Case Reuse in Textual Case-Based Reasoning

Adeyanju, Ibrahim Adepoju

A thesis submitted in partial fulfilment
of the requirements of
The Robert Gordon University
for the degree of Doctor of Philosophy

This research programme was carried out
in collaboration with the University of Aberdeen

August 2011

Bismi Allah

Dedicated to the glory of Allah, The All-Knowing, The All-Wise

and

my parents, wife (Naseemah), and sons (Abdullah and Yusuf).

Abstract

Text reuse involves reasoning with textual solutions of previous problems to solve new similar problems. It is an integral part of textual case-based reasoning (TCBR), which applies the CBR problem-solving methodology to situations where experiences are predominantly captured in text form. Here, we explore two key research questions in the context of textual reuse: firstly what parts of a solution are reusable given a problem and secondly how might these relevant parts be reused to generate a textual solution.

Reasoning with text is naturally challenging and this is particularly so with text reuse. However significant inroads towards addressing this challenge was made possible with knowledge of problem-solution alignment. This knowledge allows us to identify specific parts of a textual solution that are linked to particular problem attributes or attribute values. Accordingly, a text reuse strategy based on implicit alignment is presented to determine textual solution constructs (words or phrases) that needs adapted. This addresses the question of what to reuse in solution texts and thereby forms the first contribution of this thesis. A generic architecture, the Case Retrieval Reuse Net (CR2N), is used to formalise the reuse strategy. Functionally, this architecture annotates textual constructs in a solution as reusable with adaptation or without adaptation. Key to this annotation is the discovery of reuse evidence mined from neighbourhood characteristics. Experimental results show significant improvements over a retrieve-only system and a baseline reuse technique. We also extended CR2N so that retrieval of similar cases is informed by solutions that are easiest to adapt. This is done by retrieving the top k cases based on their problem similarity and then determining the reusability of their solutions with respect to the target problem. Results from experiments show that reuse-guided retrieval outperforms retrieval without this guidance.

Although CR2N exploits implicit alignment to aid text reuse, performance can be greatly improved if there is explicit alignment. Our second contribution is a method to form explicit alignment of structured problem attributes and values to sentences in a textual solution. Thereafter, compositional and transformational approaches to text reuse are introduced to address the question of how to reuse textual solutions. The main idea in the compositional approach is to generate a textual solution by using prototypical sentences across similar authors. While the transformation approach adapts the retrieved solution text by replacing sentences aligned to mismatched problem attributes using sentences from the neighbourhood. Experiments confirm the usefulness of these approaches through strong similarity between generated text and human references.

The third and final contribution of this research is the use of Machine Translation (MT) evaluation metrics for TCBR. These metrics have been shown to correlate highly with human expert evaluation. In MT research, multiple human references are typically used as opposed to a single reference or solution per test case. An introspective approach to create multiple references for evaluation is presented. This is particularly useful for CBR domains where single reference cases (or cases with a single solution per problem) typically form the casebase. For such domains we show how multiple references can be generated by exploiting the CBR similarity assumption. Results indicate that TCBR systems evaluated with these MT metrics are closer to human judgements.

Acknowledgements

I am greatly indebted to my supervisors, Dr. Nirmalie Wiratunga and Dr. Robert Lothian, for their guidance and support throughout the research. I am grateful to Dr. Somayajulu Sripada of University of Aberdeen for his thoughtful criticism of my work and ideas which led to some of my contributions to knowledge. I also wish to acknowledge Prof. Susan Craw and Prof. Ehud Reiter for their crucial support at the very beginning of my PhD and at other times during the research. Special thanks to Dr. Juan Recio-Garcia of Universidad Complutense de Madrid and Dr. Luc Lamontagne of Université Laval, Québec for their collaboration on some of the work presented in this thesis. I acknowledge Dr. Sutanu Chakraborti and Prof. Deepak Khemani of Indian Institute of Technology, Madras for their constructive feedback during my research visits to their university.

Many thanks to my sponsors, Northern Research Partnership (NRP), UK-India Education and Research Initiative (UKIERI) and Scottish Overseas Research Students Award Scheme (ORSAS). Gratitude also goes my colleagues, Stella, Raman, Bayo, Amandine, Jean-Claude, Ulises, Sandy, Thiery, David, Malcolm, Guofu, Ibrahim, Peter, Peng, Sadiq, Amina, Richard, Noura, Micheal, Claire, Ben, Leszek, Nuka, Olivier, Lei and Yanghui, for making CTC a convenient research environment. A final thank you to all those who made this research possible in one way or another.

Declarations

I declare that I am the sole author of this thesis and that all verbatim extracts contained in the thesis have been identified as such and sources of information specifically acknowledged in the bibliography. I have also obtained permission from the owners of third party copyrighted materials to include these materials in my thesis.

Parts of the work presented in this thesis have appeared in the following publications.

Chapter 3

- Ibrahim Adeyanju, Nirmalie Wiratunga, Robert Lothian, Somayajulu Sripada, Luc Lamontagne (2009). Case Retrieval Reuse Net (CR2N): An Architecture for Reuse of Textual Solutions, *in* L. McGinty and D.C. Wilson (eds), *Proceedings of ICCBR 2009*, LNAI 5650, Springer-Verlag, Berlin Heidelberg, pp. 14-28.

- Ibrahim Adeyanju, Nirmalie Wiratunga, Robert Lothian, Somayajulu Sripada, Susan Craw (2008). Solution reuse for textual cases, *in* M. Petridis (ed.), *Proceedings of UKCBR Workshop 2008*, CMS Press, Greenwich, pp. 54-62.

Chapter 4

- Ibrahim Adeyanju, Nirmalie Wiratunga, Juan Recio-García, Robert Lothian (2010). Learning to Author text with textual CBR. *in* H. Coelho, R. Studer and M. Wooldridge (eds), *Proceedings of ECAI 2010*. IOS Press, Amsterdam, pp. 777-782.

Chapter 5

- Ibrahim Adeyanju, Nirmalie Wiratunga, Robert Lothian, Susan Craw (2010). Applying Machine Translation Evaluation Techniques to Textual CBR. *in* I. Bichindaritz and S. Montani (eds), *Proceedings of ICCBR 2010*, LNAI 6176. Springer-Verlag, Berlin Heidelberg, pp. 21-35.

Contents

List of Algorithms

List of Figures

List of Tables

Chapter 1

Introduction

The advent of the computer age has led to the storage and communication of vast amounts of data by electronic means. Much of this is generated by professionals in various fields and typically consists of experiential knowledge. Examples include medical diagnosis reports, air traffic incident reports, weather forecasts, newspaper articles, business proposals, internal memos, emails and web pages. This has created an opportunity for experiential knowledge reuse to assist with decision support and for training purposes. Much of this content is captured in text thereby presenting interesting challenges for knowledge reuse.

Case Based Reasoning (CBR) is an artificial intelligence paradigm for solving new problems by reusing previous similar problem solving experiences (Kolodner 1993, Aamodt & Plaza 1994, Leake 1996, Lenz, Bartsch-Sporl, Burkhard & Wess 1998, Cunningham 1998, Lopez de Mantaras, McSherry, Bridge, Leake, Smyth, Craw, Faltings, Maher, Cox, Forbus, Keane, Amodt & Watson 2005) . The basic assumption is that similar problems have similar solutions and similar problems re-occur. Traditionally, CBR has been applied in domains where experiential data is structured, that is consisting of simple and complex data types such as numbers, symbols and complex design plans. This is because it is generally easier to manipulate structured data; for example, similarity between two structured problems can be computed using standard measures (e.g. euclidean distance) or other user-defined measures tailored to a specific domain. Reasoning with textual data is harder mainly due to inherent ambiguities in vocabulary. Ambiguities can come from

the same piece of information being conveyed in different ways when written down as text (synonyms) or the same piece of text having different meanings depending on the context (polysemy). This thesis focuses on aiding experiential reuse when the data is predominantly made up of textual contents. In particular, we address questions relating to what and how to reuse textual experiential data and propose answers to these questions.

The remainder of this chapter is as follows. We give an overview of Textual CBR, a sub-field of CBR that deals with textual experiential reuse, in Section 1.1. Here, the basic assumptions in CBR and the context of our text reuse are explained before discussing the challenges. Section 1.2 discusses what motivated this research and what we intend to achieve in terms of objectives. Lastly, we give an overview of the remaining chapters in this thesis in Section 1.3.

1.1 Textual Case Based Reasoning

Textual Case Based Reasoning (TCBR in short) is a sub-field of CBR that assists in the reuse of experiential documents (cases) when they are predominantly captured in textual form. The unstructured nature of textual cases makes TCBR a specialized research area and differentiates it from traditional CBR where experiences consist of structured attributes each with a set of limited possible values; for example, numeric and symbolic values or complex objects as used in planning and design. Reasoning with textual cases is therefore more challenging because of the absence of a well-defined case structure. However, TCBR upholds the same assumptions as traditional CBR and conceptually uses the same reasoning process when reusing a previous experience to solve a new problem. We therefore examine the CBR assumptions and problem-solving process in the context of TCBR and discuss its associated challenges. A comparison is then made between TCBR and other research disciplines that share similar challenges as regards text manipulation.

1.1.1 CBR assumptions and problem-solving methodology

The implicit assumptions underlying the CBR paradigm are succinctly captured by Massie (2006). These are : Regularity, Repetition, Representativeness and Experiential. Regu-

larity deals with the fact that a CBR system assumes that similar problems have similar solutions. This is key to the technique because it attempts to reuse solutions from previous similar problems to solve a new problem. The motivation here is that it is easier to modify previous solutions while solving a new similar problem rather than solving it from scratch. The assumption of repetition follows logically from the regularity assumption whereby if similar problems do not re-occur or repeat over time, then previous solutions cannot be applied. In other words, CBR can only be applied in domains where similar problems re-occur and have similar solutions. Interestingly, humans typically think about previous similar scenarios when solving new problems; this makes CBR suitable and intuitive for most domains. Next assumption, representativeness, ensures that a CBR system finds similar previous experiences that are relevant when providing solutions to a new problem. The last of the assumptions, experiential, implies that the collection of previous experiences should be the primary source of knowledge used in problem-solving. Other sources such as domain and background knowledge should therefore be auxiliary to support the problem-solving process.

The CBR problem-solving process has been conceptualized in different ways such as those described by Kolodner (1993), Leake (1997) and Watson (1997) . However, the most popular conceptual view of the CBR process was proposed by Aamodt & Plaza (1994) as shown in Figure 1.1. Central to this process is the case knowledge in the form of previous experiences which are problem-solving episodes. Each episode is referred to as a case and consists of at least two components, a problem and its solution. A case might also contain other components such as justification and result which store additional knowledge (Recio-García, Díaz-Agudo, Sánchez-Ruiz & González-Calero 2006, Massie, Wiratunga, Craw, Donati & Vicari 2007). The justification component, when used, indicates how the problem leads to the solution and why it is the best solution. The result component of a case stores the feedback from applying a solution. Such feedback can be positive or negative and aids selection of the best choice when a problem has multiple solutions. A set of cases forms the casebase which is the primary source of knowledge. A typical CBR problem-solving cycle (see Figure 1.1) has four stages: Retrieve, Reuse, Revise and Retain

commonly referred to as the 4Rs.

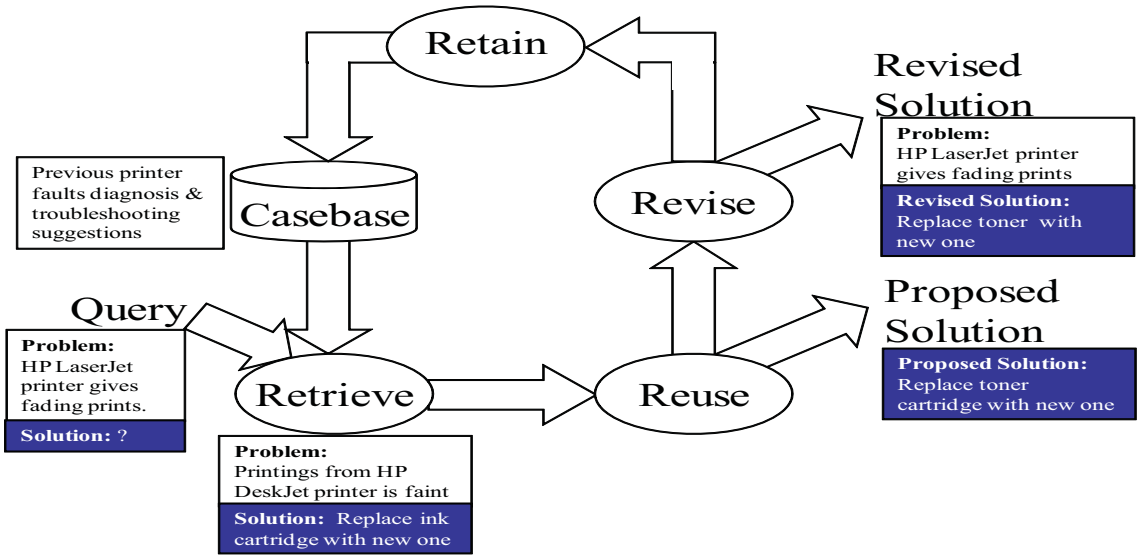


Figure 1.1: A typical CBR cycle

- Retrieve: This is the first and most important stage of the CBR process as subsequent stages depend heavily on it. Here, similar cases are retrieved from the casebase when a new problem is encountered. Defining similarity is at the heart of the retrieval process and can be done in consultation with a domain expert. Most of the earliest CBR systems such as PERSUADER (Sycara 1987, Sycara 1988), BATTLE PLANNER (Goodman 1989), CLAVIER (Hennessy & Hinkle 1992), CASCADE (Simoudis 1991, Simoudis 1992), ARCHIE-2 (Domeshek & Kolodner 1991, Domeshek & Kolodner 1992, Domeshek & Kolodner 1993) and ASK (Ferguson, Bareiss, Birnbaum & Osgood 1992), were retrieve-only systems and left the subsequent stages to human users. More detailed discussion on retrieval and similarity mechanisms appears in Section 2.2.1.

- Reuse: The information and knowledge in the retrieved similar cases are reused to solve the current problem. Such reuse can be verbatim in which the solution is applied directly to solve a new problem. However, reuse can also be by analogy where the method of obtaining the solution from the retrieved problem is replayed in the context of the new problem. Reuse by analogy requires knowledge of the

relationship or alignment between problems and solutions; the justification component of a case, see (Recio-García et al. 2006), can be used to store such alignment knowledge. A simple example of alignment knowledge is a function that accepts some of the problem attributes and whose output is the value of a particular solution attribute. The retrieval and reuse stages are sometimes combined into a single retrieval stage, typically when the retrieved solution is being used verbatim. According to Aamodt & Plaza (1994), the reuse stage consists of two kinds of tasks: copy and adapt. Copy is when a single retrieved solution is used verbatim or when the solution method is used for analogy. Adapt has to do with modifying the solution or solution method. Adaptation is typically carried out because the query and retrieved problem are similar but not identical (i.e. exactly the same). Therefore, the solution being reused from the retrieved similar case might need to be modified to compensate for problem mismatches. Adaptation takes into account the differences between the problems (current & retrieved) to guide any adjustment required in the retrieved solution or solution method. Adaptation is the 'Achilles heel' of CBR research and most of the earlier CBR systems left this task to human users, as part of the next revision stage. Nevertheless, recent works have led to the development of automated adaptation techniques but most remain domain dependent and computationally expensive (Craw, Wiratunga & Rowe 2006, Leake & Powell 2007, Cojan & Lieber 2008, Sugandh, Ontanón & Ram 2008, Badra, Cordier & Lieber 2009, Leake & Kendall-Morwick 2009, Dufour-Lussier, Lieber, Nauer & Toussaint 2010, Leake & Powell 2010, Ontanón & Plaza 2010). Further discussions on CBR reuse and adaptation can be found in Sections 1.1.2 and 2.2.

- Revise: This stage ensures that a proposed solution from a CBR system is meaningful, correct, and useful for solving the current problem. The tasks during revision are evaluation of the proposed solution and repair of any observed faults. Evaluation of the proposed solution is done by a user, applying it to the real world or using known domain models. Any faults observed in the solution during evaluation can then be repaired by a user or with other domain knowledge. The revision stage is

where a proposed solution is reviewed for faults and repaired accordingly to obtain a revised solution that is correct and useful for a user.

- Retain: The final stage of the CBR process is to retain the new case which consists of the new problem and revised solution. This can be seen as a learning stage for a CBR system, because it enables it to acquire more knowledge that can be used for problem-solving in the future. Retention over a period of time is particularly useful for industrial CBR systems because it improves representativeness and system performance. However, it is important to maintain a useful case base by not retaining redundant or noisy cases. Further details on casebase maintenance techniques can be found in (Massie 2006).

1.1.2 Text Reuse in context

We explain our view of text reuse as an integral part of TCBR. Text reuse is applicable when the solution component of a case structure used for problem-solving contains one or more textual attributes which are typically unstructured or semi-structured. However, the problem component can consist of attributes with either structure and/or textual data types. This is different from other areas where TCBR is used for text classification tasks (Wiratunga, Koychev & Massie 2004, Delany, Cunningham & Coyle 2005, Wiratunga, Lothian & Massie 2006, Chakraborti, Wiratunga, Lothian & Watt 2007, Orecchioni, Wiratunga, Massie & Craw 2008). For classification applications, the problem component is textual but the solution is a class label which can be single or multiple.

Our concept of text reuse focuses on solving a new problem by reusing solution descriptions from a textual casebase. Here, case retrieval is similar to the task of Information Retrieval (IR) systems where a user's information need is encoded as a textual query to retrieve documents that are mainly textual in content. However, TCBR works with experiential contents while IR does not. Nevertheless, the textual query in IR can be viewed as a problem and retrieved documents as the solution. Text reuse in TCBR goes beyond standard text retrieval as used in IR because the problem and solution space are separate and usually have distinct vocabularies. TCBR systems can highlight or propose sections

in the retrieved text that might need to change due to differences (implicit or explicit) between the query and retrieved problem. TCBR systems can also carry out named entity replacement during text reuse (Lamontagne & Lapalme 2004) or even combine multiple retrieved textual solutions when proposing a textual solution. These are some of the distinctive features of text reuse in TCBR; further differences between TCBR and other related research areas are given in Section 1.1.4.

Although text reuse goes beyond standard text retrieval, the output is a proposed textual solution. Revision of this proposed solution using external knowledge sources might be required to ensure that the final text is meaningful, coherent and useful; typically at the revision stage of the problem-solving process. An example of such external knowledge is a human user who can make appropriate modifications to the proposed textual solution. Revision should not be confused with adaptation which is part of the reuse stage. Textual adaptation includes any automated modifications to retrieved solutions before its presentation as a proposed solution.

In this thesis, the term 'text reuse' refers to the component of the TCBR cycle in which a textual solution is improved beyond retrieval. Improvements can include annotations of sections in the retrieved text as relevant/irrelevant to the query, removal of sections in the retrieved text to generate a partial solution, combination of solution text from multiple cases, replacement of specific terms in the retrieved text and other possible automated adaptation methods. Text reuse therefore includes adaptation, which is unlike the TCBR revision stage where proposed textual solutions are reviewed and observed faults are repaired by a human or with the aid of a domain model.

1.1.3 Challenges in Text Reuse

Real-world textual experiences are mostly unstructured or semi-structured. The nature of these experiences leads to challenges that are specific to TCBR since traditional CBR employs experiences with structured attributes. Reasoning with textual cases is more challenging, because a well-defined structure is absent (Weber, Ashley & Bruninghaus 2006).

1. Problem to Solution Alignment: When both the problem and solution components of a case have multiple attributes, values of related solution attributes are likely to depend on a specific combination of problem attributes rather than all. Such causal or alignment knowledge is invaluable for reasoning. This is because the best values for solution attributes can be retrieved from different cases with aligned problem attributes most similar to the query. This alignment knowledge allows mappings from particular problem attributes (or attribute values) to their resultant solution attributes (or attribute values) thereby leading to better retrieval accuracy. However, learning such relationships or alignment between problem and solution attributes remains a challenge when they are not explicitly expressed in the domain. This applies to text reuse where it is difficult to predict which unit of text (e.g. phrase or sentence) aligns to specific problem attributes. The only attempt at alignment in CBR and TCBR literatures (to the best of our knowledge) is an alignment approximation proposed by Lamontagne, Langlais & Lapalme (2003) which exploits word co-occurrence and machine translation techniques. We address this challenge by introducing a method for explicit alignment of problem and solution attributes in textual cases (details in Chapter 4).

2. Acquiring adaptation knowledge: Adaptation (one of the reuse tasks) remains the Achilles heel of CBR research and this also applies to TCBR. Textual adaptation includes any modification to a retrieved solution to obtain a proposed solution and is more challenging than traditional CBR adaptation. This arises from the difficulty in capturing the semantic differences between two pieces of text and how they influence the changes that need to be carried out in a retrieved textual solution. Choosing an appropriate text granularity is also important for adaptation. Smaller textual units such as keywords and phrases are more suitable for domains with short textual contents, for example with less than four sentences, but working with sentences or paragraphs might be better for cases with larger textual contents. Other questions that might need to be addressed during adaptation include the role of the grammatical and semantic information in determining difference between textual cases. Part

of speech tags, word sequence and subject-object differentiation provide information about grammatical syntax, while synonyms, negation and sense disambiguation deal with semantics. Part of this challenge is addressed by our work on annotating sections that need adaptation in a retrieved textual solution and distinguishing them from those that can be reused verbatim. We provide details of this technique in Chapter 3.

3. Automated evaluation of textual solutions: Current automated TCBR methods for evaluating textual solutions, such as precision and recall (Brüninghaus & Ashley 1998a, Lamontagne & Lapalme 2004, Asiimwe 2009), were adapted from Information Retrieval (Baeza-Yates & Ribeiro-Neto 1999). Here, a piece of text is treated as a sequence-independent set of words. During empirical evaluation, effectiveness is computed by exploiting word matches between the proposed solution and a single reference solution provided by a human expert. However, these evaluation metrics are inadequate in domains where experts use different phrases to express the same solution. This is because these metrics operate at the word granularity and are unable to capture semantically similar text at higher granularity (phrase or sentence). Therefore, TCBR evaluation results for such domains can be unreliable or misleading. Our work addresses this challenge by exploiting better text evaluation measures from other research areas, in particular machine translation (see Chapter 5). Further details on TCBR evaluation also appears in Section 2.3.

