developed. Finally, Section 3.4, described how prediction confidence can be measured and used as part of the explanation process to increase user confidence. In the next chapter we present a framework that we have developed for generating knowledge-light explanations in CBR systems.

Chapter 4

Explanation Oriented Retrieval

As previously mentioned in Section 3.3 the most common means of producing explanations in knowledge-light CBR systems is to simply display the nearest neighbour, or a certain predefined number of the nearest neighbours, to a given problem case. However, we do not believe that the most similar case will always be the must convincing case available in the case-base to use as an explanation. In this chapter we present a mechanism that we believe retrieves more useful cases for explanation. This mechanism is based on selecting cases that make strong '*a Fortiori*' arguments to support a classification. Section 4.1 explains what an '*a Fortiori*' argument is and Section 4.3 describes our framework for selecting cases to use as '*a fortiori*' arguments. Section 4.2 presents three medical domains that are used in describing the framework and in the evaluations of the framework presented in Chapter 7.

Roth-Berghofer (2004) expresses doubt in displaying the best case as being sufficient for an explanation. To address this we have developed a technique for highlighting supporting and opposing features in the case selected as the best case for explaining a classification. Techniques for emphasising relevant features based on a classification are also used in the HYPO (Ashley 1987, 1991) and CATO (Ashley and Aleven 1997) systems. It is hoped that communicating these features in free text along with the selected case will help improve users confidence in a system.

4.1 A Fortiori Arguments

Most parents are familiar with the use of *a fortiori*[1] arguments by children. *A fortiori* arguments are used to invoke compelling examples in support of a position. Let us consider an example of a child using an *a fortiori* argument to plea their case to see the latest Harry Potter movie.

Figure 4.1 shows an example of a child called Mark (the triangle) who wants to see the latest Harry Potter movie. The circles represent the children who have seen the movie and the squares

[1] "*a fortiori* - adv. for similar but more convincing reasons: if Britain cannot afford a space program, then, a fortiori, neither can India." - Collins English Dictionary

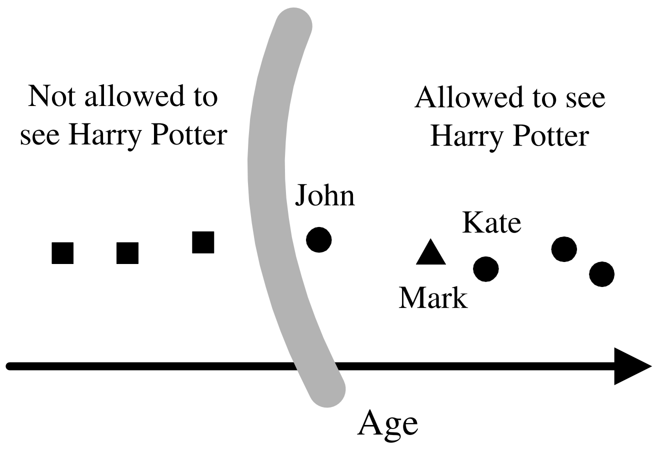


Figure 4.1: Using *a fortiori* arguments

the children who have not. Mark knows that Kate is the closest in age to him and she has seen the movie. But Mark knows that the older you are the more likely you are to be allowed to see the movie. If Mark were to use Kate as an argument to convince his parents to let him to the movie, there is a possibility that Mark's parents can argue that Mark is still a little too young to see the movie. However Mark knows that if he uses John who is younger than him as his argument to see the movie, he has a stronger case.

The above example shows that the most similar situation is not always the most convincing situation to support an argument. In the above situation, picking a child between himself and the perceived decision boundary (Mark doesn't know the exact cutoff age to be allowed see the movie, but knows the younger you are the less likely you are to be allowed to see it) increased the strength of his argument. Similar to the above example, we believe that when supporting a classification in CBR, the most similar neighbour to a query case is not necessarily the most convincing case to support the classification. For instance, if a decision is being made on whether to discharge a sick 12 month old baby from hospital, a similar example with a 9 month old baby that was discharged is more compelling than one with a 13 month old baby (based on the notion that younger babies are more likely to be kept in).

4.2 Example Problems

Our work on selecting more suitable cases in CBR systems predominately focused on medical domains. The first of these domains is the Breathalyser domain which is used to predict if a person who has consumed alcohol is under or over the legal limit to be allowed to drive in Ireland. The second domain contains data of patients with Bronchiolitis, which is a viral infection that affects young children up to two years of age. In this domain the CBR system attempts to recommend if a patient with bronchiolitis presented to an Emergency Department should be discharged or admitted. The final domain we used contains data on the suitability of diabetic patients to participate in an e-Clinic, a situation where more stable patient are more suitable for participation. This section will provide information about these three domains, along with how certain features have an influence on the classification.

4.2.1 Breathalyser Domain

When an alcoholic beverage is consumed it pases down the esophagus through the stomach and into the small intestine. Approximately 20% of alcohol is absorbed into the bloodstream through the mucous membrane (lining of the stomach) with the remaining 80% entering the blood stream through the walls of the small intestine. All blood from the stomach and intestines first goes through the liver before circulating around the whole body. So, the highest concentration of alcohol is in the blood flowing through the liver.

Liver cells contain enzymes which metabolise alcohol. The enzymes break down alcohol into other chemicals which in turn are then broken down into water and carbon dioxide. These are then passed out in urine and through the lungs. The liver cells can metabolise only a certain amount of alcohol per hour. A general rule of thumb is that it takes approximately one hour to metabolise one unit[2] of alcohol, but this rate differs slightly for everyone. So, if you drink alcohol faster than your liver can deal with it, the excess alcohol travels through your body where it accumulates in the blood and body tissues (including the brain) until it can be metabolised.

It is difficult to measure directly the amount of alcohol accumulated in the brain. As a result, blood alcohol levels were first used to assess the concentration of alcohol in a person's brain tissue. Currently in Ireland a driver is considered to be over the drink driving limit if they contain over 80 milligrams per 100 millilitres of blood. Although there are advantages when testing with blood to determine alcohol concentrations in the human body, the sample collection process can be viewed as both invasive, painful and the analysis process is both time consuming and costly.

Ethanol is volatile and as a result, an amount of alcohol, in proportion to the concentration in the blood, transfers from the blood into the alveolar air sacs in the lungs. This occurs in much the same way that carbon dioxide leaves the alveolar blood and enters the lungs for exhalation from the body. As a result, it is possible to analyse an alveolar breath sample, determine the breath alcohol concentration (BrAC) and predict with a high degree of accuracy, the blood alcohol concentration at that same point in time. Breath testing instruments were manufactured to capture a sample for analysis. Breath analysis has since evolved into a technology that offers a low cost, highly accurate, rapid analysis of a breath sample that is simple and painless to collect. Currently in Ireland a

[2]The alcohol content of drinks is measured in units. A unit is a measure of the amount of pure alcohol in a drink. In this domain a unit is considered to be the UK measure of a unit which is eight grams of pure alcohol.

sample of breath containing over 35 micrograms per 100 millilitres of breath, is considered to be over the legal drink driving limit.

There are numerous factors that influence the blood alcohol content in humans. Some of the factors that affect blood alcohol content include; age, speed of drinking, type of drinks consumed, body metabolism, percent of body that is blood, percent of body that is fat and height.

The breathalyser dataset (Cunningham, Doyle and Loughrey 2003, Doyle et al. 2004) is a real world dataset that was collected between December 2002 and January 2005 to predict blood alcohol levels based on a number of features. The dataset contains 127 cases and 5 features, The features used for determining blood alcohol levels in the Breathalyser dataset are:

Weight The more a person weighs the higher the water content in their body. As alcohol readily dilutes with water, the extra water in a heavier person has the effect of diluting the alcohol content, resulting in a lower blood alcohol level value.

Gender The process of absorbtion and metabolism of alcohol differs between men and women. Women tend to have a higher blood alcohol content after consuming the same amount of alcohol as men. The difference in blood alcohol levels between women and men has been mainly attributed to women having a smaller water content than men. This can be likened to dropping the same amount of alcohol into a smaller pail of water. An additional factor contributing to the difference in blood alcohol level is that women often contain a lower activity of the alcohol metabolising enzyme ADD in the stomach, causing a larger proportion of the ingested alcohol to reach the blood.

Meal The rate at which alcohol is absorbed depends on how quickly the alcohol can pass into the intestine. The higher the amount of food consumed, the more time this will require and the longer the process of absorption will take (Wallgren 1970, Fraser et al. 1995). One study found that subjects who drank alcohol after a meal absorbed alcohol about three times more slowly than when they consumed alcohol on an empty stomach (Jones and Jönsson 1994).

Duration of Drinking As the liver can remove approximately one unit of alcohol every hour, the duration of drinking has an influence on the blood alcohol content.

Units Consumed Needless to say this is the most important factor in predicting blood alcohol level. The more units of alcohol a person consumes, the higher their blood alcohol content.

4.2.2 Bronchiolitis Domain

Bronchioles are the non-cartilaginous tubules in the pulmonary tree and are less than 1mm in diameter (Online-Encyclopedia). Bronchiolitis means inflammation of these bronchioles. In current usage by Family Practitioners, Pediatricians and Emergency Physicians 'Bronchiolitis' generally refers to a viral illness leading to virally mediated inflammation of the bronchioles in children less than two years of age (Baker and Ruddy 2000). The effect of this inflammation is to narrow the calibre of the bronchioles. This in turn leads to decreased airflow to and from the alveoli (Baker and Ruddy 2000, Orenstein 2000).

Bronchiolitis is common, being the leading cause of hospital admission for this age group in many hospitals (Orenstein 2000). The same viruses that cause this presentation in infants cause little more than a bad cold in older children and adults.

The complications of bronchiolitis arise from airway obstruction, hypoxia, obligate nasal breathing, and the infants' need to bottle or breast feed. The proportion of lung comprised of smaller airways and the size of these directly relates to the size and age of the child. Younger children are more severely affected. Infants less than two months of age, those with co-morbidities, and premature infants are most at risk of respiratory failure and death (Baker and Ruddy 2000).

The clinical picture is of an infant or toddler with runny nose, wheezing, and laboured breathing who may be dehydrated. The diagnosis is generally straightforward. Which children should be admitted and which should be discharged is controversial. The decision to admit or discharge is termed disposition. It is one of the most important decisions an Emergency Physician makes. For many patients the disposition is clear-cut. In marginal cases, being able to draw on the collective outcome experience of the entire Emergency Department staff would be desirable. In other cases it could serve as a safety device by flagging anomalous dispositions in real time.

The Bronchiolitis Dataset is a dataset of 318 patients up to the age of 18 months that were suffering from Bronchiolitis. The data was collected by the Emergency Department of Kern Medical Center[3]. The Emergency Department is based in a county hospital in Kern County, California and has an university affiliated emergency medicine residency program. The department serves a mixed urban rural population and has annual census of approximately 46,000 patients.

For each patient enrolled in the study approximately 200 features were recorded[4]. These features are broken up into the following categories

Review of Systems: Contains information about how long the patient has been sick, changes in behaviour and eating habits.

Past Medical, Family and Social History: Investigates birth conditions of the patient, immunisations and medicines received, history of asthma in the family and the presence of smokers in the household.

Physical Examination: Contains records of vitals such as hearth rate, temperature, respiration rate, oxygen saturation, reflexes and lung sounds before any treatment is given to the patient.

Lab and X-Ray: Contains results from laboratory tests and/or x-rays performed on the patient.

[3]This research into bronchiolitis was approved by the Kern Medical Center's Institutional Review Board. Informed consent was obtained from the parents or legal guardians of all subjects.

[4]Appendix B contains the template used for collecting data for each patient

Hospital Course: Contains records of some of the vitals such as respiration, heart rate, temperature and oxygen saturation after treatment has been given to the patient. This section also contains information on the disposition of the child.

All patients attending the emergency department with bronchiolitis are seen by either a mid level provider (e.g. Nurse Practitioner) or a Resident Physician. All patients seen by residents are also seen by faculty Emergency Physicians. All faculty are board certified in Emergency Medicine. The disposition of a patient has three possible values; `Discharge`, `Admit` and `Prolonged Emergency Department Observation`. If a patient is discharged a three day telephone follow-up is performed to ensure the patient was not subsequently admitted to hospital.

Using a forward sequential search algorithm (Whitney 1971, Aha and Bankert 1994) the feature space was reduced down to twelve features for use in a CBR system. Using these features a hold one out accuracy of 72% was achievable on the original training data of 228 cases collected up until the end of January 2005. These features are:

- Age
- Birth Type
- Smoking Mother
- Hydration before treatment
- O2 Saturation before treatment
- Retraction Severity before treatment
- Heart Rate after treatment
- Overall increase in Work of Breathing after treatment
- Oxygen Saturation under 92 after treatment
- Respiratory Rate over 60 after treatment
- Temperature over 100.4 after treatment
- Work of Breathing after treatment

4.2.3 e-Clinic Domain

The final domain used is the e-Clinic domain. This is a domain for assessing suitability for participation in a diabetes e-Clinic. Patients with stable diabetes can participate in the e-Clinic whereas other patients need closer monitoring. The decision support system acts as a triage process that assesses whether the patient is stable or not. In this domain we have 300 cases collected in St. James' Hospital in Dublin by Dr. Yusof Rahman (Rahman et al. 2004).

Some of the factors for deciding if a patient is stable and suitable for the e-Clinic include: the type of diabetes they have, the treatment they are on, if they have any complications and their HbA1c level (see below for details). For example, if a patient has any complications, if they have type II diabetes or are treated by injecting insulin instead of being treated by oral hypoglycaemic agents (OHA) they would not be considered suitable for the e-Clinic. The HbA1c feature is a test that can provide an average rating for blood sugar levels over the three month period prior to testing. The lower the value for HbA1c the more likely a patient is to be stable enough to remain in the e-Clinic. However if the value for HbA1c is greater than 7- 7.5 % the patient is unlikely to be suitable for the e-Clinic.

4.3 Explanation Utility Framework

As already mentioned, many CBR systems perform explanations by simply displaying the most similar case to the end user. However, we believe that the most similar case will not necessarily always be the most convincing case to use as an explanation. Based on the '*a fortiori*' argument (Section 4.1), we believe that, a case lying between the target case and a decision boundary results in a more convincing explanation than a case that lies on the opposite side of the target case.

For example, consider the two feature problem in Figure 4.2 and the justification for the classification of query case Q. There must be a decision boundary in the solution space, however the exact location of this boundary is not known. The boundary must lie between the nearest neighbour NN and the nearest unlike neighbour NUN. Typically users will have some intuition about the decision boundary and will be less comfortable with the NN as a justification for the classification of Q if Q is considered to be closer to the decision boundary than NN. The case EC would be a more convincing example because it is more marginal. Overall the selection of a case from the shaded region in Figure 4.2 would provide a more convincing argument than a case located outside the shaded region.

We have developed a framework that attempts to select a case that lies between the target case and a decision boundary for use as an explanation. Unless otherwise specified we will refer to the case that is selected by the framework as the *Explanation Case*. The framework selects more suitable cases by using a set of explanation utility measures that are dependent on the classification.

4.3.1 Explanation Utility Measures

We believe that in order to select a suitable case to support a recommendation, the actual recommendation needs to be taken into account. In other words the metrics used in the retrieval of such a case would vary for different recommendation. In fact Bergmann et al. (2001) mentions that utility functions used for retrieval are often given by a-posteriori-criteria.

Our framework works by first of all performing a classification using a case retrieval net as

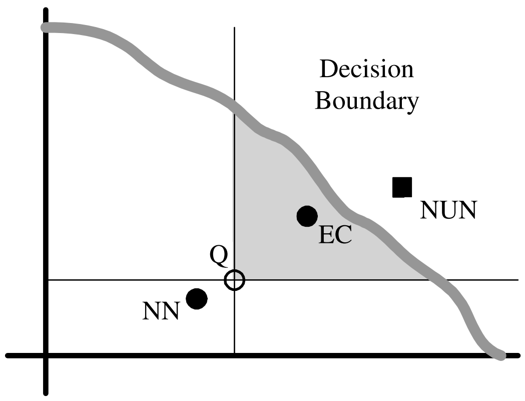


Figure 4.2: A nearest neighbour example where case EC would be a better explanation for the decision on query case Q than the nearest neighbour NN; case NUN is the nearest unlike neighbour.

described in Section 2.6. Once a classification is performed, the most similar neighbours to the query case are re-ranked in order of their explanation utility to support the classification. This re-ranking is performed using a utility measure as shown in Equation 4.1.

$$Util(Q, X, C) = \sum_{f \in F} w_f \xi(q_f, x_f, c) \tag{4.1}$$

where $\xi()$ measures the contribution to explanation utility from feature *f*. The utility measure closely resembles the similarity measure used for performing the initial nearest neighbour classification Equation 2.1 except that the $\xi()$ functions will be asymmetric compared with the corresponding $\sigma()$ functions and will depend on the class label c.

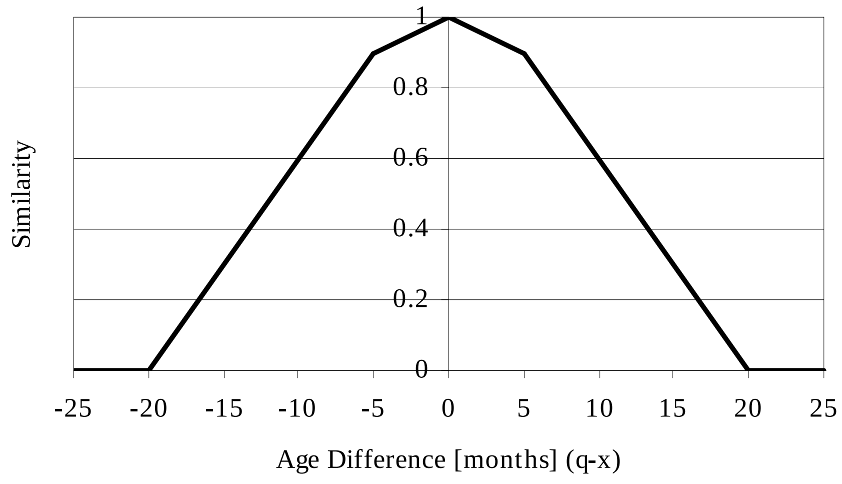


Figure 4.3: Similarity Graph for the feature `Age`

In Section 2.5.2 we presented an example of how difference graphs can be used for calculating similarity. Figure 4.3 shows a possible similarity graph that could be used for comparing the age of infants. In this situation the graph shows a high similarity for children that are aged within five months of each other. However once the age gap between two cases increases beyond five months the similarity between the two cases drops off more rapidly.

The graph used for calculating similarity can be used as a basis for generating utility measures. To generate the utility measures we need to utilise cases that support the recommendation. For example if recommending that a patient should be discharged a case that is younger than the patient and was also discharged should have a higher utility than a case that is older than the patient (based on the notion that younger babies are more likely to be kept in). Alternatively when recommending a patient should be admitted, cases that are older and also admitted should have the highest utility. Figure 4.4 shows example explanation utility graphs for both admit and discharge recommendations that captures this.

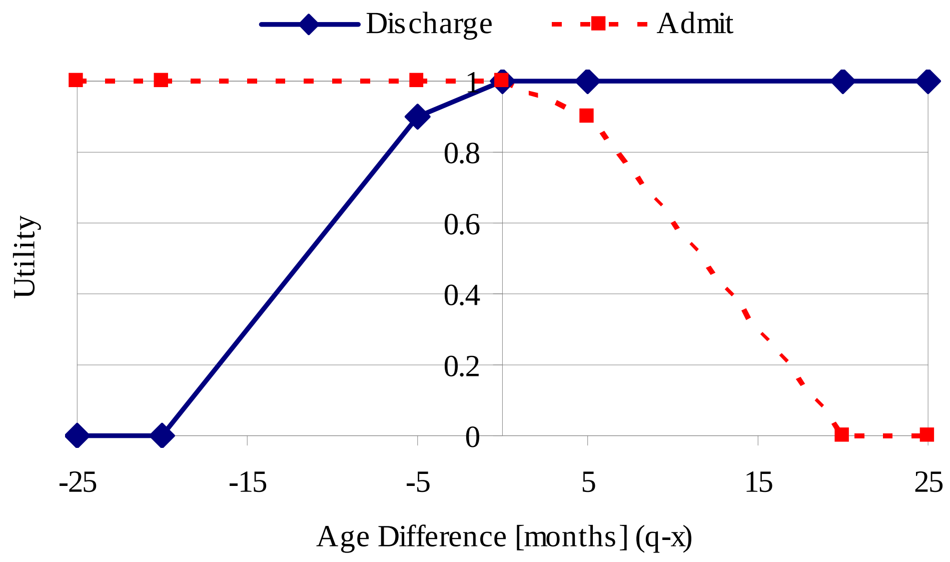


Figure 4.4: Explanation Utility for the feature Age

For example, consider a recommendation to discharge a 15 month old baby. A 20 month old case (i.e. an age difference of -5 months) that was also discharged would have an explanation utility of 0.9. However a 10 month old case (i.e. an age difference of +5 months) would have an explanation utility of 1.0. Here we can see that as the positive age difference increases the utility decreases. Alternatively, regardless of the age difference for younger cases the utility remains at a maximum of 1.0.

This method of creating explanation utility measures leaves us with one problem. In the case of `Discharge`, all examples with a positive or zero difference have an explanation utility of 1 in this dimension. This implies that the explanation utility measure is indifferent over a large range of difference values. This results in the order of the cases stored in the case base having an impact on the cases returned for explanation. For example, the three cases, C_1, C_2 and C_3 shown in Table 4.1, all have the same utility score as they all lie in the direction of the decision boundary from the query case, based on a `discharge` recommendation. As they all have the same utility score the selection of an explanation case will be based on the order of these cases, which is influenced by their original ordering in the case-base. Therefore in this situation C_1 could be selected purely because it is the first in the list of cases. A further problem with this indifference is that it also ignores the fact that a case that is three months younger could possibly be better for explaining a `discharge` recommendation than a case that is only two month younger.

Table 4.1: Case Ordering Problem

	Query Case	C_1	C_2	C_3
Heart Rate	130	150	140	160
Temp	98.5	99.2	99.9	99.5
Age	12	10	11	9

To address both of these problems the explanation utility measure is adjusted so that maximum utility is not returned at equality. There are many possible shapes that the utility measure can take on. We believe that the best shape is feature, domain and possibly user specific. Figure 4.5 shows one possible explanation utility measure for the feature `age` for a recommendation to discharge. In this situation the utility of a case older than the query case (negative age difference) is still lower than the utility of a case that is younger than the query case (positive age difference). However, this modification will now favour outlying cases over more similar cases. For example if trying to argue that a 7 month old patient should be discharged a similar case aged 4 months that was also discharged might be considered to be more useful as an argument in support of the discharge than a discharged case that is 6 months of age. Using the explanation utility measure shown in figure. 4.5 we can see that the 4 month old patient will receive a higher utility score than the 6 month old.

Although this adjustment corrects the problems associated with the utility measures shown in 4.4, this type of utility measure can still cause some problems. The most prevalent of these problems is that there is a high repetition in the cases selected as explanation cases. This problem is caused by the fact that more marginal cases are selected using this type of utility measure. For example consider the 4 query cases Q_1 - Q_4 in Fig. 4.6. In this situation there are 5 possible cases that can be used as explanation cases EC_1 - EC_5. For each of the query cases there are numerous suitable cases to select as an *a fortiori* argument. However, using the explanation utility measures described in Figure 4.5, EC_1 will always be selected as the explanation case.

Repetitively selecting the same case as an explanation case could cause users to lose confidence in the system. Furthermore a users confidence in the system can be further diminished if the user

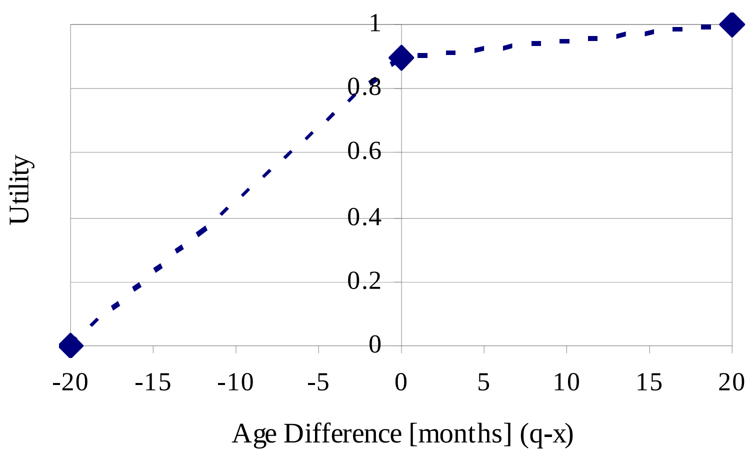


Figure 4.5: Utilising more marginal cases for a discharge recommendation

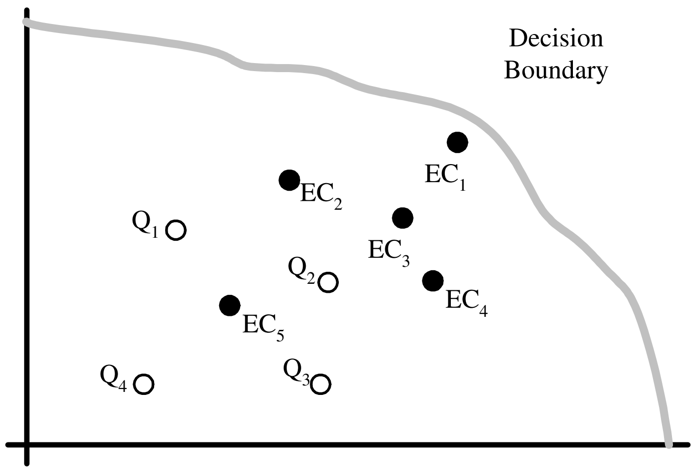


Figure 4.6: When more marginal cases are selected as explanation cases, repetition of cases can occur. In this situation EC_1 will be selected for all 4 query cases Q_1 - Q_4

is not entirely convinced that the classification of a repeatedly used explanation case is correct. By selecting more marginal cases as explanation cases there is a higher risk of selecting cases that the user is not entirely convinced is correctly classified. Another problem that can occur using this shape of utility measure is that the explanation cases that are selected can be heavily affected by one or two outlying features.

In order to select less marginal cases the explanation utility measure needs to be adjusted to favour cases that are more similar to the query case. Figure 4.7 shows how the `age` explanation utility measure to support discharge can be adjusted so that utility is maximised for younger cases that are more similar to the query case. In this example a patient 5 months younger than the query case has maximum utility when trying to support a `discharge` disposition. Once the age difference increases to more than 5 months the utility starts to fall off.

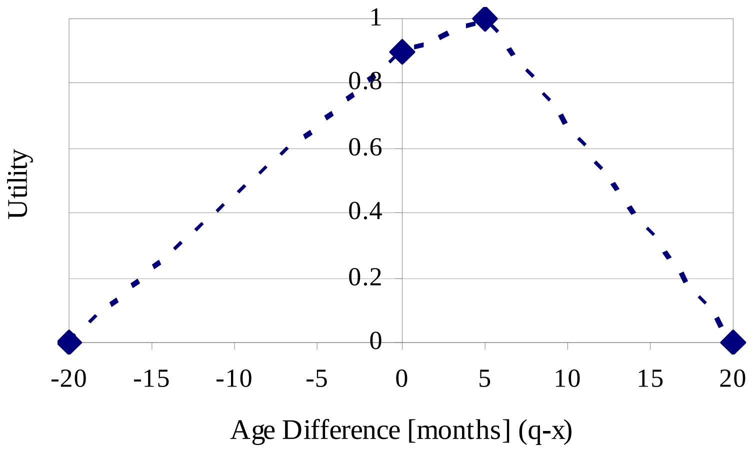


Figure 4.7: Discharge utility measure for selecting more similar/less marginal cases

In Figure 4.2 the shaded region showed the optimum area for selecting cases to use as *a fortiori* arguments. Using explanation utility measures similar to the one just described in Figure 4.7, cases closer to the query case in this optimum area are preferred over more distant cases. This preference is displayed in Figure 4.8 where the darker shaded regions are considered more suitable than the lighter regions. In this situation EC_1 would be selected as the explanation case rather than the more marginal case EC_2.

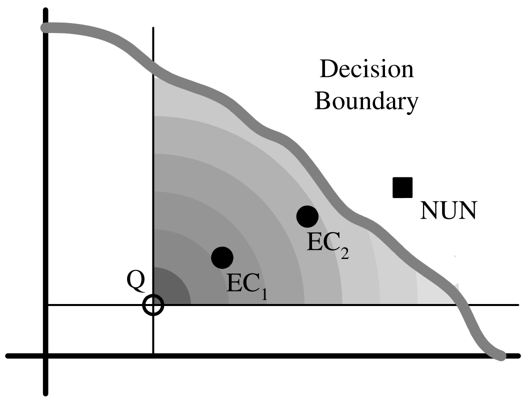


Figure 4.8: When using explanation utility measures that utilise less marginal cases, cases that are more similar to the query case receive a higher utility. In this situation a case in darker shaded region will receive a higher utility, hence EC_1 will have a higher utility than EC_2

We have just looked at two methods of improving the explanation utility of the original measure shown in Figure 4.4. These improvements consisted of two extremes; The first utilising more marginal *a fortiori* cases (Figure 4.5) while the second utilised more similar *a fortiori* cases to the query case (Figure 4.7). Although these extreme measures may be suitable for some features, they may also be too restrictive for other features. There are an infinite number of possible measures that lie between these two extremities. An example of a possible measure is shown in Figure 4.9. In this situation similar cases are preferred slightly over more marginal cases without marginal cases being penalised as much as they are using explanation utility measures similar to that shown in Figure 4.7.

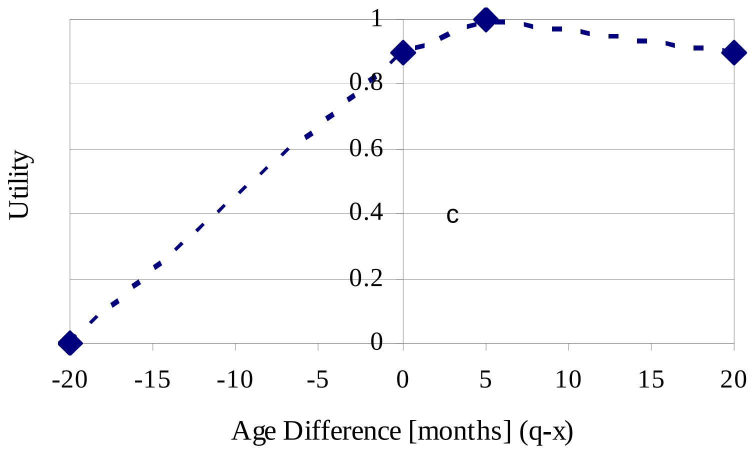


Figure 4.9: Alternative Explanation Utility Measures to Discharge for the feature Age

4.3.2 Explanation Utility Measures for Symbolic Features

So far the explanation utility measures described have focused only on numeric features. When it is possible to define a method for calculating the difference between two symbolic values, utility graphs can also be used for calculating explanation utility. In situations where symbolic features can be logically ordered, the difference could be the relative difference between the two positions in the ordered list of possible values. An example of a feature that can be ordered is the *Meal* feature in the breathalyser domain. This feature has four possible values: `None`, `Snack`, `Lunch` and `Full`. The explanation utility measure for the feature `meal` when the classification is over the limit is shown in Figure 4.10. In this situation a case that has consumed more food than the query case will have a higher utility than a case that has consumed less food. This is based on the principle that the more you eat, the slower the rate of absorption of alcohol into the blood stream. In this situation if the query case had only consumed a `snack`, the utility for a case that had consumed nothing would result in a utility of approximately 0.5 (This is based on a difference of +1 as `snack` is the 2nd item in the ordered list, while `none` is the first item in the list). Alternatively the utility of a case that was `full` would be 0.9 (This is based on a difference of -2, as `full` is the 4th item in the list).

Using a similar approach utility measures can be developed for features that contain boolean values (i.e. features that can only have `true` or `false` values). In this situation the values are

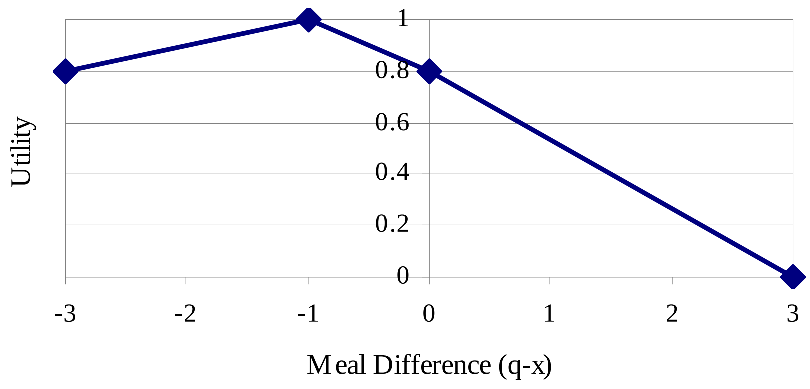


Figure 4.10: Explanation Utility measure for the feature meal for an over the limit classification in the Breathalyser domain

considered to be in an ordered list where the first value in the list is `true` and the second value is `false`. Figure 4.11 shows a possible utility measure for the feature `High Temperature` for a recommendation to discharge. If the temperature for the query case is low, i.e the feature is false, the utility for a case that has a high temperature would be 1 (based on a difference of +1 between the two relative positions).

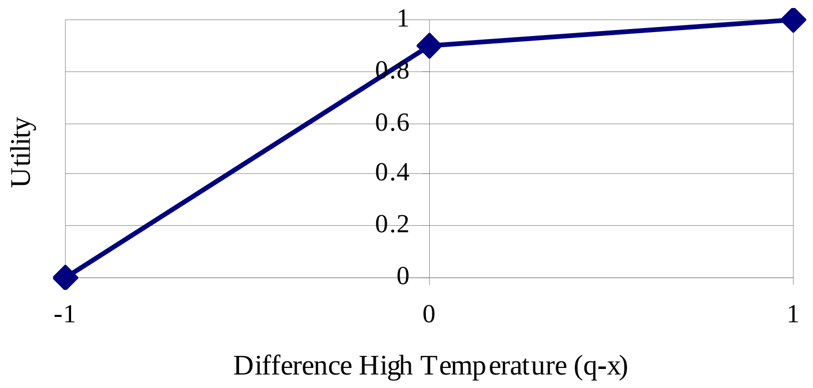


Figure 4.11: Explanation Utility measure for the feature `High Temperature` for a `discharge` recommendation.

4.3.3 Selecting Utility Measures

In Section 4.3.1 we demonstrated a number of possible utility metrics that can be used for selecting a case to use as an explanation for a given recommendation. The selection of an appropriate shape for a particular utility measure can be performed with the assistance of a domain expert. Although this may go against the general ethos of knowledge-light systems, the amount of input that is needed from a domain expert is minimal. The only information that is needed is the general direction of the decision boundary for a particular feature given a recommendation and whether marginal or more similar cases would be preferred. So for example for the feature `age` and a recommendation to `discharge` the information given by a domain expert might be that the decision boundary is in the direction of younger cases and that marginal cases are preferred. From this information and knowing the range of possible ages, an engineer developing the system can define n explanation utility measure similar to that shown in Figure 4.9.

The first problem that may occur with this approach is that once people start using the system, they might actually prefer less marginal cases for the feature `age`. There are two ways of dealing with this approach. The first way is for the explanation utility metrics to be updated manually. As the metrics are not hard-coded (Section 6.2.1) this update is a minor change without recoding. Another solution is that if a user is not happy with the explanation case that is presented they can see a list of possible explanation cases. From this list they can then select the one that they prefer as an explanation. Based on this selection the utility metrics can be automatically updated to reflect the users preference. In this situation the advice from the expert is simply used to bootstrap the utility measures. This approach can be useful in situations where different users may have different preferences. In this situation a new user could be given the original explanation utility measures, devised by the domain expert, and they will then be updated according to the preference of the user.

Another problem that may occur is that the expert may not know the direction of the decision boundary for a particular feature. As CBR is often used in weak theory domains this is a situation that needs to be addressed. Although Bridge and Cummins (2005) have developed a technique to analyse neighbouring cases to approximate the direction of the decision boundary, we believe that this approach may have two potential weaknesses. The first is that, although they have some clever techniques for dealing with noisy cases, it is possible that noisy cases can cause incorrect assumptions to be made about the direction of the decision boundary. The most important point to note is that, if for a given classification, the expert does not know the correct direction of the decision boundary for a particular feature, then the feature cannot contain any useful explanation utility for the generation of an *a fortiori* argument based on the perceived direction of the decision boundary. Also their technique does not have any method for controlling wether more or less marginal cases are preferred for certain features.

Just because a feature may not contain any useful explanation utility for the generation of an *a fortiori* argument based on the perceived direction of the decision boundary, does not mean that the feature has no use in the selection of explanation cases. It is still important for such features to have a high level of similarity between a target case and an explanation case. For this reason in situations where a utility measure cannot be generated for a feature, the original similarity measure used for retrieval is used for utility. In other words in this situation the utility of a feature is based on similarity and is independent of the predicted class.

4.4 Highlighting Features of Retrieved Cases

As mentioned by McSherry (2004) it is important for the user to make meaningful comparisons between the feature values in the query case and explanation cases. Using explanation utility measures we have developed a technique for selecting features to highlight to the user. The features that are highlighted could be both features that support and features that oppose the recommendation. Our approach is to compare the explanation utility of a feature based on the recommendation to the utility of the feature using the explanation utility measures of alternative recommendations.

In a two class problem this can be simplified to an inequality. When the following inequality holds true we consider that the feature value in the explanation case, x_f, supports the recommendation:

$$\xi(q_f, x_f, c_p) > \xi(q_f, x_f, c_n) \tag{4.2}$$

where q_f is the corresponding feature in the query case, c_p is the classification and c_n is the opposite classification.

For example consider a recommendation to discharge a patient where the explanation case is 5 months younger than the query case. Using the explanation utility measures shown in Figure 4.12 we can see that the utility associated with the feature `age` is 1.0. Alternatively if the `Admit` utility measures were used the utility would be reduced to 0.675. In this situation we can see that Inequality 4.2 holds true, hence the feature age in the explanation case is a feature that supports the recommendation. Also in this situation we can see that the difference in utility between the two utility measures is 0.325 (1.0 - 0.675). Using this difference in utility value supporting features can be ranked in order of level of support, where the greater the difference the more supportive the feature is of the recommendation.

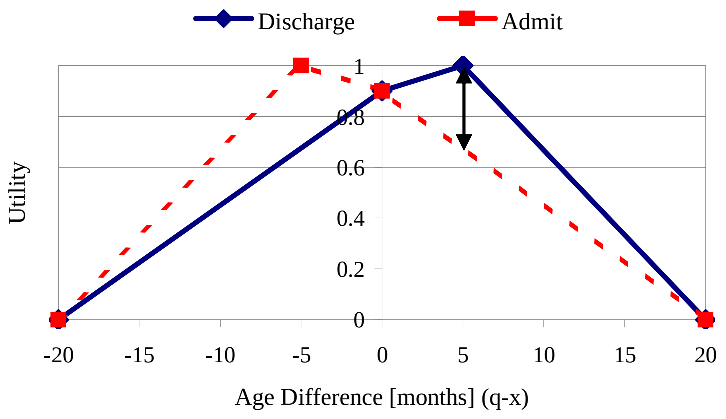


Figure 4.12: Supportive Utility Feature

In a similar manner a feature does not support the recommendation when the opposite inequality holds:

$$\xi(q_f, x_f, c_p) < \xi(q_f, x_f, c_n) \tag{4.3}$$

Staying with a recommendation to `discharge` a patient, consider an explanation case that has a heart rate of 20bpm lower than the query case. Using the utility measures for `Heart Rate` difference shown in Figure 4.13, it can be seen that the `discharge` utility measure has a lower utility than the `admit` utility measures. Therefore in this situation Inequality 4.3 holds true suggesting that this feature opposes the recommendation to `discharge`. Similar to supporting features, the greater the difference between the different utility values the more it opposes the recommendation.

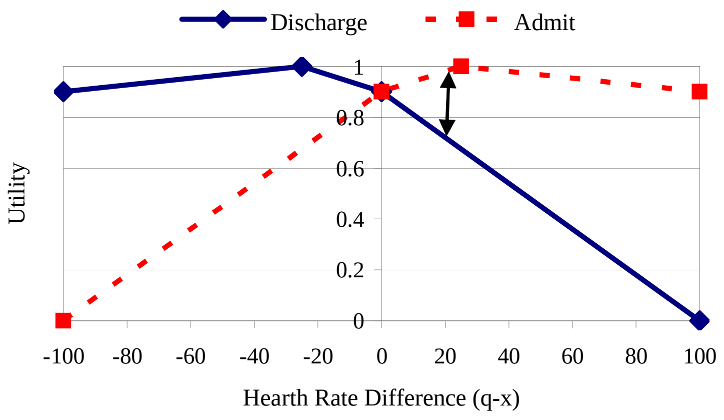


Figure 4.13: Non-Supportive Utility Feature

4.5 Explanation for Classification

In some of our reviews we received some suggestions that the explanation utility framework could be considered as a classification mechanism and should be used to perform the classification as well. We investigated the possibility of using the explanation utility measure for performing the entire retrieval process, instead of using it to simply re-rank the highest neighbours based on the classification. This is not completely straightforward as the utility metric is class dependent. This can be addressed by using the utility metrics to rank the entire case-base twice, once for each outcome class. The utility score for the k nearest neighbours for each class is summed and the class with the highest score is returned as the classification.

In order to test the effectiveness of this approach to classification, a leave-one-out validation was performed comparing this utility based classification with the standard similarity based process. The results of this comparison on three datasets is show in Table 4.2. From this table we can see a clear reduction in accuracy when using this explanation utility based classification approach to a standard nearest neighbour approach.

Table 4.2: Comparison of accuracies using explanation utility for classification compared to similarity measures.

	Nearest Neighbour	Explanation Case
Breathalyser	77	74
Bronchiolitis (Original Utility)	77	40
Bronchiolitis (Updated Utility)	72	47
e-Clinic	96	83

This shows that the requirements for classification accuracy and explanation are different and supports Wick and Thompsons (1992) idea of having an explanation utility framework that is separate from the similarity mechanism used for classification.

4.6 Conclusions

This chapter is based on the idea that, in case-based explanation, the nearest neighbours may not be the best cases to explain classifications. In classification problems there will normally be a notion of a decision surface and cases that are closer to the decision surface should be more compelling as explanations. These more compelling explanations are referred to as '*a fortiori*' arguments which are described in more detail in Section 4.1. In Section 4.3 we introduced an explanation utility framework that formalises this idea and show how it can be used to select explanation cases in knowledge-light CBR systems. In Section 4.4 we describe a technique for selecting supporting and opposing features of the selected explanation case for highlighting. Finally in Section 4.5 we showed that the use of explanation utility measures are not suitable for use in classification. In the next chapter we present a technique for estimating a binary confidence level in a classification. Using this technique and the explanation utility framework we have developed a real time decision support system for the bronchiolitis domain. This system is presented in Chapter 6.

Chapter 5

Assessing Confidence

In Section 3.4 we discussed how the level of confidence a system has in a classification can be used in explanations. In this chapter we introduce techniques for estimating binary confidence levels. Section 5.1 describes a number of possible confidence measures, that are mainly based on similarity metrics, and how they can be configured to produce binary estimations of confidence (Section 5.2). A major problem with these measures individually is that they often result in highly confident classifications occurring a low proportion of the time. Section 5.3 looks at how these individual measures can be aggregated into an ensemble to increase the occurrence of high confidence classifications.

5.1 Confidence Measures

This section describes a number of confidence measures that could be used to assess confidence in CBR systems. We concentrate on using measures appropriate for a k-NN classifier. The k-NN measures that we propose perform some calculation on a ranked list of neighbours of a target case. Many of these measures are similar to or inspired by the measures used by the case-based spam filter called ECUE (Email Classification Using Examples) (Delany, Cunningham, Doyle and Zamolotskikh 2005). The objective of the k-NN measures is to identify those cases that are 'close' (i.e. with high similarity) to cases of the same class as the target case and are 'far' (i.e. low similarity) from cases of a different class. The closer a target case is to cases of a different class, the higher the chance that the target case is lying near or at the decision boundary. Whereas the closer a case is to other cases of the same class, the higher the likelihood that it is further from the decision boundary.

For each k-NN confidence measure discussed in this section the same process occurs. Each target case is classified by the CBR system and a ranked list of neighbours of the target case is retrieved. This list of neighbours is a list of all the cases in the case-base ordered by distance from the target case. Those cases with classification equal to that of the target case are considered to be *like* cases, while cases with other classifications are considered to be *unlike* cases. The measures can use

- the distance between a case and its nearest neighbours (let $NN_i(t)$ denote the ith nearest neighbour of case t),
- the distance between the target case t and its nearest like neighbours (let $NLN_i(t)$ denote the ith nearest *like* neighbour to case t),
- the distance between a case and its nearest unlike neighbours (let $NUN_i(t)$ denote the ith nearest *unlike* neighbour to case t) and/or
- the distance between a case and its explanation cases (let $EC_i(t)$ denote the ith nearest explanation case to case t).

The number of neighbours used in each measure is adjustable and is independent of the number of neighbours used in the initial classification. All measures are constructed to produce a high score to indicate high confidence and a low score to indicate low confidence.

5.1.1 Average Nearest Unlike Neighbour Index

The Average Nearest Unlike Neighbour Index (Avg NUN Index) is a measure of how close the first k NUNs are to the target case t as given in Equation 5.1.

$$AvgNUNIndex(t,k) = \frac{\sum_{i=1}^{k} IndexOfNUN_i(t)}{k} \tag{5.1}$$

where $IndexOfNUN_i(t)$ is the index of the ith nearest unlike neighbour of target case t, the index being the ordinal ranking of the case in the list of NNs.

This is illustrated in Figure 5.1 where NLNs are represented by circles, NUNs are represented by stars and target cases are represented by triangles. For $k = 1$, the index of the first NUN to target case T_1 is 5 whereas the index of the first NUN to target case T_2 is 2, indicating higher confidence in the classification of T_1 than T_2.

5.1.2 Similarity Ratio

The Similarity Ratio calculates the ratio of the similarity between the target case t and its k NLNs to the similarity between the target case and its k NUNs, as given in Equation 5.2.

$$SimRatio(t,k) = \frac{\sum_{i=1}^{k} Sim(t, NLN_i(t))}{\sum_{i=1}^{k} Sim(t, NUN_i(t))} \tag{5.2}$$

where $Sim(a, b)$ is the calculated similarity between cases a and b.

This is illustrated in Figure 5.2 where, for $k = 1$, the similarity between the target case T_1 and its NLN is much higher than the similarity between T_1 and its NUN. Whereas the similarity

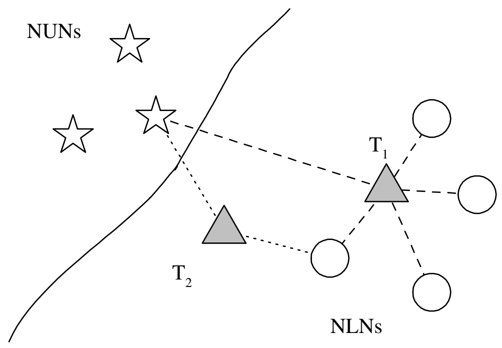


Figure 5.1: Average NUN Index Confidence Measure

between target case T_2 and its NLN is only marginally higher than the similarity between T_2 and its NUN. The ratio of these similarities for T_1 will give a higher result than that for T_2 indicating higher confidence in the classification of T_1 than T_2.

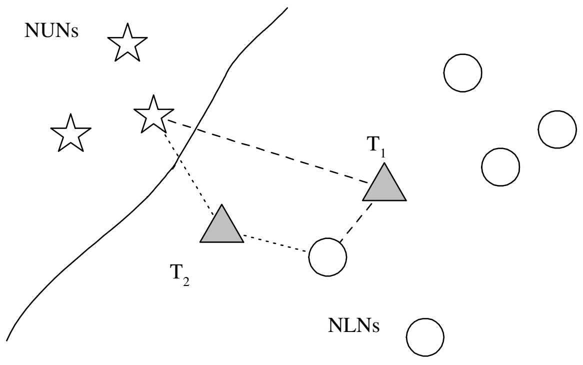


Figure 5.2: Similarity Ratio Confidence Measure

5.1.3 Explanation Case Similarity Ratio

Using the Similarity Ratio does not take into account that the target case may actually lie between the decision boundary and the nearest neighbour. Consider two possible nearest neighbours, NN_1 and NN_2 to a target case T as illustrated in Fig. 5.3. In this situation the similarity between NN_1 and T is equal to the similarity between NN_2 and T. Therefore for $k = 1$ the similarity ratio for NN_1 would be equal to the similarity ratio for NN_2. Similarly any neighbour located on the dashed circle would have an equal similarity ratio.

The Explanation Case Similarity Ratio attempts to correct this problem associated with the Similarity Ratio. The Explanation Case Similarity Ratio calculates the ratio of the similarity between the target case t and its k ECs to the similarity between the target case and its k NUNs, as given in Equation 5.3.

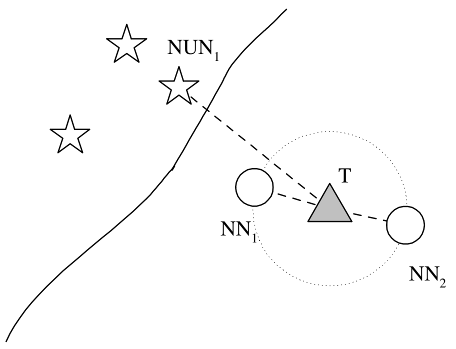


Figure 5.3: Similarity Ratio Confidence Measure Problem

$$SimRatio(t,k) = \frac{\sum_{i=1}^{k} Sim(t, EC_i(t))}{\sum_{i=1}^{k} Sim(t, NUN_i(t))} \tag{5.3}$$

where $Sim(a,b)$ is the calculated similarity between cases a and b.

This measure is similar to the Similarity Ratio described in the previous section except that the k ECs are used instead of the k NNs. This is to prioritise the use of neighbours between t and the decision boundary in calculating the ratio. This is illustrated in Figure 5.4 where, for $k = 1$, the similarity between the target case T_1 and its explanation case, EC_1, is much higher than the similarity between T_1 and its NUN. Whereas the similarity between target case T_2 and its explanation case, EC_2 is only marginally higher than the similarity between T_2 and its NUN. The ratio of these similarities for T_1 will give a higher result than that for T_2 indicating higher confidence in the classification of T_1 than T_2.

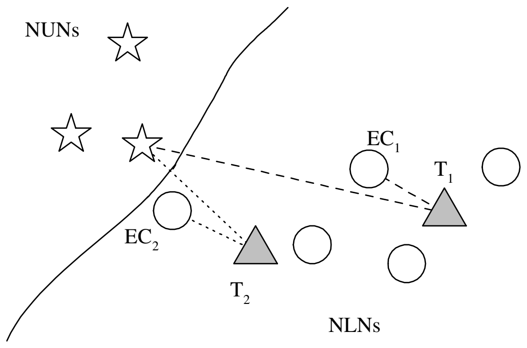


Figure 5.4: Explanation Case Similarity Ratio Confidence Measure

5.1.4 Similarity Ratio Within K

The Similarity Ratio Within K measure is similar to the Similarity Ratio as described above except that, rather than consider the first k NLNs and the first k NUNs of a target case t, it only uses the NLNs and NUNs from the first k neighbours. It is defined in Equation 5.4.

$$SimRatio(t,k) = \frac{\sum_{i=1}^{k} Sim(t, NN_i(t))1(t, NN_i(t))}{1 + \sum_{i=1}^{k} Sim(t, NN_i(t))(1 - 1(t, NN_i(t)))} \tag{5.4}$$

where $Sim(a,b)$ is the calculated similarity between cases a and b and $1(a,b)$ returns one if the class of a is the same as the class of b or zero otherwise.

This measure will attempt to reward cases that have no NUNs within the first k neighbours, i.e. are in a cluster of k cases of the same class. This is illustrated in Figure 5.5 where, considering $k = 3$, the target case T_1 has no NUNs within the first three neighbours whereas target case T_2 has two NUNs and one NLN. The Similarity Ratio Within K will be much larger for T_1 than that for T_2 indicating higher confidence in the classification of T_1 than T_2.

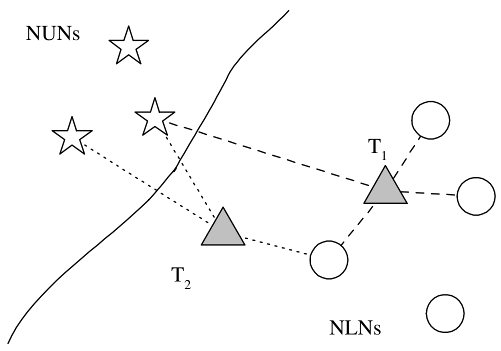


Figure 5.5: Similarity Ratio Within K Confidence Measure

If a target case t has no NUNs then Equation 5.4 is effectively Equation 5.2 with the denominator set to one.

5.1.5 Similarity Voting

The Similarity Voting measure is the total similarity of the NNs in the first k neighbours of the target case t, see Equation 5.5.

$$SumNNSim(t,k) = \sum_{i=1}^{k} 1(t, NN_i(t))Sim(t, NN_i(t)) \tag{5.5}$$

where $Sim(a,b)$ is the calculated similarity between cases a and b and $1(a,b)$ returns one if the class of a is the same as the class of b or zero otherwise.

For target cases in a cluster of cases of similar class this number will be large. For cases which are closer to the decision surface and have NUNs within the first k neighbours, this measure will be smaller. In fact for target cases with no NUNs within the first k neighbours this measure will be equal to the value of the Similarity Ratio Within K. Although this measure does not reward such cases as strongly as the Similarity Ratio Within K does as the resulting measure for the sum of the NLNs is not reduced by the influence of the NUNs.

5.1.6 Average NN Similarity

The Average NN Similarity measure is the average similarity of the NLNs in the first k neighbours of the target case t, see Equation 5.6.

$$SumNNSim(t,k) = \frac{\sum_{i=1}^{k} 1(t, NN_i(t))Sim(t, NN_i(t))}{\sum_{i=1}^{k} 1(t, NN_i(t))} \tag{5.6}$$

where $Sim(a, b)$ is the calculated similarity between cases a and b and $1(a, b)$ returns 1 if the class of a is the same as the class of b or 0 otherwise.

5.1.7 Nearest Like Neighbours Accuracy

The Nearest Like Neighbours Accuracy (NLN Accuracy) measure calculates the number of k like neighbours that would have been correctly classified by the CBR system as given in Equation 5.7.

$$NLNAccuracy(t,k) = \sum_{i=1}^{k} 1(NLN_i(t)) \tag{5.7}$$

where $1(a)$ returns one if the actual class of a is the same as the predicted class of a using a k-NN classifier or zero otherwise.

This is represented in Figures 5.6 and 5.7 where NLNs are represented by circles, NUNs by stars and target cases are represented by triangles. In Figure 5.6 we can see that the first NLN to target case T_1 would be incorrectly classified due to its close proximity to the two stars hence for $k = 1$ the number of correctly classified cases in the first k NLNs to T_1 is 0. This compares to the number of correctly classified cases in the first k NLNs to T_2 in Figure 5.7 being 1, indicating a higher level of confidence in the classification of T_2 than T_1

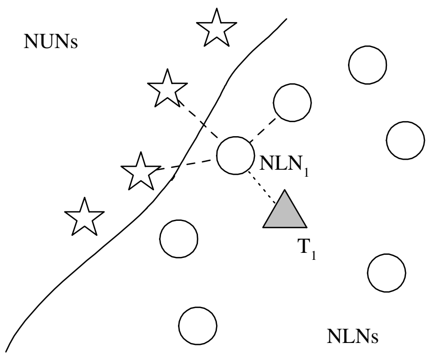


Figure 5.6: Classification of T_1 has a low confidence as NLN_1 would be incorrectly classified as a star

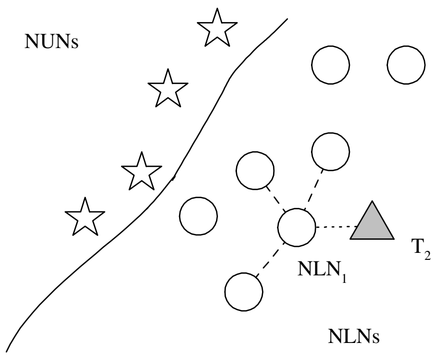


Figure 5.7: Classification of T_2 has a high confidence as NLN_1 would be correctly classified as a circle

5.2 Predicting Confidence

The previous section looked at a number of possible measures for estimating confidence in a prediction. The basis of each of these measures is that the higher the score for a measure the higher the confidence in the given prediction. In this section we look at a technique to select a threshold for each measure so that if the score is above the threshold there is a high confidence in the prediction, while if the score is lower than the threshold the system has a reasonable confidence in the prediction.