4. Mapping text into a structured representation: Traditional CBR techniques are more established than those for TCBR since they have been in use for a longer period. Therefore if textual cases can be transformed into structured equivalents, traditional CBR techniques can be seamlessly applied. Information extraction (Cowie & Lehnert 1996, Brüninghaus & Ashley 2001) has been used to identify structured content in text. PRUDENTIA (Weber, Martin & Barcia 1998) and SMILE (Brüninghaus & Ashley 2005) are TCBR systems that have applied this approach to legal texts and reported encouraging results. But transformations into a structured representation are more difficult or near impossible for many unstructured texts. This is because

these techniques for text transformation mostly rely on grammatically correct text which cannot be guaranteed for some corpora such as emails. Section 2.1.2 provides further details on techniques that have been used for mapping text into structured representations.

5. Textual case representations: Transformation from unstructured text to structured form is difficult and the basic representation of text, typically as a vector of keywords, is sometimes insufficient for reasoning. Clearly, there is a need for alternative and better representations of textual cases. One such alternative is the use of propositional clauses (Wiratunga, Lothian, Chakraborti & Koychev 2005) in case vectors in addition to keywords. A propositional clause is a logical combination of keywords into an interpretable concept such as " 'intelligent' $\wedge$ 'algorithm' $\wedge$ ('grant' $\vee$ 'application')". Other alternative representations proposed include the use of graphs (Cunningham, Weber, Proctor, Fowler & Murphy 2004), latent semantic indexing concepts (Deerwester, Dumais, Furnas, Landuer & Harshman 1990) and language trees (De Marneffe, MacCartney & Manning 2006). However, these alternatives increase the complexity of the TCBR system and sometimes offer no significant advantage in terms of system performance. More discussion on current techniques for representing textual cases can be found in Section 2.1.2.

6. Effective indexing architectures: Textual cases are typically decomposed into smaller units such as keywords, phrases or sentences for comparison to other cases. This increases memory required for computing text similarity as opposed to structured cases that need no further decomposition. Therefore, better indexing architectures are needed to allow for fast, efficient and effective textual case comparison during retrieval and reuse. Such architectures should also enable cases to be represented at different text granularities. Inverted file lists (Baeza-Yates & Ribeiro-Neto 1999) and case retrieval networks (Chakraborti, Lothian, Wiratunga, Orecchioni & Watt 2006) are indexing architectures that have been used previously for efficient and effective TCBR retrieval. Further details of indexing architectures for text reuse appear in Section 2.1.3.

7. Similarity between textually represented cases: A textual attribute can be represented as a vector of its keywords using any weighting function such as binary or term frequency. However, it is also important to consider other grammatical, semantic and domain knowledge when measuring similarity between two pieces of text. Other grammatical knowledge includes information on part of speech tags, subject-object identification and reasoning with alternative decompositions of the text such as phrases and sentences. Semantic and domain knowledge provide information about synonyms, negation and sense disambiguation, thereby correctly contextualizing each textual unit (word, phrase or sentence). The use of these knowledge sources ensures that textual similarity is meaningful and useful for the reasoning process. FAQ-Finder (Burke, Hammond, Kulyukin, Lytinen, Tomuro & Schoenberg 1997) and FallQ (Kunze & Hubner 1998) are good examples of TCBR systems that have attempted to take into account the semantic meanings of keywords using general-purpose and domain-specific ontologies. However, ontologies do not exist for most domains and general-purpose ontologies can misinterpret the contextual meanings of important keywords. Further discussion on similarity and retrieval of textual cases appears in Section 2.2.1.

1.1.4 TCBR and other text related research

CBR is related to several other research areas from which it borrows techniques for different stages of the reasoning process. Related research areas include (but are not limited to): databases, knowledge representation, machine learning, cognitive science, information retrieval and neural networks. An in-depth analysis of the relationship between CBR and these research fields can be found in (Kamp, Lange & Globig 1998). However, TCBR is closely related to other research areas that deal with manipulating or generating textual data/information. In particular, we make a distinction between TCBR and the following research areas: Information Retrieval (IR) (Baeza-Yates & Ribeiro-Neto 1999), Text Mining (TM) (Nenadic, Spasic & Ananiadou 2004, Palshikar 2007), Natural Language Generation (NLG) (Reiter & Dale 1995), Text Summarization (TS) (Neto, Freitas

& Kaestner 2002) and Machine Translation (MT) (Brown, Della Pietra, Della Pietra & Mercer 1993, Koehn 2010). A comparison between these TCBR related research areas appears in Table 1.1. We can see that while the research focus, dataset structure and real-world applications of TCBR are very different from these other research fields, they sometimes share similar methods for text processing. For instance, knowledge-light approaches to TCBR (which are more common) use shallow NLP methods (tokenisation, stemming, stop-words removal) similar to IR and Text Summarization. On the other hand, the deep NLP methods (language trees) used by TCBR knowledge-intensive approaches are very similar to those used in NLG and MT.

1.2 Research Motivation and Objectives

Majority of work on TCBR has to date focused on case indexing, representation and retrieval. Reuse and adaptation of solutions from previous similar cases is a challenge, particularly for TCBR where experiences are captured in textual form. Therefore research into techniques that can be used in the reuse phase for textual cases remains a largely unexplored area.

The aim of our research is to develop methods that can aid reuse and adaptation of textual cases with particular focus on experiences whose solutions are captured in textual form. We specifically addressed the following five objectives:-

1. *Propose a method to approximate problem-solution alignment in textual cases.*
2. *Develop an algorithm for textual solution reuse in TCBR.*
3. *Propose an algorithm for reuse guided text retrieval.*
4. *Develop techniques for transformational and compositional approaches to text reuse.*
5. *Propose semantic-aware automated evaluation measures for TCBR.*

The first three objectives address our research question of what to reuse in experiential cases with textual solutions while the fourth objective addresses the question of how to reuse. The last objective is important because it allows us to carry out comparative studies

Table 1.1: TCBR and other text related research

	Textual Case Based Reasoning	**Information Retrieval**	**Text Mining**	**Natural Language Generation**	**Text Summarization**	**Machine Translation**
Research Focus	reuse experiences documented in form of text for problem solving	satisfy a user's information need with relevant textual documents	discover implicit patterns and information from textual documents	generate useful and meaningful text descriptions of non-linguistic data	make a piece of text more concise	produce textual translations from one language to another
Dataset Structure	textual documents organised into problem and solution	textual documents (no problem/ solution identification)	textual documents (no problem/ solution identification)	non-linguistic /textual description can be seen as a problem-solution pair	textual documents (no problem/ solution identification)	parallel texts of translations can be viewed as problem-solution pairs
Nature of techniques	typically incorporates domain knowledge	general purpose and no domain knowledge incorporated	can use domain knowledge	uses domain knowledge with lots of interaction with experts	typically no domain knowledge incorporated	can use domain knowledge
Natural Language Processing	typically uses shallow NLP	shallow NLP	shallow NLP	deep NLP	shallow NLP	deep/ shallow NLP
Common Evaluation Metrics	precision, recall case alignment, case cohesion	precision, recall, f-measure	precision, recall, f-measure	edit distance, NIST, BLEU	ROUGE	BLEU, NIST
Real World Application Examples	help-desk assistants, question answering, incident reporting	internet/ intranet search	marketing, bio-informatics	weather forecast, map interpretation	abstracts/ summary generation	traveller's language assistant

of text reuse algorithms, taking into account the syntax and semantic of the texts they generate.

1.3 Thesis Overview

This chapter gave an overview of the TCBR research area in which our work on text reuse is situated and highlighted the challenges with text reuse. We also gave a scope to the research study by discussing our aims and objectives which are key towards understanding the contributions of the thesis in the subsequent chapters. In the next chapter, a critical analysis of previous relevant literatures is presented, the TCBR knowledge acquisition process is discussed and the two main approaches to text representation are identified. The merits and demerits of each approach are explained and we give reasons for the approach selected for use in our research. Also, different techniques for reasoning with problem-solving experiences are examined as well as the dependencies between similarity and retrieval performance. We then analyse current state-of-the-art methods for reusing structured and textual cases and discuss how to evaluate the performance of TCBR systems. We conclude Chapter 2 by analysing the datasets used in our experiment and their suitability for text reuse research.

Chapter 3 presents our contribution on a text reuse algorithm which identifies reusable textual constructs in a retrieved solution text relevant to a given query and differentiates them from those that need adaptation at the reuse stage of the TCBR cycle. Our text reuse algorithm introduces novel methods for calculating the reuse evidence which determines the relevance/irrelevance of each retrieved solution term to the query. It is generic because it can work with any level of text granularity (keyword, phrase, sentence or paragraph levels) and also allows the use of the entire casebase or neighbourhoods in the problem or solution space during reuse evidence computation. We propose the integration of the retrieval and reuse stages in TCBR into a single architecture called Case Retrieval Reuse Net (CR2N). The integration enables our algorithm to leverage the established capabilities of the Case Retrieval Net (CRN) in terms of effectiveness and efficiency. Also in Chapter 3, we extend CR2N's utility to guide retrieval of similar cases whose solutions are easiest to

adapt. A reuse utility score is introduced which combines retrieval similarity and the proportion of reuse. Nearest neighbours of a query are then re-ranked so that the best match is not just most similar to a query but less effort is spent to adapt its solution during reuse.

We introduce two novel concepts in relation to text reuse, text alignment and prototypical text generation, in Chapter 4. Text alignment links structured problem attributes to specific chunks of a solution text while prototypical text generation abstracts similar chunks of text into a single meaningful prototype. These concepts are generally applicable in domains where cases consists of pre-defined structured attributes along with written text. We then propose two novel text reuse techniques that generate proposed solution texts in response to a query with the predefined structured attributes. Chapter 5 discusses our work on adapting two machine translation (MT) evaluation techniques, BLEU and NIST, for TCBR. However, these metrics are typically used with multiple reference solution texts, but all TCBR datasets (to the best of our knowledge) have cases with single reference solutions. We therefore propose a novel introspective approach to generate multiple references when they do not naturally occur in a domain. We also discuss the effect of intrinsic differences in the text generation process between TCBR and MT and how these might influence the application of MT measures during TCBR evaluation.

Chapters 6 and 7 discuss the experimental set-up used to test the effectiveness of our novel techniques presented in chapters preceding them. A critical and thorough analysis of the evaluation results is then carried out to obtain evidence in support of our techniques. We conclude the thesis with a list of our major contributions and propose future extensions to our work in Chapter 8.

Chapter 2

Background

A critical appraisal of literature and previous research related to our work in Textual Case Based Reasoning (TCBR) is given in this chapter. Specifically, we address issues of knowledge acquisition, representation and reuse in TCBR. Indexing architectures that allow for efficient and effective use of the acquired knowledge are also discussed. We examine current retrieval and reuse techniques in structured CBR, where case attributes are well-defined with a limited number of attribute values, and discuss their applicability to TCBR. Evaluation is an important aspect of TCBR research as there are currently no benchmark datasets nor performance metrics to compare TCBR techniques that carry out identical tasks. We address this by exploring evaluation metrics that are used in other related fields with a view to adapting them for TCBR. Characteristics of datasets used in our comparative studies are also introduced in this chapter.

The chapter is organized as follows. We describe the TCBR knowledge acquisition process in Section 2.1 with emphasis on representation at different levels of knowledge. Section 2.2 surveys related work on techniques used for retrieval and reuse in CBR and addresses the question of why techniques for structured cases are not directly applicable to textual cases. Current techniques for reasoning with textual cases are then analysed. Section 2.3 discusses evaluation techniques (casebase complexity and performance metrics) for measuring a TCBR system's effectiveness. Datasets used in later chapters for our experimental evaluation are described in Section 2.4 and a summary appears in Section 2.5.

2.1 From Documents to Textual Cases

The advent of the computer age has several advantages. One of them is the increased use of personal computers (PCs) by experts from different industries to store important information as electronic documents rather than, or in addition to, paper documentation. These electronic documents, which include reports, proposals and email communication with customers or colleagues, typically contain problem-solving experiences of which a large part is textual. This creates an opportunity for automated reuse of such experiences, since similar problems re-occur and should have similar solutions thereby increasing the efficiency of problem-solving processes, because problems need not be solved from scratch. For instance, it should be easier to modify a previous proposal document to respond to a new proposal request, if there has been a similar request in the past.

It is important to verify that the application domain fulfils the CBR assumptions discussed in section 1.1.1. However automated experience reuse with TCBR can only take place after transforming documents into textual cases. This involves identifying problem and solution components together with their attributes that form the case structure. Case representation, knowledge containers and indexing strategies that need to be considered at the start of the problem solving process are also discussed in this section.

2.1.1 Knowledge Requirements for TCBR

Four knowledge containers have been identified as crucial for a CBR system to work effectively (Richter 1998). Each container encodes knowledge about one of the vocabulary, similarity/case comparison, casebase and solution transformation. These containers are used at the retrieval and reuse stages of the problem-solving cycle. A fifth knowledge container has been proposed (Patterson, Anand, Dubitzky & Hughes 2000, Patterson, Rooney, Galushka & Anand 2002, Iglezakis, Reinartz & Roth-Berghofer 2004, Massie 2006), which encodes maintenance knowledge to ensure that duplicate and noisy cases are not stored in the casebase at the retain stage.

1. Vocabulary: This deals with the composition of a case. A basic CBR case consists of two components: a problem and its solution. However, other components such

as justification and result (outcome) might be included as part of a case if it eases the reasoning process. Each component of a case has at least one attribute and each attribute could be a simple (e.g. integer) or a complex (e.g. graph) data structure. For TCBR, at least one of the attributes in the problem or solution component must be textual. The vocabulary for each textual attribute can be elicited as terms (words, phrases or sentences) from cases in the casebase which can then be augmented with domain knowledge and other linguistic knowledge such as synonyms, negation and sense disambiguation.

2. Casebase: The collection of problem solving experiences (cases) representative of the domain must be carefully selected as this forms the main knowledge source used for solving new problems. Any re-representation of the casebase, such as indexing, that makes reasoning more efficient/effective must be properly linked to the original cases since retrieved or adapted cases are typically shown to users in the same format as documented in the casebase. This is more important for TCBR since textual contents are typically represented differently from their original form.

3. Similarity/Case comparison: Retrieval quality is highly dependent on the measure of similarity which in turn depends on indexing and representation. This knowledge container can also include the relative importance of each attribute for retrieval when there is more than one attribute in the problem component of a case. For TCBR, the challenge is in finding an internal representation that encodes the syntactic and semantic content of a text before applying a similarity metric. This is because the same information can be written as text in several ways, sometimes with a totally different set of words. Also, the same term (word, phrase or sentence) can have different meanings depending on the context; this is known as polysemy. For instance, 'bank' can refer to a 'river bank' or a 'commercial bank'. Therefore, a lot of TCBR representation and similarity knowledge containers incorporate mechanisms that address some of these challenges.

4. Solution transformation: Refers to knowledge of how to adapt solutions during the

reuse stage of the problem solving cycle. In traditional CBR, adaptation knowledge is often encoded in rules (Leake, Kinley & Wilson 1995, Hanney & Keane 1997) or as an adaptation casebase (Craw et al. 2006). Such knowledge is manually or introspectively extracted to capture differences between similar problems and their impact on solution differences. However, this is harder for TCBR since the process of encoding semantic difference between two pieces of text is less obvious even for a human. In this thesis, we introduce novel techniques that automatically capture knowledge that aids solution transformation in textual cases.

5. Maintenance: A significant advantage of using CBR for problem solving is that the system evolves and performs better over a period of time because it learns and stores new cases that were not in the original case base. Therefore, there is a need to ensure that conflicts do not arise as a result of this learning process. Old cases might need to be deleted if they are no longer relevant while others may need to be replaced by new and improved solutions. These are part of the maintenance knowledge required to ensure that the retention of new cases does not adversely affect a CBR system. Maintenance knowledge is also important for TCBR but very little research has been done in this area. This might be due to the same challenges as with reuse of textual cases where it is difficult to completely capture all syntactic and semantic information carried by the cases. However, capturing maintenance knowledge for TCBR is beyond the scope of our work.

Although this thesis focuses on developing techniques that capture knowledge for text solution transformation, it is also important that we understand its interaction with the preceding knowledge containers; that is, vocabulary, casebase and similarity/case comparison. We do not discuss the maintenance knowledge containers further as it is used after solution transformation and beyond the scope of our study.

2.1.2 Text Representation

The internal representation of a case is crucial for reasoning to take place. It impacts case comparison which in turn influences the choice of similarity metrics. A good representation

ensures that contextually similar cases are close together in the problem search space while dissimilar cases are farther apart. Although several CBR case representation formalisms (Bergmann, Kolodner & Plaza 2005) have been proposed, the most common approach is the attribute-vector representation where cases are composed of attribute-value pairs. However, this is not directly applicable to textual cases due to the absence of pre-defined attributes and related values. As discussed earlier in Section 1.1.3, representation of textual cases without loss of syntactic and semantic information remains a challenge.

Current approaches to textual case representation can be divided into: knowledge-light and knowledge-intensive. Knowledge-light approaches (Lamontagne & Lapalme 2004, Wiratunga et al. 2006, Recio-García, Díaz-Agudo & González-Calero 2007, Orecchioni, Wiratunga, Massie, Chakraborti & Mukras 2007) are adapted from Information Retrieval (Baeza-Yates & Ribeiro-Neto 1999) research where texts are represented using the vector-space model. In this approach also known as bag of words (BOW), textual contents are treated as sequence-independent textual units. In this thesis, we call this bag of terms (BOT) instead of BOW where a term (textual unit) can be a (key)word, phrase, sentence or concept depending on the level of representation granularity. A separate vocabulary is created for each textual attribute using its terms across all cases in the casebase. Terms in this vocabulary are then used as attributes in the vector space. The value for each attribute in a case can be binary (0 or 1) indicating the absence or presence of particular terms, term frequency (tf), inverse document frequencies (idf) or tf*idf which is a combination of term frequency and inverse document frequency.

The decomposition of text into smaller units such as words and phrases leads to the curse of high dimensionality with a huge number of terms that can have an adverse effect on the efficiency of retrieval. Therefore, there is a need to reduce the high dimensionality while maintaining an accurate representation. This can be done using feature selection and extraction techniques. Feature selection is the process of choosing a subset of the terms to accurately represent text and can be based on term frequency (TF), inverse document frequency (IDF), information gain (IG) and χ^2-test (CHI) (Yang & Pedersen 1997, Liu, Liu, Chen & Ma 2003). On the other hand, feature extraction transforms a representation

into a smaller set of new features by combining terms using generalization (Wiratunga et al. 2004) or dimensionality reduction techniques such as Latent Semantic Indexing (LSI) (Deerwester et al. 1990), Principal Component Analysis (PCA) (Zhao, Chellapa & Krishnaswamy 1998) or Propositional Semantic Indexing (PSI) (Wiratunga et al. 2005).

Another common characteristic of knowledge-light approaches to text representation is minimal use of natural language processing (NLP) tools resulting in shallow NLP for lexical (stemming, lemmatization, stop-word removal, part-of-speech tagging) and semantic (subject-object identification) tasks. A major advantage of knowledge-light approaches is that they are intrinsically domain-independent and can therefore be used across several domains. Although this approach seems too simplistic for reasoning with texts, relatively good performance has been reported in the literature (Lamontagne & Lapalme 2004, Recio-García et al. 2007). Nevertheless, adaptation of a retrieved textual solution during text reuse is more difficult since most of the contextual information is lost. Substitution of specific terms, typically keywords, is the most common form of TCBR adaptation (Zhang, Hu, Namee & Delany 2008, Adeyanju, Craw, Ghose, Gray & Wiratunga 2008) while extensive semantic adaptation is ideally left to the user during the revision stage of the problem-solving cycle.

Knowledge-intensive approaches to reuse in TCBR include attributes/information extraction (Cowie & Lehnert 1996, Brüninghaus & Ashley 2001), named entities identification (Lamontagne & Lapalme 2004) and generation of ontologies (Badra, Bendaoud, Bentebibel, Champin, Cojan, Cordier, Despres, J-Daubias, Lieber, Meilender, Mille, Nauer, Napoli & Toussaint 2008), formal concept trees (Asiimwe, Craw, Wiratunga & Taylor 2007b) and taxonomies (Recio-García & Wiratunga 2010) using a combination of shallow and deep NLP tools. Deep NLP tools include parsers for building language trees (De Marneffe et al. 2006) and their manipulation for reasoning with text. These TCBR techniques typically require active participation of, or extensive assistance from, domain experts for knowledge elicitation and are therefore domain-dependent and not easily transferable to other domains. Although adaptation of a retrieved textual solution is expected to be less difficult using this approach, a human user or natural language generation system

(Reiter & Dale 1995) might be needed to transform the formalism (structured attributes or ontology) back into text so it is in the same format as existing cases.

Lenz (1998) proposed a seven layer hierarchical representation of text which incorporates characteristics from each of the knowledge-light and knowledge-intensive approaches and can be seen as mid-way between them. Each layer contains knowledge about one of keyword, phrase, thesaurus, glossary, attribute value, domain structure and information extraction. The keyword layer is at the bottom of the representation with least complexity and minimal semantics while the information extraction layer at the top is most complex in structure and contains the semantic contents of the text. The bottom three layers (keyword, phrase and thesaurus) are often used in the knowledge-light approaches to text representation. A knowledge-intensive approach will emphasize use of the top four layers (glossary, attribute value, domain structure and information extraction). However, the minimum requirement is that the values for one of the layers are available across all cases including the query. The idea is that with more layers being used, the more semantic the representation, therefore the better the retrieval process.