The calculation of a threshold involves performing a leave-one-out validation on a dataset for each measure. We evaluated each measure using k neighbours from $k = 1$ up to $k = 30$ and identified the confidence threshold, over all the k values, that gave us the highest proportion of confident and correctly classified cases while minimising the number of confident incorrect predictions. This is illustrated in Figure 5.8.

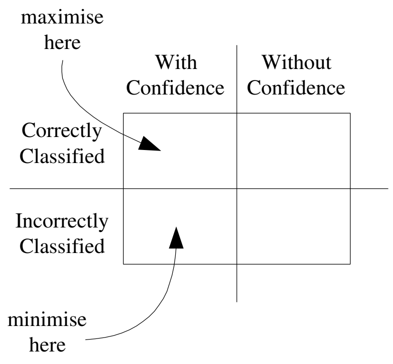


Figure 5.8: Criteria used to identify the best confidence threshold level

This was achieved by recording the confidence measure results for each target case $c_i, i = 1 \ldots N$, that was classified by a nearest neighbour algorithm. The results recorded included the number of neighbours k used in the measure, whether the target case was classified correctly or not and the measure calculated, m_{ik}. Setting the threshold t_k equal to the minimum value of m_{ik} for a given k and varying the threshold in small units ($t_k = t_k + .01$) up to the maximum value of m_{ik}, the number classified correctly with confidence (CC_k), the number classified incorrectly with confidence (CI_k), the number classified correctly without confidence (NCC_k) and the number classified incorrectly without confidence (NCI_k) as given by Equations 5.8, 5.9, 5.10 and 5.11 , were calculated.

$$CC_k = \sum_{i=1}^{N} \mathrm{gte}(m_{ik}, t_k)1(c_i) \tag{5.8}$$

$$CI_k = \sum_{i=1}^{N} \mathrm{gte}(m_{ik}, t_k)(1 - 1(c_i)) \tag{5.9}$$

$$NCC_k = \sum_{i=1}^{N} \mathrm{lt}(m_{ik}, t_k)1(c_i) \tag{5.10}$$

$$NCI_k = \sum_{i=1}^{N} \mathrm{lt}(m_{ik}, t_k)(1 - 1(c_i)) \tag{5.11}$$

where $\mathrm{gte}(a, b) = 1$ if $a >= b$ and $\mathrm{gte}(a, b) = 0$ otherwise, $\mathrm{lt}(a, b) = 1$ if $a < b$ and $\mathrm{lt}(a, b) = 0$ otherwise and $1(a) = 1$ if a is correctly classified and $1(a) = 0$ otherwise.

The selected threshold value, for a given k is the threshold t_k that maximises the odds ratio as given by Equation 5.12.

$$OddsRatio_k = \frac{\frac{CC_k}{CI_k}}{\frac{NCC_k}{NCI_k}} \tag{5.12}$$

Table 5.1 shows values for CC_k, CI_k, NCC_k and NCI_k for a range of similarity ratio thresholds in the Breathalyser domain. Also shown in this table is the odds ratio, calculated using Equation 5.12, for each of the possible thresholds. From the table we can see that there is a sudden increase in the odds ratio when a threshold of 1.18 is used, after this the odds ratio starts to fall off as the threshold increases. Figure 5.9 plots the odds ratio over a range of possible similarity ratio threshold values.

Table 5.1: Odds ratio for different similarity ratio thresholds

Similarity Ratio Threshold	CC_k	CI_k	NCC_k	NCI_k	Odds Ratio
1.10	52	2	50	23	12.0
1.11	47	2	55	23	9.8
1.12	42	2	60	23	8.1
1.13	41	1	61	24	16.1
1.14	39	1	63	24	14.9
1.15	36	1	66	24	13.1
1.16	34	1	68	24	12.0
1.17	33	1	69	24	11.4
1.18	32	0	70	25	$1.1*10^8$
1.19	32	0	70	25	$1.1*10^8$
1.20	30	0	72	25	$1.0*10^8$
1.21	24	0	78	25	$7.6*10^7$
1.22	17	0	85	25	$5.0*10^7$
1.23	11	0	91	25	$3.0*10^7$

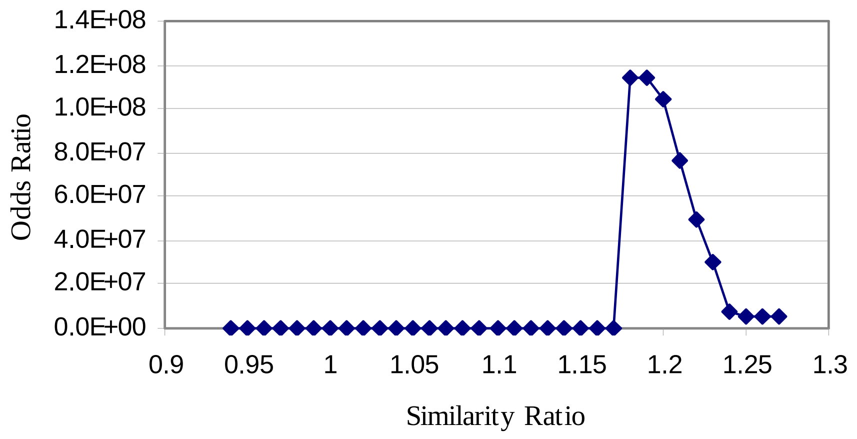


Figure 5.9: Odds Ratio for different Similarity Ratio Thresholds

5.3 Aggregated Confidence Measure

Although the primary importance of using confidence measures is to have a near if not perfect accuracy when there is a high level of confidence, it is also important to have a high confidence a high proportion of the time. Very often using a single confidence measure will result in a high confidence being achieved only a small proportion of the time. An approach to increasing the proportion of confident predictions is to use an aggregation approach that involves combining the results from individual confidence measures. Some possible aggregation approaches include:

(i) Summing the results from each of the 5 individual measures evaluated at the same value of k and comparing the sum against a threshold;

(ii) Using the best threshold for each individual measure and indicating confidence if a certain number of the measures indicate confidence;

(iii) Using a fixed k across all measures and indicating confidence if a certain number of the measures indicate confidence.

We found that the simplest and most effective method of aggregating the results is to assign confidence to a prediction if any of the individual measures indicated that the prediction was confident as in (ii) above. We call this measure the Aggregated Confidence Measure. The algorithm for using this measure has two stages:

(i) calculation of the constituent measure threshold values in a pre-classification stage,

(ii) determination of the aggregated confidence measure during classification.

The pre-classification stage involves pre-processing of the case-base to identify the best threshold for each individual constituent measure. This is performed in the manner described in Section 5.2.

A threshold consists of two values; the k value indicating the number of neighbours to use in the calculation and the actual threshold value above which the prediction is considered confident. These constituent measure thresholds are stored.

The aggregated confidence measure is then determined during classification for each target case that is classified. This is performed by first calculating the actual score for each individual constituent measure for the target case. The aggregated confidence measure specifies that if at least one of the calculated scores for the individual measures is equal to or greater than the stored threshold value for that measure, confidence is expressed in the classification.

5.4 Confidence Evaluation

In this section we present some of the results from an evaluation of the assessment of confidence in CBR systems. These evaluations examine confidence accuracy on unseen data and the effects of the addition of cases to a case base on confidence prediction.

5.4.1 Real-time Confidence

The first analysis on confidence measures was to check the accuracy of the confidence measures that were being used during the realtime evaluation of the developed Bronchiolitis Decision Support System (Section 6.3 provides a description of the developed system). In the first part of this evaluation 9 out of the 46 predictions had a high confidence while in the second part of the evaluation 12 out of the 65 predictions had a high confidence. However, when the recommendation of each of these cases were compared to the final disposition[1] the recommendation was correct 8 out of the 9 times in the first part of the evaluation and 10 out of the 12 times in the second part. From these results in the bronchiolitis domain we can see that we were achieving a high confidence only 18-20% of the time. This level of achieving a high confidence is extremely low as it is only occurring at most one out of every five predictions.

5.4.2 Confidence in relation to Case-base size

To try and address this low level of confidence we performed an analysis to see how the confidence varies in relation to the case-base size. The first part of this analysis was on the Bronchiolitis domain. Figure 5.10 shows the accuracy of the system and confidence levels for the aggregated confidence measure and the individual confidence measures for case-base sizes varying in intervals of 10 from 100 to 330 using a leave-one-out validation. It should be noted that as the case-base increases in size the cases that are added are such that the original ordering of the case-base is

[1]If a patient is discharged a follow up call is made three days later to make sure the patient was not subsequently admitted to hospital with bronchiolitis. If the patient was subsequently admitted their final disposition is an admission.

maintained. This is to mimic how the system would have operated if the confidence measures were retrained each time after 10 cases were added to the system in real life.

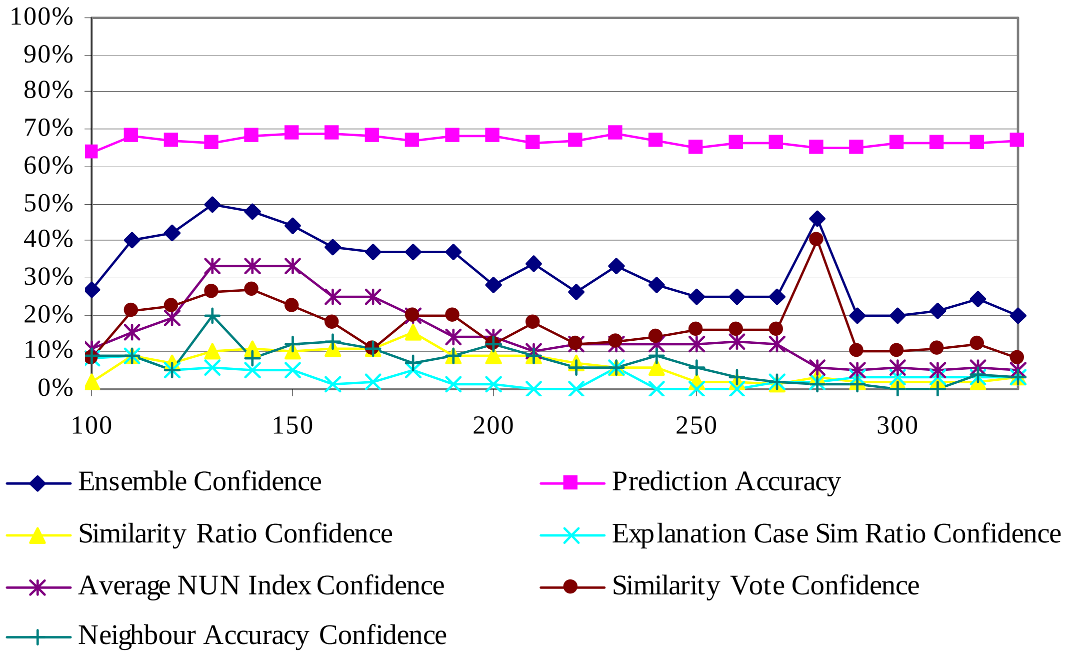


Figure 5.10: Accuracy and Confidence levels in relation to case-base size for the Bronchiolitis domain

Overall the accuracy remains relatively stable as cases are added to the case-base with the accuracy varying by 5% throughout the evaluation with only a slight fall off in accuracy as the case-base contains more than 230 cases. The results from the confidence measures present a more interesting picture. The general trend for the measures are to initially increase in the percentage of the time that they yield a high confidence and then to decrease as the case-base continues to grow. This evaluation was repeated on a randomised version of this case-base to see if this characteristic was an artefact of the order of the cases. Although the overall confidence level never reached the same maximum value, the same trend of rising initially and then falling off could still be seen.

To check if this trend was due to noise in the case-base, noisy cases were removed from the case-base. The techniques we used for noise reduction were Repeated Edited Nearest Neighbour (Tomek 1976) and Blame Based Noise Reduction (Delany and Cunningham 2004). The results for confidence and accuracy after two different runs of each of the noise reduction techniques are shown in Table 5.2. After noise reduction the confidence levels and accuracy levels have actually decreased.

This evaluation was also performed on the Breathalyser domain. In a similar manner the confidence level and accuracy was calculated for different sizes of the case-base from 50 to 120 cases. The results of this evaluation are shown in figure 5.11. In this situation significantly higher

Table 5.2: Confidence and Accuracy after noise reduction

	Number of Cases	Aggregated Confidence Level	Accuracy Level	Original Accuracy
RENN run 1	271	15%	65%	66%
RENN run 2	291	15%	63%	65%
BBNR run 1	277	19%	64%	65%
BBNR run 2	308	15%	64%	66%

values for the confidence level were obtained, however the overall trend of a decreasing confidence level as the case-base grows is again apparent.

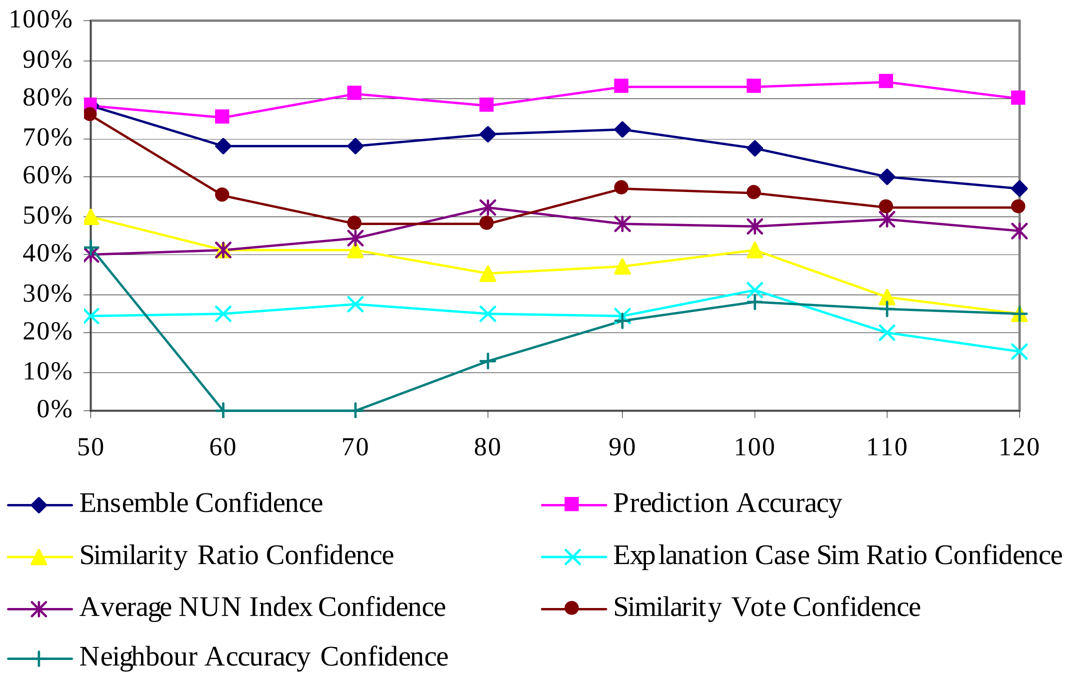


Figure 5.11: Accuracy and Confidence levels in relation to case-base size for the Breathalyser domain

These results show how sensitive confidence measures are to new cases being added to a case-base. Further evidence of this sensitivity is shown by the variance in parameters for the different confidence measures when retrained after new cases are added. Table 5.3 shows the optimum k and threshold values for each of the measures for the different case-base sizes for the Bronchiolitis domain. The variance in these values further show that the confidence measures are sensitive to new cases.

While it is clear that it is useful to produce estimates of confidence, it is clear that generating reliable estimates is not straightforward. In this section we saw that the addition of cases to a

Table 5.3: *k* and threshold, *t*, levels for different case-base sizes in the Bronchiolitis domain.

Number of Cases	Similarity Ratio		Exp Case Similarity Ratio		Average NUN Index		Similarity Vote		Nearest Neighbour Accuracy	
	k	*t*	*k*	*t*	*k*	*t*	*k*	*t*	*k*	*t*
100	5	1.21	4	1.04	13	24	23	1829	27	22
110	13	1.11	11	1.02	17	27	4	467	23	20
120	13	1.11	11	1.02	26	35	3	362	19	15
130	12	1.12	12	1.02	19	24	4	414	29	21
140	7	1.13	13	1.02	27	32	4	414	17	14
150	12	1.12	12	1.01	27	33	5	518	18	14
160	29	1.11	29	1.09	1	6	4	486	8	7
170	1	1.17	17	1.05	1	6	30	2474	14	12
180	14	1.10	30	1.05	1	7	4	486	13	12
190	17	1.11	18	1.05	14	33	4	486	17	15
200	18	1.11	19	1.05	13	33	30	2654	10	9
210	29	1.11	29	1.08	7	24	4	486	22	20
220	6	1.14	30	1.08	11	31	5	616	13	12
230	6	1.14	1	1.06	7	28	5	616	15	14
240	6	1.14	30	1.07	8	31	5	616	13	12
250	3	1.23	26	1.05	10	32	5	591	14	13
260	3	1.23	26	1.05	12	35	5	591	17	16
270	2	1.23	23	1.04	9	33	5	591	25	24
280	1	1.22	27	1.02	17	51	1	142	15	14
290	1	1.22	1	1.04	14	47	3	420	16	15
300	1	1.22	1	1.04	14	48	3	420	17	16
310	1	1.22	1	1.04	14	50	3	420	17	16
320	1	1.22	1	1.04	16	55	3	420	23	22
330	1	1.22	1	1.04	18	59	4	560	14	13

case-base can have detrimental effects on confidence estimation. These detrimental effects can still occur even if the confidence measures are updated after the addition of cases. These findings agree with those of Cheetham and Price (2004) who showed that it is difficult to avoid the situation where predictions labelled as confident prove to be incorrect. They also emphasise that the confidence estimation mechanism will need to be updated over time as the nature of the problems being solved can change.

5.4.3 Confidence in Spam Filtering

These confidence measures were also used for generating estimates of classification confidence for a case-based spam filter called ECUE (Email Classification Using Examples) (Delany, Cunningham and Coyle 2004, Delany, Cunningham, Doyle and Zamolotskikh 2005). ECUE has the advantage of being very effective at tracking concept drift but this requires the user to identify false positives and false negatives so that they can be used to update the case-base. Concept drift occurs as a result of the constantly evolving nature of spam. Identifying false negatives is not a problem because they turn up in the Inbox (i.e. spam that has been allowed through the filter). Identifying false positives involves monitoring a spam folder to identify legitimate email that has been classified as spam. Our objective here is to be able to partition this class so that the user need only monitor a subset - the set for which the confidence is low.

A straightforward success criterion in this regard is the proportion of positives for which prediction confidence is high and the prediction is correct (clearly there cannot be any false positives in this set). A mechanism that could label more than 50% of the positive class (i.e. classified as spam) as confident and have no false positives in this set would be useful. The lower-confidence positives could be allowed into the Inbox carrying a *Maybe-Spam* marker in the header or placed in a *Maybe-Spam* folder that would be checked periodically.

In order to assess the performance of the confidence measures in the spam domain we evaluated each of them on a number of spam datasets. Five datasets were used. Each consisted of legitimate and spam emails received by a single individual over a period of time. Each dataset represents a different period of time for a single individual. Two different individual's mail were used over all datasets. The legitimate emails in the datasets include a mixture of business, personal and mailing list emails. Case-bases were built from each of the five original datasets. Case representation details are available in (Delany, Cunningham and Coyle 2005, 2004).

The evaluation involved performing a leave-one-out validation on each dataset for each measure. We evaluated each measure using k neighbours from $k = 1$ up to $k = 15$ and identified the confidence threshold, over all the k values, that gave us the highest proportion of correctly predicted spam emails when there were no incorrect predictions (i.e. false positives). This is illustrated in Figure 5.12.

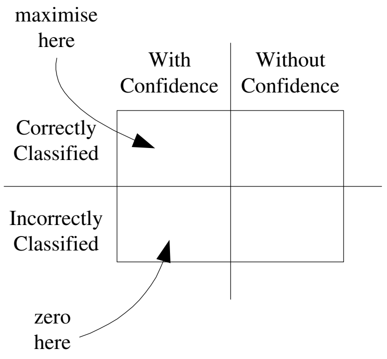


Figure 5.12: Criteria used to identify the best confidence threshold level in the spam domain

The selection of the optimum parameters, the k and threshold values, were selected using a slightly modified technique to that described in Section 5.1. In this situation there is no tolerance for any false positives, so instead of using the parameters that maximised the odds ratio the selected threshold value was the one that maximised the number of spam correctly predicted with high confidence, while the number of incorrect predictions with high confidence was zero.

The results of this evaluation are presented in rows 1 to 5 of Table 5.4. It details for each measure the highest percentage confidence that can be achieved on each dataset. This is the proportion of *spam* predictions that are made with high confidence. In all situations no highly confident incorrect predictions were made so no false positives are included in this proportion. In effect, this proportion of the spam can be ignored by the user, whereas the remaining percentage would have to be checked by the user. Row 6 of this table shows the highest percentage confidence that is achievable when the aggregated confidence measure is used. Here we can see that the aggregated measure resulted in an increase in accuracy in each dataset compared to using a single measure to assess confidence.

Table 5.4: Best percentage confidence achievable for each spam dataset using different confidence measures

Confidence Measure	Dataset 1	Dataset 2	Dataset 3	Dataset 4	Dataset 5	Average
Similarity Ratio	46%	84%	50%	49%	16%	49.0%
Average NUN Index	23%	76%	75%	41%	44%	51.8%
Similarity Vote	21%	29%	68%	91%	58%	53.4%
Similarity Ratio Within k	21%	29%	71%	91%	57%	54.8%
Average NN Similarity	20%	29%	49%	91%	60%	49.8%
Aggregated Confidence	55.4%	85.4%	83.8%	93.7%	77.3%	79.1%

To evaluate the ACM on unseen data involved building confidence thresholds for the ACM constituent measures on the initial case-base and then classifying the remaining emails using the ACM to determine how confident the *spam* predictions are (Delany, Cunningham, Doyle and Zamolotskikh 2005). In this way, the test emails were not used in the determination of the confidence thresholds in any way.

The test emails were presented in date order for classification. Since this email data is subject to concept drift, ECUE's case-base update policy was applied to allow the classifier to learn from the new types of spam and legitimate email presented. The update policy has a number of components; an immediate update of the case-base with any misclassified emails when a FP occurred, a daily update of the case-base with any other misclassified emails that occurred that day, and a monthly feature reselection process to allow the case representation to take any new predictive features into account. In order to keep the confidence thresholds in line with the updates to the case-base an update policy for the confidence thresholds was also applied. This policy had two components; the confidence thresholds were updated whenever a confident FP email occurred and also after a monthly feature reselect.

Tables 5.5 and 5.6 show the results of testing the performance of the ACM on unseen data using the two datasets 6 and 7. The tables present the accumulated monthly results for each dataset listing the total number and types of emails that were classified, the percentage of incorrect spam predictions (i.e. FPs) made (labeled *%FP classified*) and the percentage of incorrect spam predictions made with high confidence (labeled *%Confident FPs*). The table also gives the total percentage of spam predictions with high confidence (labeled *%Confidence*).

Table 5.5: Performance of ACM on unseen data using Dataset 6

Month	1	2	3	4	5	6	7	8	Overall
Number of emails	772	542	318	1014	967	1136	1370	1313	7382
Number of Spam	629	314	216	925	917	1065	1225	1205	6496
Number of Non Spam	93	228	102	89	50	71	145	108	886
%FPs classified	4.3%	2.6%	1.0%	1.1%	6.0%	1.4%	0.0%	1.9%	2.0%
%Confident FPs	0.0%	0.9%	0.0%	1.1%	0.0%	0.0%	0.0%	0.9%	0.5%
%Confidence	70%	87%	76%	94%	89%	73%	77%	99%	85%

Table 5.6: Performance of ACM on unseen data using Dataset 7

Month	1	2	3	4	5	6	Overall
Total emails classified	293	447	549	693	534	495	3011
Number of Spam	142	391	405	459	406	476	2279
Number of Non Spam	151	56	144	234	128	19	732
%FPs classified	0.7%	3.6%	3.5%	2.6%	1.6%	0.0%	2.2%
%Confident FPs	0.0%	3.6%	0.7%	0.4%	1.6%	0.0%	0.8%
%Confidence	95%	95%	87%	64%	89%	88%	85%

In both datasets predictions of confidence are high, averaging 85% in both cases with a lowest monthly level of 64%. This is the percentage of spam predictions that can be ignored by the user, the remaining spam predictions can either be flagged in the Inbox as *Maybe Spam* or placed in a separate *Maybe Spam* folder for the user to check.

However in some of the months the ACM has resulted in confident incorrect predictions. Although the actual numbers of emails are low (four emails for Dataset 6 and six emails for Dataset 7) the ideal situation is one where all incorrect predictions have low confidence and will be flagged for the user to check. FPs flagged as confident will end up in the *spam* folder and may be missed by the user. Examining the confident FPs, three are emails from mailing lists and two are responses to Web registrations which users may not be too concerned with missing. The remaining five are important, some work related and one even a quotation in response to an online car hire request.

It is clear that we are approaching the limits of the accuracy of machine learning techniques in this domain. We see two possibilities for addressing these FPs. Close examination of such emails may identify domain specific characteristics that could be used as a feature or number of features in the case representation. Secondly, most deployed spam filtering solutions do not rely on one approach for filtering spam, they combine a number of techniques including white and black listing, rules, collaborative and learning approaches. Incorporating additional techniques into ECUE to add to its case-based approach could help in catching these outlier FPs.

5.5 Conclusions

In this chapter we introduced a number of confidence measures (Section 5.1) and how they can be amalgamated into an ensemble to increase the occurrence of high confidence predictions (Section 5.3). Section 5.4 presented the results from evaluations on confidence measures. In the next chapter, Chapter 6, we describe the implementation of a system that use these measures and the explanation utility framework, described in Chapter 4, in a real-time decision support system for the Bronchiolitis domain.

Chapter 6

Implementation

In this chapter we implement a real time CBR system for predicting dispositions and producing explanations for patients presented to the Emergency Department of the Kern Medical Center with Bronchiolitis. This system uses the CBR techniques described in Chapter 2, the explanation utility framework presented in Chapter 4 and the confidence measures described in Chapter 5. The Bronchiolitis Decision Support System (Section 6.3) is implemented in Java based on the Fionn framework which uses the CBML markup language. CBML and Fionn are described in more detail in Sections 6.1 and 6.2 respectively.

6.1 CBML

XML (eXtensible Mark-up Language) is currently commonly used for marking up structured, knowledge-rich data. XML is a description language that supports meta-data descriptions for particular domains and these meta-descriptions allow applications to interpret data marked-up according to this format. Some useful advantages of using XML to represent data include: interoperability, ease of reuse, as well as the application-independent exchange of data over existing network protocols and access for developers to the entire XML tool-set. This tool-set includes fast, reliable document parsers, e.g. SAX and DOM, validating documents e.g. DTDS and XML-Schema, and document transformers, e.g. XSLT.

As previously mentioned in Section 2.4 there have been numerous XML-based CBR representation formats. The earliest work in the CBR community on an XML-based case representation language was the introduction of CBML (Case Based Markup Language) by Hayes et al. (1998) The main motivation behind this first version of CBMLv1 was to facilitate the storage and distribution of case data over a network, thus allowing the creation of distributed CBR applications.

CBMLv1 was represented by two documents; the case document and the structure document. The case document contained the contents of the case and the structure document contained the definition of the case as well its constituent features. It was possible to define symbol, integer and string feature types. The case structure also contained a simple similarity measure representation containing feature weights and some simple local similarity measures in the case structure definition.

6.1.1 CBMLv3

CBML has continued to develop to the point where it currently is at CBMLv3[1] Coyle et al. (2003, 2004). The maturity of XML-Schema led to the redevelopment of CBML and now the description of CBML is stored in an XML-Schema document — the **CBML Schema**. The CBML Schema is shown in Appendix C.1. Only documents that follow this schema exactly can be considered valid CBML.

CBMLv3 continued the tradition of separating structure from content. New feature types were added, the ability to restrict case data and support for hierarchical cases along with flat cases were also added. We also separated the similarity measure description from the structure document and stored it in a new CBML document — the *similarity profile* document. Transferring structural and similarity information away from the case document resulted in a significant simplification and condensing of the case content document. The remainder of this section will give an overview of each of the CBML documents with emphasis on the similarity profile document. Example case structure, case content and similarity documents for the Breathalyser domain can be found in Appendix D.

6.1.2 Case Structure Document

The case structure document defines the hierarchy and cardinality of its features, their types and possible restrictions on their values. A case structure document is made up of feature structure definitions describing the features that can occur in a case. The feature structure defines the feature's type, its value restrictions, and other attributes. Table 6.1 describes the four attributes of a feature structure description. If any of these attributes is omitted from a feature structure definition the parser assumes the default values.

Table 6.1: Description of a CBML Feature Structure

Attribute	**Definition**	**Default**
`name`	The name of the feature — used for identification	N/A
`discriminant`	If `true` this feature is used in the CBR Process	`true`
`mandatory`	If `true` this feature is required to appear in every case	`true`
`solution`	If `true` this feature is the solution component of the case. N.B. only one feature can be defined as a solution feature in a case.	`false`

[1]The CBML web page is located at `http://www.cs.tcd.ie/research_groups/mlg/CBML/`

Figure 6.1 shows an example feature structure definition. This feature is called `Units` and is the units consumed feature in the Breathalyser domain (Section 4.2.1). Because the definition does not have an explicit value for the `discriminant` and `solution` attributes it takes the default values (`true` and `false` respectively). The tag `<double/>` defines the feature's type as a double feature with the enclosed tags further restricting the feature to only contain values in the range of 0-30. There are seven possible feature types in CBML: `symbol`, `integer`, `double`, `boolean`, `string`, `taxonomy` and `complex`. More information on defining other feature types can be found in Coyle et al. (2003).

```
<feature name="Units">
    <double>
        <minInclusive value="0.0"/>
        <maxInclusive value="30.0"/>
    </double>
</feature>
```

Figure 6.1: An Example Feature Structure Definition

The Case parser requires the case structure document to understand and convert a case content document into a case or case-base. If it parses a case from a content document that does not follow the structure definition exactly it will fail. In effect, this means that the case structure document is used to *validate* the case content document. This validation serves to protect the CBR system from an unexpected failure due to bad case or feature content. The Case Structure document is itself validated against the CBML Schema document by the CBML document parser.

6.1.3 Case Content Representation

The syntax of a case is described in CBML as follows: feature values are encased by a pair of XML tags with the feature name. This makes them easy to read and easy to translate into other formats since their tags do not contain attributes. Figure 6.2 shows an example case from the Breathalyser domain in CBML format. The values of each feature are enclosed within the feature tags, i.e. the value for the feature `Units` is `5.2`.

```
    <case name="xxxxxxx">
        <Kgs>51.0</Kgs>
        <Duration>120.0</Duration>
        <Gender>Female</Gender>
        <Meal>Lunch</Meal>
        <Units>5.2</Units>
        <BAC>Over</BAC>
    </case>
```

Figure 6.2: An Example CBML Case

6.1.4 Similarity Measure Representation

In Section 2.5 we defined the similarity between two cases as the amalgamation of the local similarities of features common between two cases:

$$Sim(Q, X) = \sum_{f \in F} w_f \ \sigma(q_f, x_f) \tag{6.1}$$

where w_f is the feature relevance weight and $\sigma(q_f, x_f)$ is the local similarity measure for feature f. In order to provide a representation of the similarity measure it is therefore necessary to represent both a relevance weight and a description of the local similarity measure for every feature. These are defined in the similarity profile document Coyle et al. (2004). Similarity profile documents are defined by the CBML Schema and are validated by the CBML parsers in the same way as case structure documents.

Feature similarities have attributes containing their name and the relevance weight attached to them in the amalgamation function. The relevance weight is simply an attribute called `weight` that can have any floating-point value. Within each feature-similarity description there is also the description of the local similarity measure. These fall into one of four categories: `exact`, `difference-based`, `array` or `complex`. These local similarity types are described in the following sections.

Exact Similarity Measures

This type of similarity measure is based on exact matching. Similarity is assigned the value `1` if two feature values are equal, otherwise it is assigned the value `0`. We represent this type of function as an `exact` type similarity function. Figure 6.3 shows the representation of a local similarity measure for a feature called `Gender` that uses an exact similarity measure. `Gender` is a symbol feature used in the Breathalyser domain. `Gender` is defined as having a relevance weight of `0.25`, and that it uses an `exact` similarity function.

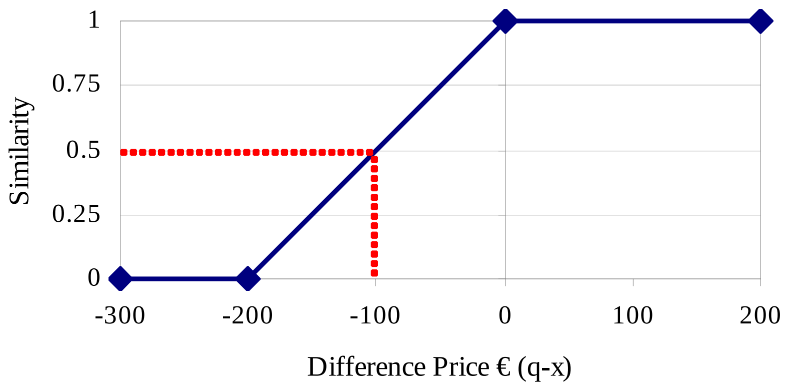
```
<feature name="Gender" weight="0.25">
    <exact/>
</feature>
```

Figure 6.3: An Example Exact Similarity Measure Definition

Difference-based Similarity Measures

In Section 2.5.2 we described difference based similarity measures. The similarity graphs are defined in CBML by points that are on the graph. By defining a suitable set of points, any piece-wise linear relationship between similarity and difference can be represented. This graph may be symmetrical (the default) or asymmetrical. A symmetrical graph only deals with absolute difference values.

Figure 6.4 shows the representation of an asymmetrical `difference` function similarity measure for the feature `price` from the PC Sales domain (Section 2.5.3). `Price` is a numeric (`double`) feature that represents the price of a PC. `Price` is defined as having a relevance weight of `1.0`. A graph of similarity versus difference is defined with a number of point definitions. For demonstration purposes, this graph is also plotted. From the graph, the calculated similarity between two prices with a negative difference of €100 is `0.5` (whereas a difference of positive €100 would have yielded a similarity of `1`).



```
<feature name="price" weight="1.0">
    <graph type="asymmetrical">
        <point difference="-Infinity" similarity="0"/>
        <point difference="-200" similarity="0"/>
        <point difference="0" similarity="1"/>
        <point difference="Infinity" similarity="1"/>
    </graph>
</feature>
```

Figure 6.4: An Example Difference-based Similarity Measure Definition

Array Similarity Measures

The `array` similarity measure (Section 2.5.1) is defined by specifying the exact similarity for each combination of feature value. If a similarity value is not defined in this array, the `exact` similarity measure will be used. Figure 6.5 shows the CBML representation of an `array` similarity measure for the feature `meal`. `Meal` is a symbolic feature used in the Breathalyser domain. `Meal` is defined as having a relevance weight of `0.75`. The similarity definition also defines an array of every possible feature value combination with a similarity value for each, e.g. $\sigma_{Meal}('none', 'snack') = 0.8$. The array as defined is also shown in this figure.

```
<feature name="meal" weight="0.75">
    <array>
        <primary name="none">
            <secondary name="snack" value="0.8"/>
            <secondary name="lunch" value="0.4"/>
        </primary>
        <primary name="snack">
            <secondary name="none" value="0.8"/>
            <secondary name="lunch" value="0.8"/>
            <secondary name="full" value="0.4"/>
        </primary>
        <primary name="lunch">
            <secondary name="none" value="0.4"/>
            <secondary name="lunch" value="0.8"/>
            <secondary name="full" value="0.8"/>
        </primary>
        <primary name="full">
            <secondary name="snack" value="0.4"/>
            <secondary name="lunch" value="0.8"/>
        </primary>
    </array>
</feature>
```

	none	snack	lunch	full
none	1	0.8	0.4	0
snack	0.8	1	0.8	0.4
lunch	0.4	0.8	1	0.8
full	0	0.4	0.8	1

Figure 6.5: An Example Array Similarity Measure Definition

Complex Similarity Measures

It is impossible to provide for a representation scheme that could cover every possible type of similarity measure, e.g. polynomial or exponential. If the number of possible values is finite, it may be appropriate to calculate all possibilities a priori and store them using the `array` similarity definition. If this is impossible or impractical it is possible to define a similarity measure externally from the similarity profile document, e.g. in a Java function or MathML document, and refer to it using the `complex` similarity definition. Figure 6.6 shows the representation of a complex similarity measure for the feature `sepal-length` that contains a reference to a predefined local similarity measure (`iris.similarity.SepalLength`). `sepal-length` is a numeric (`double`) feature used in Fisher's Iris domain (Fisher 1936). It defines `sepal-length` as having a relevance weight of `0.25`. It also tells the Similarity Profile to use the `iris.similarity.SepalLength` local similarity function.

`iris.similarity.SepalLength` actually refers to a Java class that implements a CBML Similarity Measure Interface (shown in Figure 6.7). Any Java class that implements this interface can be referred to in the CBML Similarity Profile document. It will then be used by the similarity

```
<feature name="sepal-length" weight="0.25">
    <measure name="iris.similarity.SepalLength"/>
</feature>
```

Figure 6.6: An Example Complex Similarity Measure Definition

profile to calculate the similarity between `feature1` and `feature2`. In this way CBML similarity profiles are capable of calling any user-defined local similarity measure. The interface itself contains a single method that takes in two features and returns a double — the similarity value. This ensures a level of interoperability. However, since `complex` similarity measures depend on external resources that are outside the CBML core specification their use is discouraged.

```java
package cbml.cbr;

public interface SimilarityMeasure extends java.io.Serializable{
    double calculateSimilarity(Feature feature1, Feature feature2);
}
```

Figure 6.7: The Local Similarity Measure Java Interface

6.2 Fionn

Based on CBML we have also developed a machine learning workbench called *Fionn*. Fionn provides a range of machine learning techniques while creating a level of abstraction from the CBML syntax. Figure 6.8 shows the architecture of Fionn. The core specifications of Fionn are used by all the other components and CBML makes up an important part of this. Although case-bases are stored in CBML documents they can also be imported from comma separated files and arff files (files that are compatible with the Weka toolkit[2]).

The other main components of the Fionn core are a suite of classifiers (including k-NN, support vector machines, neural nets, logistic regression, linear regression and naïve Bayes) and an evaluation framework which supports a variety of validation schemes and a selection of error functions.

Much of the current work being done in the Fionn Framework is in developing the CBR Explanations, Feature Selection and Weighting, and Noise Reduction components. There are three applications currently in development that use the Fionn Framework:

Medical Decision Support As described in this thesis

Spam Filtering Application Delany, Cunningham and Coyle are working on a spam filtering application called ECUE (E-mail Classification Using Examples) that dynamically adapts to the changing nature of spam e-mails (Cunningham, Nowlan, Delany and Haahr 2003, Delany,

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

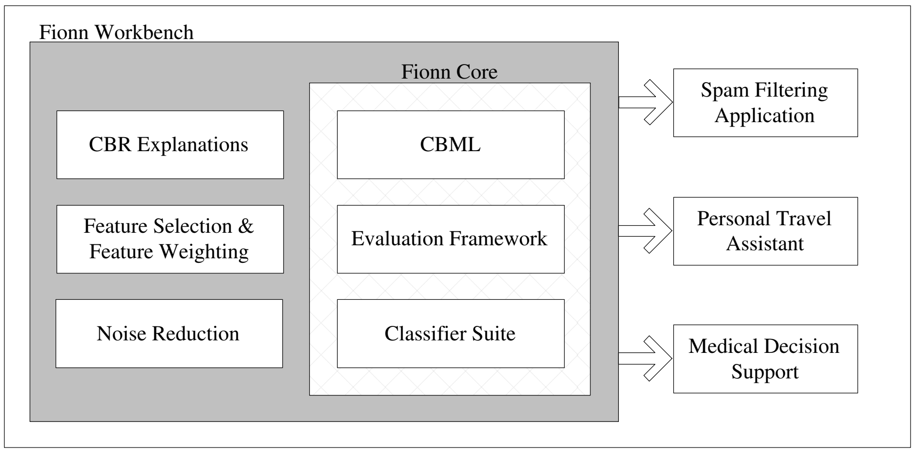


Figure 6.8: The Fionn Workbench

Cunningham, Tsymbal and Coyle 2004). Because of the volume of spam e-mail and its evolving nature their application uses several case-base maintenance techniques that remove noisy and redundant cases (Delany and Cunningham 2004). Their application uses many features of the Fionn framework including the k-NN classifiers and evaluation framework.

The Personal Travel Assistant The Personal Travel Assistant is a flight recommender application that uses CBR. It allows users to search multiple flights based on their individual travel preferences. These preferences are implicitly learned from observations of user behaviour after a flight is purchased (Coyle 2004).

The CBR explanations component of Fionn is most relevant to our research. This component contains mechanisms for using the explanation utility measures to select explanation cases (Section 4.3) and for generating discursive text based on the selected features for highlighting (Section 4.4) and the confidence in a classification (Section 5.1). Following on from using XML, in the form of CBML, these mechanisms are also based on XML representation schemes. The remainder of this section will briefly describe the representation schemes for defining explanation measures and the discursive text.

6.2.1 Explanation Utility Profile

The representation for explanation utility measures extends the similarity representation of CBML. In this situation there is a set of feature-level explanation utility measures for each possible classification. The individual measures have the same syntax as the local similarity measures defined in Section 6.1.4. Figure 6.9 shows part of an explanation document from the Bronchiolitis domain. In this example there are two sets of explanation utility measures. One for a `Discharge` classification and the other for an `Admit` classification. Explanation profile documents are also defined by a schema document and are validated by the parsers in the same way as case structure and similarity documents. The explanation schema document is given in Appendix C.2.

6.2.2 Explanatory Text Profile

The explanatory text profile is used to describe the syntax of textual parts of an explanation. The explanatory text profile is also represented using XML. There are a number of components that make up this profile. The main component is the actual text to display when a classification is performed. This component can contain normal text along with dynamic text that is based on the actual classification and comparisons between the query case and the retrieved explanation case. The comparisons can highlight features in the explanation case that are supportive of the classification and features that are not supportive of the classification. The following is a list of components that are used in generating the dynamic section of the discursive text:

Descriptive Feature Names As certain characters are restricted from being used in XML tags[3] this component creates a mapping from feature names used by the system to a more user friendly syntax. For example in the Breathalyser domain the feature "`Units`" could be mapped to "`units of alcohol consumed`". This can also be used when displaying a table containing case details (similar to Table 2.1).

Utility Text This component is used to express differences in feature values in a natural language. For example if there is a positive difference in age between two cases this could be expressed as "`older`" instead of "positive age difference".

Classification Text text to display based on the classification. For example a predicted disposition to `discharge` could be mapped to "`discharged from hospital`"

Confidence Settings and Confidence Text This component contains information on what measures and associated k and threshold values to use when calculating confidence. Also contained in this section is the text to display depending on the level of confidence.

The explanatory text profile is also validated against a schema document which is included in Appendix C.3.

6.3 Bronchiolitis Decision Support System

The Bronchiolitis Decision Support System was developed in Java using the Fionn framework. The initial use of Fionn was for feature selection (Loughrey and Cunningham 2004) and noise reduction (Pasquier et al. 2005). The system generates recommendations and explanatory text for patients presented to the Emergency Department with bronchiolitis. For the most part this text supports

[3]Tags cannot contain spaces and non alpha-numeric values

```
<expdoc>
    <explanation classification="Discharge">
        <feature name="Age" weight="1.0">
            <graph type="asymmetrical">
                <point difference="-20.0" similarity="0.0"/>
                <point difference="0.0" similarity="0.9"/>
                <point difference="2.0" similarity="1.0"/>
                <point difference="3.0" similarity="0.9"/>
                <point difference="20.0" similarity="0.0"/>
            </graph>
        </feature>
        .
        .
        .
    </explanation>
    <explanation classification="Admit">
        <feature name="Age" weight="1.0">
            <graph type="asymmetrical">
                <point difference="-20.0" similarity="0.0"/>
                <point difference="-3.0" similarity="0.9"/>
                <point difference="-2.0" similarity="1.0"/>
                <point difference="0.0" similarity="0.9"/>
                <point difference="20.0" similarity="0.0"/>
            </graph>
        </feature>
        .
        .
        .
    </explanation>
</expdoc>
```

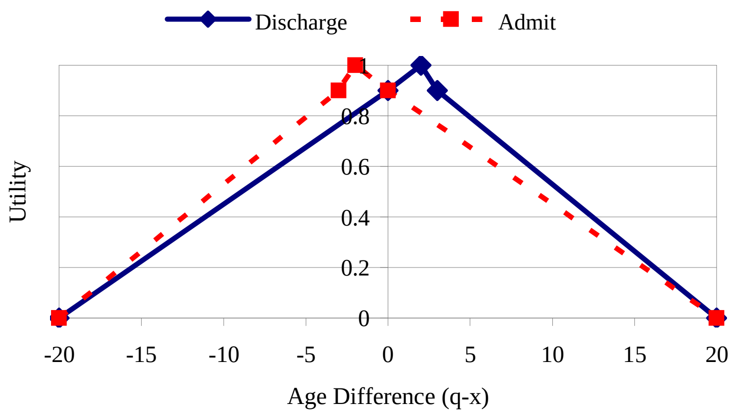


Figure 6.9: An Example Explanation Utility Document Definition

the recommendation; however it can also highlight issues that might support a different disposition (see Fig. 6.10). The system operates as follows:

- A new patient suffering from bronchiolitis is entered into the emergency department's records in the normal way.
- The Bronchiolitis Decision Support System generates a recommendation and supporting explanation (Figure 6.10). In this system the three most similar cases to a presented target case are considered. If all three of the most similar cases were discharged than the target case is classified as a discharge; otherwise an admission is recommended. This biases the system towards recommending admission.
- The recommended disposition and supporting explanation are presented to any Attending Doctors, Resident Doctors or Nurse Practitioners dealing with the patient, who then fill out an evaluation on the recommendation and the explanation. The results from this evaluation will be presented in Chapter 7.

Figure 6.10 shows an example recommendation and explanatory text for a particular patient. The first line of the text shows that the system recommends that the patient should be discharged from hospital. The next paragraph displays the features that it believes supports its recommendation. The third paragraph is included if the system believes that certain features should be considered before a user accepts the recommendation. Finally the system indicates that it has a high level of confidence in the recommendation.

6.4 Conclusions

In this chapter we presented the Bronchiolitis Decision Support System that we implemented for predicting dispositions and producing explanations for patients admitted to the Emergency Department of the Kern Medical Center with Bronchiolitis. This system was developed using Fionn and CBML which were described in more detail in Sections 6.1 and 6.2 respectively. In the next chapter we present an evaluation of this system along with evaluations of the explanation utility framework described in Chapter 4 and the confidence measures described in Chapter 5.

Features	Patient	Explanation Case
Age (months)	10.2	6.3
Birth	Vaginal	Vaginal
Smoking Mother	No	Yes
Hydration before treatment	5% dehydrated	Normal
O2 saturation before treatment	95	95
Retraction severity before treatment	Mild	Mild
Heart rate after treatment	136	155
Overall increase in work of breathing after treatment	Mild	Mild
Oxygen saturation under 92 after treatment	No (99.0)	No (98.0)
Respiratory rate after treatment	No (30)	No (35)
Temperature over 100.4 after treatment	No (98.0)	Yes (101.0)
Work of breathing after treatment	Improved	Same
Disposition		Discharge

We suggest that this patient should be discharged from hospital.

In support of this recommendation we have the Explanation Case that appears to have been sicker than this patient (due to smoking mother, higher temperature after treatment and worse work of breathing after treatment) but was still discharged from hospital.

However it should be noted that the patients better hydration before treatment in relation to the Explanation Case is a feature that goes against our argument that the explanation case is sicker than the patient.

We have a high confidence in our recommendation

Figure 6.10: Example Case output from the deployed Bronchiolitis Decision Support System.

Chapter 7

Evaluation

Since the foundations of CBR, CBR researchers have argued that case-based explanations are more convincing than other types of explanations. The main argument that case-based explanations are considered to be more convincing is that explanations are based on actual prior cases (Leake 1996). This is a process that is often present in explanations generated by humans. The first part of our evaluation was to perform an empirical analysis to test this hypothesis. The results of this analysis are presented in 7.1.

With empirical evidence that supports the appropriateness of using case-based explanation the usefulness of the explanation utility framework we introduced in Chapter 4 needed to be assessed. The following sections of this chapter presents the results from a number of analyses on the framework. These analyses include:

Usefulness of the Explanation Case (Section 7.2) This analysis was performed on the recommendations and explanations produced by Bronchiolitis Decision Support System described in Section 6.3.

Explanation Case and Nearest Neighbour Comparison (Section 7.3) In this section we performed direct comparisons between the explanations using the explanation case and explanations using the nearest neighbour in the e-Clinic, Breathalyser and Bronchiolitis domains.

Explanation Case Occurrence (Section 7.4) It is possible that the explanation case can also be the nearest neighbour. Regardless of the quality of explanations produced using the explanation case the benefits are not justifiable unless the explanation case is generally not also the nearest neighbour. Here we investigated how often the explanation case isn't one of the three nearest neighbours.

Usefulness of Counter Example (Section 7.5) Including counter examples with explanations can have the possible benefit of allowing users to realise if an incorrect recommendation has been made. In this section we describe an evaluation in which we test if the inclusion of a counter examples results in any significant benefits for a user.

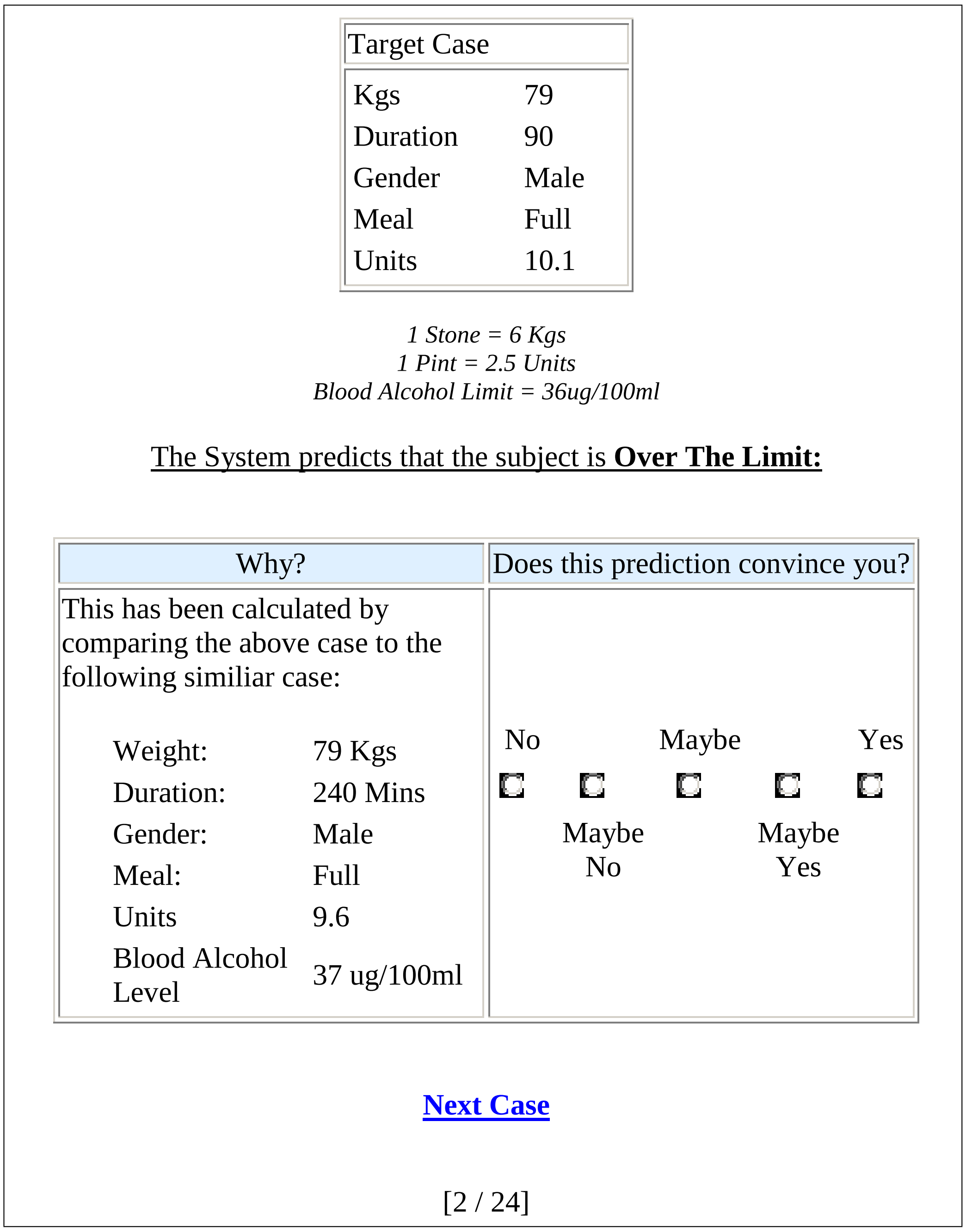
7.1 Analysis of Case Based Explanations

The first part of our evaluation was to evaluate the usefulness of typical knowledge-light CBR explanations compared with rule based explanations or no explanations. This evaluation was performed using an online evaluation with the Breathalyser domain (Section 4.2.1). Eight unique problem cases were used in the experiment. 37 subjects were presented with each of these problem cases three times, once with classifications and case-based explanations, once with classifications and rule-based explanations and once with classifications only without explanation. The rule-based and case-based explanations were presented together but the order was varied to avoid any bias due to familiarity. The format in which the cases and explanations were presented to the user is shown in Figure. 7.1 and figure 7.2. Figure 7.1 shows the case-based explanation while Figure 7.2 shows the rule-based explanation.

The Weka toolkit provides the J48 algorithm, a decision tree-learning algorithm, which is an extension of the C4.5 algorithm (Quinlan 1993). This code was used to produce the decision tree from which the rules were extracted (see Figure 7.3). Weka provides code for automatically extracting rules from a decision tree. This code was not used as the rules it produces are designed to be applied in order. Because of this, rules late in the order are incomplete if used as explanations. Instead we extracted complete rules with a comprehensive rule describing each of the possible paths from the root to the leaves of the tree shown in Figure 7.3. When a new case is passed to the resulting rule-based system for classification, the classification is produced from the rule that covers it and the rule is also returned as an explanation(see figure 7.2). A 10-fold cross validation assessment of the accuracy of the classification system produced a figure of 80%.

The case-based explanation system was also developed on top of Weka. In the similarity metric used, nominal values such as Gender and Meal simply contribute binary similarity scores. The accuracy of the case-based classification was assessed using 10-fold cross validation. Using a single nearest neighbour for classification yielded an accuracy of 81%. In the evaluation, the nearest neighbour was returned as an explanation of the classification.

The subjects were asked to score how convinced they were by the explanations on a 5-point scale (No, Maybe No, Maybe, Maybe Yes, Yes). In the evaluation of the results these scores were interpreted as numeric values from 1-5. The target cases were presented in turn to the subjects and the subjects were able to backtrack to change their score. The subjects were all staff and postgraduate students in the Computer Science Department of Trinity College Dublin and it was explained to them that the objective of the experiment was to compare the usefulness of case-based and rule-based explanation.