This research employs a knowledge-light approach to represent textual cases as it eases the transferability of our text reuse techniques across domains. Domain knowledge is also incorporated where possible to improve the reasoning process. The results we obtained using this approach have been very good as retrieved cases are sufficiently similar to queries; discussion of our results are given in Chapters 6 and 7. However, the implication for text reuse is that proposed solutions are sometimes incomplete. Also, suggestions for adaptation of a retrieved solution text might be limited to the granularity of decomposition (i.e. word, phrase or sentence level).

2.1.3 Indexing Architectures

When the casebase is relatively large, retrieval efficiency can be adversely affected, since a new problem is compared to all cases to determine the best match. This problem is exacerbated further when there is high dimensionality of attributes. Several indexing architectures have therefore been employed to improve retrieval efficiency and this

is typically carried out as a pre-processing step prior to the start of the problem solving process. Architectures that have previously been used for indexing casebases include decision/KD trees (Wess, dieter Althoff & Derwand 1994, Selvamani & Khemani 2005), shared-feature/discrimination networks (Kolodner 1993), neural nets (Reategui, Campbell & Leaob 1996) and case retrieval networks (Lenz & Burkhard 1996a). There is also a strong relationship between indexing architecture and case representation used for reasoning. For example, case (problem) attributes with poor information gain values are automatically deemed unimportant when indexing with (pruned) decision trees. Therefore, it is important to use an architecture that is able to encode the case representation required for reasoning.

For TCBR, indexing is crucial since textual comparison typically involves decomposing the texts into smaller units (e.g. bag of keywords) or mapping it into a more structured form (e.g. ontology). Therefore, indexing architectures are used in TCBR even when the casebase is not too large since a textual case can decompose into several smaller units and this needs to be managed efficiently. Although, most CBR indexing structures can be adapted for TCBR, the case retrieval network (CRN) is commonly used as the knowledge-light approach to text representation maps directly onto it. The CRN architecture and its extensions such as Fast CRN (Chakraborti et al. 2006) have been found to be very efficient and flexible for retrieval (Chakraborti 2007) providing easy integration of most textual similarity measures such as Cosine similarity, Euclidean distance, Hamming distance and Edit distance (Levenshtein distance). One of the contributions of this thesis is the extension of CRN for reuse of textual solutions. The extended architecture called Case Retrieval Reuse Net (CR2N) is introduced in Chapter 3.

2.2 Reasoning with Problem-Solving Experiences

CBR is a lazy approach to solving problems. It does not seek to build universal models (like model-based learners) or encode generic domain rules (like rule-based expert systems) but attempts to solve problems on a case by case basis thereby narrowing the context and reusing similar experiences from local neighbourhoods. However, it has been tradition-

ally applied to domains with structured attributes whose values are finite (e.g. numbers, symbols or complex design/plan objects). The structured nature of attributes in these traditional CBR domains eases the reasoning process, because formal knowledge representations from other established research fields such as databases and data mining are applicable. Nevertheless some effort is required during knowledge acquisition to identify useful case attributes, prioritize their importance and utilize them effectively for retrieving similar cases. However, comparison of new problems to existing cases is straight-forward since each structured attribute carries some semantic information.

Increasingly, experiential knowledge is being captured in semi-structured or unstructured form as reports, emails, frequently asked questions/answers and web blogs. Traditional CBR techniques cannot be directly applied to such textual experiential data because of the challenges associated with reasoning with text (details in Section 1.1.3). Accordingly, the aim of TCBR research is to reason with textual cases to solve new problems without necessarily transforming them into traditional structured format. Textual cases are compared during retrieval using techniques similar to those in other related research fields involving text analysis such as Natural Language Processing, Text Mining and Information Retrieval. In this section, we examine and analyse techniques that have been proposed and employed for retrieval and subsequent reuse of both structured and textual cases.

2.2.1 Similarity and Retrieval

The notion of similarity is at the heart of the CBR process since one of the basic assumptions is that similar problems have similar solutions. Similarity must be defined and encoded with appropriate methods/metrics for effective retrieval. There are several similarity metrics that have been proposed for CBR retrieval depending on the domain and case representation. Some of these metrics are standard mathematical functions such as Euclidean, Cosine and Manhattan (Wilson & Martinez 1997) while others have been ad-hoc and defined based on requirements of the domain. The representation of the problem component of a case with respect to the attribute data type greatly influences the simi-

larity definition. For example, while Euclidean distance is a suitable similarity metric for attribute-value representations, it cannot be directly applied to complex representations like design/plan objects (Smyth & Cunningham 1993, Muñoz Avila & Cox 2008) where the sequence of components (attributes) is important.

Similarity between textual cases can also be measured using these mathematical functions when using a knowledge light approach with the textual attribute represented as a vector of terms or concepts. A concept is formed from several terms as a higher level of semantic representation. This improves retrieval because it takes into account seemingly different terms that are used in the same context. TCBR techniques need to take relationships such as synonyms, hypernyms (super-ordinate or more generic word) and hyponyms (subordinate or more specific word) between the vocabulary terms into account. These relationships are usually incorporated into the retrieval mechanism to ensure that sufficiently similar cases are not excluded due to differences in the choice of terms. Domain glossaries, thesaurus or general-purpose ontologies like WordNet (Fellbaum 1998) have previously been used to elicit these term-term relations (Kunze & Hubner 1998, Lenz, Hubner & Kunze 1998, Ahmed, Begum, Funk, Xiong & Schéele 2008). Some retrieval structures such as Case Retrieval Network (CRN) also enable the encoding of these term-term relations. Another approach to ensure effective retrieval of similar textual cases while minimizing the effect of differences in choice of terms is to normalize the casebase vocabulary (Asiimwe, Craw, Wiratunga & Taylor 2007a). Normalisation in this context means choosing a single term to represent a set of synonymous terms and replacing them with this representative term in all cases.

Retrieval in CBR is typically based on comparison of a new problem (query) to the problem component of previous cases in the casebase. However, solutions of cases most similar to the query are not always the easiest to adapt during reuse. Therefore, techniques have been proposed for adaptation-guided retrieval (Smyth & Keane 1994, Smyth & Keane 1996, Díaz-Agudo, Gervás & González-Calero 2003, Nouaouria & Boukadoum 2010) to improve the adaptability of retrieved solutions. Here, the main idea is to elicit an adaptability criterion from the relationship between problem and solution components of existing

cases using techniques such as Formal Concept Analysis (Wille 1992) or other statistical models (Brown et al. 1993, Manning & Schütze 1999). This criterion encodes information about the ease of transforming one solution to another relative to the similarity of their problems. The adaptability criterion is incorporated into the similarity measure to influence the retrieval process. Similar cases with easily adaptable solution are then ranked higher than others whose solutions are less adaptable.

Lamontagne, Langlais & Lapalme (2003) applied an adaptation guided approach for the retrieval of cases where both problem and solution are textual. Here, the idea is to allow retrieval similarity to be influenced by expected solution similarity. A shadow solution to a new problem is generated using problem-solution associations of keywords by exploiting word co-occurrence and probabilistic generative models. The final retrieval similarity is then an aggregation of the problem and solution similarities.

A knowledge light approach to text representation and retrieval is used in our research. We compare the textual components of our cases using cosine similarity metric because it does not penalize the absence of terms and minimises the effect of the text length with its normalization. Our retrieval mechanism used the CRN architecture since it allows for easy integration of term-term relation apart from increased efficiency. We also propose a method for reuse guided retrieval of cases with textual solution which unlike the one propose by Lamontagne et al. (2003) can be used with both structured and textual problems (details are given in Section 3.4).

2.2.2 Reuse and Adaptation of Structured Cases

Reuse is central to the CBR problem-solving process, because the primary aim is to reuse previous solutions to solve a new similar problem. Adaptation is an integral part of reuse to deal with modification to a retrieved solution to address differences between the new and retrieved cases. In this thesis, the term 'reuse' refers to any of the reuse tasks, that is verbatim copy or adaptation as explained in Section 1.1.1. In broad terms, there are two main categories of reuse: generative and non-generative. Figure 2.1 shows the categories of reuse, their subsets and related citations. Each category carries a distinctive theme

that differentiates it from its alternative. When a reuse technique combines themes across different categories, we label it with only one theme which is most central for simplicity. An important distinction between the two main categories is in the form of knowledge used from retrieved cases. Non-generative reuse techniques are more common and make use of the actual contents in the retrieved solutions which they attempt to modify during reuse. On the other hand, generative reuse techniques attempt to produce a new solution by retracing the method used to obtain the retrieved solution in the context of the new problem. Generative reuse, also known as replay or constructive reuse, is less commonly used primarily due to the overheads attached to storing and reasoning with problem-solution traces. One of the few works in generative CBR reuse involves a search-based approach proposed for configuration tasks (Plaza & Arcos 2002). Solutions in this domain consist of a complex structure of elements which are captured as *states*; a domain-specific representation of a partially specified solution. A solution to a given problem is therefore generated by searching the space of solutions guided by the retrieved solution. Wilke and Bergmann (1998) also described how this approach might be applied to obtain a suitable configuration for a PC having specified its utility for attributes such as games and music.

There are three approaches to non-generative reuse shown in Figure 2.1: verbatim, transformational and compositional. Verbatim reuse includes all CBR systems where no form of adaptation is carried out after retrieval; these are otherwise called retrieve-only systems. In such systems, adaptation is completely left to the human users and most of the earliest CBR systems such as BATTLE PLANNER (Goodman 1989), ARCHIE-2 (Domeshek & Kolodner 1991) and ASK (Ferguson et al. 1992) fall in this category. Transformational and compositional reuse both carry out some form of adaptation but their major difference lies in the manner in which they use their retrieved solutions. The transformational approach to reuse typically selects the solution from the retrieved best case and attempts to adapt its contents. On the other hand, compositional approach attempts to combine sub-solutions from several similar cases to produce a new solution during adaptation. Nevertheless, the transformational approach could adapt its best solution using the contents from other similar cases. In this scenario, the line between transformational and

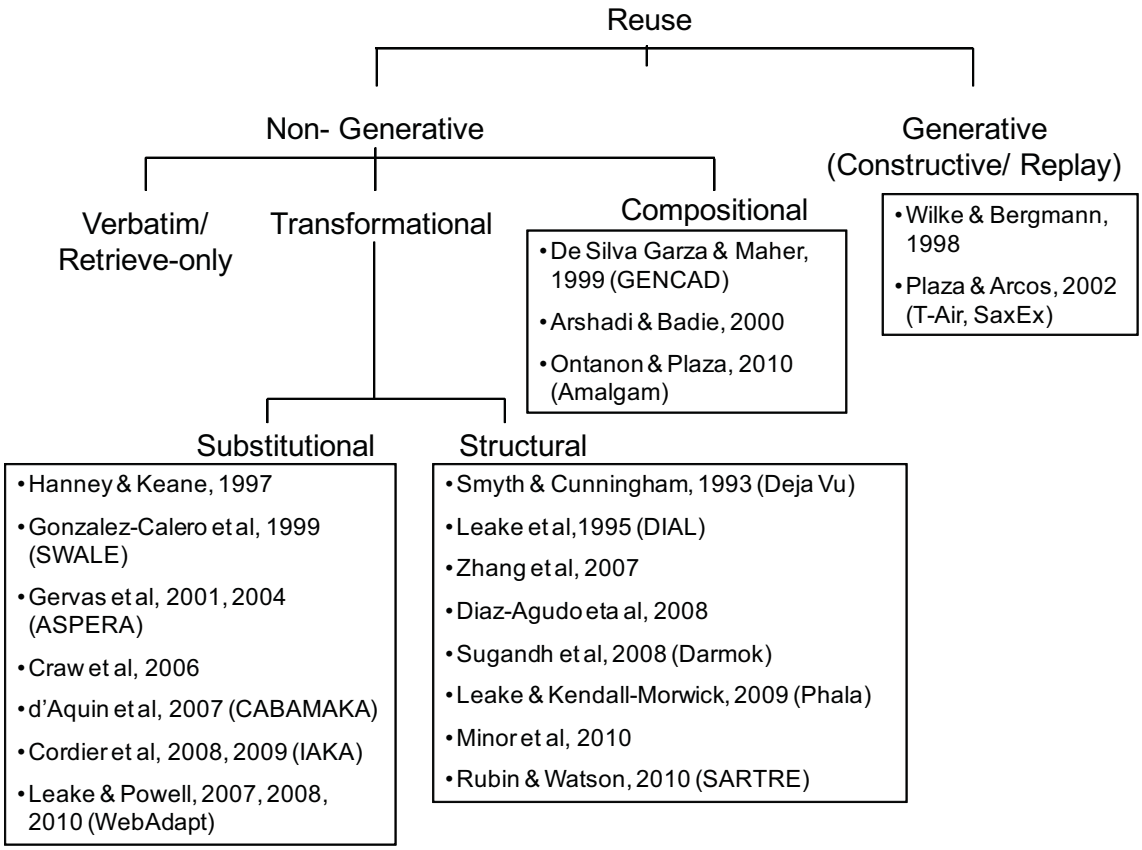


Figure 2.1: Types of Reuse in CBR (Adapted from Wilke and Bergmann, 1998)

compositional reuse becomes blurred. A compositional approach to reuse was proposed for tutoring library system (Arshadi & Badie 2000) where chapters from different textbooks are combined into a new book in response to a user request. The user request consist of a set of keywords and structured information like searching area, user's current knowledge level and desired status of knowledge while the solutions consist of book chapters, authors and their year of publication. Similar approaches to compositional reuse have also been proposed for use in architectural designs (De Silva Garza & Maher 1999) and formalised in description logic (Ontanón & Plaza 2010).

The two categories of transformational reuse (substitutional and structural) are quite similar and encompass majority of CBR adaptation techniques. The main distinction has to do with whether there is a change in the overall structure of the retrieved solution or not during adaptation. Substitutional adaptation is commonly used with attribute-value case representations where adaptation only involves modifying the value of attributes. In gen-

eral, CBR adaptation techniques where attribute values are replaced with optimal, better or improved values will fall under this category of substitutional adaptation (Hanney & Keane 1997, González-Calero, Gómez-Albarran & Díaz-Agudo 1999, Gervás 2001, Gervás, Díaz-Agudo, Peinado & Hervás 2004, Craw et al. 2006, D'Aquin, Badra, Lafrogne, Lieber, Napoli & Szathmary 2007, Leake & Powell 2007, Leake & Powell 2008, Cordier, Fuchs, de Carvalho, Lieber & Mille 2008, Badra et al. 2009, Leake & Powell 2010). Structural adaptation approach on the other hand involves re-organization of the solution structure through deletion, insertion and/or substitution operations. It is more suitable for case representations in domains such as planning and design where case attributes are more complex and the sequence of attributes is important for meaningful problem-solving (Smyth & Cunningham 1993, Muñoz Avila & Cox 2008, Sánchez-Ruiz, Gómez-Martín, Díaz-Agudo & González-Calero 2008, Sugandh et al. 2008, Lee-Urban & Munoz-Avila 2009, Minor, Bergmann, Grg & Walter 2010, Rubin & Watson 2010). However, approaches that can be categorised under structural adaptation has also been applied in several other domains (Leake et al. 1995, Zhang, Louvieris & Petrou 2007, Díaz-Agudo, Plaza, Recio-García & Arcos 2008, Leake & Kendall-Morwick 2009).

Previous research in transformational reuse employed general and manually abstracted rules to adapt a past solution given a new problem (Leake et al. 1995). These rules are mined from the case base using differences between previous problems to form antecedent and corresponding solution differences to form adaptation actions as consequence (Hanney & Keane 1997). The use of an adaptation case base has also been explored in transforming a previous solution when solving a new problem. Such adaptation case base is used to store a trace of the steps involved when adapting new cases (Leake et al. 1995) or learnt introspectively from the original case base using previous problem differences and corresponding solution differences as an adaptation case (Craw et al. 2006). Self Organising Maps and Neural networks have also been used to learn relations between problem and solution differences in a high dimensional solution space (Zhang et al. 2007). Other techniques that have been proposed for CBR adaptation include genetic algorithm (De Silva Garza & Maher 1999), constraint satisfaction (Purvis & Pu 1995) and revision

theory (Lieber 2007).

Several CBR techniques have been proposed for case adaptation during reuse as discussed in the preceding paragraphs. However, these adaptation/reuse techniques were applied in domains with structured cases and are not directly applicable to textual cases especially when the solution is also textual. The next section discusses relevant works on reuse of textual cases though very few of them incorporate any form of adaptation.

2.2.3 Reusing Textual Cases

A textual case has at least one of its attributes (either problem, solution or both components) in free text form. As discussed at the beginning of Section 2.2, retrieval deals mainly with the problem component of cases to determine their similarity to a new problem while reuse deals with the solution components. Text reuse is applicable when the solution is in free text form. The categorization in Figure 2.1 also applies to text reuse techniques; this is reproduced in Figure 2.2 but now with citations of relevant TCBR literatures. Note that the work by Lamontagne & Lapalme (2003,2004) is cited in two different categories since it proposes substitutional and structural text reuse techniques. A key difference between the reuse categories shown for traditional and textual CBR is the unavailability of any current TCBR compositional reuse technique. This requires the combination of textual sub-solutions from several similar cases into a single meaningful solution in response to a query. Such combination of text from several cases is difficult to automate without loss of coherence and overall contextual meaning. The closest work we found related to compositional text reuse is the tutoring library system (Arshadi & Badie 2000) discussed earlier in Section 2.2.2. Here, chapters from different textbooks are combined into a new book in response to a user request. However, we regard this as structured CBR reuse because it utilises a pre-defined topic label (symbolic attribute) attached to each book chapter during reuse rather than its textual contents.

The substitutional form of transformational reuse has been applied extensively to textual cases especially when minimal adaptation is required. This involves the identification of specific terms in a retrieved textual solution and proposing suitable modifications

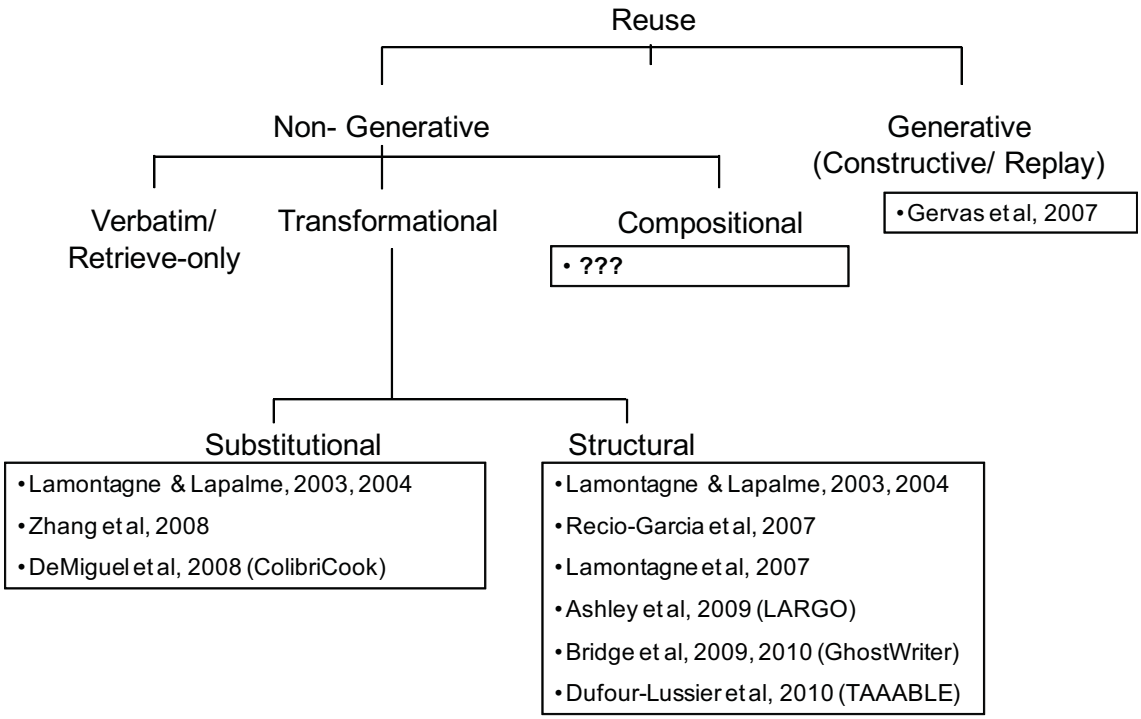


Figure 2.2: Categories of Text Reuse approaches

due to observed differences between the query and retrieved problem. This approach was used for a substitution based adaptation in some TCBR applications that deal with modification of ingredients to recommend recipes that satisfy a user query (Adeyanju et al. 2008, DeMiguel, Plaza & Díaz-Agudo 2008, Zhang et al. 2008). It was also used to makes suggestions for named entities substitution in a TCBR application for automated email response (Lamontagne & Lapalme 2003, Lamontagne & Lapalme 2004). Substitutional form of text reuse has also been used in some machine translation techniques. For instance, Example Based Machine Translation (EBMT) (Nagao 1984, Sumita, Lida & Kohyama 1990, Brown 1996, Zhang, Brown & Frederking 2001) retrieves similar textual contents that have previously been translated given some text to translate into another language. A dictionary is then used to substitute mismatched words or phrases between the problem and retrieved text in the newly generated translation.