Target Case	
Kgs	79
Duration	90
Gender	Male
Meal	Full
Units	10.1

1 Stone = 6 Kgs
1 Pint = 2.5 Units
Blood Alcohol Limit = 36ug/100ml

The System predicts that the subject is **Over The Limit:**

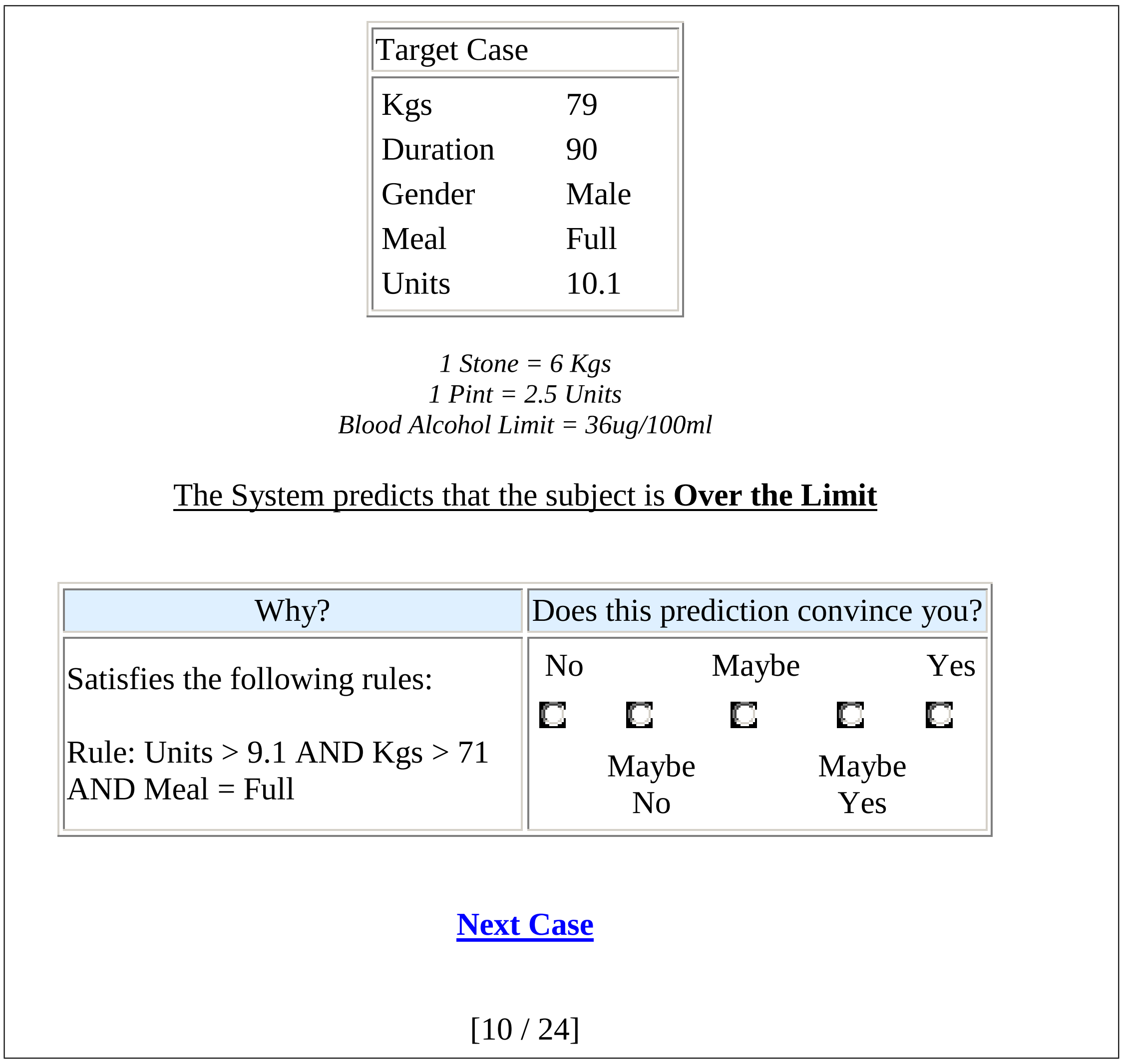
Why?	Does this prediction convince you?
This has been calculated by comparing the above case to the following similiar case:	No ◻ ◻ Maybe No ◻ Maybe ◻ Maybe Yes ◻ Yes

Weight:	79 Kgs
Duration:	240 Mins
Gender:	Male
Meal:	Full
Units	9.6
Blood Alcohol Level	37 ug/100ml

Next Case

[2 / 24]

Figure 7.1: An example case-based classification and explanation from the experiment



Target Case	
Kgs	79
Duration	90
Gender	Male
Meal	Full
Units	10.1

1 Stone = 6 Kgs
1 Pint = 2.5 Units
Blood Alcohol Limit = 36ug/100ml

The System predicts that the subject is **Over the Limit**

Why?	Does this prediction convince you?
Satisfies the following rules: Rule: Units > 9.1 AND Kgs > 71 AND Meal = Full	No ☐ ☐ Maybe No ☐ Maybe ☐ Maybe Yes ☐ Yes

Next Case

[10 / 24]

Figure 7.2: An example rule-based classification and explanation from the experiment

Out of the eight unique problem cases an incorrect classification coupled with a poor explanation was included in each category to help assess the attention paid by subjects to the evaluation. The average rating for these poor classifications was 1.5 while the average for the other classifications was 3.9 (on a scale of 1-5). The ratings for these poor classifications were not considered further in the evaluation. The averages of the remaining ratings are shown in Figure 7.4.

We used the *Student's paired t-test* (Gosset 1908) to assess the statistical significance of the usefulness of case-based explanations over rule-based explanations and against not using explanations at all. This distribution is useful in estimating significance when the sample size is small. This is a parametric test that looks at the evaluation differences between two techniques. Let x_i and y_i be the evaluation value for user i using techniques A and B respectively and d_i be the difference between the evaluations. Using Equation 7.1 we can calculate a value for the paired t:

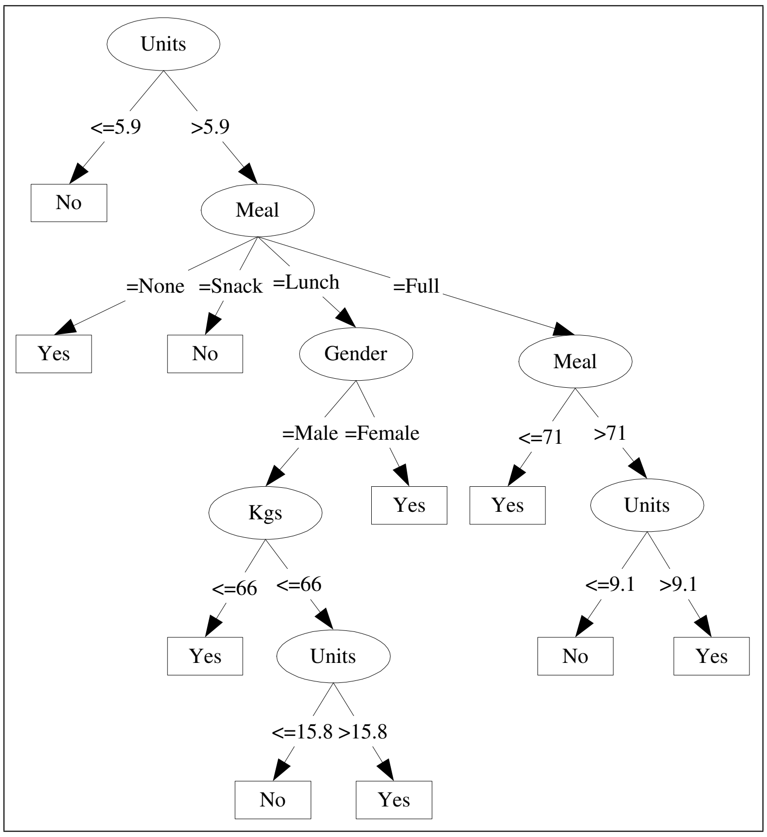


Figure 7.3: The decision tree on which the rule-based explanation system was built

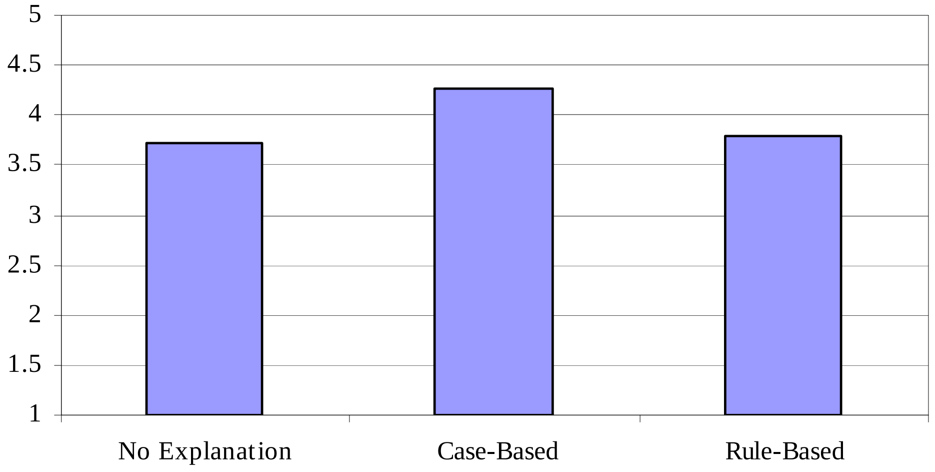


Figure 7.4: The average ratings of the three alternative classification and explanation systems

$$t = \frac{\bar{d}}{\sigma/\sqrt{N}} \tag{7.1}$$

where $\bar{d}$ is the mean difference for all the evaluations, N is the number of evaluations and σ is the standard deviation of the differences as given by Equation 7.2.

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N} (d_i - \bar{d})^2}{N - 1}} \tag{7.2}$$

By using the t value, the number of degrees of freedom df, ($df = N - 1$), and the t test table (Appendix E) we can test the confidence level that technique A is significantly preferred to technique B.

The first test was to check that the case-based explanation was statistically better than using no explanation. Table 7.1 shows the results from this analysis. Here we can see that there is a 99.75% confidence level that case-based explanation is better than no explanation.

Table 7.1: Summary Statistics for case-based explanation against no explanation

Statistic	Value
Number Of Samples (N)	37
Degrees Of Freedom (df)	36
Sample Standard Deviation (σ)	7.1
Paired Student's t (t)	3.32
99.75 Percentile Student's t Distribution ($t_{.9975}$)	2.75

Table 7.2 shows a summary of the statistical analysis of using case-based explanations compared to rule-based explanations. This time there is a 99.95% confidence that the case-based explanations are statistically better than the rule-based explanations. Further analysis of the case-based explanations and rule-based explanations showed that the case-based explanations outperformed rule-based explanations 105 times, while rule-based explanations outperformed case-based explanations 48 times with 106 draws between the two techniques.

Table 7.2: Summary Statistics for case-based explanation against rule-based explanation

Statistic	Value
Number Of Samples (N)	37
Degrees Of Freedom (df)	36
Sample Standard Deviation (σ)	4.75
Paired Student's t (t)	4.22
99.95 Percentile Student's t Distribution ($t_{.9995}$)	3.65

Although these results show a significant benefit in using case-based explanations over rule-based explanations, it also shows that there is room for further improvement for knowledge-light case-based explanations.

7.2 Analysis of the Explanation Case

In the previous section we found empirical evidence that case-based explanations are more useful than rule-based explanations. In this section we evaluate how useful using an explanation case selected by the explanation utility framework (Section 4.3) and explanatory text are in supporting classifications by CBR systems in a real life situation. This evaluation was performed using the Bronchiolitis Decision Support System that is described in Section 6.3. The system was prospectively validated during the months February to April 2005. After completion of patient care, the users were asked to fill out an evaluation for each recommendation and explanation that was produced. The evaluation consisted of three questions:

Question One Do you agree with the suggested course of action?

Question Two Did you find the explanation case useful?

Question Three Did you find the explanatory text useful?

Each of these questions had five options to select from; `Definitely Not`, `No`, `Maybe`, `Yes` and `Absolutely`. Also included in the evaluation was a facility for the user to add any comments they may have. As a result of feedback from these comments the system was updated half way through the evaluation period. Therefore the evaluation was split up into two parts. Section 7.2.1 describes the initial part of the evaluation and the updates made to the system. Section 7.2.2 describes the results of the evaluation after the update.

7.2.1 Results: Part 1

This part of the evaluation ran until mid March 2005. During this period the system was used for 46 recommendations with a classification accuracy of 72%. In total 82 evaluations were collected; 27 from Residents, 43 from Attendings and 12 from Physician Assistants / Nurse Practitioners (PAs/NPs)[1]. Figure 7.5 shows the results from question one of the evaluation. Here we can see that the users agreed with the recommendations a majority of the time.

The results from question two (Figure 7.6) show that both the Residents and the PAs/NPs found the explanation case useful a vast majority of the time; the Residents found the explanation case to be useful 66% of the time while the PAs/NPs found it useful 100% of the time. However an analysis of the results from the Attendings for this question show that they did not find the explanation case as useful as the Residents or the PAs/NPs. In this situation the Attendings only

[1]Attendings have the highest level of expertise. PAs/NPs have less training but more experience than residents.

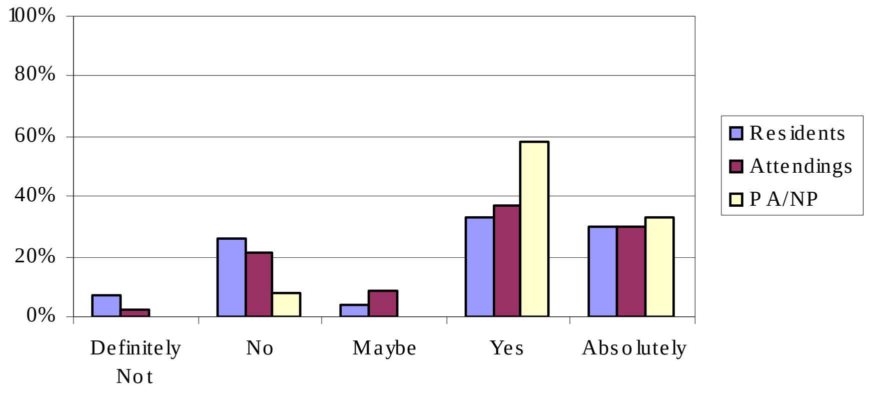


Figure 7.5: Q1: Do you agree with the suggested course of action?

found the case to be useful 46% of the time compared to not useful 37% of the time. This pattern is repeated in the results for question three which examines the usefulness of the explanatory text (Figure 7.7).

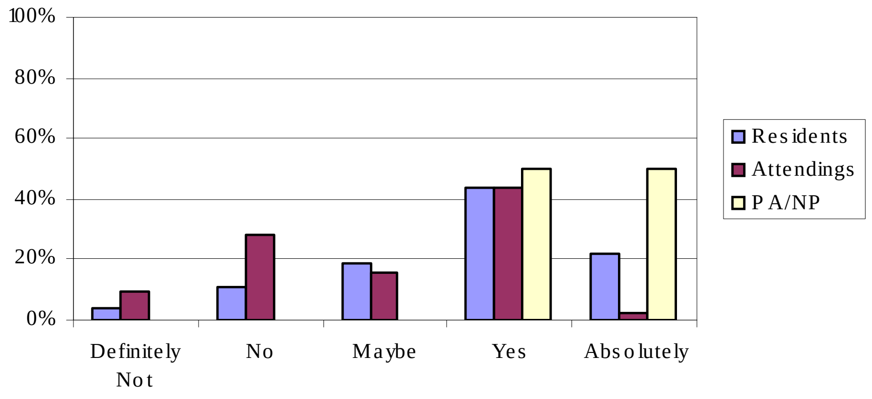


Figure 7.6: Q2: Did you find the explanation case useful?

An analysis of the comments helped to explain why the Attendings did not find the explanation case as useful as the Residents or the PAs/NPs. There were a number of comments by Attendings that complained about the age difference being too great between the explanation case and the target case to allow a suitable comparison between the two cases. In the comments the Attendings often suggested that a less marginal case with respect to the patients `age` would be more useful.

These comments suggested that we should modify the shape of the utility graph for the age feature to reduce the tendency to invoke extreme example cases. The original explanation utility measure for age when the classification is `discharge` is shown in figure 7.8. In this scenario consider a query case, q, with `Age` equal to 15 months and a retrieved case, x, with `Age` equal to 10 months. The difference between these two values $(q - x)$ is +5. By looking up the graph it can be seen that a difference of +5 returns an explanation utility of 1.

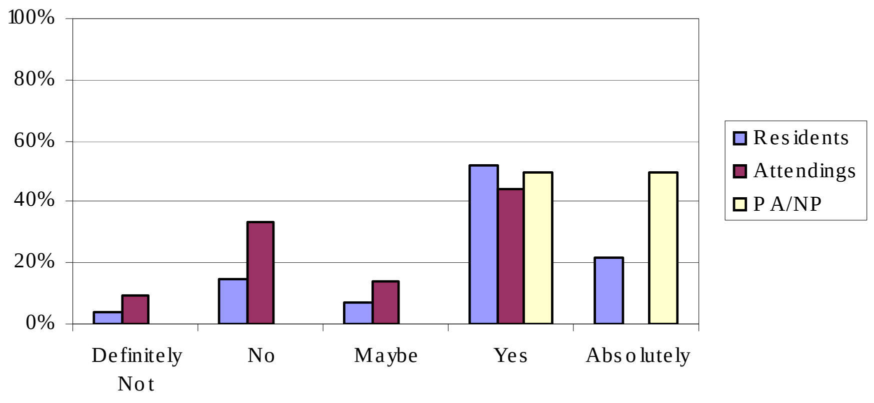


Figure 7.7: Q3: Did you find the supporting explanatory text useful?

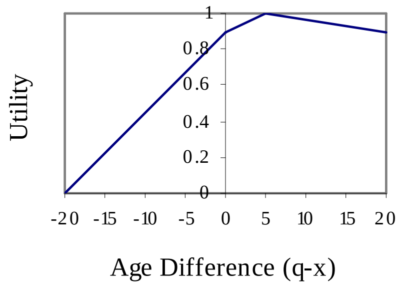


Figure 7.8: Original age explanation utility measure for a discharge disposition

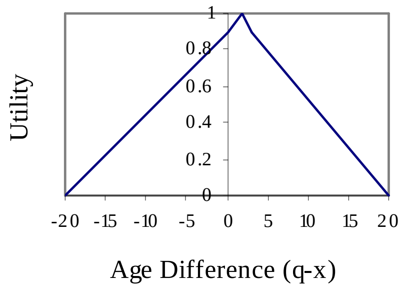


Figure 7.9: Updated age explanation utility measure for a discharge disposition

An explanation utility of 1 is high for an age difference of +5 months if we wish to retrieve cases that are closer in age to the patient. Figure 7.9 shows the updated explanation utility metric. Here we can see that the utility of a difference of +5 has now dropped, while smaller differences remains high.

7.2.2 Results: Part 2

The second part of the evaluation used the updated utility measures as described in the previous section and ran from mid March until the end of April. A predication accuracy of 80% was achieved on the 65 patients presented with bronchiolitis during this part of the evaluation. In this part there were 106 evaluations; 39 from Residents, 59 from Attendings and 8 from PAs/NPs. In question one we looked at the users overall confidence in the recommendation generated by the system. Figure 7.10 shows that over 85% of each group had either some confidence or total confidence in the recommendation.

In question two we were checking to see if the users found the explanation case to be a suitable case for explaining the recommendation. The results for this question are shown in Figure 7.11. Here we can see that 54% of Residents, 56% of Attendings and 100% of PAs/NPs found the

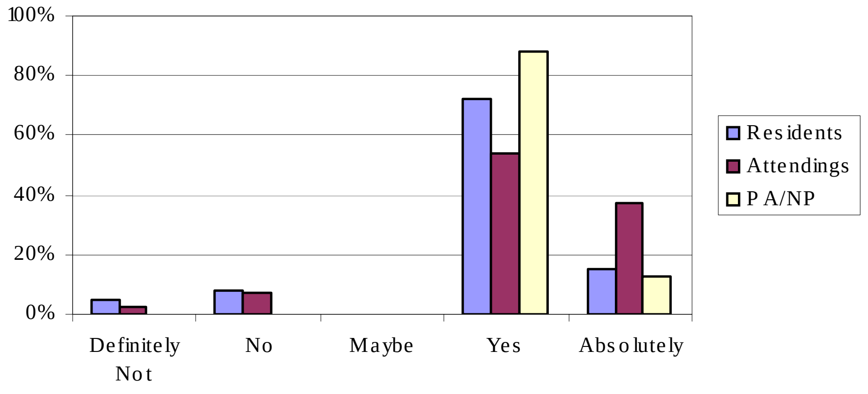


Figure 7.10: Q1: Do you agree with the suggested course of action?

explanation case to be useful. It is the responses on this question from the Attendings that had the most significant change when compared to the first part of the evaluation. In the first part 44% of Attendings answered `yes` to this question, while only 2% answered `absolutely` while in the second part this increased to 49% and 7% respectively. Although it could be argued that this increase could be partially attributed to an increase in accuracy of the system from 73% to 80%, it appears that the increase is mainly due to patients of a more similar age being used as explanation cases. This is shown by a substantial reduction in comments complaining about an excessive age gap. As more cases are added to the case base this problem should be completely eliminated. However it should be noted that the increase in acceptance of the Explanation Case by the Attendings is traded off by a reduction in acceptance with Residents and PAs/NPs. This shows that different utility measures could be used depending on the persons level of expertise. This agrees with Sørmo and Cassens (2004) who argue that explanations are both domain and user dependent, and this should be reflected in the case retrieval process.

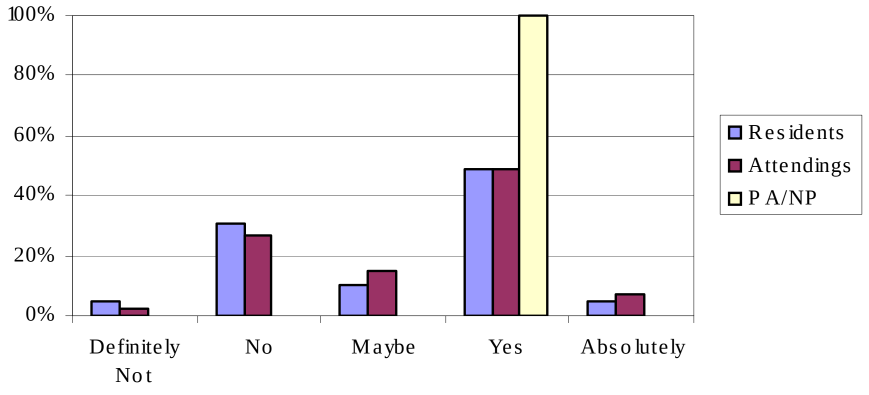


Figure 7.11: Q2: Did you find the explanation case useful?

Finally in question three we are interested in the quality of the generated explanatory text.

Figure 7.12 shows the results for this part of the evaluation. As with the explanation case the majority were happy with the explanatory text. Also in a similar manner to question two there was an increase in acceptance by Attendings but also a decrease in acceptance by Residents and PA/NPs when compared to the results from the first part of this evaluation.

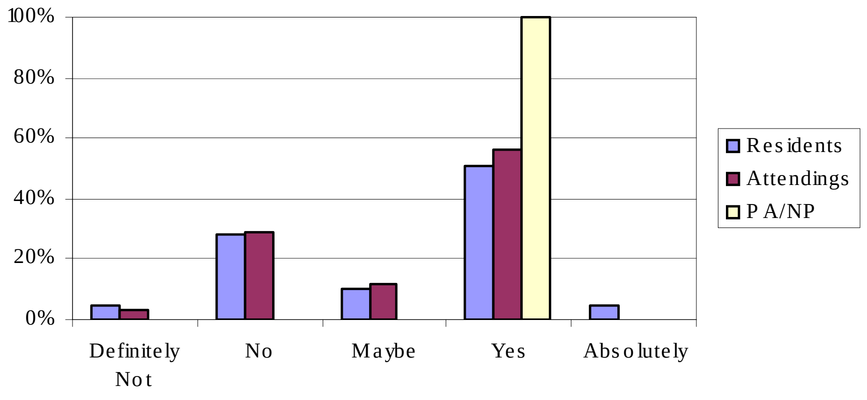


Figure 7.12: Q3: Did you find the supporting explanatory text useful?

7.3 Comparison of Explanation Case to Nearest Neighbour

In the last section we presented results that showed that explanation cases, retrieved by explanation utility measures, are a suitable technique for generating explanations in a knowledge-light CBR system. In order to determine the full usefulness of these cases a direct comparison against the use of the nearest neighbour as an explanation needs to be performed. This evaluation was performed by evaluating domains with a low number of directional features separate to domains with a higher number of directional features. Table 7.3 shows the number of directional features in the three domains that we used. From this table we can see that the e-Clinic and Breathalyser domains have a low number of directional features, having three and four directional features respectively, compared to the Bronchiolitis domain which has ten directional features. In this section we will present the results from evaluations on domains with low and high numbers of directional features.

Table 7.3: Directional Features

Domain	Number of Features	Number of Directional Features
e-Clinic	5	3
Breathalyser	5	4
Bronchiolitis	12	10

7.3.1 Low Number of Directional Features

In order to support the assertion that our explanation cases are in fact better explanations than the nearest neighbour, we asked an expert in the e-Clinic domain to evaluate some of the results. The expert was presented with nine target cases and associated nearest neighbour and explanation cases - labelled as Explanation 1 and Explanation 2. In eight out of the nine cases the domain expert indicated that the case selected by the explanation utility measure was more convincing than the nearest neighbour.

Due to a difficulty in obtaining experts in the e-Clinic domain to further our evaluation we performed a similar evaluation using the breathalyser domain. In this situation ten unique problem cases were used in the experiment; 5 cases over the legal drink driving limit and 5 cases under the limit in a random order. Each problem had an associated recommendation and two supportive cases; the nearest neighbour and an explanation case retrieved using the explanation utility measures. To remove bias these two cases were labelled Case 1 and Case 2 and their ordering randomised. The subjects were asked to select the case that they felt was most supportive of the recommendation. They had the option to indicate that they found both cases to be equally supportive or if they found neither case to be supportive. An example case is displayed in figure 7.13.

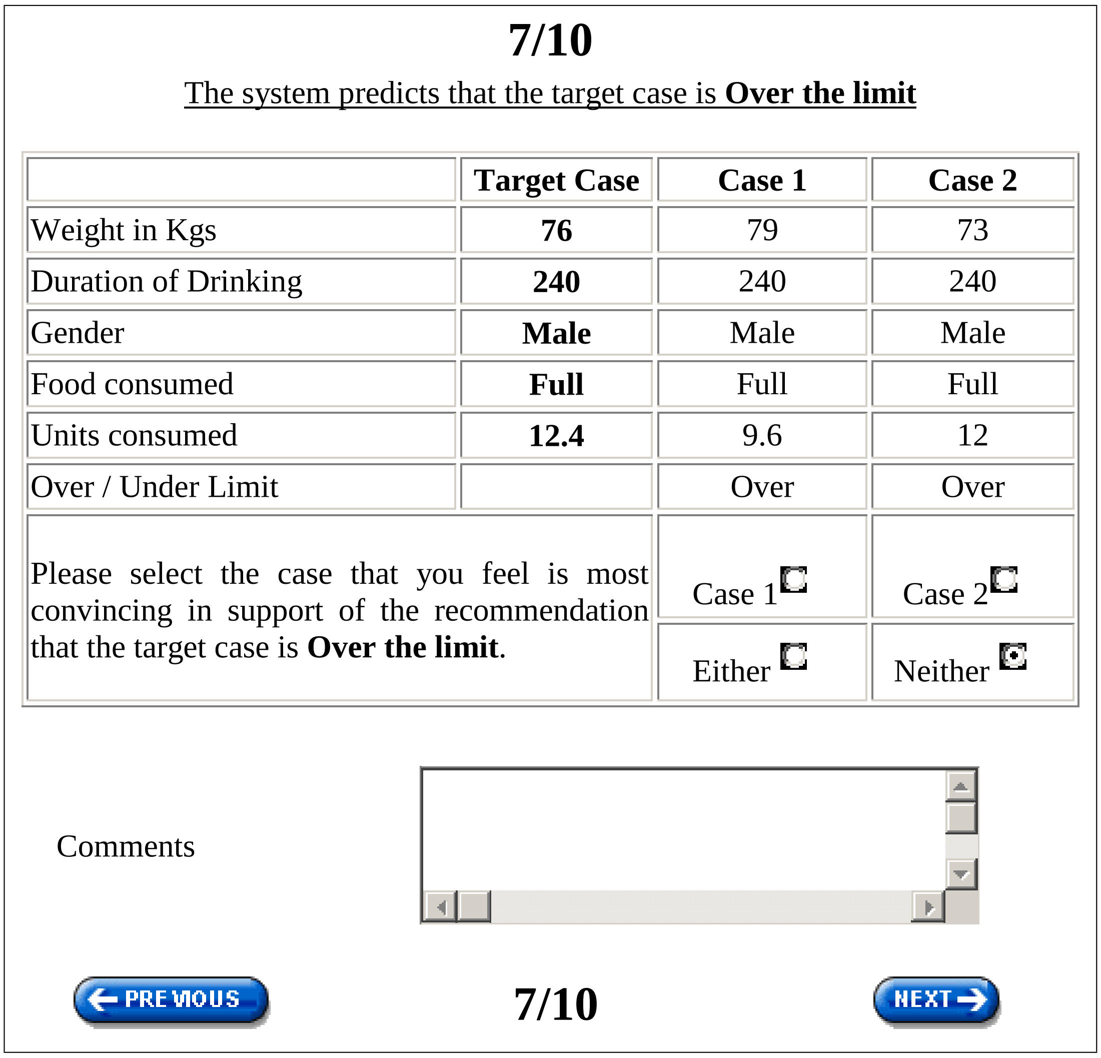
In total 13 subjects from the Computer Science Department of Trinity College Dublin performed the evaluation[2]. The time spent per question was recorded to help assess the attention paid by subjects to the evaluation. The average time taken to complete the evaluation was over ten minutes. Any users spending less than five minutes performing the evaluation were removed from the evaluation.

Table 7.4 shows a summary of the results from this evaluation. Here we can see that the explanation case was found to be more supportive of the recommendation over five times as often as the nearest neighbour. On further analysis we can see that the majority of the subjects selected the explanation case over the nearest neighbour a majority of the time. The only exception to this was subject 3 who showed a strong preference for the nearest neighbour.

7.3.2 High Number of Directional Features

Both the Breathalyser and the e-Clinic domains have a small number of directional features to be considered. It has in total five features with the units consumed being the most dominant feature by far. The Bronchiolitis domain was therefore used to compare the explanation case to the nearest neighbour. The layout of this evaluation was the same as the evaluation used with the Breathalyser domain. Again 10 unique problem cases were selected; five with recommendations to discharge and five with recommendations to admit.

[2]The subjects were from the Image Synthesis Group and the Centre for Telecommunications Value-Chain-Driven Research (Subjects were selected to exclude members of the Machine Learning Group).



7/10

The system predicts that the target case is **Over the limit**

	Target Case	**Case 1**	**Case 2**
Weight in Kgs	**76**	79	73
Duration of Drinking	**240**	240	240
Gender	**Male**	Male	Male
Food consumed	**Full**	Full	Full
Units consumed	**12.4**	9.6	12
Over / Under Limit		Over	Over

Please select the case that you feel is most convincing in support of the recommendation that the target case is **Over the limit**.

Case 1 ○ Case 2 ○ Either ○ [✓ Neither]

Comments

Figure 7.13: Evaluation of the Explanation Case compared to the Nearest Neighbour in the Breathalyser Domain.

Table 7.4: Summary results of Explanation Case v's Nearest Neighbour in the Breathalyser domain.

Subject	Explanation Case	Nearest Neighbour	Neither	Either
1	8	2	0	0
2	6	1	1	2
3	0	8	0	2
4	7	0	2	1
5	8	1	0	1
6	7	1	2	0
7	9	1	0	0
8	9	1	0	0
9	9	0	0	1
10	10	0	0	0
11	5	2	0	3
12	9	0	0	1
13	9	1	0	0
Total	96	18	5	11

Table 7.5: Summary results of Explanation Case v's Nearest Neighbour in the Bronchiolitis domain.

Subject	Explanation Case	Nearest Neighbour	Neither	Either
1	6	4	0	0
2	2	8	0	0
3	1	9	0	0
4	4	1	1	4
5	1	5	0	4
6	1	8	0	1
7	1	7	0	2
8	1	8	0	1
9	6	2	0	2
10	0	8	1	1
11	1	1	3	5
12	7	2	1	0
13	1	2	1	6
14	0	7	0	3
Total	32	72	7	29

The evaluation was performed by 14 subjects. The subjects were all staff from the Emergency Department of Kern Medical Center, Bakersfield, California. Table 7.5 shows a summary of the results from this evaluation. In this situation the nearest neighbour was considered to be more supportive of the recommendation than the explanation case.

It is clear that the explanation framework works well for the e-Clinic and Breathalyser domains but is not favoured in the Bronchiolitis domain. We feel this is because of the increased complexity of the Bronchiolitis cases which have 10 directional features compared to three and four respectively in the e-Clinic and Breathalyser domains.

7.4 Explanation Case Occurrence

An important question is, how often will a case selected using the explanation utility measures actually be different to the nearest neighbour. In case-based explanation using nearest neighbours, it is reasonable to assume that explanation will be based on the top cases retrieved. This evaluation involved performing a leave-one-out validation on the Breathalyser, Bronchiolitis and e-Clinic datasets to see how often cases selected by the explanation utility measure were not among the nearest neighbours (the top three were considered). The results of this evaluation are shown in Table 7.6. Here we can see that the explanation case was found outside the three nearest neighbours a majority of the time. Thus, useful explanation cases - according to this framework - are not necessarily nearest neighbours and would not normally be presented to the user.

Table 7.6: Explanation Case Usage

	1st Nearest Neighbour	2nd Nearest Neighbour	3rd Nearest Neighbour	Total first 3 Neighbours
Breathalyser	10%	7%	14%	31%
Bronchiolitis	18%	13%	10%	41%
e-Clinic	4%	3%	3%	10%

7.5 Counter Example

The explanation utility measures attempt to retrieve cases that lie between a problem case and the decision boundary. A possible improvement to just displaying the explanation case is to also display a counter example. The HYPO system (Ashley 1989) contests arguments by also citing a past case as a counter example. The motivation for showing the counter example is to give the user a sense of the "robustness" of the classification. If the nearest counter example is quite different to the query case then the user can have some confidence that the recommendation is correct. If the counter example is close to the query then classification may be more marginal. The counter example could play an important role when classification is incorrect as it could show that the situation is marginal. However one problem with displaying the counter example is that it adds to the amount of information a user has to process when presented with a recommendation and explanation. This can cause information overload for the user and end up confusing the user.

To test the usefulness of displaying a counter example we set up an evaluation using the Bronchiolitis domain. The evaluation was made up of ten unique problem cases; 5 with correct recommendations and 5 with incorrect recommendations randomly order. For each of the problem cases subjects were presented with the problem case, recommendation, explanation case and counter example. The counter example was selected as the most similar case with a different classification to the recommendation. The evaluation comprised of two questions:

Question One Do you consider the recommendation to be correct?

Question Two Do you think the counter example was useful?

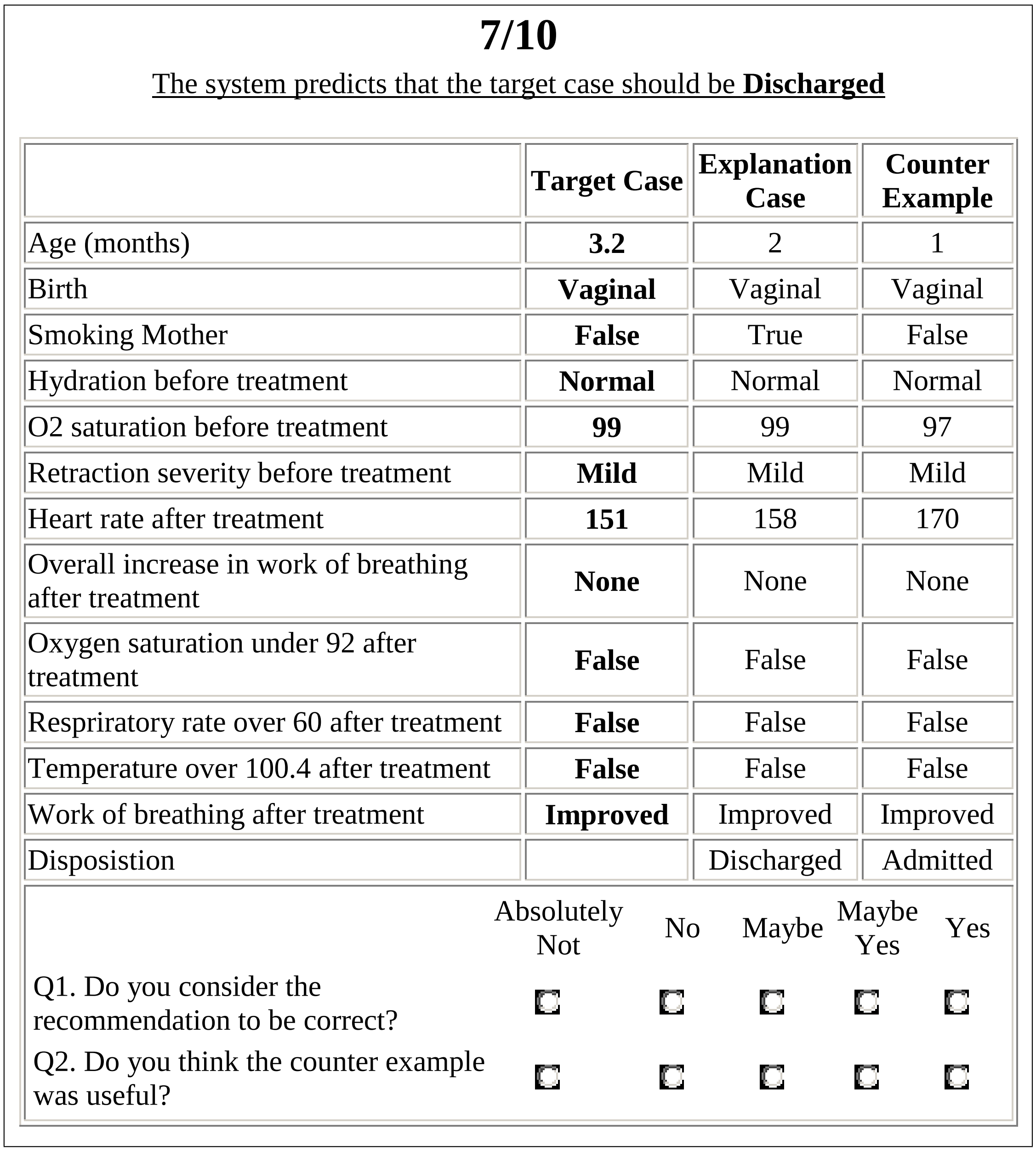
Each of these questions were scored on a 5-point scale using five options to select from; `Absolutely Not`, `No`, `Maybe`, `Maybe Yes` and `Yes`. Also included in the evaluation was a facility for the user to add any comments they may have and the ability to backtrack to change their score. In the evaluation of the results these scores were interpreted as numeric values from 1-5. Figure 7.14 shows an example from this evaluation. The subjects were again staff from the Emergency Department of Kern Medical Center, Bakersfield, California. The time spent per question was recorded to help assess the attention paid by subjects to the evaluation.

In total 12 subjects performed the evaluation. Table 7.7 shows the average ratings on a scale of 1-5 for both questions amongst the 12 subjects for the correct and incorrect recommendations. The results from question one show that users had a good understanding of the accuracy of the recommendations. The average rating for correct recommendations is 4.6 out of a possible max of 5. On the other hand incorrect recommendations had an average rating of 2.9. The results from question two present a more interesting picture. The average rating for the incorrect recommendations is higher than for correct recommendations. This shows that users found the inclusion of the counter example more useful when they believed that the case had an incorrect recommendation than when it had a correct recommendation. In spite of the subjects finding the counter example useful for detecting incorrect recommendations overall they did not find the counter example very useful. This is shown by the poor overall ratings for question two where the average value of usefulness of the counter example for incorrect recommendations of 3.3 equating to a slightly positive maybe and the average rating for correct recommendations of 2.9 equating to a slightly negative maybe.

Table 7.7: Results from the evaluation of the usefulness of using a counter example in explanations.

	Q1	Q2
Average Rating Correct Recommendations	4.6	2.8
Average Rating Incorrect Recommendations	2.9	3.3

We conclude from this that any benefits the counter example might offer in providing insight on incorrect recommendations will be more than offset by creating confusion around correct recommendations.



7/10

The system predicts that the target case should be **Discharged**

	Target Case	**Explanation Case**	**Counter Example**
Age (months)	**3.2**	2	1
Birth	**Vaginal**	Vaginal	Vaginal
Smoking Mother	**False**	True	False
Hydration before treatment	**Normal**	Normal	Normal
O2 saturation before treatment	**99**	99	97
Retraction severity before treatment	**Mild**	Mild	Mild
Heart rate after treatment	**151**	158	170
Overall increase in work of breathing after treatment	**None**	None	None
Oxygen saturation under 92 after treatment	**False**	False	False
Respriratory rate over 60 after treatment	**False**	False	False
Temperature over 100.4 after treatment	**False**	False	False
Work of breathing after treatment	**Improved**	Improved	Improved
Disposistion		Discharged	Admitted

	Absolutely Not	No	Maybe	Maybe Yes	Yes
Q1. Do you consider the recommendation to be correct?	○	○	○	○	○
Q2. Do you think the counter example was useful?	○	○	○	○	○

Figure 7.14: Evaluation of the usefulness of using a counter example in explanations.

7.6 Conclusion

The first part of the evaluation performed in this chapter was to evaluate the effectiveness of case-based explanation over rule-based explanations or not using explanations at all. This evaluation found that case-based explanations were statistically more convincing than the alternatives.

The next stage was to evaluate the explanations produced by the explanation utility framework described in Chapter 4. This evaluation was performed using the Bronchiolitis Decision Support System described in Section 6.3. The results from this evaluation showed a high acceptance for both the explanation case and the explanatory text. The most significant finding from this evaluation was the need to update the utility metrics for the feature `Age`. This shows that a major benefit of using the utility measures is the ease of changing the retrieval preference without a major workload.

The results of an evaluation that compared the usefulness of this explanation case to the simple alternative of selecting the nearest neighbour showed that in the absence of explanatory text the nearest neighbour was found to be more useful than the explanation case in a domain with a high number of directional features. While the benefit of the explanation case over the nearest neighbour is lost in domains with a high number of directional features the knowledge-light explanation framework still offers the benefit of the explanatory text.

The final part of the analysis of the explanation utility measures was to determine how often the explanation case is one of the nearest neighbours. The results on three domains showed that for over a majority of the time the explanation case was not one of the three nearest neighbours and at most was the nearest neighbour 18% of the time.

The final part of out evaluation was on the usefulness of the counter example in explanations. Although the counter example proved to be of some benefit for users when an incorrect recommendation was made overall it did not show any significant benefits as it seemed to be a hinderance when correct recommendations were made.

Chapter 8

Conclusions and Future Work

8.1 Overview

This thesis described the Bronchiolitis Decision Support System, an application for recommending if a child presented to an Emergency Department should be admitted or discharged. These type of systems often fail as users are not convinced by their recommendations. Using case-based reasoning we have developed an explanation utility framework and confidence assessment system that can be used to produce more convincing explanations.

Although our explanation utility framework and confidence assessment system described in this thesis were developed with the Bronchiolitis Decision Support System in mind, we believe that these techniques can be applied to other systems. This is supported by our evaluations of the techniques in both the e-Clinic and Breathalyser domains. Also the success of the confidence measures in the spam domain, shows their potential for use in a wide range of CBR Systems.

8.2 Case-Based Explanation

The first stage of our work was to show that case-based explanations are in fact a convincing method for developing explanations. We performed an evaluation of case-based explanations against rule-based explanations and having no explanations. The results of this evaluation showed, with a statistical significance, that case-based explanations are more convincing than the alternative approaches. This provided concrete evidence to the usefulness of case-based explanations which has been argued by the CBR community for years.

8.3 Explanation Utility Framework

The explanation utility framework we developed consists of two parts. The first part is the retrieval of the most suitable case to use as an *a fortiori* argument. A real time evaluation of these explanation cases showed that they are useful for producing convincing explanations. The second part of the framework is the generation of explanatory text to accompany the explanation case as the part of the overall explanation. Evaluations showed that this text is useful in supporting explanations. In particular the explanatory text is more beneficial as the number of directional features increases.

8.4 Assessing Confidence

We implemented confidence measures for use in case-based systems. These measures are generally based on the similarity measures used for retrieval. The measures attempt to approximate the distance of a problem case to the decision boundary. The greater the distance the higher the confidence level. Although the confidence measures worked well in the Spam and Breathalyser domains they did not perform well in the Bronchiolitis domain. In our evaluations we discovered that these types of measures are highly influenced by the inclusion of new cases to the case-base. Unfortunately these influences generally have a detrimental affect on confidence levels.

8.5 Bronchiolitis Decision Support System

We developed a prototype system, called the Bronchiolitis Decision Support System. This system incorporated the explanation utility framework and confidence measures. The results from an evaluation of this system showed that overall the medical professionals using the system were satisfied with its explanations. As the Bronchiolitis season only runs for a small number of months a year, typically 3-4 months, many medical professionals do not see many cases of Bronchiolitis. It is hoped that this system could also be a useful training system helping resident doctors to retrieve suitable past cases.

8.6 CBR Representation

In Section 6.1 we presented CBML, an XML-based CBR representation language. CBML separates three of Richter knowledge containers, the domain knowledge, case data and similarity knowledge, from application code into ASCII documents. This makes it easier for developers to alter the CBR process by changing the CBML documents without having to recompile any code.

CBML is an important component of the Bronchiolitis Decision Support System. It assisted in easy updating of the Bronchiolitis Decision Support System once it was active. This was an important aspect considering the distance between Dublin and Bakersfield California. As a further demonstration of the usefulness of CBML we also mentioned work been done by other researchers in a diverse range of research areas to represent their CBR data. These areas include recommender systems, spam filtering and feature selection.

8.7 Future Work

Explanation Utility Framework

As a result of CBML being separate to application code this allows for each user to have their own set of explanation utility measures. This is useful in situations where different users have different preferences for what they consider to be convincing explanation cases. Some users may prefer marginal cases while other users may prefer less marginal cases. In order to implement such a system new users would start off with the original measures defined by the domain expert. As they use the system if they don't find a case convincing they would then select what they consider to be a more convincing case. Based on the selected case the utility measures would be adjusted to favour the selected case and to reduce the utility for the originally retrieved case.

Confidence Measures

Although the confidence measures were useful in some domains, the introduction of new cases into the case-base resulted in the measures becoming unstable. These findings correspond with the findings of Cheetham and Price (2004). Further work is needed in the area of developing confidence measures for case-based systems that remain stable after the addition of new cases to the case-base.

CBML

The current implementation of CBML can represent three of Richter's four knowledge containers — the case base, the similarity measure and the domain vocabulary. It is clear that if CBML is to develop further, the next step should be the development of a representation format for the fourth container — adaptation knowledge.

Bibliography

Aamodt, A.: 1991, *A Knowledge-Intensive, Integrated Approach to Problem Solving and Sustained Learning*, PhD thesis, Norwegian Institute of Technology, Department of Computer Science, Trondheim.
URL: *http://www.idi.ntnu.no/grupper/su/publ/phd/aamodt-thesis.pdf*

Aamodt, A. and Plaza, E.: 1994, Case-based reasoning: Foundational issues, methodological variations, and system approaches., *Artificial Intelligence Communications* **7**(1), 39–59.

Achinstein, P.: 1983, *The Nature of Explanation*, Oxford University Press, Oxford.

Aha, D. and Bankert, R.: 1994, Feature selection for case-based classification of cloud types: An empirical comparison, *in* D. Aha (ed.), *Case-Based Reasoning: Papers from the 1994 Workshop (Technical Report WS-94-01)*, AAAI Press., Menlo Park, CA:.

Aha, D., Breslow, L. and Munoz-Avila, H.: 2001, Conversational case based reasoning, *Applied Intelligence, special issue on Interactive CBR* **14(1)**, 9–32.

Aha, D. W.: 1997, Special issue on lazy learning, *Artificial Intelligence Review* **11**, 7–10.

Alpaydin, E.: 2004, *Introduction to Machine Learning*, The MIT Press, chapter 8, pp. 173–196.

Althoff, K.-D. and Richter, M.: 1999, *Similarity and Utility in Non-Numerical Domains*, Physika-Verlag, pp. 403–413. http://www.cbr-web.org/documents/RichterSimilarity99.pdf.

Andrews, R., Diederich, J. and Tickle, A.: 1995, A survey and critique of techniques for extracting rules from trained artificial neural networks, *Knowledge Based Systems* **8**, 187–202.

Armengol, E., Palaudries, A. and Plaza, E.: 2001, Individual prognosis of diabetes long-term risks: A CBR approach, *Methods of Information in Medicine: Special issue on prognostic models in Medicine* **40**, 46–51.

Ashley, K.: 1991, Reasoning with cases and hypotheticals in hypo., *International Journal of Man-Machine Studies* **34**, 753–796. Academic Press. New York.

Ashley, K. and Aleven, V.: 1997, Reasoning symbolically about partially matched cases, *In Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97)*, Morgan Kaufmann: San Francisco, Nagoya, Japan, pp. 335–341.

Ashley, K. D.: 1987, *Modelling Legal Argument: Reasoning with Cases and Hypotheticals*, PhD thesis, Department of Computer and Information Science, University of Massachusetts.

Ashley, K. D.: 1989, Defining salience in case-based arguments, *in* N. Sridharan (ed.), *Eleventh International Joint Conference on Artificial Intelligence (IJCAI-89)*, Morgan Kaufmann, San Mateo, CA, pp. 537–542.

Baker, M. and Ruddy, R.: 2000, Pulmonary emergencies, *in* G. Fleischer and S. Ludwig (eds), *Textbook of Pediatric Emergency Medicine*, 4 edn, Lippincott Williams & Wilkins, Philadelphia, pp. 1067–1087.

Bergmann, R.: 1993, Integrating abstraction, explanation-based learning from multiple examples and hierarchical clustering with a performance component for planning, *in* E. Plaza (ed.), *Proceedings of the ECML-93 Workshop on Integrated Learning Architectures (ILA-93)*, Vienna, Austria.

Bergmann, R.: 1998, The use of taxonomies for representing case features and local similarity measures, *in* L. Gierl and M. Lenz (eds), *6th German Workshop on CBR.*

Bergmann, R.: 2002, *Experience Management: Foundations, Development Methodology, and Internet-Based Applications.*, Vol. 2432 of *Lecture Notes in Artificial Intelligence*, Springer-Verlag Berlin Heidelberg New York.

Bergmann, R., Richter, M. M., Schmitt, S., Stahl, A. and Vollrath, I.: 2001, Utility-oriented matching: A new research direction for case-based reasoning, *Proceedings of the 9th German Workshop on Case-Based Reasoning, GWCBR'01, Baden-Baden, Germany*, pp. 264–274.

Bergmann, R. and Stahl, A.: 1998, Similarity measures for object-oriented case representations, *in* B. Smyth and P. Cunningham (eds), *Proceedings of EWCBR 1998*, Vol. 1488 of *Lecture Notes in Artificial Intelligence*, p. 2536.

Bridge, D. and Cummins, L.: 2005, Knowledge lite explanation oriented retrieval, *to appear AAAI Fall Symposium on EXPLANATION-AWARE COMPUTING (ExaCt 2005)*, Washington DC (USA).

Brighton, H. and Mellish, C.: 2002, Advances in instance selection for instance-based learning algorithms., *Data Mining and Knowledge Discovery* **6**(2), 153–172.

Brzillon, P., Pomerol, J. (eds.), C. and Hall, pp44-60, .: 1996, Misuse and nonuse of knowledge-based systems: The past expreiences revisited, *in* P. Humphreys, L. Bannon, A. McCosh,

P. Migliarese and J. Pomerol (eds), *Implementing Systems for Supporting Management Decisions*, Chapman and Hall, pp. 44–60.

Carney, M., Cunningham, P., Dowling, J. and Lee, C.: 2005, Predicting probability distributions for surf height using an ensemble of mixture density networks, *In Proceedings of the 22nd International Conference in Machine Learning*, ACM, Bonn, Germany, pp. 113–121.

Cestnik, B., Kononenko, I. and Bratko, I.: 1987, Assistant 86: a knowledge-elicitation tool for sophisticated users, *Progress in Machine Learning*, Sigma, Wilmslow, England, pp. 31–45.

Chandrasekaran, B., Tanner, M. C. and Josephson, J. R.: 1989, Explaining control strategies in problem solving, Vol. 4, IEEE Educational Activities Department, Piscataway, NJ, USA, pp. 9–24.

Cheetham, W.: 2000, Case-based reasoning with confidence., *in* E. Blanzieri and L. Portinale (eds), *Advances in Case-Based Reasoning, 5th European Workshop, EWCBR 2000, Trento, Italy, September 6-9, 2000, Proceedings*, Vol. 1898 of *Lecture Notes in Computer Science*, Springer, pp. 15–25.

Cheetham, W. and Price, J.: 2004, Measures of solution accuracy in case-based reasoning systems., *in* P. Funk and P. A. González-Calero (eds), *Advances in Case-Based Reasoning, 7th European Conference, ECCBR 2004, Madrid, Spain, August 30 - September 2, 2004, Proceedings*, Vol. 3155 of *Lecture Notes in Computer Science*, Springer, pp. 106–118.

Coyle, L.: 2004, *Making Personalised Flight Reccommendations using Implicit Feedback*, PhD thesis, University of Dublin, Trinity College.

Coyle, L., Doyle, D. and Cunningham, P.: 2004, Representing similarity for CBR in XML, *in* P. A. G. Calero and P. Funk (eds), *Advances in Case-Based Reasoning, 7th European Conference on Case-Based Reasoning, ECCBR 2004*, Vol. 3155, Springer, Madrid, Spain, pp. 119–127.

Coyle, L., Hayes, C. and Cunningham, P.: 2003, Representing cases for CBR in XML, *Expert Update* **6**(2), 7–13.

Craw, S., Wiratunga, N. and Rowe, R.: 1998, Case-based design for tablet formulation., *in* B. Smyth and P. Cunningham (eds), *Advances in Case-Based Reasoning, 4th European Workshop, EWCBR-98, Dublin, Ireland, September 1998, Proceedings*, Springer, pp. 358–369.

Cunningham, P.: 1998, CBR: strengths and weaknesses, *in* A. del Pobil, J. Mira and M. Ali (eds), *Proceedings of 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, number 1416 in *LNAI*, Springer, pp. 517–523.

Cunningham, P., Doyle, D. and Loughrey, J.: 2003, An evaluation of the usefulness of case-based explanation, *in* K. D. Ashley and D. G. Bridge (eds), *Case-Based Reasoning Research* *and Development, 5th International Conference on Case-Based Reasoning*, Vol. 2689 of *Lecture Notes in Computer Science*, Springer, ICCBR 2003, Trondheim, Norway, pp. 122–130.

Cunningham, P., Nowlan, N., Delany, S. J. and Haahr, M.: 2003, A case-based approach to spam filtering that can track concept drift, *In The ICCBR'03 Workshop on Long-Lived CBR Systems, Trondheim, Norway, June 2003. Available: http://www.cs.tcd.ie/publications/tech-reports/reports.03/TCD-CS-2003-16.pdf.*

Davis, R.: 1982, Expert systems: Where are we? and where do we go from here?, *AI Magazine* **3**(2), 3–22.

Davis, R. and Buchanan, B.: 1985, Meta level knowledge, *in* F. Hayes-Roth, D. A. Waterman and D. B. Lenat (eds), *Rule-Based Expert Systems*, Addison-Wesley, London, pp. 507–530.

Davy, M.: 2005, Active learning in text classification, *Technical report*, Department of Computer Science, Trinity College Dublin.

Delany, S. J. and Cunningham, P.: 2004, An analysis of case-base editing in a spam filtering system., *in* P. Funk and P. A. González-Calero (eds), *Advances in Case-Based Reasoning, 7th European Conference, ECCBR 2004, Madrid, Spain, August 30 - September 2, 2004, Proceedings*, Vol. 3155 of *Lecture Notes in Computer Science*, Springer, pp. 128–141.

Delany, S. J., Cunningham, P. and Coyle, L.: 2004, An assessment of case-based reasoning for spam filtering, *in* L. McGinty and B. Crean (eds), *Proceedings of the Fifteenth Irish Conference on Artificial Intelligence and Cognitive Science (AICS'2004)*, pp. 9–18.

Delany, S. J., Cunningham, P. and Coyle, L.: 2005, An assessment of case-based reasoning for spam filtering, *Artificial Intelligence* p. (to appear).

Delany, S. J., Cunningham, P., Doyle, D. and Zamolotskikh, A.: 2005, Generating estimates of classification confidence for a case-based spam filter, *in* H. Muñoz-Avila and F. Ricci (eds), *6th International Conference on Case-Based Reasoning*, Vol. 3620 of *LNAI*, Springer, pp. 177–190.

Delany, S. J., Cunningham, P., Tsymbal, A. and Coyle, L.: 2004, A case-based technique for tracking concept drift in spam filtering, *in* A. Macintosh and T. Ellis, R. amd Allen (eds), *Applications and Innovations in Intelligent Systems XII, Procs. of AI-2004*, Springer, pp. 3–16.

Doyle, D., Cunningham, P., Bridge, D. and Rahman, Y.: 2004, Explanation oriented retrieval, *in* P. Funk and P. Calero (eds), *Advances in Case-Based Reasoning (Procs. of the Seventh European Conference on Case-Based Reasoning)*, Springer, pp. 157–168.

Falkman, G.: 2002, The use of a uniform declarative model in 3d visualisation for case-based reasoning, *ECCBR '02: Proceedings of the 6th European Conference on Advances in Case-Based Reasoning*, Springer-Verlag, London, UK, pp. 103–117.