The structural form of transformational reuse in the context of TCBR involves proposing suggestions for adaptation to a textual solution that goes beyond substitution. In other words, there might be suggestions to delete and/or insert terms into specific sections of

a retrieved textual solution without necessarily replacing other sections thereby changing the overall structure of the solution. More sophisticated strategies might also consider the impact of these structural changes to other solution parts. A form of structural text reuse was proposed for report writing applied to the air travel incidents investigation (Recio-García et al. 2007); we denote this technique as jCOLIBRI-Reuse for ease of comparison to others. Here, incident reports have common headings for each section of one or more paragraphs. Text reuse is facilitated by presenting clusters of similar text from other documents for each section while a user is modifying the best match case's solution. This enables the manual reuse of text from several documents technique thereby altering the overall structure of each section. Though an intuitive form of text reuse, this approach is restrictive since it cannot be used in the absence of common sectional headings across the casebase. Lamontagne and Lapalme (2004) demonstrated Case Grouping (CG), a form of structural text reuse on a semi-automated email response application. This involves the reuse of previous email messages to synthesize new responses to incoming requests. Sentences in a retrieved solution are labelled as reusable or not depending on whether there is sufficient evidence that previous similar problems contain such sentences. Reuse evidence for each sentence is computed by comparing the centroid of two clusters (support and reject) to the query. Only cases that have a similar sentence in their solution belong to the support cluster while the reject cluster contains all other cases. Although the use of similarity knowledge to guide text reuse is novel, CG uses the entire casebase to determine if a sentence can be reused. This will be computationally expensive and seems counter-intuitive since cases with no similarity to the query nor retrieved solution will contribute to reuse evidence. However, such an approach is likely to guide reuse towards generic solutions. CG is also more generic than jCOLIBRI-Reuse, since its clusters are not restricted to domains with a common template structure. The GhostWriter systems (Bridge & Waugh 2009, Healy & Bridge 2010) aid text reuse by suggesting features and values or phrases during authoring of a product description for trading (Bridge & Waugh 2009) or its review after purchase (Healy & Bridge 2010). Features, feature values or noun phrase suggestions are iteratively extracted from top previous similar cases using

manually-defined regular expressions or shallow NLP techniques. The list of suggestions is limited using a set of criteria such as ensuring their absence in solution being authored at that point. Another criteria ensures that the length of the phrases are not too short. This approach is similar to jCOLIBRI-Reuse in the sense that suggestions for text reuse change continuously while a user is authoring a new solution. However, users start with an empty solution in the GhostWriter systems rather than a retrieved solution text. Another commonality between GhostWriter and jCOLIBRI-Reuse is that the problem and solution share a common vocabulary. This is unlike CG where the problem and solution vocabularies are separate though they might share some common terms. Other forms of structural text reuse have involved the use of translation models for word alignment in incident reports (Lamontagne, Bentebibel, Miry & Despres 2007), diagrammatic representation for legal texts (Ashley, Lynch, Pinkwart & Aleven 2009) and formal concept analysis for adapting recipes (Dufour-Lussier et al. 2010).

Verbatim Reuse is the most common form of text reuse and previous TCBR systems that fall under these category includes ExperienceBook (Kunze & Hubner 1998), FallQ (Lenz & Burkhard 1997, Lenz, Hubner & Kunze 1998), FAQ Finder (Burke et al. 1997), CATO (Brüninghaus & Ashley 1998b), PRUDENTIA (Weber et al. 1998), DRAMA (Wilson 2000), InRet (Wilson, Carthy, Abbey, Sheppard, Wang, Dunnion & Drummond 2003) and SMILE (Brüninghaus & Ashley 2005). With these systems, the indexing and retrieval mechanism encode most of the semantics in the textual cases, so that retrieved cases are semantically similar, but give no assistance on how to adapt the retrieved solution. To the best of our knowledge, there is only one publication (Gervás, Hervás & Recio-García 2007) that can be viewed as constructive or generative TCBR reuse. Here, the main idea is to use Natural Language Generation (NLG) during TCBR reuse. This is done by first transforming textual cases into a structured representation, reuse and adapt using any of the traditional CBR techniques and finally apply NLG to convert the adapted solution (in structured form) back into natural language. However, no experimental evidence was reported to justify the success of this approach. The approach is very knowledge intensive with limited applicability because not all textual domains (e.g.

email management) can be transformed into the semantic structured representation that is utilized by traditional CBR techniques. Also, such structured representations must meet NLG requirements for text generation. For example, a corpus containing sample structured data and human-authored textual equivalents must be available for NLG domain analysis (Reiter & Dale 1995).

Our research makes contributions that fall under different categories of text reuse. We propose a structural text reuse technique similar to CG (Lamontagne & Lapalme 2004) which identifies portions of a retrieved textual solution that need adaptation (details in Chapter 3). The technique is formalised with a generic architecture that can cater for different text granularity levels. Novel transformational and compositional text reuse methods whose aims are similar to those used in the GhostWriter systems (Bridge & Waugh 2009, Healy & Bridge 2010) are also proposed for text authoring (see Chapter 4). These techniques are meant to fill some of the gaps we observed in TCBR literature.

2.3 Evaluation techniques

Evaluation is a critical aspect in the design and analysis of any new technique since it helps to demonstrate the technique's effectiveness and limitations. New techniques can also be compared with existing ones used for identical tasks based on their evaluation results. This kind of evaluation which we call 'performance evaluation' is applicable to most research disciplines (TCBR inclusive). For TCBR, performance evaluation involves assessing proposed solutions for their suitability to solve a given set of problems. Traditional CBR evaluation metrics such as accuracy are used when the solution is structured and adapts information retrieval evaluation metrics for textual solutions.

Another form of evaluation peculiar to CBR, especially TCBR, is the measurement of the casebase complexity. Here, an experiential corpus is assessed for its conformity to the similarity assumption which is core to the CBR paradigm. In other words, casebase complexity metrics show how well similar problems have similar solutions in a given corpus. However, the complexity of a casebase is dependent on the case representation and similarity measure. Therefore, these metrics can be used to assess different configurations to

determine the most suitable for a domain without necessarily designing a complete CBR system. In this section, we examine these two forms of evaluation (Casebase complexity and performance evaluation) and critically analyse previous work related to TCBR.

2.3.1 Casebase complexity evaluation

The basic assumption in TCBR as with CBR is that similar problems have similar solutions. Therefore TCBR is not suited to domains in which this assumption does not hold. However, the definition of similarity is highly dependent on the representation which implicitly captures the focus and context of the problem-solving scenario. For example, two pieces of texts that mean the same thing but expressed differently might be incorrectly judged to be dissimilar if the system does not capture variability in vocabulary in its representation (e.g. using text normalisation). Different configurations (representations and similarity metrics) might need to be compared to determine which best captures the information needed for reasoning while designing a TCBR system. A qualitative evaluation although ideal is clearly impractical due to cost and time implications. An alternative is to develop casebase complexity metrics that measure the similarity of solutions in similar problem neighbourhoods. In other words, how well does a cluster of similar problems align to the cluster of their solutions? High values of casebase complexity indicate that the dataset (or domain) is well suited for the CBR paradigm. The hypothesis is that casebases with high complexity values will give better performance evaluation than those with low casebase complexity.

Lamontagne (2006) proposed case cohesion as a casebase complexity measure to aid selection of an appropriate configuration for a TCBR system. Case cohesion estimates how well the problem in a case has similar solutions using its neighbourhoods in the problem and solution spaces. A similar measure called case alignment evaluates the competence of different system configurations (Massie et al. 2007). The major difference between case cohesion and case alignment is in the level of granularity; cohesion uses the number of neighbours common to the problem and solution spaces but alignment uses similarity values of such neighbours. In contrast, global alignment quantifies the overall alignment of all cases

by analysing similarity between problem and solution clusters to determine casebase complexity for alternate configurations (Mudambi-Ananthasayanam, Wiratunga, Chakraborti, Massie & Khemani 2008, Mudambi-Ananthasayanam, Chakraborti & Khemani 2009).

These three casebase complexity metrics (i.e. case cohesion, case alignment and global alignment) were applied to our experimental datasets to determine the suitability of TCBR. The metrics determine their values by approximating the implicit alignment between problems and solutions using case similarity. This is similar to the notion of alignment used in this thesis for determining what to reuse in a textual solution (details in Chapter 3). However, we also propose another method for explicitly aligning problem attributes to some portions of the solution text as discussed in Chapter 4.

2.3.2 Performance evaluation

The need to evaluate natural language texts is common to several research areas in computer science. These areas include (but are not limited to) Information Retrieval (IR) (Lenz 1998b, Baeza-Yates & Ribeiro-Neto 1999), TCBR (Brüninghaus & Ashley 2005, Weber et al. 2006), Natural Language Generation (Sripada, Reiter & Hawizy 2005, Belz 2005) and Machine Translation (White & Connell 1994, Hovy 1999). Generally, we can divide text evaluation techniques into two broad categories: qualitative and quantitative.

Qualitative techniques involve user trials (experts and non-experts) to determine the quality of some text produced by a machine. The resulting user feedback is then aggregated using statistical methods to judge the average quality of such texts. The major disadvantages are that these techniques are very expensive especially when expert knowledge is required and identical results are not reproducible as human judgement is subjective. Nevertheless, qualitative techniques have been used for evaluation across many application domains involving natural language processing and generation. For example, Sripada et al. (2005), Belz & Reiter (2006), Zhang et al. (2008), DeMiguel et al. (2008) and Hanft et al. (2008) all report experimental results using qualitative evaluation techniques.

On the other hand, quantitative techniques involve the comparison of machine texts to one or more gold standards written by humans (usually experts). Here quality of the

method is gauged according to similarity at the syntactic or semantic level. Quantitative techniques are typically less reliable as most of them depend on finding matching string patterns between the machine-produced texts and gold standards. However, such techniques can be automated, are less expensive and are easily reproducible. This also allows for easy comparison across several algorithms that are designed for the same purpose.

Precision and Recall are two basic quantitative metrics (Brüninghaus & Ashley 1998a, Baeza-Yates & Ribeiro-Neto 1999, Lamontagne & Lapalme 2004) widely used for text evaluation across several disciplines especially IR and TCBR. The basic idea is to regard a piece of text as a bag of (key)words and to count common words between the machine and human texts. Proportions of these common words to the machine and human texts give a metric of precision and recall respectively. A major drawback is that the sequence of words in a piece of text is ignored and this can adversely affect the grammatical and semantic meaning. In other words, a machine text with high precision and recall might not necessarily be grammatically and/or semantically correct.

The edit distance, also called Levenshtein distance (Levenshtein 1966), has also been used for text evaluation; for example in (Belz 2005). This technique takes the sequence of words into account and is calculated in the simplest form as the number of delete, insert and substitute operations required to change the machine text into its human solution equivalent. Typically, different costs are associated with each of these edit operations. Nevertheless, the edit distance can be misleading because the same piece of text can be written in several ways without loss of meaning. In particular, machine texts with a longer length will be unfavourably penalized by this technique.

We used the standard TCBR evaluation metrics of precision and recall in most of our empirical experimental evaluations. However, we found out that this was at times inadequate and misleading as these metrics do not correlate well with human qualitative judgements. Therefore, one of the contributions of this thesis is the application of some machine translation (MT) evaluation metrics to TCBR. These evaluation metrics have been tested to correlate highly with human judgements and are used widely for empirical evaluation not just in MT but in other fields such as natural language generation and text

summarization. Further details on automated TCBR evaluation and how it is used in this thesis can be found in Chapters 5, 6 and 7.

2.4 Datasets

This thesis focuses on developing techniques for reusing textual solutions irrespective of the nature of their problem attributes, that is, problems can be structured or textual. Comparative studies were conducted on four datasets: weather forecast revision, hospital health & safety incident reporting, hotel review authoring and academic coursework feedback authoring. Apart from meeting the CBR requirements of being experiential and representative of their domains, the solution is entirely textual in these datasets and therefore suitable for our text reuse research.

2.4.1 Weather forecast revision corpus

Weather forecasts are typically generated from numerical weather prediction models and are written in textual form for use by non-weather experts. Generating weather forecast texts is an established application for Natural Language Generation (NLG) research (Reiter & Dale 1995) which involves producing understandable texts from non-linguistic data. However, forecast texts generated by NLG systems are inspected and sometimes edited by weather experts to ensure they are completely error-free before being passed on to users. The aim of applying CBR in this domain is therefore to reuse the editing experiences of the weather experts to automate the revision of forecast texts generated by NLG systems. The assumption is that similar forecast texts generated by an NLG system will be edited in a similar way by the human experts. In other words, similar machine-generated forecast texts will be similarly edited by human experts. Therefore enabling reuse of the editing experiences of the human experts by a TCBR system to automatically edit NLG system generated forecasts.

We obtained a weather forecast revision corpus from the developers of SUMTIME-MOUSAM (SM) (Sripada, Reiter, Hunter & Yu 2002, Sripada, Reiter & Davy 2003), an NLG system that generates weather forecast texts using numerical weather prediction

models. The system was developed by a research group at the University of Aberdeen in collaboration with a local weather forecast company. This corpus consists mainly of weather texts from SM and their edited equivalents by Human experts. Weather parameters generated as forecast texts include wind, wave, outlook, visibility, temperature and cloud predictions edited by nine domain experts. Each forecast record has a timeID, section which represents a particular day in a 5-day forecast, weather parameter, unedited text, edited text as well as the name of the domain expert that edited the text.

The portion of the corpus with the wind weather parameter was solely used for our experimental evaluation as it was determined as least noisy, most accurate and reliable after consultation with NLG experts involved in the development of SM. Forecast texts from other weather parameters were highly unreliable because the human experts made significant changes to the original data used for text generation by SM. These changes led to inequivalent texts as the generated and edited texts seem to come from different data. Each NLG system generated text (Unedited Text) is used as problem and its revised form by domain experts (Edited text) as solution and together form a single case in our case base. We extracted 5,011 wind cases after removing duplicates from the original 14,690 records. However, the number of wind states in each unedited forecast text ranged from one to seven across these extracted cases. A basic wind state consists of the wind speed and direction. The casebase used for our experimental evaluation was limited to 2,414 cases with only two wind states in the unedited text. This was to avoid retrieval failures in cross-validation empirical evaluation because weather experts suggested that forecast texts are only similar if they have the same number of wind states.

We observed the following peculiar properties in the datasets extracted from SM weather corpus.

- The problem text is more similar to its solution text in a single case than to any problem text from other cases. This means that the problem & solution vocabularies are identical unless forecasters introduce new terminology.

- The indexing vocabulary is relatively small; there are 71/140 keywords in problem/solution vocabulary respectively.

- The problem (Unedited text) is consistent because it is generated by an NLG system with abstracted rules. But the solution (Edited text) is less consistent due to stylistic variations among experts and may contain typing errors (e.g. "middnight" instead of "midnight", "acking" rather than "backing", "deceasing" instead of "decreasing").

The first two characteristics are unusual and have not been reported nor observed in datasets previously used in TCBR research; such as the European Space Agency incident reports (Massie et al. 2007), medical health and safety incident reports (Mudambi-Ananthasayanam et al. 2008), email datasets (Lamontagne & Lapalme 2003, Delany et al. 2005, Orecchioni et al. 2008), air traffic incidents investigation reports (Orecchioni et al. 2007, Lamontagne et al. 2007), legal jurisprudence reports (Brüninghaus & Ashley 1998b, Brüninghaus & Ashley 2001, Brüninghaus & Ashley 2005) and product review authoring datasets (Bridge & Waugh 2009, Healy & Bridge 2010). However, we expect that similar edit operations are applicable on generated texts that are similar thereby making the TCBR paradigm suitable for the domain.

2.4.2 Medical health and safety incident reports

Health care personnel are obliged to record any health and safety incidents related to in-patients (on admission at the hospital). These are naturally maintained as incident reports. This corpus consists of such incident reports from hospitals in the Aberdeen area of Scotland provided by National Health Service Grampian (NHS-G). Staff on duty complete a form with specified attributes describing each incident. These attributes include id, care stage code, adverse event code, detail code, incident description, action taken and lessons learnt among others. The incident description, action taken and lessons learnt attributes are textual while the three codes (care stage, adverse event and detail) have symbolic values. Each symbolic attribute has a limited number of values that covers the documentation of all possible incidents. There are 17, 149 and 334 possible values for care stage, detail and adverse event attributes respectively. NHS-G also provides guidance notes describing when each possible code value should be used. We observed that the occurrence of similar incidents required similar actions to be applied and that similar

incidents do re-occur. Therefore the CBR paradigm and techniques are suitable for this domain. The aim is therefore to design and develop a TCBR system that proposes actions to be taken when a new incident occurs using previous reports of similar incidents. This can assist health personnels, especially new recruits, to resolve and/or document incidents with access to previous similar experiences. Therefore, our case structure consists of the incident description as problem component and the actions taken as solution.

A total of 4011 reports are originally available but this is reduced to 3024 cases after some data cleaning such as removal of duplicate reports in which the textual contents of the incident description and action taken are identical. We also removed reports with blank entries in their action taken attribute, because they are incomplete and will not be useful for empirical experiments. 983 of the 3024 reports belong to a single care stage code (ACCID) which identifies a group of records that are related to personal injuries and incidents during treatment or procedures. This category was chosen because it had the largest number of reports compared to the other sixteen care stage codes. The textual vocabulary of the reports in this category also contained few domain specific terms thereby enabling the exploitation of general-purpose knowledge sources during reasoning. We further discovered that size of the textual content for both incident description and action taken varied greatly, ranging between one to eight sentences. This is due to stylistic differences across report authors and the variation in verbosity. A casebase of 362 cases was therefore extracted for our experimental evaluation where there were two sentences or less in the problem and solution textual attributes to reduce the vocabulary variability during textual reuse. Another reason for excluding cases whose problem or solution text have more than two sentences is to ensure consistency with our weather dataset where we worked with only two wind phrases. We also intend to test our reuse algorithms at keyword granularity which is only reasonable when the solution texts are small. However, it is expected that the algorithms will scale up to bigger levels of granularity (e.g. phrases and sentences) albeit with very little modification.

Unlike the weather forecast revision domain, medical health and safety incident reporting is a typical TCBR application where problem and solution vocabulary share little

in common and indexing vocabulary is relatively large. For example, there are 567/307 keywords in problem/solution vocabulary of the 362 cases extracted casebase; in contrast to 71/140 keywords in 2414 cases for the weather forecast revision domain. Both problem and solution texts may also contain typing errors since they are manually recorded by humans.

2.4.3 Reviews of Hotels

User generated experiential content is readily available on the world wide web in the form of blogs, forum posts, reviews and other social applications. This provides an opportunity to reuse these experiences for similar web related tasks such as search and browse, review generation and other forms of problem-solving (Plaza 2008, O'Mahony & Smyth 2009, Healy & Bridge 2010). However, reuse will only make sense if there are several experiences authored about similar/identical objects (or problems). Hotel reviews are particularly useful in this regard, because several reviews are available for the same, or indeed, similar hotels. Apart from being freely available, they are also less likely to be biased since they are generally authored by persons who have no commercial interest in the hotels. Each review typically has attributes involving a qualitative scale (e.g. 1-5) and accompanying text. Hotel reviews are generally suitable for the text reuse focus of our TCBR research as the assumption is that authors with similar ratings will use similar explanatory feedback text. However, such review texts are prone to grammatical errors since authors rarely use spell checkers. They also contain a lot of verbose details that might not be directly related to hotels (since unedited reviews are uploaded).

We downloaded several reviews from a hotel recommender website[1] where each review is written by an author who visited a hotel and presents his/her opinion. The 13 attributes shown in Table 2.1 were originally extracted for each review; however, some of these attributes were absent in some reviews. In total, $39,870$ reviews were extracted across $6,564$ hotels in 104 different countries (or states in USA). Initial analysis of the corpus showed that the downloaded corpus contained a small number of reviews (< 50) per hotel or author. The overall rating of a hotel is also an average of authors' ratings and not

[1] www.tripadvisor.co.uk

those given by regulators such as ISO (International Organization for Standardization). It therefore seemed more intuitive to reuse similar reviews across all hotels instead of within a single hotel or a single author. 641 cases across different hotels were extracted for our experimental evaluation.

Table 2.1: Complete list of possible attributes extracted for each hotel review

1. Hotel name	2. Hotel town	3. Hotel country (or US state)
4. Overall rating	5. Review Title	6. Author **rating** of the hotel
7. Author ID	8. Author location	9. Trip type (solo, couple ...)
10. **Review text**	11. Date of stay	12. Recommend to friend(y/n)
13. **Sub-ratings** for value, room, location, cleanliness & service		

Another finding from the corpus analysis is that the rating and sub-rating attributes have the greatest effect on the contents of a review text because most authors enter values for these attributes. We therefore form our case structure with the problem component consisting of the hotel rating and sub-ratings for cleanliness, location, rooms, service and value while the review text is the only solution component. This implies that the hotel reviews in our casebase can be reused across a wider range of authors and not limited to a single hotel or author. The rating (and sub-rating) attributes are completed on a qualitative scale of terrible (1), poor (2), average (3), very good (4) and excellent (5).

2.4.4 University Coursework Feedback

Academic staff in teaching institutions are generally required to provide written feedback in addition to grades while assessing students' courseworks (or assignments). It is relatively easier to assign grades to students than the additional feedback. This is because a marking scheme or rubric (Goodrich 1996, Mansilla, Duraisingh, Wolfe & Haynes 2009) which elicits the quality expectations of the answers must be prepared along with a coursework. A rubric divides the coursework into a limited number of criteria and summarises what is expected from the student to obtain a specific grade in each criterion. Different weights might also be assigned to each criterion and an average (possibly weighted) gives a student's final grade in the coursework.

Our focus in this domain is to support the marking of courseworks by automatically

proposing feedback text from previously assessed students with similar assigned grades. Feedback text of the courseworks from the same course in the previous academic session could also be used as a casebase if the rubric contents are very similar. Here, the assumption is that students with similar grades will be given similar feedback text. Therefore, feedback text can be reused across several students having similar grades with minimal modifications. This should increase efficiency by allowing more students to be graded and given feedback in less time.

Table 2.2: Partial Rubric structure for the data mining coursework

	Criteria	**Description of required quality**
1.	Task 1.1	Identify the class attribute, numeric attribute with mostly equal distribution and numeric/ nominal attributes with skewed distribution
2.	Task 1.2	Tree generated and depth no more than 7 achieved, Tree parameters specified and important attributes identified
3.	Task 2.1	List 4 important parameters for experimental design
4.	Task 2.2	Classifier results obtained and comparison with one or more baselines and significance discussed
5.	Task 3.1	Boosted classifiers specified, evaluation methodology and trials, effectiveness and efficiency criteria
6.	Task 3.2	Effectiveness & efficiency comparison results and explanation and significance discussed
7.	Task 3.3	Discussion on result presentation
8.	Task 4	ARFF preamble extended and new values created

We obtained the rubric, coursework grades and feedback text related to a single 'data mining' coursework for computing postgraduate students in a UK higher institution[2]. The rubric consists of four main tasks with three of them having 2 or 3 sub-tasks giving a total of eight criteria. Table 2.2 lists these criteria along with a short description of the required tasks that should be completed to achieve maximum grade. Each criteria represents a problem attribute in our case structure while the feedback text is the solution. There are a total of forty-six (46) students assessed for this course meaning that only 46 cases are available. The casebase size is therefore the smallest in comparison to the other datasets described in Sections 2.4.1, 2.4.2 and 2.4.3 used for our experimental evaluation. The implication is that our experiment design will differ slightly for this dataset. For

[2]www.comp.rgu.ac.uk

instance, a leave-one-out test will be more appropriate for this dataset rather than a cross validation. Analysis of the dataset also showed that the feedback texts are generally small. The longest feedback, with seven sentences, occurs in just three cases with an average of about four sentences across all cases. This is relatively small when compared to an average of ten sentences in the feedback text for each of our hotel reviews.

2.4.5 Comparison of dataset features

A common feature of all four datasets used in our text reuse experiments is the presence of a textual solution. This is very important as our research focuses on reuse of textual solutions. Table 2.3 shows a summary comparing the major differences across our evaluation datasets. Two of the datasets, weather forecast revision corpus and hospital incident reports, also have a textual attribute as the main constituent of the problem component in their case structure. On the other hand, all attributes in the problem component of the hotel review and coursework feedback dataset are structured.

Table 2.3: Comparison of our evaluation datasets

		Weather forecast revision	**Medical Incident reports**	**Hotel Reviews**	**University Coursework Feedback**
No. of Cases		2414	362	641	46
Problem		Textual	Textual	Structured	Structured
Solution		Textual	Textual	Textual	Textual
Nature of text		Semi-structured	Unstructured	Unstructured	Unstructured
Unique keywords in problem/solution vocabulary		71/140	567/307	NA/11,336	NA/157
Average keywords in problem/solution text		6.3/6.2	7.0/4.3	NA/263.1	NA/52.4
Case-base Complex-ity	**Global Alignment**	0.8868	0.3769	NA (Not Applicable)	NA (Not Applicable)
	Avg. Case Alignment	0.7978	0.3046	0.4644	0.5639
	Case Cohesion	0.3276	0.0177	0.0019	0.1348

The textual content is semi-structured in the weather forecast revision domain but unstructured in the other domains. Semi-structured textual contents should be easier

to reuse than unstructured text, since information extraction can be applied to identify entities for substitution. Casebase complexity values indicate how well the similarity assumption holds in the datasets and estimate the relative performance expected from applying CBR techniques. Better casebase alignment (higher values) should lead to an above-average effectiveness even if no adaptation is carried out, that is, retrieved solutions are reused verbatim. But lower casebase alignment values imply that some form of adaptation might be required during reuse in order to propose an effective solution. In Table 2.3, the case cohesion and average alignment values are computed using three nearest neighbours. We expect that the best retrieval effectiveness will be obtained from the weather forecast dataset since its complexity values across the various metrics are higher than other datasets.

2.5 Chapter Summary

In this chapter, we critically analysed previous literature related to text reuse in TCBR. The knowledge acquisition process was discussed in terms of the knowledge containers and the importance of each container for text reuse. Representation of textual cases was reviewed and two main groups were identified. These are the knowledge-light and knowledge-intensive approaches to text representation. We exposed the merits and demerits of each approach and explained why we opted for the knowledge-light approach in our research. Indexing architectures that help to increase the efficiency of CBR systems were highlighted and emphasis was made on why they are more important for TCBR.

We examined different techniques for reasoning with problem-solving experiences. In particular, we discussed the concept of similarity and how retrieval performance is highly dependent on it. Current methods for reusing structured and textual cases were then analysed and discussed in relation to text reuse challenges. Evaluation of TCBR systems was discussed and we observed that automated evaluation of textual solutions is common to other disciplines. We concluded this chapter by analysing available experimental datasets and their suitability for our text reuse research.

Chapter 3

Learning What to Reuse

Case reuse in traditional CBR (with structured attributes) typically involves proposing a value for one or more solution attributes where the number of attributes is pre-defined as part of the case structure. For instance, a CBR system for tablet formulation (Craw et al. 2006) for predicting the type and amount of filler, binder, dis-integrant, lubricant and surfactant based on the required physical and chemical properties (e.g. dose, solubility, yield pressure etc.) will be expected to propose a solution for all ten (i.e. type & amount $\times$ 5 [filler, binder,...]) attribute values from retrieved similar cases. This is because all solution attributes have been pre-assessed to be important and relevant to formulating a tablet. Therefore, the main focus of structured CBR reuse is to determine suitable values for solution attributes that solves a new problem. This form of reuse is not always applicable to TCBR when the solution attribute is textual.

For a textual solution, its decomposition into constituent parts (tokens, phrases, sentences or even paragraphs) can be viewed as distinct attributes of the solution. However, these solution attributes cannot be pre-defined in unstructured text, since they are likely to vary across similar cases due to differences in their problem attributes. Consequently, some attributes in a retrieved similar textual solution might be irrelevant to a new problem. For example, a TCBR system can be used to aid the reuse of company bids (which are largely textual) in response to calls for tenders. In this scenario, retrieved similar bids will contain attributes (sections or paragraphs) that are irrelevant or cannot be reused

for the tender. Also new text might need to be inserted apart from irrelevant contents being deleted or substituted. Therefore, an additional task (other than copy and adapt discussed in Section 1.1.1) with textual solution reuse involves the identification of solution attributes relevant to a new problem. This identification task can also be viewed as an intermediate step between copy and adapt, since deletion, substitution as well as insertion are adaptation tasks. In other words, TCBR reuse might involve deletion of attributes or insertion of new attributes when the solution is textual.

Chapter 2 revealed a wide gap in TCBR techniques available for case reuse particularly with unstructured or semi-structured textual solutions. In this Chapter, we address this gap by presenting a novel technique designed to aid text reuse by identifying reusable textual attributes in a retrieved solution text. This is done by annotating a solution text so that reusable attributes are distinguished from those that need adaptation. Our text reuse technique is called Case Retrieval Reuse Net (CR2N) because of its formalization as an extension to a Case Retrieval Net (CRN) (Lenz & Burkhard 1996a). Although the technique can be used independently of the CRN, the new architecture gives a conceptual and graphical view of the technique. The availability of similarity arcs is also expected to increase the effectiveness of our text reuse technique.

Section 3.1 discusses details of the CR2N architecture and a conceptual understanding of how it is applied to text reuse. Accordingly, we detail the CRN and how it is extended into a CR2N. Although the casebase provides the evidences for determining which textual attributes are relevant or irrelevant, there are several ways of using these evidences. A comparative evaluation of different approaches is presented in Section 3.2. This is then followed by an in-depth analysis of the CR2N with details of the algorithm.

3.1 Case Retrieval Reuse Net (CR2N)

We present an architecture that aids text reuse by identifying relevant and irrelevant textual units (e.g. keywords, phrases, sentences) in the solution text of a retrieved similar case. Here, 'relevant' means that they can be reused without adaptation while 'irrelevant' indicates that such textual units need adaptation due to differences between the new and

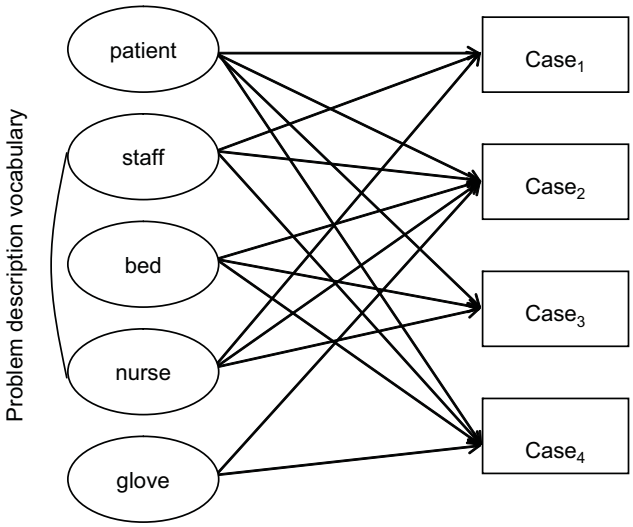
retrieved similar problems. Our architecture called CR2N extends the Case Retrieval Net (CRN) which has been established as an effective and efficient retrieval architecture. This section gives important details of the CRN and how it is extended into CR2N for text reuse.

3.1.1 Case Retrieval Net

Case Retrieval Net (CRN) is a memory model that can efficiently retrieve a relatively small number of relevant cases from a case base. The model in its basic form was proposed by Lenz & Burkhard (1996) although several extensions to the basic CRN such as the lazy propagation CRN (Lenz & Burkhard 1996b), Microfeature CRN (Lenz & Burkhard 1996b) and Fast CRN (Chakraborti et al. 2006) have been proposed. CRN uses a net-like case memory to spread activation for retrieval of similar cases to a query. It consists of four components: case nodes, Information Entities nodes (IEs), relevance arcs and similarity arcs. An IE consists of an attribute-value pair and a case consists of a set of IEs. Relevance arcs connect IEs to cases and show the presence and strength of an IE in a case while a similarity arc connects two IEs and indicates how similar an IE is to another. Case retrieval is performed by activating IE nodes which occur in a given query, propagating this activation according to similarity through the net of IEs using relevance and similarity arcs, and aggregating activation in the associated case nodes to rank cases for retrieval.

The CRN is very efficient during retrieval of similar cases because it avoids exhaustive memory search by not activating case nodes that have no similarity to the query. It also handles partially specified queries efficiently since the similarity arcs enable the expansion of such partial queries in to better and more useful queries. This retrieval architecture is also complete because it assures that every similar case in memory is found during retrieval with the use of the similarity arcs. It is flexible as there are no inherent restrictions concerning the circumstances under which a particular piece of knowledge can be recalled and this is particularly useful for text. When used in TCBR, each IE node is used to represent a single textual unit (keyword, phrase or sentence) depending on the granularity of indexing and similarity matching. Similarities between the textual units are then

captured by the similarity arcs.



	patient	staff	bed	nurse	glove
$Case_1$	1	2	0	2	0
$Case_2$	2	1	2	2	2
$Case_3$	1	0	1	1	0
$Case_4$	2	1	2	0	1

Figure 3.1: Partial CRN for our medical health and safety dataset

A partial CRN built for our medical incident reporting application is illustrated in Figure 3.1 with its corresponding matrix representation for the relevance arc weightings. The figure shows how health & safety keywords relate to incident cases. A relevance arc connects an IE to a case when the keyword associated with the IE is contained in the case. For example the keywords "patient", "staff", "bed", and "glove" occur in $Case_4$. The weight on the arc typically denotes the importance of the keyword in a case. Such relevance arc weights can be stored in a table as shown in Figure 3.1. Here, we use term frequency weights and each row in the matrix relates to a case represented as a feature vector. The similarity arc between "staff" and "nurse" indicates that the two keywords are similar and could be learnt from a glossary, thesaurus or domain ontology. Such similarities between IEs can also be learnt introspectively from the corpus using techniques based on word co-occurrences such as Latent Semantic Indexing (LSI). The values on the similarity arcs can also be stored in a table. Aggregation of network activations across the relevance and similarity arcs can be easily implemented using matrix multiplication as utilized by Chakraborti et al. (2006) if the weightings are stored in tables.

3.1.2 From CRN to CR2N

CRN is used solely for the retrieval of similar cases as explained in Section 3.1.1. Here we discuss how it is extended to aid reuse of textual solutions. The CR2N architecture consists of two CRN structures. These are the original Case Retrieval Net (CRN) (Lenz & Burkhard 1996a), which indexes the problem vocabulary, and a second CRN-like structure, which we refer to as a Case Reuse Net, for the solution vocabulary. Figure 3.2 illustrates the CR2N architecture and its components.

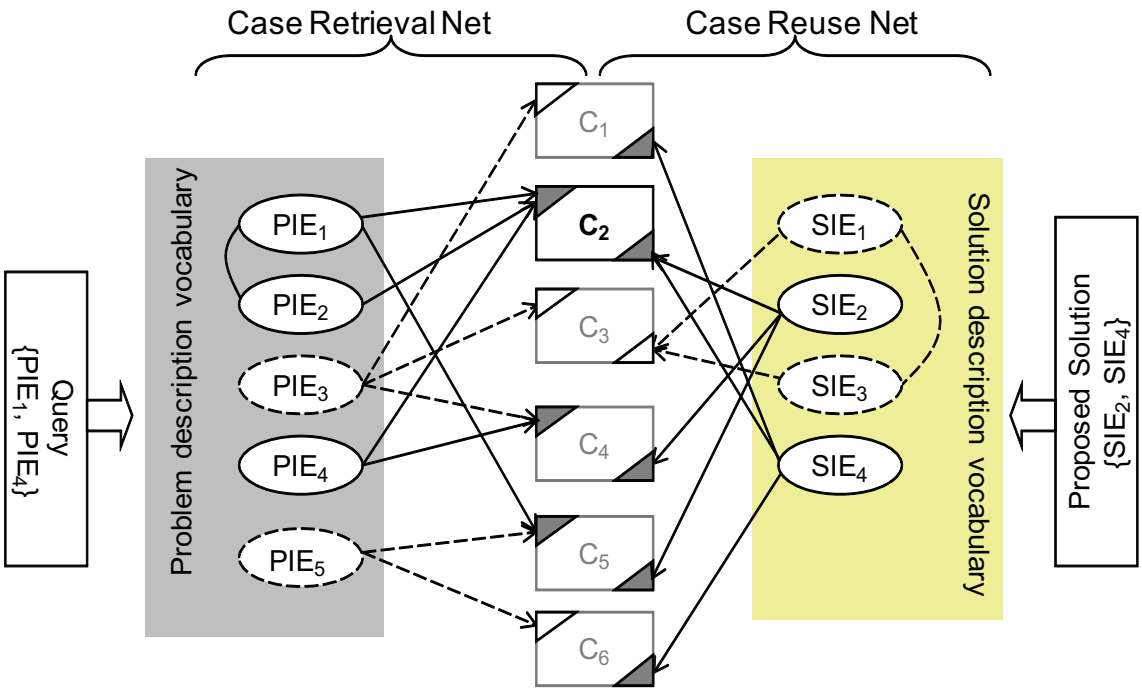


Figure 3.2: The Case Retrieval Reuse Net (CR2N) architecture

The CRN retrieves the most similar case(s) to a query while the Case Reuse Net aids text reuse by identifying information entities (IEs) in the retrieved solution that are relevant or irrelevant to the query. Figure 3.2 shows a trivial example with a case base of six cases and five/four IEs in the problem/solution vocabulary respectively. CRN represents the problem vocabulary of indexed cases as a mapping between IE nodes and cases containing such IEs. Case nodes are denoted as C_i and the problem description IEs are denoted as PIE_i where i is a unique integer. Mapping of IEs onto cases are shown as relevance arcs, while the similarity arcs indicate the similarity between IEs. Solution

description IEs in the Case Reuse Net are denoted as SIE to differentiate these from problem description IEs. It should be noted here that the PIEs can come from structured or textual problems that use a feature vector representation. PIEs can therefore be unique attribute-value pairs in structured problem attributes or terms (keywords or phrases) in textual problem attributes. On the other hand, we assume SIEs are only from textual solution attributes because all attributes in a structured solution must be relevant if they are included in the case structure. Nevertheless, this architecture might be useful in identifying structured solution attribute values that need adaptation, but this is beyond the scope of this thesis as our focus is on reusing textual solutions.

In the CR2N architecture, the problem solving process begins with a new problem (query) spreading activation in the CRN through its PIEs; activated relevance and similarity arcs are shown as solid lines while inactive arcs are shown as dashed lines. Active and inactive IEs for the current query also have solid and dashed outlines respectively, while cases activated are shown with grey coloured triangles at their top left corner and the triangles for inactive cases remain unshaded. The most similar case has the highest aggregation of activations (C_2). Each SIE from the most similar case then spreads activation in the Case Reuse Net to determine its relevance to the query. We decide the relevance of an SIE by comparing two retrieval sets: RS_1, the set of cases activated in the CRN by a query; and RS_2, the set of cases activated by the SIE in the Case Reuse Net. A large intersection between RS_1 and RS_2 implies relevance (verbatim reuse) of SIE otherwise some adaptation is required. In other words, a specific SIE is relevant if the majority of the cases it activates in the Case Reuse Net have already been activated in the CRN. Intuitively, we want to establish if cases with similar problem descriptions to the query also contain the SIE of interest. For instance in Figure 3.2, C_2 (most similar to the query) contains SIE_2 & SIE_4. SIE_2 is determined to be relevant because all cases (C_2, C_4 & C_5) activated by (similar to) the query in the CRN are also activated by (contain) the SIE_2 node. On the other hand, SIE_4 is likely to need adaptation because it only activates one (C_2) out of the three cases activated by the query in the CRN. From another perspective, only one of the three cases (C_1, C_2, & C_6) activated by SIE_4 is similar to the query;

therefore, there is insufficient evidence to suggest that SIE_4 is relevant to the query.

3.2 Evidence for Reuse or Adapt

Our illustration of the CR2N architecture for text reuse with Figure 3.2 gives a simplistic view. This is because the casebase was assumed to contain only six cases which were considered in the creation of the sets RS_1 and RS_2 whose overlap's size determines the relevance of a retrieved solution attribute (SIE). However, an average casebase, even for experimental evaluation, will contain many more cases. Since these sets provide evidence for our annotation judgements, we need to establish if there is a link between the quality of evidence and the size of these sets. Although these sets can be formed from the entire casebase, it is more efficient and effective to use cases in local neighbourhoods during the set formation for casebases with a large number of cases. When using local neighbours, it is typical to utilize the query's neighbourhood in the problem space. However, there are other possible neighbourhoods that might be more suitable such as the retrieved case's neighbours in the problem or solution space. In this section, we give an analysis of the various configurations that can be used to compute evidence for reuse of an attribute in a retrieved textual solution. A comparison is made between using the entire casebase or restricting attention to local neighbourhoods when gathering evidence for the reuse of solution attributes. We then discuss the relative merits of using the problem or solution space in the computation of the reuse evidence.

3.2.1 Local or Global Neighbours

Figure 3.3 illustrates the notion of local and global neighbourhoods with Q representing the query, the shaded circle as the best match case and other circles as cases in the casebase. Local neighbourhood includes nearest neighbours of the query based on similarity while the global neighbourhood consist of all cases in the casebase. The use of all cases to gather evidence for the reuse of a single solution attribute is clearly computationally expensive and may also be ineffective. Global neighbourhoods are typically less effective than local neighbourhoods because local neighbours give a similarity context which is lost as the

neighbourhoods grow to include distant cases. We expect that local neighbourhoods with a small number of similar cases will be more efficient for reuse evidence computation than using the entire casebase (global neighbourhood). Efficiency becomes even more important since the reuse computation is repeated for each solution attribute in the retrieved solution.

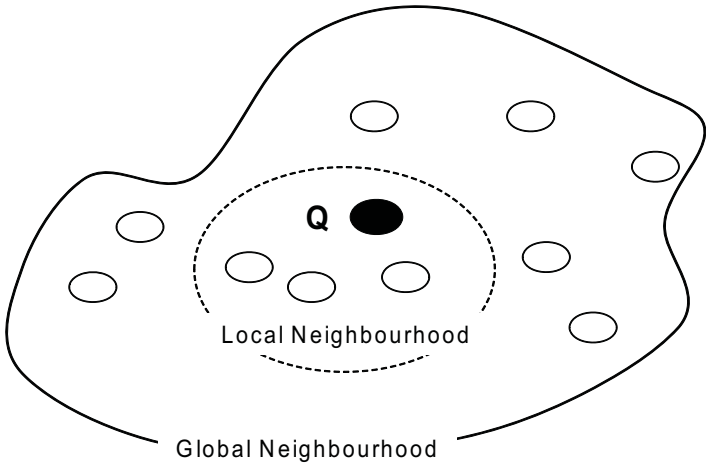


Figure 3.3: Local and Global Neighbourhoods

On the other hand, global neighbourhoods might help to properly discriminate generic attributes which are used across many cases from specialized ones which can be reused only in a specific context. Frequent solution attributes in cases notwithstanding their dissimilarity are more likely to be reused verbatim as opposed to specialized attributes that will need adaptation. The initial similarity computation to identify local neighbours is not required when all cases are used for evidence computation. However, similarity information might already be available from the retrieval component. Further discussion on the influence of local and global neighbourhoods appears in Section 6.2, which analyses results from empirical evaluation of our reuse algorithm using different neighbourhoods.

3.2.2 Problem versus Solution space

It is natural to use the problem space neighbourhood (query's neighbours) for reuse evidence computation, since similar problems should have similar solutions. We will therefore expect that other cases similar to the query should contain solution attributes similar to the retrieved solution and can be used to determine the relevance of the retrieved solu-

tion attributes. For instance in Figure 3.4, Q appearing in the problem space represents a query and the shaded circle represents the best match case. The reuse evidence can be obtained from the four cases nearest to the query as indicated by the dotted outline around Q in the problem space.