Finch, I.: 1999, Knowledge-based systems, viewpoints and the world wide web., *in* A. N. Kumar and I. Russell (eds), *Proceedings of the Twelfth International Florida Artificial Intelligence Research Society Conference, May 1-5, 1999, Orlando, Florida, USA*, AAAI Press, pp. 37–41.

Finlay, P. N.: 1994, *Introducing decision support systems*, NCC Blackwell; Blackwell Publishers, Oxford, UK Cambridge, Mass.

Fisher, R. A.: 1936, The use of multiple measurements in taxonomic problems, *The Annals of Eugenics 7*, pp. 109–122.

Fraser, A., Rosalki, S., Gamble, G. and Pounder, R.: 1995, Inter-individual and intra-individual variability of ethanol concentration-time profiles: Comparison of ethanol ingestion before or after an evening meal., *British Journal of Clinical Pharmacology* **40**, 387–392.

Freund, Y., Seung, H. S., Shamir, E. and Tishby, N.: 1997, Selective sampling using the query by committee algorithm, *Machine Learning* **28**(2-3), 133–168.

Fu, K. S.: 1968, *Sequential Methods in Pattern Recognition and Machine Learning*, Academic Press, New York.

Gosset, W.: 1908, *On the Probable Error of a Mean*, Vol. 6(1), Biometrika.

Hammond, K.: 1988, Case-based planning: Viewing planning as a memory task, *Proceedings of the DARPA Case-Based Reasoning Workshop*, Morgan Kaufmann.

Harman, G.: 1965, The inference to the best explanation, *The Philosophical Review* **74(1)**, 88–95.

Hayes, C., Cunningham, P. and Doyle, M.: 1998, Distributed CBR using XML, *in* W. Wilke and J. Schumacher (eds), *Proceedings of the KI-98 Workshop on Intelligent Systems and Electronic Commerce.*

Hempel, C.: 1965, *Aspects of Scientific Explanation*, NewYork : FreePress.

Hempel, C. and Oppenheim, P.: 1948, Studies in the logic of explanation, *Philosophy of Science* **15**, 135–175.

Inselberg, A.: 1985, The plane with parallel coordinates., *The Visual Computer* **1**(2), 69–91.

Jackson, P.: 1986, *Introduction to expert systems*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Jones, A. and Jönsson, K.: 1994, Food-induced lowering of blood-ethanol profiles and increased rate of elimination immediately after a meal., *Journal of Forensic Sciences* **39(4)**, 1084–1093.

Kass, A. M. and Leake, D. B.: 1988, Case-based reasoning applied to constructing explanations, *Proc. of a Workshop on Case-Based Reasoning*, Holiday Inn, Clearwater Beach, FL, pp. 190–208.

Kira, K. and Rendell, L.: 1992a, The feature selection problem: Traditional methods and a new algorithm, *Tenth International Conference on Artifial Intelligence*, MIT Press, pp. 129–134.

Kira, K. and Rendell, L.: 1992b, A practical approach to feature selection, *Proceedings of the Proceedings of the Ninth International Conference on Machine Learning*, Morgan Kaufmann, pp. 249–256.

Kohavi, R., Langley, P. and Yun, Y.: 1997, The utility of feature weighting in nearest-neighbor algorithms, *in* W. Gerstner, A. Germond, M. Hasler and J.-D. Nicoud (eds), *9th European Conference on Machine Learning ECML-97, Prague, Czech Republic*, pp. 213–220.

Kolodner, J.: 1983, Reconstructive memory, a computer model, *Cognitive Science* **7 (2)**, 281–328.

Kolodner, J.: 1996, Making the implicit explicit: Clarifying the principles of case-based reasoning, *in* D. Leake (ed.), *Case-Based Reasoning: Experiences, Lessons and Future Directions*, MIT Press, pp. 349–370.

Kolodner, J. L.: 1991, Improving human decision making through case-based decision aiding, *AI Magazine* **12**(2), 52–68.

Kononenko, I.: 1994, Estimating attributes: Analysis and extensions of relief, *Proceedings of European Conference on Machine Learning*, pp. 171–182.

Kononenko, I., Bratko, I. and Roskar, E.: 1984, Experiments in automatic learning of medical diagnostic rules, *Technical report*, Jozef Stefan Institute, Ljubljana, Yugoslavia.

Leake, D.: 1995a, Goal-driven learning, *Goal-Based Explanation Evaluation* pp. 251–285.

Leake, D.: 1996, Cbr in context: The present and future, *in* D. Leake (ed.), *Case-Based Reasoning: Experiences, Lessons & Future Directions*, AAAI Press/MIT Press, Menlo Park, California, chapter 1, pp. 3–30.

Leake, D. B.: 1995b, Abduction, experience, and goals: A model of everyday abductive explanation, *Journal of Experimental and Theoretical Artificial Intelligence* **7**, 407–428.

Lenat, D., Davis, R., Doyle, J., Genesereth, M., Goldstein, I. and Schrobe, H.: 1983, Reasoning about reasoning, *in* F. Hayes-Roth, D. A. Waterman and D. B. Lenat (eds), *Building Expert Systems*, Addison-Wesley, London, pp. 219–239.

Lenz, M.: 1993, Cabata - a hybrid case-based reasoning system, *in* M. Lenz, B. Bartsch-Spörl, H.-D. Burkhard and S. Wess (eds), *In Proceed. of the First European Conference on Case- Based Reasoning. SEKI Report SR-93-12. University of Keiserslautern*, pp. 204–209.

Lenz, M.: 1999, *Case Retrieval Nets as a model for building flexible information systems*, PhD thesis, Faculty of Mathematics and Natural Sciences, Humboldt University, Berlin.

Lenz, M. and Ashley, K. (eds): 1998, *Proceedings of the AAAI98 Workshop On Textural Case-Based Reasoning*, AAAI press.

Lenz, M. and Burkhard, H.-D.: 1996, Case retrieval nets: Basic ideas and extensions, *in* G. Gorz and S. Holldobler (eds), *KI-96: Advances in Artificial Intelligence*, number 1137 in *Lecture Notes in Artificial Intelligence*, Springer Verlag, pp. 227–239.

Lewis, D. D. and Gale, W. A.: 1994, A sequential algorithm for training text classifiers, *in* W. B. Croft and C. J. van Rijsbergen (eds), *Proceedings of Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retireval*, Springer-Verlag, London, pp. 3–12.
URL: *citeseer.ist.psu.edu/lewis94sequential.html*

Loughrey, J. and Cunningham, P.: 2004, Overfitting in wrapper-based feature subset selection: The harder you try the worse it gets, *in* M. Bramer, F. Coenen and T. Allen (eds), *Research and Development in Intelligent Systems XXI, Proceedings of AI-2004, The Twenty-fourth SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence*, Springer, pp. 33–43.

Majchrzak, A. and Gasser, L.: 1991, On using artificial intelligence to integrate the design of organixational and process change in us manufacturing, *AI and Society* **5**, 321–338.

Manago, M., Bergmann, R., Wess, S. and Traphoener, R.: 1994, Casuel: A common case representation language - version 2.0. ESPRIT-Project INRECA, 1994, available from http://wwwagr.informatik.unikl.de/ bergmann/casuel/.

Mark, W., Simoudis, E. and Hinkle, D.: 1996, Case-based reasoning: Expectations and results, *in* D. Leake (ed.), *Case-Based Reasoning: Experiences, Lessons, & Future Directions*, AAAI Press/The MIT Press.

Massie, S., Craw, S. and Wiratunga, N.: 2004a, Visualisation of case-base reasoning for explanation, *Workshop Proceedings of the 7th European Conference on Case-Based Reasoning*, pp. 135–144.

Massie, S., Craw, S. and Wiratunga, N.: 2004b, A visualisation tool to explain case-base reasoning solutions for tablet formulation, *Proceedings of the 24th Annual International Conference of the British Computer Society's Specialist Group on Artificial Intelligence*, pp. 222–234.

McArdle, G. and Wilson, D.: 2003, Visualising case-base usage, *Workshop Proceedings of the 5th International Conference on Case-Based Reasoning*, pp. 47–55.

McKenna, E. and Smyth, B.: 2000, Competence-guided editing methods for lazy learning, *in* C. Mellish (ed.), *14th European Conference on Artificial Intelligence, Proceedings.*

McLaren, B. M. and Ashley, K. D.: 2001, Helping a CBR program know what it knows., *in* D. W. Aha and I. Watson (eds), *Case-Based Reasoning Research and Development, 4th International Conference on Case-Based Reasoning, ICCBR 2001, Vancouver, BC, Canada, July 30 - August 2, 2001, Proceedings*, Vol. 2080 of *Lecture Notes in Computer Science*, Springer, pp. 377–391.

McSherry, D.: 1999, Dynamic and static approaches to clinical data mining., *Artificial Intelligence in Medicine* **16**(1), 97–115.

McSherry, D.: 2001, Interactive case-based reasoning in sequential diagnosis, *Applied Intelligence* **14**, 65–76.

McSherry, D.: 2003a, Explanation in case-based reasoning: an evidential approach, *8th UK Workshop on Case-Based Reasoning*, pp. 47–55.

McSherry, D.: 2003b, Explanation in case-based reasoning: an evidential approach, *in* B. Lees (ed.), *8th UK Workshop on Case-Based Reasoning*, pp. 332–346.

McSherry, D.: 2004, Explaining the pros and cons of conclusions in CBR, *in* P. A. G. Calero and P. Funk (eds), *Advances in Case-Based Reasoning, 7th European Conference, ECCBR 2004 Madrid, Spain, August 30th through Sep 2nd, 2004*, Vol. 3155 of *LNAI*, Springer, pp. 317–330.

Minsky, M.: 1961, Steps toward artificial intelligence, *Proceedings of the Institute of Radio Engineers*, Vol. 49, p. 830.

Mitchell, T. M.: 1997, *Machine Learning*, McGraw-Hill, chapter 3, pp. 52–80.

Mucciardi, A. N. and Gose, E. E.: 1971, A comparison of seven techniques for choosing subsets of pattern recognition properties, *IEEE Transaction on Computers* **20**, 1023–1031.

Nugent, C. and Cunningham, P.: 2004, A case-based explanation system for 'black-box' systems, *in* P. Cunningham and D. McSherry (eds), *ECCBR 2004 Workshop Proceedings*, pp. 155–164.

Nugent, C., Cunningham, P. and Doyle, D.: 2005, The best way to instil confidence is by being right; an evaluation of the effectiveness of case-based explanations in providing user confidence, *in* H. Muñoz-Avila and F. Ricci (eds), *6th International Conference on Case-Based Reasoning*, Vol. 3620 of *LNAI*, Springer, pp. 368–381.

Ong, L., Sheperd, B., Tong, L., Seow-Choen, F., Ho, Y., Tong, L., Y.S, H. and Tan, K.: 1997, The colorectal cancer recurrence support (cares) system., *Artificial Intelligence in Medicine* **11**(3), 175–188.

Online-Encyclopedia: 2005. http://en.wikipedia.org/wiki/Bronchiole accessed 5/5/05.

Orenstein, D.: 2000, *Bronchiolitis*, 16 edn, Saunders Philadelphia, pp. 1285–1287.

Pasquier, F.-X., Delany, S.-J. and Cunningham, P.: 2005, Blame-based noise reduction: An alternative perspective on noise reduction for lazy learning, *Technical Report TCD-CS-2005-29*, Department of Computer Science, Trinity College Dublin.

Pews, G. and Wess, S.: 1993, Combining model-based approaches and case-based reasoning for similarity assessment and case adaptation in diagnostic applications, *in* M. Richter, S. Wess, K. Althoff and F. Maurer (eds), *Preprints of the First European Workshop on Case-Based Reasoning (EWCBR-93)*, Vol. 2, University of Kaiserslautern, pp. 325–328.

Plaza, E.: 1997, An overview of the noos representation language. available from http://www.iiia.csic.es/∼enric/noos/Overview.

Quinlan, J.: 1987, Rule induction with statistical data - a comparison with multiple regression, *Journal of the Operational Research Society* **38**, 347–352.

Quinlan, J.: 1993, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Francisco.

Quinlan, J. R.: 1986, Induction of decision trees, *Machine Learning* **1**, 81–106.

Rahman, Y., Knape, T., Gargan, M., Power, G., Hederman, L., Wade, V., Nolan, J. and Grimson, J.: 2004, e-clinic: An electronic triage system in diabetes management through leveraging information and communication technologies, *Medinfo* **11(Pt 1)**, 246–250.

Richards, D.: 2003, Knowledge-based system explanation: The ripple-down rules alternative, *Knowledge and Information Systems* **5**(1), 2–25.

Richter, M. M.: 1995, The knowledge contained in similarity measures. Invited talk at ICCBR95, http://www.cbr-web.org/documents/Richtericcbr95remarks.html.

Richter, M. M.: 1998, Introduction, *in* M. Lenz, B. Bartsch-Spörl, H.-D. Burkhard and S. Wess (eds), *Case-Based Reasoning Technology, From Foundations to Applications*, Springer, pp. 1–16.

Riesbeck, C.: 1988, An interface for case-based knowledge acquisition, *in* J. Kolodner (ed.), *Proceedings of the DARPA Case-Based Reasoning Workshop*, Morgan Kaufmann, Calif., US.

Riesbeck, C. and Schank, R.: 1989, *Inside Case-Based Reasoning, Erlbaum*, Erlbaum.

Roth-Berghofer, T.: 2004, Explanations and case-based reasoning: Foundational issues., *in* P. Funk and P. A. González-Calero (eds), *Advances in Case-Based Reasoning, 7th European Conference, ECCBR 2004, Madrid, Spain, August 30 - September 2, 2004, Proceedings*, Vol. 3155 of *Lecture Notes in Computer Science*, Springer, pp. 389–403.

Salmon, W.: 1971, *Statistical Explanation*, Pittburgh University Press, Pittburgh, chapter The Nature and Function of Scientific Theories, pp. 173–231.

Schank, R. and Abelson, R.: 1977, *Scripts, Plans, Goals, and Understanding: An Inquiry into Human Knowledge Structures*, Lawrence Erlbaum, Hillsdale, N.J.

Schank, R. C.: 1982, *Dynamic Memory: A Theory of Learning in Computers and People*, Cambridge University Press, Yale University.

Schank, R. C.: 1986, *Explanation Patterns: Understanding Mechanical and Creatively*, Lawrence Erlbaum Associates, Inc., Mahwah, NJ, USA.

Seung, H. S., Opper, M. and Sompolinsky, H.: 1992, Query by committee, *COLT '92: Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, ACM Press, pp. 287–294.

Shankar, R. and Musen, M.: 1999, Justification of automated decision-making: Medical explanations as medical arguments, *Proc AMIA Symposium*, pp. 395–399.

Shimazu, H.: 1998, A textual case-based reasoning system using xml on the world-wide web, *in* B. Smyth and P. Cunningham (eds), *Advances in Case-Based Reasoning: Proceedings of the Fourth European Workshop on Case-Based Reasoning*, Vol. 1488 of *Lecture Notes in Computer Science*, Springer, pp. 274–285.

Shortliffe, E. H.: 1976, *Computer Based Medical Consultations: MYCIN*, American Elsevier, New York (1976.

Sørmo, F. and Cassens, J.: 2004, Explanation goals in case-based reasoning, *1st Workshop on Case-Based Explanation (Proceedings of the ECCBR 2004 workshops)*, pp. 165–174.

Sørmo, F., Cassens, J. and Aamodt, A.: 2005, Explanation in case-based reasoning perspectives and goals, *To appear in: Artificial Intelligence* **24**.

Stahl, A.: 2002, Defining similarity measures: Top-down vs. bottom-up., *in* S. Craw and A. D. Preece (eds), *Advances in Case-Based Reasoning, 6th European Conference, ECCBR 2002 Aberdeen, Scotland, UK, September 4-7, 2002, Proceedings*, Springer, pp. 406–420.

Swartout, W. and Moore, J.: 1993, Explanation in second generation expert systems, *in* R. S. J. David, J. Krivine (ed.), *Second Generation Expert Systems*, Springer Verlag, pp. 543–585.

Swartout, W. R.: 1984, Explaining and justifying expert consulting programs, *in* W. J. Clancey and E. H. Shortliffe (eds), *Readings in Medical Artificial Intelligence: The First Decade*, Addison-Wesley, Reading, MA, pp. 382–398.

Tickle, A.and Andrews, R., Golea, R. and Diederich, J.: 1998, The truth will come to light: Directions and challenges in extracting rules from trained neural networks, *IEEE Transactions on Neural Networks* **9**, 1057–1068.

Tomek, I.: 1976, An experiment with the nearest neighbor rule, *IEEE Transactions on Systems, Man and Cybernetics* **6 (6)**, 448–452.

Toulmin, S.: 1958, *The Uses of Argument*, Cambridge University Press.

Toulmin, S., Rieke, R. and Janik, A.: 1984, *An Introduction to Reasoning (2nd ed.)*, Macmillan.

Turing, A.: 1950, Computing machinery and intelligence, *Mind*, Vol. 59, pp. 433–460.

vanFraassen, B.: 1980, *The Scientific Image*, Oxford:Clarendon Press.

von Neumann, J. and Morgenstern, O.: 1944, *Theory of Games and Economic Behavior*, Princeton Univ. Press.

Wallgren, H.: 1970, Absorption, diffusion, distribution and elimination of ethanol: Effect on biological membranes, *International Encyclopedia of Pharmacology and Therapeutics* **1**, 161–188.

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

Watson, I. and Gardingen, D.: 1999, A distributed case-based reasoning application for engineering sales support., *in* T. Dean (ed.), *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, IJCAI 99, Stockholm, Sweden, July 31 - August 6, 1999. 2 Volumes, 1450 pages*, Morgan Kaufmann, pp. 600–605.

Wettschereck, D., Aha, D. W. and Mohri, T.: 1997, A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms, *Artificial Intelligence Review* **11**(1-5), 273–314.

Whitney, A.: 1971, A direct method of non parametric measurement selection, *IEEE Transactions on Computing*, Vol. C-20.

Wick, M.: 1992, Expert system explanation in retrospect: A case study in the evaluation of expert system explanation, *Journal of Systems and Software* **19(2)**, 159–169.

Wick, M. R. and Slagle, J. R.: 1989, An explanation facility for today's expert systems, *IEEE Expert: Intelligent Systems and Their Applications* **4**(1), 26–36.

Wick, M. R. and Thompson, W.: 1992, Reconstructive expert system explanation, *Artificial Intelligence* **54(12)**, 3370.

Wilson, D. L.: 1972, Asymptotic properties of nearest neighbor rules using edited data, *IEEE Transactions on Systems, Man and Cybernetics* **2 (3)**, 408–421.

Zhou, Z.-H. and Jiang, Y.: 2003, Medical diagnosis with c4.5 preceded by artificial neural network ensemble, *IEEE Transactions on Information Technology in Biomedicine* **7**(1), 37–42.

Appendix A

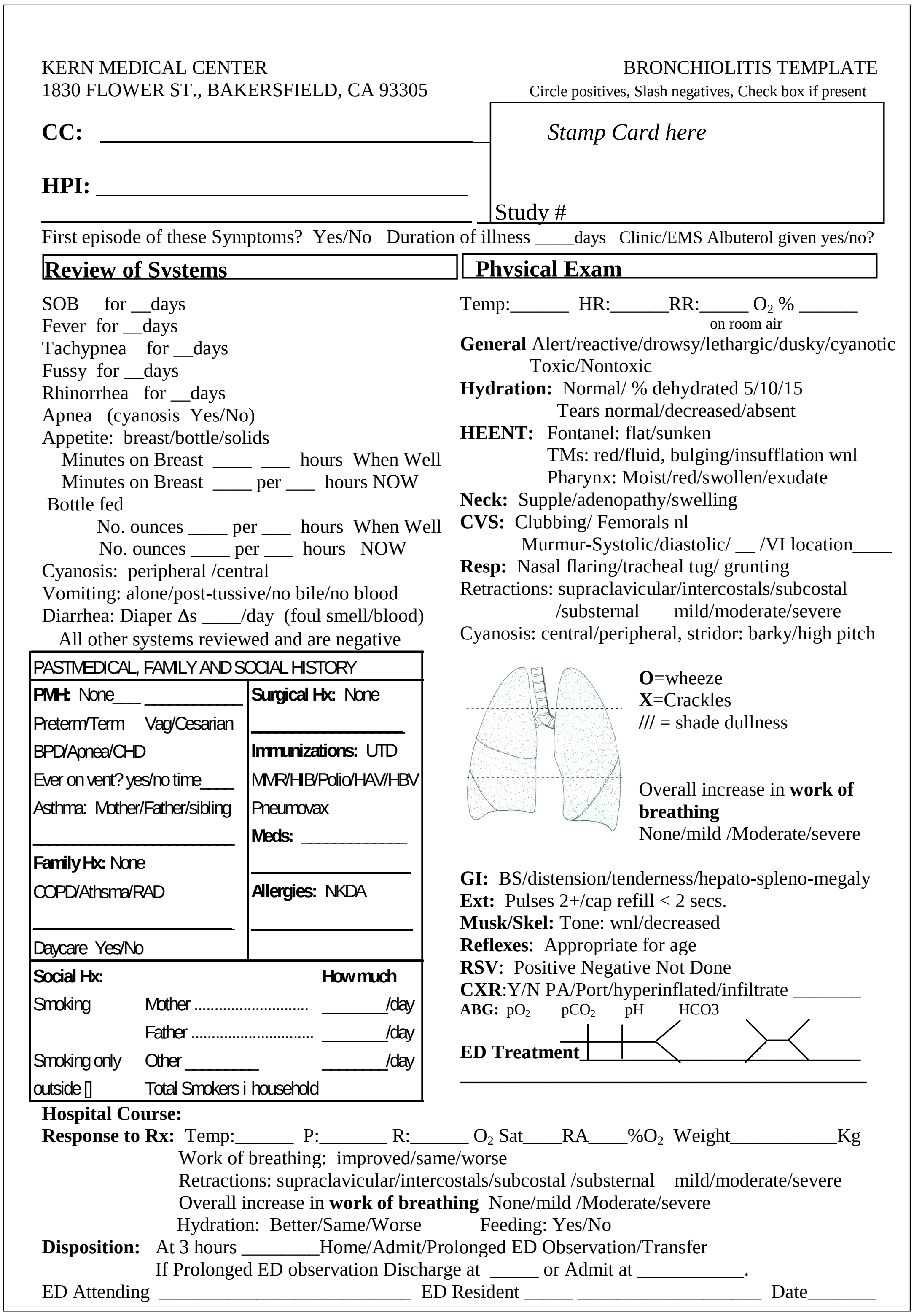
Tennis Dataset

Table A.1: Tennis Dataset

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Appendix B

Bronchiolitis Questionnaire



KERN MEDICAL CENTER
1830 FLOWER ST., BAKERSFIELD, CA 93305

BRONCHIOLITIS TEMPLATE
Circle positives, Slash negatives, Check box if present

Stamp Card here

Study #

CC: ________________________________

HPI: ________________________________
____

First episode of these Symptoms? Yes/No Duration of illness ____days Clinic/EMS Albuterol given yes/no?

Review of Systems

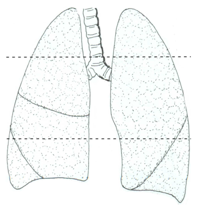
SOB for __days
Fever for __days
Tachypnea for __days
Fussy for __days
Rhinorrhea for __days
Apnea (cyanosis Yes/No)
Appetite: breast/bottle/solids
Minutes on Breast ____ ___ hours When Well
Minutes on Breast ____ per ___ hours NOW
Bottle fed
No. ounces ____ per ___ hours When Well
No. ounces ____ per ___ hours NOW
Cyanosis: peripheral /central
Vomiting: alone/post-tussive/no bile/no blood
Diarrhea: Diaper Δs ____/day (foul smell/blood)
All other systems reviewed and are negative

PASTMEDICAL, FAMILY AND SOCIAL HISTORY	
PMH: None___ __________	**Surgical Hx:** None
Preterm/Term Vag/Cesarian	______________
BPD/Apnea/CHD	**Immunizations:** UTD
Ever on vent? yes/no time____	MMR/HIB/Polio/HAV/HBV
Asthma: Mother/Father/sibling	Pneumovax
______________	**Meds:** __________
Family Hx: None	______________
COPD/Athsma/RAD	**Allergies:** NKDA
______________	______________
Daycare Yes/No	

Social Hx:		**How much**
Smoking	Mother	_______/day
	Father	_______/day
Smoking only	Other ________	_______/day
outside []	Total Smokers in household	

Physical Exam

Temp:______ HR:______RR:_____ O_2 % ______
on room air
General Alert/reactive/drowsy/lethargic/dusky/cyanotic
Toxic/Nontoxic
Hydration: Normal/ % dehydrated 5/10/15
Tears normal/decreased/absent
HEENT: Fontanel: flat/sunken
TMs: red/fluid, bulging/insufflation wnl
Pharynx: Moist/red/swollen/exudate
Neck: Supple/adenopathy/swelling
CVS: Clubbing/ Femorals nl
Murmur-Systolic/diastolic/ __ /VI location____
Resp: Nasal flaring/tracheal tug/ grunting
Retractions: supraclavicular/intercostals/subcostal
/substernal mild/moderate/severe
Cyanosis: central/peripheral, stridor: barky/high pitch



O=wheeze
X=Crackles
/// = shade dullness

Overall increase in **work of breathing**
None/mild /Moderate/severe

GI: BS/distension/tenderness/hepato-spleno-megaly
Ext: Pulses 2+/cap refill < 2 secs.
Musk/Skel: Tone: wnl/decreased
Reflexes: Appropriate for age
RSV: Positive Negative Not Done
CXR:Y/N PA/Port/hyperinflated/infiltrate _______
ABG: pO_2 pCO_2 pH HCO3

ED Treatment________________________
____

Hospital Course:
Response to Rx: Temp:______ P:_______ R:______ O_2 Sat____RA____%O_2 Weight___________Kg
Work of breathing: improved/same/worse
Retractions: supraclavicular/intercostals/subcostal /substernal mild/moderate/severe
Overall increase in **work of breathing** None/mild /Moderate/severe
Hydration: Better/Same/Worse Feeding: Yes/No
Disposition: At 3 hours ________Home/Admit/Prolonged ED Observation/Transfer
If Prolonged ED observation Discharge at _____ or Admit at ___________.
ED Attending __________________________ ED Resident _____ ___________________ Date_______

Figure B.1: Bronchiolitis data collection form.