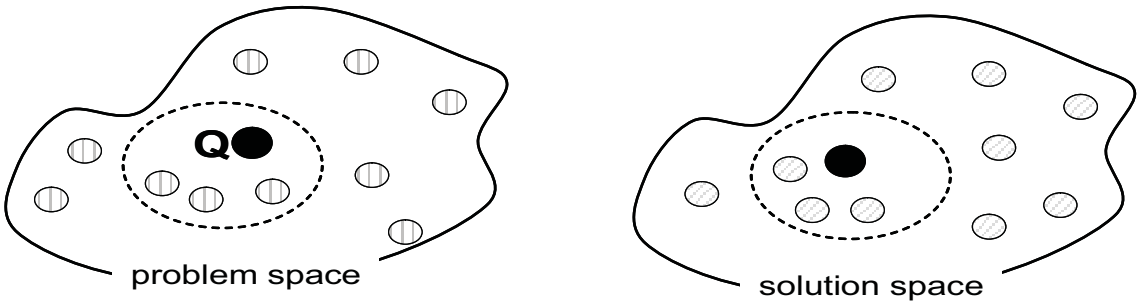


Figure 3.4: Neighbourhoods in the problem and solution spaces

However, we could also focus on the solution space neighbourhood consisting of the retrieved solution's neighbours. The use of this neighbourhood in the solution space allows each retrieved solution attribute (SIE) to be put in the proper context of the entire solution during reuse computation. This might be more important in domains where a solution attribute, whether keyword, phrase or sentence, can have different contextual meaning even within the same domain. Consider an example from medical health and safety domain, the keyword 'plate' might refer to 'food plate' for serving patients. However its context will be different in a 'plate kit' used during surgical operations. Therefore, the reuse evidence gathered from solution space might be more effective because of its ability to contextualise solution attributes by using nearest neighbours of the retrieved solution. We conducted experiments to evaluate this hypothesis that the neighbourhoods from the solution space might give more reliable reuse evidence than nearest neighbours in the problem space. Analysis of the results obtained from our evaluation on different domains is discussed in Section 6.2.

3.3 Text Reuse with CR2N

Text reuse in our context refers to the reuse of experiential cases whose solution components consist mainly of textual contents (unstructured or semi-structured). As discussed

in Chapter 2, very little work has been done in this area. We therefore propose to address this in the Case Retrieval Reuse Net (CR2N) by annotating a solution text so that reusable sections are identifiable from those that need adaptation. The CR2N is formalized as an algorithm independent of the Case Retrieval Net and we explain its capabilities and novel features.

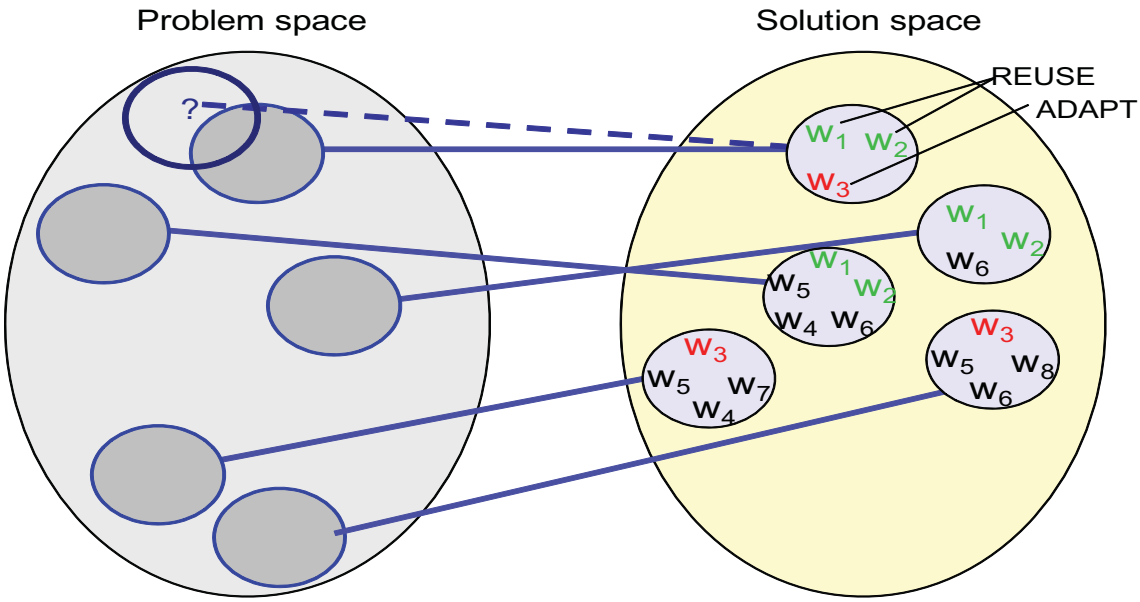


Figure 3.5: Conceptual view of the CR2N for text reuse

Figure 3.5 provides a conceptual view of how the CR2N can be used to aid text reuse. The problem space consist of problem components of cases in the casebase shown as empty circles while their corresponding solution components are in the solution space. A case consist of problem and solution components; a solid line links each problem in the problem space to its solution. Each solution attribute is labelled as w_i and can represent a keyword, phrase or sentence depending on the text granularity being used. It can be observed in the figure that the number of solution attributes varies from one solution to another; this is common in TCBR domains with textual solutions. The circle in the problem space with an inscribed '?' represents a query and it overlaps with its most similar case. The retrieved solution therefore contains w_1, w_2 and w_3. Solution attributes w_1 and w_2 will be annotated as 'reuse' because the three cases containing identical attributes are also similar to the query as shown by their nearness in the problem space. However, w_3 is more likely

to need adaptation because the two cases containing it are farther from the query in the problem space.

Our text reuse technique emphasizes the use of local neighbourhoods for computation of the reuse evidence. We formalise the CR2N architecture as an algorithm shown in Algorithm 3.1 to automatically generate text reuse annotations (i.e. 'reuse verbatim' or 'requires adaptation') for each retrieved solution attribute (information entity). Although the algorithm is based on the simplistic CR2N architecture explained in Section 3.1.2, it is independent of the Case Retrieval Net (CRN) architecture and incorporates more complex mechanisms for calculating the reuse evidence. However, we retain CRN terminologies like Information Entity (IE) since it allows for a more flexible representation at different levels of text granularity. IEs can also be used to represent distinct attribute-values if the problem component is structured.

Algorithm 3.1 CR2N algorithm

Require: CB= $\{C_1,\ldots,C_n\}$, set of cases in the case base
Require: V_p= $\{pie_1,\ldots,pie_m\}$, set of problem IEs in CB
Require: V_s= $\{sie_1,\ldots,sie_l\}$, set of solution IEs in CB
Require: C= $\{P, S\}, where(C \in CB) \wedge (P \subset V_p) \wedge (S \subset V_s)$
Require: Q= a query, where Q $\subset V_p$
Require: k= local neighbourhood size used for reuse evidence computation, where k<= n
Require: σ= similarity threshold between a retrieved solution attribute & other solutions

1: $C_{best} \leftarrow$ SelectK(RET(V_p, Q),1)
2: $RS_1 \leftarrow$ SelectK(RET(V_s, C_{best}), k),default is solution space neighbourhood
<!– replace RET(V_s, C_{best}) with RET(V_p, Q) for problem space neighbourhood –>
3: **for each** ($\{sie_i\} \in C_{best}$) **do**
4: $RS_2 \leftarrow$ SelectT(RET(V_s, $\{sie_i\}$), σ)
5: AS $\leftarrow RS_1 \cap RS_2$
6: BS $\leftarrow RS_1 \backslash RS_2$
7: $\overline{S}_A \leftarrow \frac{1}{|AS|} \sum_{a \in AS} SIM(a, Q)$
8: $\overline{S}_B \leftarrow \frac{1}{|BS|} \sum_{b \in BS} SIM(b, Q)$
9: **if** $\overline{S}_A \geq \overline{S}_B$ **then**
10: REUSE $\{sie_i\}$ <!– relevant to the query –>
11: **else**
12: ADAPT $\{sie_i\}$ <!– irrelevant to query –>
13: **end if**
14: **end for**

One of the important mechanisms in the algorithm is the provision of a parameter that allows for the use of local neighbourhoods in the problem or solution space. The

local neighbourhood size can also be varied up to the casebase size. Another feature of the algorithm is that groups of solution attributes (SIEs) can be used together during reuse evidence computation. For example, if each SIE represents a keyword, then the relevance of a sentence can be determined by activating the sentence's keywords simultaneously in the Case Reuse Net. This allows IEs to be represented (in the CR2N) with the lowest possible text granularity (keyword level) but reuse evidence can be computed at the same granularity level or with higher levels such as phrase, sentence or even paragraph granularity.

The algorithm uses a generic retrieval (RET) function to retrieve cases given a partial case description and an indexing vocabulary. There are two RET function calls, with the first retrieving over the problem vocabulary, V_p, on Line 1 and the second over the solution vocabulary, V_s on Lines 2 and 4. The retrieval sets returned by the RETs are qualified by two further Select functions: SelectK returns the top k cases as in Lines 1 and 2, and SelectT returns all cases with similarity above a specified threshold, σ as used in Line 4. The use of a generic function (RET) in our text reuse algorithm makes the technique independent of the retrieval mechanism. However, we employed the CRN in our experiments, because of its efficiency with applications having very large casebase sizes and because its similarity arcs allow for more semantic retrieval. Therefore, RET can be replaced with CRN anywhere in the CR2N algorithm.

The best match case C_{best}, is identified by retrieving over V_p in response to a query Q (Line 1). Here Q is a case consisting of just the problem description and RS_1 on Line 2 is the resultant retrieval set by retrieving over V_s with the retrieved solution from C_{best}. The reuse stage involves iterating over the proposed textual solution content (i.e. C_{best}'s solution) to identify and annotate relevant parts; see Lines 3-14 of the algorithm. Like the second RET call, the third RET on Line 4 retrieves cases over the solution vocabulary given some partial solution text, which is formally denoted as a set of solution IEs or $\{sie_i\}$ in Algorithm 3.1. The resultant retrieval set is RS_2. It should be noted that $\{sie_i\}$ must be a subset of C_{best}'s solution.

A solution IE is reusable by the query if cases containing it are similar to the query. In

other words we want to establish if cases with similar problem descriptions to the query also contain the solution IE(s) of interest, $\{\text{sie}_i\}$. For this purpose the retrieval sets RS_1 and RS_2 are compared. The intersection of these sets contain cases (AS) that have similar solution to the retrieved solution and also contain the $\{\text{sie}_i\}$, whilst the set difference identifies cases (BS) that are similar to the retrieved solution but not containing $\{\text{sie}_i\}$ as shown on Lines 5 and 6. The annotation is determined by the average similarity of the query to cases in the intersection ($\overline{S}_A$) relative to that of the set differences ($\overline{S}_B$) listed on Lines 7 and 8. The solution is determined to be reusable if $\overline{S}_A$ is greater than or equals to $\overline{S}_B$ else it needs adaptation.

The SelectK(RET(V_s, C_{best}), k) function retrieves k cases similar to the retrieved solution. The function thereby allows the retrieved solution's overall context to be taken into account when annotating each $\{\text{sie}_i\}$. Alternatively, neighbours of the query can be used to measure the effect of ignoring the context of the retrieved solution during reuse evidence computation. To use the query neighbourhood, RET(V_s, C_{best}) on Line 2 is replaced with RET(V_p, Q). The use of a specified k-neighbourhood should decrease the computational cost of the algorithm (especially for large casebases) since a smaller number of cases are used for reuse evidence computation. Small values of k ensure that the reuse evidence comes from a local neighbourhood and remove the influence of cases with little similarity to the retrieved solution.

The CR2N algorithm is generic because IEs can represent any form of textual units (keywords, phrases, sentences etc). Also the algorithm could still be used if each IE represents a keyword and we want to annotate larger textual units like phrases or sentences. This is done by using all keywords in the large textual unit as a set for activation in the function SelectT(RET($\text{V}_s, \{sie_i\}), \sigma$) (Line 4). The best values for parameters k and σ on a given domain must be established empirically.

CR2N is similar to Case Grouping (CG), a text reuse strategy reviewed in Section 2.2.3 in that both exploit the latent relation between a query and each attribute in a retrieved solution text by forming two sets of cases; AS/support & BS/reject in CR2N/CG. However, there are some distinguishing features that make the CR2N stand out from CG.

- Local versus Global Neighbourhoods: CR2N uses knowledge from a specified local neighbourhood to compute the reuse evidence that determines the relevance or otherwise of a textual solution's attribute to a query. CG on the other hand uses the entire case base. The use of local neighbours removes the influence of cases that are dissimilar to the retrieved case during reuse evidence computation.

- Average similarity versus Centroid vector: Average similarity of cases in each group (AS or BS) to the query is employed for reuse/adapt evidence in CR2N rather than centroid vectors used in CG. This is more intuitive since it takes into account similarity to the query of each case individually rather than as a group of cases as used when centroid vectors are created. In other words, CR2N uses the average similarity of cases in a group to the query while CG uses the similarity of the average case (centroid) in the group to the query.

- Novel architecture and formalized algorithm: Unlike CG, retrieval and reuse stages of the TCBR cycle are integrated into a single architecture in CR2N. The technique is also formalized as an algorithm which makes it easier to implement, test and utilize by other researchers.

3.4 Extending CR2N for Reuse Guided Retrieval

Reuse is heavily influenced by the quality of retrieval since TCBR attempts to adapt retrieved cases to solve a new problem. Therefore the quality and accuracy of reuse depends on the quality of cases retrieved. In scenarios where only the most similar case is adapted during reuse, such best match case might not necessarily be the easiest to adapt. This is because most retrieval mechanisms focus solely on the problem space while reuse is done within the solution space. The similarity of cases in the problem space is typically not the same as their similarity in the solution space. In other words, two cases most similar to each other with respect to their problem components might have a third case similar to both but whose solution is more similar to one of the cases than their solution components are to each other.

Figure 3.6 illustrates this concept better with three hypothetical cases in the problem and solution spaces. Here, though case A can be seen as more similar to B than C in the problem space (P_A and P_B), its solution is more similar to C since the distance between S_A and S_C is smaller than that between S_A and S_B. Therefore, if P_A were a query, P_B will be retrieved as the best match. We then try to adapt or transform S_B into S_A. However, it should be easier to transform S_C into S_A since they are closer. The main hypothesis here is that a specific nearest neighbour of a query other than the best match might be more easily adapted to an accurate solution.

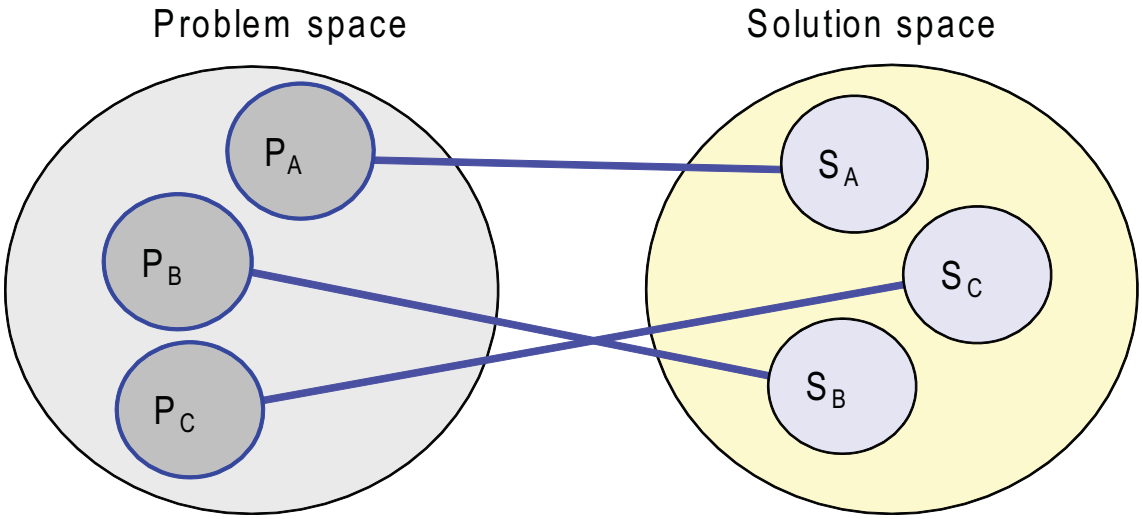


Figure 3.6: Better adaptable solutions from other nearest neighbours

We introduce a technique to determine a single specific neighbour, which can also be the best match case with respect to retrieval similarity, whose solution is best adaptable to solve a new problem. This technique is based primarily on the CR2N reuse technique as detailed earlier in Section 3.3. A reuse metric is also proposed which encodes how easily reusable or adaptable the solution from a particular nearest neighbour is to a query. Further discussion on the use of single or multiple nearest neighbours is given in Section 3.4.1 followed by a detailed explanation of reuse guided retrieval algorithm in Section 3.4.2.

3.4.1 Single vs. Multiple Nearest Neighbours

Solution to a new problem (query) is typically proposed in CBR by adapting the solutions from one or more nearest neighbours. When a single nearest neighbour is used, only the

solution from the best match case is adapted for proposing a new solution. However, the use of multiple neighbours is preferable as it diminishes the effect of noisy cases. A new solution can be proposed for structured solution attributes such as symbols or numbers by combining the solutions from k nearest neighbours. For example, number-type attributes in such solutions can be combined using an average (weighted or not) while attributes with symbolic values or class labels can be combined using voting (weighted or not). The similarity of each case to the query can also be used as weights during such combination.

Nearest neighbourhood's optimal size (k) depends on the deviation of similarity values between cases in the casebase which might in turn depend on the similarity metric and application domain. Generally, smaller values of k like 3, 5 or 7 give better results in CBR as the retrieved cases will be in the query's local neighbourhood and therefore more likely to have the same context. Larger neighbourhoods will include cases with less similarity to the query though their effect can be eliminated or greatly reduced with a weighted combination of the solutions.

For TCBR applications where the solution is textual, it is typical to use one nearest neighbour (1NN) or the best match. This is mainly because of the difficulty in combining texts from different cases without loss of meaning, context, syntax and semantics. This remains a challenge for TCBR; albeit, one of the contributions of this thesis is the introduction of techniques that attempt to combine textual solutions from several nearest neighbours discussed later in Chapter 4. Therefore 1NN is more common in TCBR as it is easier to adapt textual contents from a single case than from multiple cases. However, the other nearest neighbours can be used to influence or support the reuse/adaptation stage as utilised in the CR2N technique discussed in Section 3.3. Also, any of the nearest neighbours other than the best match might be easier to reuse since the relative similarity of cases in the problem space might differ from that of the solution space and this applies to the query as well.

3.4.2 Reuse Guided Retrieval (RGR)

The basic idea in our reuse guided retrieval technique is to determine the best match to a query using not just its retrieval similarity in the problem space but also how much of its solution (in the solution space) can be reused without adaptation. This should generally increase the overall effectiveness of the TCBR system since the new proposed solution will be similar to the actual solution but also easiest to adapt. In other words, retrieval performance is improved by determining a case in the query's neighbourhoods whose solution is easiest to adapt. Our best match case in this scenario can be a case other than the nearest neighbour (1NN). A new metric is introduced which determines the utility of a retrieved case for solving a query by combining the reusable proportion of its solution and its similarity to the query. Intuitively, this metric assigns a high score to a retrieved case whose problem is very similar to the query and whose solution can be reused with very little adaptation. We leverage the CR2N to determine what proportion of the solution of a query's neighbour is reusable; therefore, our technique (just like the CR2N) is limited to domains with textual solutions. However, the idea is applicable to other domains with non-textual solutions if techniques similar to the CR2N are available to determine the ease of reuse or adaptability of such solutions relative to the query.

Algorithm 3.2 lists the pseudo codes for the reuse guided retrieval technique. This uses function *getReuseUtilityScore* which returns the reuse utility score for any case in the query's specified neighbourhood. We show the pseudo codes for this function in Algorithm 3.3 separately to allow an explanation of how it incorporates the CR2N algorithm. Most of the parameters required by our reuse guided retrieval algorithm are identical to those in the CR2N algorithm. These includes casebase (CB), problem/solution vocabulary (V_p/V_s), query (Q) and similarity threshold to assess a case solution as containing a similar term (attribute) to a retrieved solution term (σ). The reuse neighbourhood size (rs_k) here also serves the same purpose as k in CR2N can be tuned to obtain the best performance. We expect the best value for rs_k to be identical or very similar to the optimal k value obtained from CR2N empirical evaluations in any domain. Additional parameters in the reuse guided algorithm are the size of the query's neighbourhood (ret_k) from which

Algorithm 3.2 Reuse guided retrieval algorithm

Require: CB= $\{C_1, \ldots, C_n\}$, set of cases in the case base
Require: V_p= $\{pie_1, \ldots, pie_m\}$, set of problem IEs in CB
Require: V_s= $\{sie_1, \ldots, sie_l\}$, set of solution IEs in CB
Require: C= {P, S}, where $(C \in CB) \wedge (P \subset V_p) \wedge (S \subset V_s)$
Require: Q= a query, where $Q \subset V_p$
Require: ret_k= query's retrieval neighbourhood
Require: rs_k= optimal reuse neighbourhood size
<!– based on empirical evaluation of CR2N on best match case –>
Require: σ= similarity threshold between a retrieved solution attribute and other solutions
Require: α= weight on retrieval similarity in reuse utility score
Require: β= weight on reuse proportion in reuse utility score
<!– where $\alpha + \beta = 1$ –>
1: $CB_{local} \leftarrow$ SelectK(RET(V_p, Q), ret_k)
2: Initialise RM $\leftarrow \{rm_1, \ldots, rm_{ret\_k}\}$, reuse utility score for each retrieved case
3: **for each** ($C_i \in CB_{local}$) **do**
4: $rm_i \leftarrow$ getReuseUtilityScore(C_i,Q,V_p,V_s,rs_k,σ, α, β)
5: **end for**
6: max= getMaxValue(RM)
7: index= getIndex(max, RM)
8: **return** C_{index} as C_{best}, where $C_{index} \in CB_{local}$

one is selected as best match, weight of the retrieval similarity (α) and reuse proportion (β) in reuse utility score. We expect ret_k to be a very small value, typically less than 10. This is because cases ranked lower during retrieval are likely to be less similar and therefore less reusable except the query belongs to a densely-populated cluster with very close similarity values between the query and several cases in its neighbourhood. The *RET* and *SelectK* functions are also identical to those in the CR2N algorithm.

The reuse guided retrieval algorithm begins in line 1 with the retrieval of nearest neighbours of the query in the problem space using the *RET* function which incorporates the similarity metric. The top ret_k neighbours (CB_{local}) are then selected by the *SelectK* function from which one of them will be assessed to be the best match. The remainder of the algorithm on Lines 2-8 calculates a reuse utility score for each case from these top neighbours and selects the case with the highest reuse utility score as the best match. It should be noted that the case returned as the best match might be the same as the retrieval best match (1NN) if the other neighbours are not found to have a better utility score.

Algorithm 3.3 Function *getReuseUtilityScore*

Input: V_p= {pie$_1$,. . . ,pie$_m$}, set of problem IEs in the casebase
Input: V_s= {sie$_1$,. . . ,sie$_l$}, set of solution IEs in the casebase
Input: Q= a query, where Q $\subset V_p$
Input: C_{nn}= $\{P, S\}, where(P \subset V_p) \wedge (S \subset V_s)$
<!– A case in the nearest neighbourhood of Q –>
Input: rs_k= reuse neighbourhood size (optimal)
Input: σ= similarity threshold between a retrieved solution attribute and other solutions
Input: α= weight on similarity between Q and C_{nn}
Input: β= weight on reuse proportion in C_{nn}, where $\alpha + \beta = 1$
Output: Proportion of reusable terms in C_{nn}

1: rc $\leftarrow$ 0, counter for number of reusable terms in solution of C_{nn}
2: $RS_1 \leftarrow$ SelectK(RET(V_s, C_{nn}), rs_k)
3: **for each** ({sie$_i$} $\in C_{nn}$) **do**
4: $RS_2 \leftarrow$ SelectT(RET(V_s, {sie$_i$}), σ)
5: AS $\leftarrow RS_1 \cap RS_2$
6: BS $\leftarrow RS_1 \backslash RS_2$
7: $\overline{S}_A \leftarrow \frac{1}{|AS|} \sum_{a \in AS} SIM(a, Q)$
8: $\overline{S}_B \leftarrow \frac{1}{|BS|} \sum_{b \in BS} SIM(b, Q)$
9: **if** $\overline{S}_A \geq \overline{S}_B$ **then**
10: rc $\leftarrow$ rc+1, [REUSE attribute {sie$_i$}]
11: **end if**
12: **end for**
13: size $\leftarrow$ getSizeSolutionTerms(C_{nn}), total number of terms in solution of C_{nn}
14: ret_sim $\leftarrow SIM(C_{nn}, Q)$
15: rs_prop $\leftarrow$ rc $\div$ size, reuse proportion
16: score $\leftarrow \alpha * ret\_sim + \beta * rs\_prop$
17: **return** score

The reuse utility score is calculated as a weighted average of retrieval similarity and reuse proportion values; see Algorithm 3.3. Lines 2-12 of function *getReuseUtilityScore* are identical to those in the CR2N (Algorithm 3.1) and this is where we compute the number of reusable terms in the solution of a given nearest neighbour (C_{nn}) of the query. Although we used the absolute values of the reuse proportion in our utility score computation, other functions such as binary logarithm as used in information entropy (Shannon 1948, Ash 1990) might also be used to minimize the effects of the solution length (*size* in Algorithm 3.3) on the computed score. One advantage of absolute reuse proportion value is that it ensures that the cost of inserting new terms during reuse/ adaptation is higher than the cost of deleting terms in the proposed solution. This is intuitive because deleting terms from a piece of text is generally easier (less costly) than adding

other new terms when modifying the text. For instance, if there were two solutions with 3 and 4 terms out of which 2 and 3 terms were determined to be reusable respectively. These will give reuse proportion values of 0.67 (2/3) and 0.75 (3/4) while logarithmic equivalent will be 0.578 ($\log_2 2/3$) and 0.415 ($\log_2 3/4$). But the solution with 3 reusable out of 4 terms will typically be preferred since the cost of deleting a term from this solution (if required) will be less than adding a term to the other solution text with 2 reusable out of 3 terms.

3.5 From What to Reuse to How to Reuse

CR2N aims to address our research question of "what to reuse". An important outcome of the CR2N and other reuse strategies is the alignment between the problem and solution space. In other words, retrieved solution terms annotated as "reuse" must be aligned to one or more attributes or terms in the query while the "adapt" terms are unaligned. However, this alignment is implicit or latent as there is no knowledge of the specific query attributes aligned to each reuse solution terms. The question of how to reuse might be easier to address if these problem-solution alignments were made explicit. This is because the solution for specific unaligned query attributes might then be selected from other nearest neighbours apart from the best match case.

Explicit alignment is far more difficult to determine when both problem and solution are textual. The challenge is compounded further when working at the keyword granularity where context is minimal. We expect that solution terms will be explicitly aligned to phrases or sentences in a textual problem rather than a single keyword. However with structured problems and textual solutions, explicit alignment should be easier to formulate. This is because each problem attribute contains a specific context that might easily be matched to phrases or sentences in the solution space. Hence, our second research question of how to reuse is addressed by investigating datasets (e.g. hotel reviews) with structured problems and textual solutions. Our concept of explicit alignment and how it is used to address the questions of what and how to reuse are discussed in Chapter 4.

3.6 Chapter Summary

Firstly, we propose the reuse stage in TCBR as identification of reusable textual constructs in a retrieved solution text followed by adaptation of constructs that have been determined to be irrelevant to the query. Here, each textual construct is called a solution attribute or term and its size can vary depending on the granularity being used. Text granularity can be at the keyword, phrase, sentence or paragraph levels. The CR2N algorithm introduces novel methods for calculating the reuse evidence which determines the relevance/ irrelevance of each retrieved solution term to the query. The algorithm is generic because it can work with any level of text granularity and also allow the use of the entire casebase as well as neighbourhoods in the problem or solution space.

Another contribution of the work in this chapter is the integration of the retrieval and reuse stages in TCBR into a single architecture called CR2N. This is unique and novel because the Case Retrieval Net (CRN) has been extended or improved for retrieval but not for the reuse stage. The integration enables the CR2N algorithm to leverage the renowned capabilities of the CRN in terms of effectiveness and efficiency. We then extend CR2N's utility to guide retrieval of similar cases whose solutions are easiest to adapt. A reuse utility score is introduced which combines retrieval similarity and the proportion of reuse; this is used to re-rank nearest neighbours of a query so that the best match is not just most similar to a query but less effort is spent to adapt its solution during reuse. The reuse utility score and reuse guided retrieval techniques are also formalized as algorithms which makes them easier to reproduce for comparison with other similar techniques. Finally, we discuss how the CR2N leads to implicit alignment between the problem and solution space and the extension required to make it explicit. Our research question of how to reuse is easily addressed with the presence of explicit alignments.

Chapter 4

Text Authoring with TCBR

Textual Case-Based Reasoning (TCBR) is typically employed to assist with the reuse of problem-solving experiences documented as text. Such problem-solving experiences are reused to assist new users by adapting previous solutions whose problems are similar to the current problem. However, authoring of textual documents can be viewed as a problem solving task especially when textual contents underpin values assigned to structured attributes. This is because users generally find it easier to document their experiences by only completing structured formats such as assigning a percentage score, ticking checked boxes, rating things on a qualitative scale or just picking from a list instead of authoring an experience in natural language texts even when it is meant to explain chosen structured values. For example, it will be easier for a guest who just stayed at a hotel to rate it as bad, average or excellent than to describe specific things he/she likes or dislikes about the hotel in free text. Here, we propose TCBR as a strategy to assist text authoring by suggesting textual contents from previous cases with similar ratings.

The task of authoring documents that include pre-defined attributes along with some textual content is common to several domains. Such documents include reviews, student feedback, medical notes and incident reports. In this chapter, we investigate the role of text reuse, an integral part of TCBR, for text authoring applications that involve feedback or review generation. Generally providing feedback in the form of assigning a rating from a Likert scale is far easier compared to articulating explanatory feedback as text. When

previous feedback generated about the same or similar objects are maintained as cases, there is opportunity for knowledge reuse. Our focus is to assist authors to write better and more comprehensive reviews by proposing useful text which they can easily edit to their taste. Such reuse can be improved by identifying parts of the feedback text that refer to a particular rating attribute. It is also important to avoid irrelevant specialised details that are not easily reusable across several authors. We introduce a method for text alignment which enables the identification of sentences in the feedback text that are related to specific rating descriptors. Transformational and compositional adaptation techniques are then proposed for the reuse of these feedback texts given a similar set of structured rating attributes. Details of our text alignment method are given in Section 4.1 followed by an explanation of our text reuse techniques in Section 4.2. We conclude with a summary of the main contributions of this Chapter in Section 4.3.

4.1 Text Alignment

The problem and solution of an experiential case can both have multiple attributes. In such a scenario, each solution attribute might depend on a specific combination of problem attributes rather than all. Knowledge of such problem-solution attribute alignment should allow for better retrieval accuracy. This is because the best values for solution attributes can be retrieved from different cases with aligned problem attributes most similar to the query. However, learning such relationships or alignment between problem and solution attributes remains a challenge when they are not explicitly expressed in the domain. This applies to TCBR where it is difficult to predict which section of a text (e.g. sentence) in the solution relates to specific problem attributes and attribute values.

We propose a mechanism called text alignment for approximating the implicit relationship between specific structured problem attributes and particular chunks (sections) of a textual solution. Text alignment consists of two basic processes: generation of seeds and alignment approximation discussed in Sections 4.1.1 and 4.1.2 respectively. Although our alignment process is discussed in the context of hotel reviews, it is generally applicable to other domains with similar characteristics (i.e. structured problem and textual

solution). The problem-solution alignment discussed here is explicit in the sense that each problem attribute is aligned to a specific textual solution section unlike the CR2N whose alignments are implicit.

4.1.1 Seed Generation

Text alignment is an attempt to bridge the vocabulary in the problem and solution spaces. Our assumption is that each part of a solution text is related to at least one structured problem attribute. This is because the solution text in our application domains are complementary and contain reasons for the values chosen for each structured attribute in the problem space. Therefore it should be possible to identify terms (seeds) in the text that are typically used when giving explanatory feedback for each structured attribute. The idea is that an author is most likely to use one of the seeds for an attribute when writing text related to that attribute. In other words, each seed for an attribute will have a high probability of occurrence in the section of text describing the attribute and its value. We use the name 'seed' because such terms form the core of any text related to the attribute and can be expanded (or germinated) into bigger texts. A seed in our context is therefore a term (keyword or key-phrase) that is crucial when describing a particular structured attribute. Such seeds are therefore mostly nouns, noun phrases and verbs but can also be adjectives or adverbs. Seeds can also be viewed as terms that carry sentiments (neutral, positive or negative) about structured attributes.

A seed list for a particular attribute can be thought of as a descriptive expansion of the attribute and its possible values. This list can be generated for each structured attribute using its main descriptor name with other terms having the same meaning (synonyms, hypernyms or hyponyms) and opposite meanings (antonyms) for negative sentiments. The synonyms and antonyms can be learnt from dictionaries, web resources like Wikipedia[1], general-purpose ontologies like WordNet (Fellbaum 1998) and/ or domain-specific ontologies. When the seed terms are learnt automatically from any of these sources, it is important to ensure that they are applicable in the same sense as used in the attribute descriptor. Word sense disambiguation (Brown, Pietra, Pietra & Mercer 1991, Yarowsky 2000, Agirre

[1] http://en.wikipedia.org/wiki/Wiki

& Edmonds 2006, Navigli 2009) algorithms can be used for this task of identifying seeds having the same context as the attribute descriptor. However a manual verification of the seed list should still be carried out since none of the available sense disambiguation algorithms are 100% accurate. For instance, manual verification of the seeds generated for our hotel reviews dataset led to the removal of cleanliness rating seeds such as 'blank', 'fair' and 'light'. These seeds though related to cleanliness in the general sense were deemed unuseful to describe a hotel's cleanliness.

Table 4.1: Sample seeds used for text alignment in hotel reviews authoring

Rating descriptor	**Sample seeds list**
cleanliness	clean, neat, tidy, dirty, unclean, filthy
location	locality, location, position, place, region
room	room, bedroom, toilet, bed, table, chair
service	disservice, help, serve, service, reception, star
value	esteem, rate, value, worth, worthless

We illustrate the generation of seeds using our hotel review authoring domain. Here, the structured attributes are qualitative ratings on a scale of 1 to 5 for five attributes with descriptor names: cleanliness, location, room, service and value. Seeds are generated for each attribute by automatically obtaining the descriptor's synonyms, antonyms (opposite meanings for negative sentiments), immediate hypernyms (generalized form) and hyponyms (specialized form) from WordNet. Each list is then manually refined to remove terms with irrelevant word sense and adding other relevant terms that are missing. A sample list of seeds generated for each rating attributes in our hotel reviews is given in Table 4.1. It can be observed from the table that seed terms for some structured attributes such as 'location' and 'room' rating seem to have neutral sentiments because the sentiment-rich words that will be used along with their seeds, such as 'good', 'bad', and 'terrible', are applicable across all rating descriptors. This is unlike the 'cleanliness', 'service' and 'value' attributes, which have unique seeds that might show positive or negative sentiments. The names of the components of a rating descriptor (where possible) are also very useful seeds. For instance, a hotel room will contain at least a bed, chair and table which might be referred to in the feedback text. These additional seeds are more likely to co-occur in the

same sentences as the original seeds obtained from rating descriptors. Techniques such as Latent Semantic Analysis (Deerwester et al. 1990) and Mutual Information (Manning & Schütze 1999) that help to identify strong relationships between words based on their co-occurrences will therefore be very useful for learning additional seeds. But seeds learnt from these techniques could introduce more errors into the text alignment process. These errors can be reduced by setting a high threshold for the co-occurrence score as well as manual inspection to remove incorrect seeds.

4.1.2 Alignment Link Generation

Textual solutions must be parsed into suitable chunks for text alignment to take place. The size of each text chunk could be a clause, sentence, paragraph or even a section of several paragraphs depending on the average size of the textual solutions and typical style of writing in the domain. For instance, if each solution texts contain several paragraphs, it would be more reasonable to align each structured attribute to specific paragraphs rather than sentences. We approximate the alignment between each structured problem attribute and textual solution chunks using the generated seed lists. A chunk of text is aligned to a specific structured attribute if any of the attribute's seeds occur in the text. In other words, the presence of any term from a structured attribute's seed list in a section (chunk) of text indicates some relationship (alignment) between the text and problem attribute.

For our hotel review dataset, we worked at the sentence level of text granularity. Each review text is parsed into sentences for text alignment to take place. A review sentence is aligned to a particular rating attribute if it contains any of the terms in its seed list. This is illustrated in Figure 4.1, which shows a review and its alignment of sentences to the five rating attributes. Here, seeds within each sentence are shown in bold and six (6) of the nine (9) sentences are aligned to the pre-defined ratings. Notice how most of the aligned sentences are semantically related. For example, sentence 1 is about the proximity of the hotel to rail station and is correctly aligned to *location* rating. However, sentence 5 might be better aligned to *location* than *service* rating since it highlights the hotel's proximity to restaurants and local shops. The unaligned sentences (i.e. 2, 3 & 4) seem related to

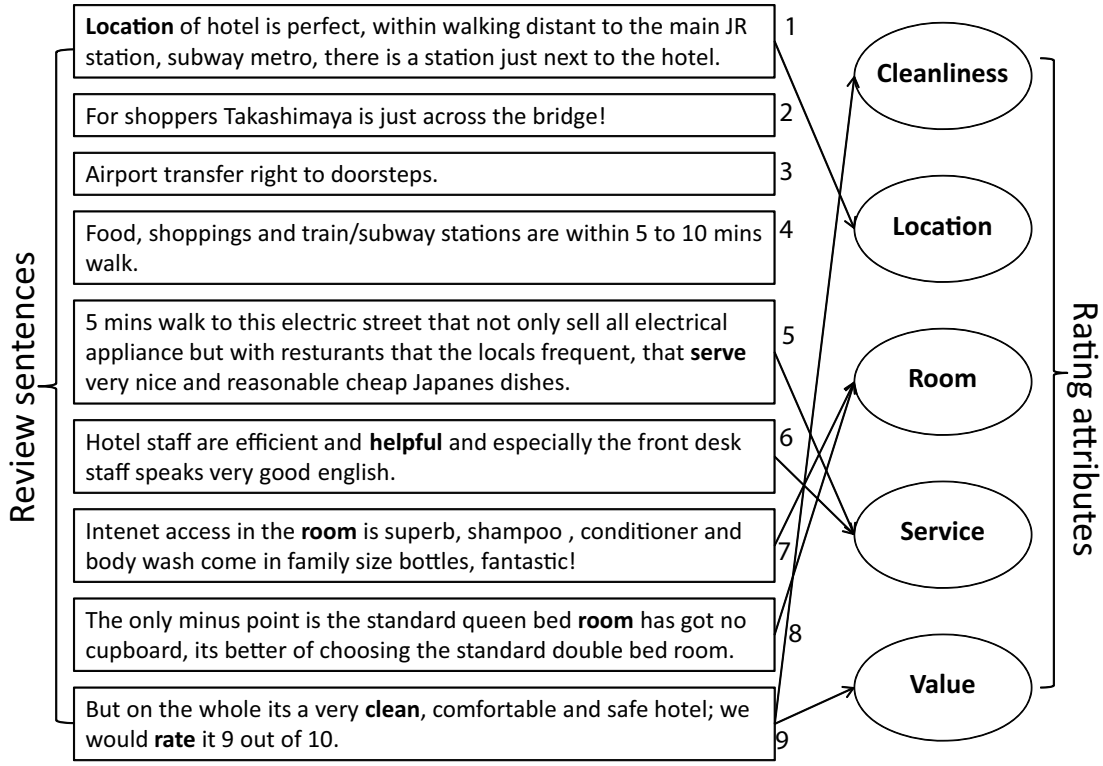


Figure 4.1: Text alignment in hotel reviews authoring domain

location, but were not linked, because they do not contain any of the seeds generated from our method as explained in Section 4.1.1. Relevant terms in these sentences such as 'shop', 'airport' and 'train' can be manually added to the seed list. Additional seeds can be manually detected by inspecting the alignment links across a small random sample of cases to improve the alignment process. The alignment generation in this example seems reasonable with about 67.7% accuracy; that is 6 out of 9 sentences were correctly aligned. Unaligned review sentences might be viewed as verbose details (sometimes unrelated to the hotel) that cannot be easily reused without alignment evidence. For example, not every hotel will be in a town with an airport, which means that sentence 3 might not be very useful to authors giving feedback on such a hotel. Nevertheless, the seed generation might be improved further with access to domain ontologies, which currently are not available for this domain.

The importance of the quality of seeds cannot be over-emphasized as this has a direct influence on text alignment. It is expected that alignment accuracy results will vary when using different subsets of the seeds list. Such subset of seeds can be created by separating seeds generated for the same structured attribute from different knowledge sources. This enables the determination of the knowledge source which gives seeds of the best quality and this should vary across different domains.

Alignment in TCBR can be viewed as a many-to-many relationship, since a sentence can belong to more than one rating and vice versa. This is illustrated in Figure 4.1 where sentences 7 & 8 are linked to *room* rating, whereas sentence 9 is linked to *cleanliness* and *value* ratings. A many-to-many relationship implies that the same sentence might be selected more than once when assembling a proposed solution from the sentences of the best match attribute values to a new problem. This also applies if the relationships from sentence to rating attributes were one-to-many but will not be applicable if each sentence was aligned to only one rating attribute, that is one-to-one or many-to-one relationships. A simple remedy is to use data structures (e.g. set) that do not store duplicate sentences of the proposed solution or to remove such duplicates after solution assembly. Another remedy, albeit less desirable, might be to enforce that each sentence is aligned to only one rating attribute during the alignment generation using a sensible heuristic. One heuristic might be to parse any sentence aligned to multiple attributes into clauses where possible. For instance, sentence 9 in Figure 4.1 could be broken into two meaningful clauses using the semi-colon as a delimiter so that each clause is aligned to only one attribute (cleanliness or value). Another heuristic that can be used to enforce a one-to-one relationship if a sentence is aligned to several rating attributes is to align the sentence to an attribute (out of the several aligned attributes) having no other aligned sentence or the fewest aligned sentences. The sole alignment might also be assigned to the attribute having the highest number of seed in the sentence. Ties from any of these heuristics can be resolved by random selection of one aligned attribute.

4.2 Reuse Algorithms for Text Authoring

We propose reuse techniques for aiding text authoring in domains that involve feedback or review generation. Two novel algorithms for text reuse are introduced in addition to a retrieve-only system which serves as baseline. These techniques make use of our text alignment approach between structured problem attributes and textual solutions as explained in Section 4.1. The differences between the techniques are mainly in terms of what neighbourhood of a new problem (query) is used and how chunks of text from several similar cases are assembled to form a proposed solution text. Although our techniques are similar to those proposed for substitutional, transformational and compositional adaptation in traditional CBR, they have not been used (to the best of our knowledge) for proposing textual solutions. We adapt these traditional CBR techniques for TCBR when assembling chunks of text from the nearest neighbours of a query. The baseline technique is explained in Section 4.2.1 while transformational and compositional approaches to text reuse are discussed in Sections 4.2.2 and 4.2.3 respectively.

4.2.1 Baseline Text Reuse

The baseline (BASE) reuse technique is a retrieve-only system which employs the use of our text alignment method. Given a new set of structured problem attributes (query), BASE retrieves the nearest neighbour and reuses its textual solution. However, it keeps only chunks of text that are aligned to the structured attributes as identified by text alignment. In other words, the proposed textual solution by BASE will not contain chunks of the retrieved text that have no alignment to any of the structured attributes. Unaligned chunks of text are not reused when assembling the new solution.