Appendix C

Schema Documents

C.1 CBMLv3 Schema Document

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="case">
    <xs:complexType>
      <xs:choice>
        <xs:element name="structure" maxOccurs="1">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="feature" maxOccurs="unbounded" type="structure_feature"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="similarity" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="feature" minOccurs="0" maxOccurs="unbounded" type="similarity_feature"/>
            </xs:sequence>
            <xs:attribute name="username" type="xs:ID" use="required"/>
            <xs:attribute name="extends" type="xs:IDREF" use="optional"/>
          </xs:complexType>
        </xs:element>
      </xs:choice>
      <xs:attribute name="domain" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>

  <xs:complexType name="similarity_feature">
    <xs:choice>
      <xs:element name="feature" minOccurs="1" maxOccurs="unbounded" type="similarity_feature"/>
      <xs:element name="array" minOccurs="1" maxOccurs="1">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="primary" minOccurs="0" maxOccurs="unbounded">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="secondary" minOccurs="0" maxOccurs="unbounded">
                    <xs:complexType>
                      <xs:attribute name="name" type="xs:string" use="required"/>
                      <xs:attribute name="similarity" type="xs:double" use="required"/>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
                <xs:attribute name="name" type="xs:string" use="required"/>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="graph"  minOccurs="1" maxOccurs="1">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="point" minOccurs="0" maxOccurs="unbounded">
              <xs:complexType>
                <xs:attribute name="difference" type="xs:double" use="required"/>
                <xs:attribute name="similarity" type="xs:double" use="required"/>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
          <xs:attribute name="type" type="graphType" use="required"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="measure" minOccurs="1" maxOccurs="1">
        <xs:complexType>
          <xs:attribute name="name" type="xs:string" use="required"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="exact" minOccurs="1" maxOccurs="1"/>
    </xs:choice>
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="weight" type="positiveDouble" use="required"/>
  </xs:complexType>

  <xs:simpleType name="positiveDouble">
    <xs:restriction base="xs:double">
      <xs:minInclusive value="0"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="normalisedDouble">
    <xs:restriction base="xs:double">
      <xs:minInclusive value="0"/>
      <xs:maxInclusive value="1"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="graphType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="symmetrical"/>
      <xs:enumeration value="asymmetrical"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="structure_feature">
    <xs:choice>
      <xs:element name="complex">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="feature" minOccurs="0" maxOccurs="unbounded" type="structure_feature"/>
          </xs:sequence>
          <xs:attributeGroup ref="references"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="symbol">
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="enumeration" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attributeGroup ref="references"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="string">
        <xs:complexType>
          <xs:attributeGroup ref="references"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="taxonomy">
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="node" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attributeGroup ref="references"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="integer">
        <xs:complexType>
          <xs:all>
            <xs:element name="maxInclusive" minOccurs="0">
              <xs:complexType>
                <xs:attribute name="value" type="xs:int" use="required"/>
              </xs:complexType>
            </xs:element>
            <xs:element name="minInclusive" minOccurs="0">
              <xs:complexType>
                <xs:attribute name="value" type="xs:int" use="required"/>
              </xs:complexType>
            </xs:element>
          </xs:all>
          <xs:attributeGroup ref="references"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="double">
        <xs:complexType>
          <xs:all>
            <xs:element name="maxInclusive" minOccurs="0">
              <xs:complexType>
                <xs:attribute name="value" type="xs:double" use="required"/>
              </xs:complexType>
            </xs:element>
            <xs:element name="minInclusive" minOccurs="0">
              <xs:complexType>
                <xs:attribute name="value" type="xs:double" use="required"/>
              </xs:complexType>
            </xs:element>
          </xs:all>
          <xs:attributeGroup ref="references"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="boolean">
        <xs:complexType>
          <xs:attributeGroup ref="references"/>
        </xs:complexType>
      </xs:element>
    </xs:choice>
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="discriminant" use="optional" default="true" type="xs:boolean"/>
    <xs:attribute name="solution"     use="optional" default="false" type="xs:boolean"/>
    <xs:attribute name="mandatory"    use="optional" default="true" type="xs:boolean"/>
    <!--Although manditory will be accepted, it will be removed in future versions, so should not be used -->
    <xs:attribute name="manditory"    use="optional" default="true" type="xs:boolean"/>
  </xs:complexType>

  <xs:attributeGroup name="references">
    <xs:attribute name="name" type="xs:ID" use="optional"/>
    <xs:attribute name="ref" type="xs:IDREF" use="optional"/>
  </xs:attributeGroup>
  <xs:element name="enumeration">
    <xs:complexType>
      <xs:attribute name="value" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="node">
    <xs:complexType>
      <xs:sequence>
```

```
        <xs:element ref="node" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="name" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

C.2 Explanation Schema Document

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xi="http://www.w3.
    org/2001/XMLSchema">
  <xs:include schemaLocation="http://www.cs.tcd.ie/research_groups/mlg/CBML/
      Schema/cbmlv3.xsd"/>
  <xs:element name="expdoc">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="1" maxOccurs="unbounded" name="explanation">
          <xs:complexType>
            <xs:sequence>
              <xs:element minOccurs="0" maxOccurs="unbounded" name="feature"
                  type="similarity_feature" />
            </xs:sequence>
            <xs:attribute name="classification" type="xs:ID" use="required" />
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="domain" type="xs:string" use="required" />
    </xs:complexType>
  </xs:element>
</xs:schema>
```

C.3 Explanatory Text Schema Document

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xi="http://www.w3.
    org/2001/XMLSchema">
  <xs:element name="explanation_dialogue">
    <xs:complexType>
      <xs:choice maxOccurs="unbounded">
        <xs:element name="utility">
      <xs:complexType>
        <xs:sequence>
          <xs:element minOccurs="1" maxOccurs="unbounded" name="feature">
            <xs:complexType>
              <xs:sequence>
                <xs:element minOccurs="0" maxOccurs="unbounded" name="difference
                    ">
                  <xs:complexType>
                    <xs:attribute name="change" type="differenceString" use="
                        required"/>
                    <xs:attribute name="dialogue" type="xs:string" use="required
                        "/>
                  </xs:complexType>
                </xs:element>
              </xs:sequence>
              <xs:attribute name="name" type="xs:string" use="required"/>
            </xs:complexType>
          </xs:element>
          <xs:element name="dialogue" minOccurs="1" maxOccurs="1">
            <xs:complexType mixed="true">
          <xs:choice maxOccurs="unbounded">
          <xs:element ref="replace" minOccurs="0" maxOccurs="unbounded"/>
                <xs:element minOccurs="0" maxOccurs="unbounded" name="
                    conditional_replace">
                  <xs:complexType mixed="true">
                  <xs:choice maxOccurs="unbounded">
              <xs:element ref="replace" minOccurs="0" maxOccurs="unbounded"/>
              <xs:element minOccurs="0" maxOccurs="unbounded" name="br"/>
              </xs:choice>
                    <xs:attribute name="condition" type="conditionalReplace" use="required"/>
                    <xs:attribute name="value" type="xs:integer" use="required"/>
                    <xs:attribute name="equality_type" type="equality" use="required"/>
                  </xs:complexType>
                </xs:element>
                <xs:element minOccurs="0" maxOccurs="unbounded" name="br"/>
          </xs:choice>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element minOccurs="1" maxOccurs="unbounded" name="similarity">
          <xs:complexType>
        <xs:sequence>
          <xs:element minOccurs="1" maxOccurs="unbounded" name="feature">
            <xs:complexType>
              <xs:sequence>
                <xs:element minOccurs="0" maxOccurs="unbounded" name="similar">
                  <xs:complexType>
                    <xs:attribute name="dialogue" type="xs:string" use="required"/>
                  </xs:complexType>
                </xs:element>
              </xs:sequence>
              <xs:attribute name="name" type="xs:string" use="required"/>
            </xs:complexType>
          </xs:element>
          <xs:element name="dialogue" minOccurs="1" maxOccurs="1">
            <xs:complexType mixed="true">
          <xs:choice maxOccurs="unbounded">
          <xs:element minOccurs="0" maxOccurs="unbounded" name="replace">
                  <xs:complexType>
                    <xs:attribute name="value" type="replaceString" use="required"/>
                  </xs:complexType>
                </xs:element>
          </xs:choice>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element minOccurs="1" maxOccurs="unbounded" name="solution">
          <xs:complexType>
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="unbounded" name="classification">
            <xs:complexType>
              <xs:attribute name="value" type="xs:string" use="required"/>
              <xs:attribute name="classification_dialogue" type="xs:string" use="required"/>
              <xs:attribute name="utility_shift" type="xs:string" use="required"/>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element minOccurs="1" maxOccurs="unbounded" name="mappings">
            <xs:complexType>
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="unbounded" name="feature">
            <xs:complexType>
              <xs:attribute name="name" type="xs:string" use="required"/>
              <xs:attribute name="mapping" type="xs:string" use="required"/>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element minOccurs="0" maxOccurs="1" name="confidence">
      <xs:complexType>
        <xs:choice maxOccurs="unbounded">
        <xs:element minOccurs="0" maxOccurs="unbounded" name="confidenceMeasure"
            >
          <xs:complexType mixed="true">
          <xs:attribute name="threshold" type="xs:double" use="required"/>
          <xs:attribute name="measure" type="confidenceMeasures" use="required"/
              >
          <xs:attribute name="neighbours" type="xs:integer" use="optional"
              default="3"/>
          </xs:complexType>
        </xs:element>
        <xs:element minOccurs="0" maxOccurs="unbounded" name="confidenceDialogue
            ">
          <xs:complexType mixed="true">
            <xs:attribute name="level" type="confidence" use="required"/>
          </xs:complexType>
        </xs:element>
        </xs:choice>
      </xs:complexType>
    </xs:element>
    </xs:choice>
    <xs:attribute name="domain" type="xs:string" use="required" />
  </xs:complexType>
  </xs:element>


  <xs:simpleType name="differenceString">
    <xs:restriction base="xs:string">
      <xs:enumeration value="positive"/>
      <xs:enumeration value="equal"/>
      <xs:enumeration value="negative"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="replaceString">
    <xs:restriction base="xs:string">
      <xs:pattern value="target_case_name"/>
      <xs:pattern value="classification_dialogue"/>
      <xs:pattern value="nun_classification_dialogue"/>
      <xs:pattern value="utility_case_name"/>
      <xs:pattern value="similarity_case_name"/>
      <xs:pattern value="nun_case_name"/>
      <xs:pattern value="utility_shift"/>
      <xs:pattern value="positive_utilty_features_[0-9]{0,3}"/>
      <xs:pattern value="positive_utilty_features_[0-9]{0,3}"/>
      <xs:pattern value="negative_utilty_features_[0-9]{0,3}"/>
      <xs:pattern value="strong_similarity_features_[0-9]{0,3}"/>
      <xs:pattern value="negative_nun_features_[0-9]{0,3}"/>
      <xs:pattern value="num_negative_nun_features_[0-9]{0,3}"/>
      <xs:pattern value="num_negative_nun_features_words_[0-9]{0,3}"/>
      <xs:pattern value="confidence"/>
      <xs:pattern value="system_date_.+"/>
      <xs:pattern value="include_.+"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="conditionalReplace">
    <xs:restriction base="xs:string">
      <xs:pattern value="positive_utilty_features"/>
      <xs:pattern value="negative_utilty_features"/>
      <xs:pattern value="strong_similarity_features"/>
      <xs:pattern value="negative_nun_features"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="equality">
    <xs:restriction base="xs:string">
      <xs:pattern value="greater_than"/>
      <xs:pattern value="less_than"/>
      <xs:pattern value="equal_to"/>
```

```
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="confidence">
      <xs:restriction base="xs:string">
        <xs:pattern value="confident"/>
        <xs:pattern value="not_confident"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="confidenceMeasures">
      <xs:restriction base="xs:string">
        <xs:enumeration value="similarity_ratio"/>
        <xs:enumeration value="explanation_ratio"/>
        <xs:enumeration value="average_nun_index"/>
        <xs:enumeration value="similarity_vote"/>
        <xs:enumeration value="neighbour_accuracy"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:element name="replace">
    <xs:complexType>
      <xs:attribute name="value" type="replaceString" use="required"/>
    </xs:complexType>
    </xs:element>
</xs:schema>
```

Appendix D

Breathalyser CBML Documents

D.1 Breathalyser Structure Document

```xml
<?xml version="1.0" encoding="UTF-8"?>
<case domain="breathalyser2002" xsi:noNamespaceSchemaLocation="http://www.cs.tcd
    .ie/research_groups/mlg/CBML/Schema/cbmlv3.xsd" xmlns:xsi="http://www.w3.org
    /2001/XMLSchema-instance">
  <structure>
    <feature name="Kgs">
      <double>
        <minInclusive value="47.0"/>
        <maxInclusive value="101.0"/>
      </double>
    </feature>
    <feature name="Duration">
      <double>
        <minInclusive value="5.0"/>
        <maxInclusive value="435.0"/>
      </double>
    </feature>
    <feature name="Gender">
      <symbol>
        <enumeration value="Male"/>
        <enumeration value="Female"/>
      </symbol>
    </feature>
    <feature name="Meal">
      <symbol>
        <enumeration value="None"/>
        <enumeration value="Snack"/>
        <enumeration value="Lunch"/>
        <enumeration value="Full"/>
      </symbol>
    </feature>
    <feature name="Units">
      <double>
        <minInclusive value="0.0"/>
        <maxInclusive value="31.2"/>
      </double>
    </feature>
    <feature name="BAC" solution="true" discriminant="false">
      <symbol>
        <enumeration value="Under"/>
        <enumeration value="Over"/>
      </symbol>
    </feature>
  </structure>
</case>
```

D.2 Breathalyser Similarity Document

```
<?xml version="1.0"?>
<case domain="breathalyser2002" xsi:noNamespaceSchemaLocation="http://www.cs.tcd
    .ie/research_groups/mlg/CBML/Schema/cbmlv3.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <similarity username="default">
    <feature name="Kgs" weight="1.0">
      <graph type="symmetrical">
        <point difference="0.0" similarity="1.0"/>
        <point difference="5.0" similarity="0.9"/>
        <point difference="10.0" similarity="0.8"/>
        <point difference="40.0" similarity="0.0"/>
      </graph>
    </feature>
    <feature name="Duration" weight="0.25">
      <graph type="symmetrical">
        <point difference="0.0" similarity="1.0"/>
        <point difference="10.0" similarity="0.9"/>
        <point difference="20.0" similarity="0.8"/>
        <point difference="100.0" similarity="0.0"/>
      </graph>
    </feature>
    <feature name="Gender" weight="1.0">
      <exact/>
    </feature>
    <feature name="Meal" weight="1.0">
      <graph type="symmetrical">
        <point difference="0.0" similarity="1.0"/>
        <point difference="1.0" similarity="0.8"/>
        <point difference="2.0" similarity="0.4"/>
        <point difference="3.0" similarity="0.0"/>
      </graph>
    </feature>
    <feature name="Units" weight="1.0">
      <graph type="symmetrical">
        <point difference="0.0" similarity="1.0"/>
        <point difference="5.0" similarity="0.9"/>
        <point difference="30.0" similarity="0.0"/>
      </graph>
    </feature>
  </similarity>
</case>
```

D.3 Breathalyser Case Base Document

```
<?xml version="1.0"?>
<casebase domain="breathalyser2002">
  <case name="n0">
    <Kgs>76.0</Kgs>
    <Duration>60.0</Duration>
    <Gender>Male</Gender>
    <Meal>Full</Meal>
    <Units>2.9</Units>
    <BAC>Under</BAC>
  </case>
  <case name="n1">
    <Kgs>60.0</Kgs>
    <Duration>60.0</Duration>
    <Gender>Female</Gender>
    <Meal>Full</Meal>
    <Units>2.6</Units>
    <BAC>Under</BAC>
  </case>
  <case name="n2">
    <Kgs>63.0</Kgs>
    <Duration>90.0</Duration>
    <Gender>Female</Gender>
    <Meal>Full</Meal>
    <Units>1.2</Units>
    <BAC>Under</BAC>
  </case>
  <case name="n3">
    <Kgs>63.0</Kgs>
    <Duration>120.0</Duration>
    <Gender>Male</Gender>
    <Meal>Full</Meal>
    <Units>5.2</Units>
    <BAC>Under</BAC>
  </case>
  <case name="n4">
    <Kgs>51.0</Kgs>
    <Duration>120.0</Duration>
    <Gender>Female</Gender>
    <Meal>Lunch</Meal>
    <Units>5.2</Units>
    <BAC>Over</BAC>
  </case>
  .
  .
  .
  <case name="n124">
    <Kgs>73.0</Kgs>
    <Duration>155.0</Duration>
    <Gender>Male</Gender>
    <Meal>Full</Meal>
    <Units>9.8</Units>
    <BAC>Over</BAC>
  </case>
  <case name="n125">
    <Kgs>79.0</Kgs>
    <Duration>205.0</Duration>
    <Gender>Male</Gender>
    <Meal>Full</Meal>
    <Units>12.2</Units>
    <BAC>Over</BAC>
  </case>
  <case name="n126">
    <Kgs>76.0</Kgs>
    <Duration>130.0</Duration>
    <Gender>Male</Gender>
    <Meal>Full</Meal>
    <Units>9.8</Units>
    <BAC>Over</BAC>
  </case>
</casebase>
```

D.4 Breathalyser Explanation Profile Document

```
<?xml version="1.0"?>
<expdoc domain="breathalyser2002" xsi:noNamespaceSchemaLocation="http://www.cs.
    tcd.ie/research_groups/mlg/FIONN/Schema/explanation.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <explanation classification="Under">
    <feature name="Kgs" weight="1.0">
      <graph type="asymmetrical">
        <point difference="-40.0" similarity="0.0"/>
        <point difference="-10.0" similarity="0.7"/>
        <point difference="-5.0" similarity="0.85"/>
        <point difference="0.0" similarity="0.9"/>
        <point difference="5.0" similarity="0.92"/>
        <point difference="10.0" similarity="0.95"/>
        <point difference="40.0" similarity="1.0"/>
      </graph>
    </feature>
    <feature name="Duration" weight="0.25">
      <graph type="asymmetrical">
        <point difference="-100.0" similarity="0.0"/>
        <point difference="-20.0" similarity="0.6"/>
        <point difference="-10.0" similarity="0.8"/>
        <point difference="0.0" similarity="0.95"/>
        <point difference="10.0" similarity="0.97"/>
        <point difference="20.0" similarity="0.98"/>
        <point difference="100.0" similarity="1.0"/>
      </graph>
    </feature>
    <feature name="Gender" weight="1.0">
      <exact/>
    </feature>
    <feature name="Meal" weight="1.0">
      <graph type="asymmetrical">
        <point difference="-3.0" similarity="0.0"/>
        <point difference="-2.0" similarity="0.25"/>
        <point difference="-1.0" similarity="0.5"/>
        <point difference="0.0" similarity="0.8"/>
        <point difference="1.0" similarity="0.9"/>
        <point difference="2.0" similarity="0.95"/>
        <point difference="3.0" similarity="1.0"/>
      </graph>
    </feature>
    <feature name="Units" weight="1.0">
      <graph type="asymmetrical">
        <point difference="-10.0" similarity="0.75"/>
        <point difference="-5.0" similarity="1.0"/>
        <point difference="0.0" similarity="0.85"/>
        <point difference="10.0" similarity="0.0"/>
      </graph>
    </feature>
  </explanation>
  <explanation classification="Over">
    <feature name="Kgs" weight="1.0">
      <graph type="asymmetrical">
        <point difference="-40.0" similarity="1.0"/>
        <point difference="-10.0" similarity="0.95"/>
        <point difference="-5.0" similarity="0.92"/>
        <point difference="0.0" similarity="0.9"/>
        <point difference="5.0" similarity="0.85"/>
        <point difference="10.0" similarity="0.7"/>
        <point difference="40.0" similarity="0.0"/>
      </graph>
    </feature>
    <feature name="Duration" weight="0.25">
      <graph type="asymmetrical">
        <point difference="-100.0" similarity="1.0"/>
        <point difference="-20.0" similarity="0.98"/>
        <point difference="-10.0" similarity="0.97"/>
        <point difference="0.0" similarity="0.95"/>
        <point difference="10.0" similarity="0.8"/>
        <point difference="20.0" similarity="0.6"/>
        <point difference="100.0" similarity="0.0"/>
      </graph>
    </feature>
    <feature name="Gender" weight="1.0">
      <exact/>
    </feature>
    <feature name="Meal" weight="1.0">
      <graph type="asymmetrical">
        <point difference="-3.0" similarity="1.0"/>
        <point difference="-2.0" similarity="0.95"/>
        <point difference="-1.0" similarity="0.9"/>
        <point difference="0.0" similarity="0.8"/>
        <point difference="1.0" similarity="0.5"/>
        <point difference="2.0" similarity="0.25"/>
        <point difference="3.0" similarity="0.0"/>
      </graph>
    </feature>
    <feature name="Units" weight="1.0">
      <graph type="asymmetrical">
        <point difference="-10.0" similarity="0.0"/>
        <point difference="0.0" similarity="0.85"/>
        <point difference="5.0" similarity="1.0"/>
        <point difference="10.0" similarity="0.75"/>
      </graph>
    </feature>
  </explanation>
</expdoc>
```

D.5 Breathalyser Explanatory Text Document

```xml
<?xml version="1.0"?>
<explanation_dialogue domain="Bronc2004-Evaluation" xsi:noNamespaceSchemaLocation="http://www.cs.tcd.ie/research_groups/mlg/FIONN/Schema/dialogue.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <utility>
    <feature name="Age">
      <difference change="positive" dialogue="being older"/>
      <difference change="equal" dialogue="same age"/>
      <difference change="negative" dialogue="being younger"/>
    </feature>
    <feature name="Crs.HR">
      <difference change="positive" dialogue="higher heart rate after treatment"/>
      <difference change="equal" dialogue="same heart rate after treatment"/>
      <difference change="negative" dialogue="lower heart rate after treatment"/>
    </feature>
    <feature name="Crs.IncrWOB">
      <difference change="positive" dialogue="higher overall increase in work of breathing after treatment"/>
      <difference change="equal" dialogue="same overall increase in work of breathing after treatment"/>
      <difference change="negative" dialogue="lower overall increase in work of breathing after treatment"/>
    </feature>
    <feature name="Crs.OxSAT_Under_92">
      <difference change="positive" dialogue="higher oxygen saturation after treatment"/>
      <difference change="equal" dialogue="same oxygen saturation after treatment"/>
      <difference change="negative" dialogue="lower oxygen saturation after treatment"/>
    </feature>
    <feature name="Crs.RR_Over_60">
      <difference change="positive" dialogue="lower resp rate over 60 after treatment"/>
      <difference change="equal" dialogue="same resp rate over 60 after treatment"/>
      <difference change="negative" dialogue="higher resp rate over 60 after treatment"/>
    </feature>
    <feature name="Crs.Temp_Over_100.4">
      <difference change="positive" dialogue="lower temperature after treatment"/>
      <difference change="equal" dialogue="same temperature after treatment"/>
      <difference change="negative" dialogue="higher temperature after treatment"/>
    </feature>
    <feature name="Crs.WOB">
      <difference change="positive" dialogue="worse work of breathing after treatment"/>
      <difference change="equal" dialogue="same work of breathing after treatment"/>
      <difference change="negative" dialogue="better work of breathing after treatment"/>
    </feature>
    <feature name="PE.Hydr">
      <difference change="positive" dialogue="worse hydration before treatment"/>
      <difference change="equal" dialogue="same hydration before treatment"/>
      <difference change="negative" dialogue="better hydration before treatment"/>
    </feature>
    <feature name="PE.O2">
      <difference change="positive" dialogue="higher O2 saturation before treatment"/>
      <difference change="equal" dialogue="same O2 saturation before treatment"/>
      <difference change="negative" dialogue="worse O2 saturation before treatment"/>
    </feature>
```

```
      <feature name="PE.Rtr.Sev">
        <difference change="positive" dialogue="better retraction severity before
            treatment"/>
        <difference change="equal" dialogue="same retraction severity before
            treatment"/>
        <difference change="negative" dialogue="worse retraction severity before
            treatment"/>
      </feature>
      <feature name="PE.Gen1">
        <difference change="positive" dialogue="different birth"/>
        <difference change="equal" dialogue="same birth"/>
        <difference change="negative" dialogue="different birth"/>
      </feature>
      <feature name="PastMFS.SmokeMother">
        <difference change="positive" dialogue="none smoking mother"/>
        <difference change="equal" dialogue="same smoking mother"/>
        <difference change="negative" dialogue="smoking mother"/>
      </feature>
      <dialogue>
        <br/>We suggest that this patient should be <replace value="
            classification_dialogue"/> hospital.

        <br/><br/>In support of this prediction we have the Explanation Case that
            appears to have been
        <replace value="utility_shift"/> than this patient (due to <replace value=
            "positive_utilty_features_3"/>)
        but was still <replace value="classification_dialogue"/> hospital.

        <conditional_replace condition="negative_utilty_features" value="0"
            equality_type="greater_than">
         <br/><br/>However it should be noted that the patients <replace value="
             negative_utilty_features_2"/>
         in relation to the Explanation Case
        </conditional_replace>
        <conditional_replace condition="negative_utilty_features" value="1"
            equality_type="equal_to">
        is a feature that goes
        </conditional_replace>
        <conditional_replace condition="negative_utilty_features" value="1"
            equality_type="greater_than">
        are features that go
        </conditional_replace>
        <conditional_replace condition="negative_utilty_features" value="0"
            equality_type="greater_than">
         against our argument that the explanation case is <replace value="
             utility_shift"/> than the patient.
        </conditional_replace>


        <br/><br/><replace value="confidence"/>
        <br/><br/><replace value="include_data/questionnaire.htm"/>
        <br/><br/>Explanation type 2 generated <replace value="system_date_dd MMMM
             yy HH:mm"/> for patient <replace value="target_case_name"/>,
            Explanation Case: <replace value="utility_case_name"/>
      </dialogue>
    </utility>
    <solution>
      <classification value="Discharge" classification_dialogue="discharged from"
          utility_shift="sicker"/>
      <classification value="Admit" classification_dialogue="admitted to"
          utility_shift="healthier"/>
    </solution>
    <mappings>
      <feature name="Age" mapping="age"/>
      <feature name="Crs.HR" mapping="heart rate after treatment"/>
      <feature name="Crs.IncrWOB" mapping="overall increase in work of breathing
          after treatment"/>
      <feature name="Crs.OxSAT_Under_92" mapping="oxygen saturation under 92 after
           treatment"/>
      <feature name="Crs.RR_Over_60" mapping="respriratory rate over 60 after
          treatment"/>
      <feature name="Crs.Temp_Over_100.4" mapping="temperature over 100.4 after
          treatment"/>
      <feature name="Crs.WOB" mapping="work of breathing after treatment"/>
      <feature name="PE.Hydr" mapping="hydration before treatment"/>
```

```xml
    <feature name="PE.O2" mapping="O2 saturation before treatment"/>
    <feature name="PE.Rtr.Sev" mapping="retraction severity before treatment"/>
    <feature name="PastMFS.Birth" mapping="birth"/>
    <feature name="PastMFS.SmokeMother" mapping="smoking Mother"/>
    <feature name="Crs.HR_Over_150" mapping="Heart Rate over 150 after treatment
        "/>
    <feature name="PE.RR_Over_70" mapping="respriratory rate over 70 before
        treatment"/>
    <feature name="Crs.Disp.3Hr" mapping="disposistion"/>
  </mappings>
  <confidence>
    <confidenceMeasure measure="average_nun_index" neighbours="3" threshold="
        19.1"/>
    <confidenceMeasure measure="similarity_ratio" neighbours="1" threshold="1.18
        "/>
    <confidenceMeasure measure="similarity_vote" neighbours="3" threshold="
        1050.46"/>
    <confidenceMeasure measure="neighbour_accuracy" neighbours="12" threshold="
        12"/>
    <confidenceMeasure measure="explanation_ratio" neighbours="30" threshold="
        1.01"/>
    <confidenceDialogue level="confident">
      We have a high confidence in our prediction
    </confidenceDialogue>
    <confidenceDialogue level="not_confident">
      We have a reasonable confidence in our prediction
    </confidenceDialogue>
  </confidence>
</explanation_dialogue>
```

Appendix E

Student's t-Distribution

Table E.1: Critical values of t at various levels of probability (t test)

For any particular *df* the observed value of t is significant at a given level of significance if it is *equal to* or *larger than* the critical values shown in the table

	Level of significance for one tailed test					
df	*0.10*	*0.05*	*0.025*	*0.005*	*0.0025*	*0.0005*
1	3.078	6.314	12.706	31.821	63.657	636.619
2	1.886	2.920	4.303	6.965	9.925	31.598
3	1.638	2.353	3.182	4.541	5.841	12.941
4	1.533	2.132	2.776	3.747	4.604	8.610
5	1.476	2.015	2.571	3.365	4.032	6.859
6	1.440	1.943	2.447	3.143	3.707	5.959
7	1.415	1.895	2.365	2.998	3.499	5.405
8	1.397	1.860	2.306	2.896	3.355	5.041
9	1.383	1.833	2.262	2.821	3.250	4.781
10	1.372	1.812	2.228	2.764	3.169	4.587
11	1.363	1.796	2.201	2.718	3.106	4.437
12	1.356	1.782	2.179	2.681	3.055	4.318
13	1.350	1.771	2.160	2.650	3.012	4.221
14	1.345	1.761	2.145	2.624	2.977	4.140
15	1.341	1.753	2.131	2.602	2.947	4.073
16	1.337	1.746	2.120	2.583	2.921	4.015
17	1.333	1.740	2.110	2.567	2.898	3.965
18	1.330	1.734	2.101	2.552	2.878	3.922
19	1.328	1.729	2.093	2.539	2.861	3.883
20	1.325	1.725	2.086	2.528	2.845	3.850
21	1.323	1.721	2.080	2.518	2.831	3.819
22	1.321	1.717	2.074	2.508	2.819	3.792
23	1.319	1.714	2.069	2.500	2.807	3.767
24	1.318	1.711	2.064	2.492	2.797	3.745
25	1.316	1.708	2.060	2.485	2.787	3.725
26	1.315	1.706	2.056	2.479	2.779	3.707
27	1.314	1.703	2.052	2.473	2.771	3.690
28	1.313	1.701	2.048	2.467	2.763	3.674
29	1.311	1.699	2.045	2.462	2.756	3.659
30	1.310	1.697	2.042	2.457	2.750	3.646
40	1.303	1.684	2.021	2.423	2.704	3.551
60	1.296	1.671	2.000	2.390	2.660	3.460
120	1.289	1.658	1.980	2.358	2.617	3.373
∞	1.282	1.645	1.960	2.326	2.576	3.291

N.B. When required *df* is not shown use the next lowest number, except for very large *dfs* (well over 120), when you can use the row for infinity (∞).