Algorithm 4.1 lists pseudo-code for the BASE technique. Here, the function *RET* in line 1 returns the most similar case (C_{best}) from the casebase given a new problem Q, which consists of a set of structured attribute values. In the hotel review dataset, these are Likert values ($1 \ldots 5$) for the cleanliness, location, room, service and value rating attributes. The rest of the algorithm deals with obtaining chunks of text from the retrieved textual solution ($SolutionText$) which are aligned to any of the structured attributes. The

Algorithm 4.1 Baseline text reuse algorithm (BASE)

Require: CB= $\{C_1, \ldots, C_n\}$, set of cases in the case base
Require: $R = \{r_1, \ldots, r_p\}$, set of structured attributes e.g. ratings in hotel reviews
Require: $V = \{v_1, \ldots, v_q\}$, possible values for each structured attribute e.g. rating values
Require: IE= information entity consisting of a structured attribute with distinct value, where $(attribute(IE) \in R) \wedge (attributeValue(IE) \in V)$
Require: $C_i = \{\text{IE}_{i1}, \ldots, \text{IE}_{ip}, \text{SolutionText}_i\}$, where $(i \in \{1 \ldots n\})$
i.e. a case consists of p attribute values and a solution text
Require: $Q = \{\text{IE}_1, \ldots, \text{IE}_p\}$, a query with p attribute values
1: $C_{best} \leftarrow RET(\text{CB}, Q, 1)$, retrieve most similar case
2: $SOLN = \{\}$,
<!– to contain chunks of text in the proposed solution text –>
3: SolutionText$\leftarrow$ $getSolutionText(C_{best})$
4: **for each** $\text{IE}_j \in C_{best}$ **do**
5: $r_j \leftarrow attribute(\text{IE}_j)$
6: $S_j \leftarrow selectAlignedTextChunks(r_j, \text{SolutionText})$
7: $addTextChunks(S_j, SOLN)$
8: **end for**
9: Aggregate chunks of text in $SOLN$ for reuse

proposed solution text denoted as $SOLN$ is a set structure thereby ensuring that duplicate chunks of text are ignored when they are put together. This is the simplest remedy to the many-to-many relationship, as discussed earlier in Section 4.1.2, since each chunk of text can be aligned to more than one problem attribute. Lines $4 - 8$ of the algorithm deals with the removal of unaligned chunks of text in C_{best}, the best match case. Each chunk of text is a sentence from a review text in the hotel review dataset. Therefore, function $selectAlignedTextChunks$ identifies aligned sentences for each of the five pre-defined rating attributes while $addTextChunks$ adds the aligned sentences to the proposed solution. BASE will generate five or more sentences in a proposed solution text in the hotel review domain assuming that each attribute is aligned to at least one distinct sentence in the review text.

4.2.2 Transformational Text Reuse

We introduce a transformational (XFRM) approach to text reuse which uses multiple nearest neighbours of a new problem to assemble a solution text, rather than a single nearest neighbour as used in our baseline technique (see Section 4.2.1). The main idea is to use

other nearest neighbours to progressively adapt the solution text from the best match case into a more accurate solution. This takes place only if a new problem (query) and its best match are not identical; that is there are mismatches. Therefore some attribute values in the query will be different from those in the best match case. Transformational text reuse occurs by replacing chunks of text aligned to mismatched attributes with those from other nearest neighbours having better attribute values than the query; 'better' here means their values are closer to the query than those from the best match case. Such chunks of text are only replaced if they are not aligned to any other problem attribute. Also, the search space for cases with better attribute values can be controlled by limiting the neighbourhood parameter (k). This approach to text reuse is similar to CBR transformational adaptation (Chang, Cui, Wang & Hu 2004), where solution elements are re-organised through add and delete operations. It is also similar to substitutional adaptation (Wilke & Bergmann 1998, González-Calero et al. 1999, Adeyanju et al. 2008), when viewed as successive replacement of aligned chunks of text in solution obtained from the best match case.

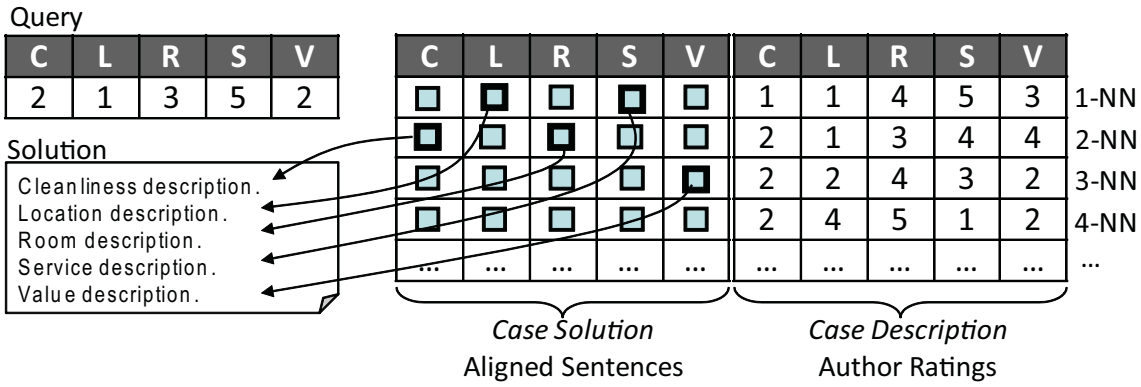


Figure 4.2: Transformational text reuse with hotel reviews

Transformational text reuse is illustrated in Figure 4.2 where it is applied to our hotel review dataset. Given a query consisting of values for the five rating attributes, specified $k-$nearest neighbours are retrieved. Here $k = 4$, therefore four nearest neighbours are shown with their problem attribute values and aligned sentences as shaded squares. Only sentences aligned to the 'location' and 'service' ratings are chosen from the best match (1-NN) since they have attribute values identical to the query's. Mismatched attribute

values are resolved by utilizing aligned sentences from other similar cases in the neighbourhood (2-NN & 3-NN) with closer values to the query than 1-NN. The aligned sentences used for assembling a proposed solution are shown in the diagram as squares with dark outlines. Note that if no better values for the mismatches are found in the other neighbours, sentences from the best match case aligned to these rating attributes are retained in the assembled solution. Therefore, there will always be five or more sentences in the assembled solution, assuming all cases in the casebase have at least one unique sentence aligned to each attribute.

Algorithm 4.2 Transformational text reuse algorithm (XFRM)

Require: CB= $\{C_1, \ldots, C_n\}$, set of cases in the case base
Require: $R = \{r_1, \ldots, r_p\}$, set of structured attributes e.g. ratings in hotel reviews
Require: $V = \{v_1, \ldots, v_q\}$, set of possible values for each structured attribute
e.g. rating values
Require: IE= information entity consisting of a structured attribute with distinct value, where $(attribute(IE) \in R) \wedge (attributeValue(IE) \in V)$
Require: $C_i = \{\text{IE}_{i1}, \ldots, \text{IE}_{ip}, \text{SolutionText}_i\}$, where $(i \in \{1 \ldots n\})$
i.e. a case consists of p attribute values and a solution text
Require: $Q = \{\text{IE}_1, \ldots, \text{IE}_p\}$, a query with p attribute values
1: $SOLN = \{SOL_1, \ldots, SOL_p\}$,
<!– set of proposed sentences for each problem attribute –>
2: $\text{CB}_{local} \leftarrow RET(\text{CB}, Q, k)$, retrieve k similar cases
3: **for each** $\text{IE}_j \in Q$ **do**
4: $qr = attribute(\text{IE}_j)$
5: $qv \leftarrow attributeValue(\text{IE}_j)$
6: $dv \leftarrow 1000$ <!– initialise difference between query and case attribute values to determine best match –>
7: **for each** $\text{C}_i \in \text{CB}_{local}$ (in order of decreasing similarity) **do**
8: SolutionText$\leftarrow getSolutionText(C_i)$
9: $r_j \leftarrow attribute(\text{IE}_j, C_i)$
10: $v_j \leftarrow attributeValue(\text{IE}_j, C_i)$
11: **if** $qr = r_j \wedge |qv - v_j| < dv$ **then**
12: $dv \leftarrow |qv - v_j|$
13: $clear(SOL_j)$
14: $S_j \leftarrow selectAlignedTextChunks(r_j, \text{SolutionText})$
15: $addTextChunks(S_j, SOL_j)$
16: **end if**
17: **end for**
18: **end for**
19: Aggregate all chunks of text in $SOLN$ for reuse

The transformational text reuse pseudo code is listed in Algorithm 4.2. Here, function

RET (in line 2) returns k nearest neighbours CB_{local} given a query. The aligned chunks of text for each problem attribute are then extracted from the nearest neighbours whose attribute values are most similar to the query's. Text chunks are extracted from retrieved cases in decreasing order of similarity to the query, because similarity reflects the overall closeness to the query. This means that chunks of text from a case with a higher overall similarity should be more reusable by a new author with little or no modifications. The conditional statement '$qr = r_j$ AND $|qv - v_j| < dv$' on line 11 ensures that aligned chunks of text for each attribute are only extracted from the first similar case within the specified neighbourhood whose attribute values best match the query's. The difference between the query and case attribute values, dv, should be initialised to any value bigger than the difference between the lowest and highest possible values of each attribute. This ensures that the difference between the attribute values of the query and the best match case is smaller than the initialised dv value. Therefore the conditional statement will be true at least once and sentences from the best match case will be the default in the assembled solution if better values are not found in the neighbourhood for any particular attribute. When the algorithm is applied for hotel review authoring, function *attribute* returns one of the five rating attributes (e.g. location) while $attributeValue$ returns an integer between 1 and 5 (e.g. 4) showing the rating value.

4.2.3 Compositional/ Constructive Text Reuse

Here, a textual solution is assembled in response to a query by combining chunks of text from several similar cases. Hence it is called compositional (COMP) or constructive text reuse because of its similarity to CBR's compositional (Chang et al. 2004, Bentebibel & Despres 2006) or constructive (Plaza & Arcos 2002) adaptation where a solution is obtained by combining solution elements of several partially similar cases. An important assumption for this approach is that we consider chunks of text to be contextually similar if they are aligned to the same problem attribute and have identical attribute values. For example in our hotel reviews domain, all chunks of text (sentences) aligned to a cleanliness rating of 3 are regarded as similar. These similar chunks of text can be extracted from the

query's k nearest neighbours (COMP_k) with $k \leq n$ where n is the size of the casebase. Therefore the maximum value of k here relates to using all cases in the casebase which we denote as COMP_N.

Algorithm 4.3 Algorithm to determine a Prototypical text chunk

Require: S= $\{s_1, \ldots, s_n\}$, similar text chunks for which a prototype is to be extracted
KW= $\{kw_1, \ldots, kw_m\}$, to contain set of keywords in all text chunks
for each $s_i \in$ S **do**
  keywords= $getKeywords(s_i)$
  add(keywords, KW)
end for
V= $\{v_1, \ldots, v_n\}$ where $length(v_i)$= $size$(KW)=m,
<!– to contain term frequency vectors for all text chunks –>
for each $s_i \in$ S **do**
  keywords= $getKeywords(s_i)$
  v_i= $createVector$(keywords, KW)
end for
cv= cv[1] ... cv[m] where $length$(cv)= $size$(KW), to contain centroid vector
for $j = 1$ to m **do**
  sum $\leftarrow$ 0
  for each $v_i \in$ V **do**
    sum $\leftarrow$ sum + v_i[j]
  end for
  cv[j] $\leftarrow$ sum $\div$ n
end for
maxSim $\leftarrow$ -1, similarity of best match to centroid
maxIndex $\leftarrow$ -1, index of best match to centroid
for each $v_i \in$ V **do**
  sim $\leftarrow$ $getSimilarity$(v_i, cv)
  if sim > maxSim **then**
    maxSim $\leftarrow$ sim
    maxIndex $\leftarrow$ i
  end if
end for
return $s_{maxIndex}$

We introduce a method that combines several similar chunks of text into a single meaningful and equivalent chunk of text named the prototype or prototypical chunk of text. Aggregating several pieces of similar text chunks into a single meaningful prototype is not trivial. In our methodology, concatenation will be inappropriate since it leads to tautology. This is because the text chunks will be expressing very similar or identical opinions about the same thing. Such concatenated text will be repetitive, unintuitive

and boring to a new author. Summarization methods would have been ideal for this task but they are generally applied to identify the central theme of textual contents from a single author. The central theme, which is what makes them similar, in our text chunks is already known, but is expressed using varying lexical forms by different authors. Also, summarization techniques will not be efficient, because deep natural language processing capabilities are typically required (Mitra, Singhal & Buckley 1997). The method we propose here uses the same idea as extractive summary generation (Sparck-Jones 1999, Neto et al. 2002), where a subset of the sentences of the original text are identified as the central theme. However, our text chunks already have identical central theme, but we select the subset (representative) whose syntactic construct is most generic and therefore easily reusable by other authors. Another similarity to extractive summaries is that our prototypes do not guarantee a good narrative coherence when assembled for different problem attributes, but they are sufficient as a starting point for feedback text generation, since the author can edit when required.

Algorithm 4.3 lists the pseudo-code for generating a prototypical text chunk. The algorithm takes a group of similar text chunks, S, and returns one of them as the representative or prototype. Prototypes are generated by first creating a term frequency vector (v_i) for each similar chunk of text; lines $7-9$ of the algorithm listing. The length of each vector is the size (m) of unique keywords in all similar text chunks for which a prototype is being determined. We then compute a centroid vector, cv, which consists of the average value across the cells representing the same keyword in each vector. Lines $11-18$ give details of how the centroid is computed. A prototype is determined as the chunk of text whose term vector is most similar to the centroid vector as indicated on lines $19-28$ of the algorithm. Intuitively, a prototype will contain the commonly used keywords across all similar chunks of text. This is because values of such keywords in the prototype's vector will be closer to the average. It should be noted that other term weighting functions, such as binary or normalised term frequency, can also be used to create the vectors.

The generation of prototypes or prototypical sentences for hotel review authoring is illustrated in Figure 4.3. Aligned sentences across the specified reviews (local or global)

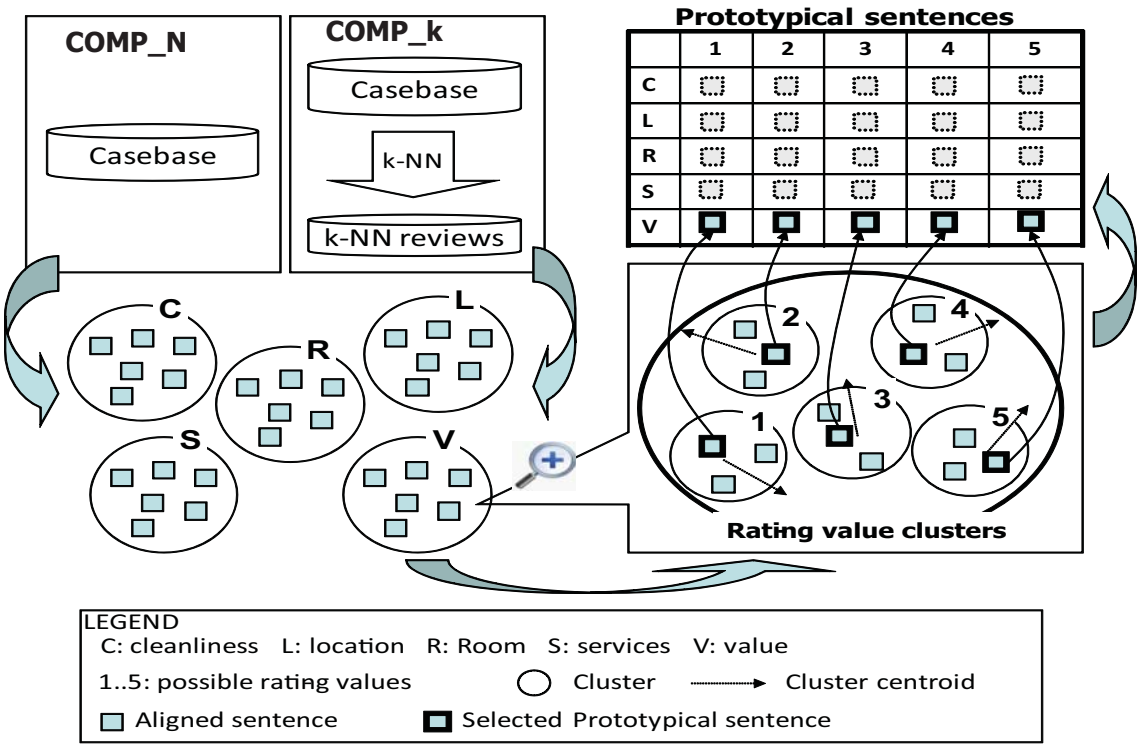


Figure 4.3: Generating prototypical sentences in hotel reviews

are grouped into five natural clusters which maps directly to the possible rating attributes. Each cluster is then further re-clustered into five groups using their rating value (i.e. 1 to 5). The smaller group of clusters shown for the value rating attribute also applies to the other four attributes. The outcome of this clustering process is twenty-five smaller clusters and a prototypical sentence per cluster.

Algorithm 4.4 shows the compositional text reuse algorithm. This uses the same convention as the baseline and transformational techniques listed in Algorithms 4.1 and 4.2 respectively; that is, CB represents the casebase, Q is a query and C_i is a case in the casebase. This approach is then illustrated in Figure 4.4 with our hotel review authoring domain where the query has five attribute values (p in algorithm): $2, 1, 3, 5, 2$ for cleanliness, location, room, service and value ratings respectively. Five sentences are then obtained from the prototypical sentences with identical rating values to the query and aggregated as proposed text ($SOLN$). In the algorithm, each prototypical sentence is generated from an element in the matrix (G) having $p \times q$ elements where each element is a cluster of similar text chunks. The pseudo-code for determining a prototype in our

Algorithm 4.4 Compositional text reuse algorithm (COMP)

1:

$$G = \left\{ \begin{array}{l} g_{11}, \ldots, g_{1q} \\ g_{21}, \ldots, g_{2q} \\ \ldots \\ g_{p1}, \ldots, g_{pq} \end{array} \right\}, \quad \begin{array}{l} \text{set of clustered similar text chunks;} \\ \text{each cluster belongs to a pair} \\ \text{from } p \text{ problem attributes and } q \text{ attribute values} \end{array}$$

2: $\text{CB}_{local} \leftarrow RET(\text{CB}, Q, k)$
retrieve k similar cases; for COMP_N, $k \leftarrow n$ the size of CB
3: $SOLN = \{\}$,
<!– to contain text chunks in the proposed solution text –>
4: **for each** $C_i \in \text{CB}_{local}$ (in order of decreasing similarity) **do**
5: SolutionText$\leftarrow getSolutionText(C_i)$
6: **for each** $\text{IE}_j \in C_i$ **do**
7: $r_j \leftarrow attribute(\text{IE}_j)$
8: $v_j \leftarrow attributeValue(\text{IE}_j)$
9: $g_j \leftarrow getClusteredSimilarTextChunks(G, r_j, v_j)$
<!– $g_j \equiv g_{r_j v_j}$ –>
10: $S_j \leftarrow selectAlignedTextChunks(r_j,$ SolutionText$)$
11: $addTextChunks(S_j, g_j)$
12: **end for**
13: **end for**
14: **for each** $\text{IE}_k \in Q$ **do**
15: $r_k \leftarrow attribute(\text{IE}_k)$
16: $v_k \leftarrow attributeValue(\text{IE}_k)$
17: $g_k \leftarrow getClusteredSimilarTextChunks(G, r_k, v_k)$
18: $ps_k \leftarrow getPrototypicalTextChunk(g_k)$
19: $addTextChunks(ps_k, SOLN)$
20: **end for**
21: Aggregate all chunks of text in $SOLN$ for reuse

context is listed in Algorithm 4.3 and has been explained earlier. Lines $4 - 13$ of Algorithm 4.4 show how cluster of similar text chunks are selected from cases in the query's neighbourhood, while lines $14 - 20$ use the clusters to assemble a solution composed of prototypical text chunks. A major difference between COMP_k that uses solution texts from neighbours and COMP_N that uses all solution texts in the casebase is that COMP_k might propose less than p text chunks in its solution if no prototypes are generated for any attribute value in the query. This can occur when k is small and there are no cases in the neighbourhood with this particular attribute value.

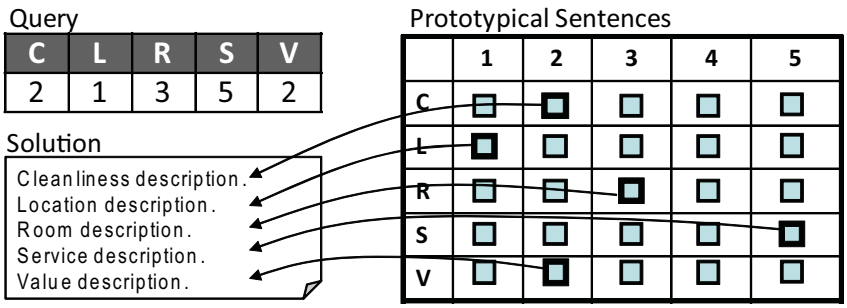


Figure 4.4: Compositional text reuse with hotel reviews

4.3 Chapter Summary

This chapter introduced two novel concepts in relation to text reuse: text alignment and prototypical text generation. Text alignment links structured problem attributes to specific chunks of a solution text while prototypical text generation abstracts similar chunks of text into a single meaningful prototype. These concepts are generally applicable in domains where cases consists of pre-defined structured attributes along with written text. We then propose two novel text reuse techniques and a third retrieve-only baseline that generate proposed solution texts related to the problem attributes. Transformational text reuse progressively changes the solution text from the best match case into a more accurate solution, using texts from other nearest neighbours when the query and its best match are not identical. Compositional text reuse on the other hand, proposes a solution text constructed by aggregating chunks of text from several similar cases. Although hotel review authoring was used to illustrate our text reuse techniques throughout this chapter, the formalised algorithms have the advantage of being domain-independent and therefore applicable in any domain containing cases with both pre-defined structured attributes and complementary textual content.

Chapter 5

Evaluating TCBR with Machine Translation Techniques

Evaluation is the key to measuring the capabilities, effectiveness and efficiency of any proposed technique or algorithm. Although qualitative evaluation is generally regarded as the best method of evaluation, automated evaluation methods where user intervention is minimal, are preferable for initial and intermediate testing during development of any new technique. This is because user evaluations are more expensive and repetition of such evaluations with different users with similar expertise may vary greatly due to subjective user judgements. Therefore automated evaluation methods are more commonly used for research experiments to allow for inexpensive, faster testing and comparison to current state-of-the-art techniques. Automated evaluation methods are typically encoded as quantitative metrics which give single numeric values for the evaluation of each test situation.

Our research work proposes novel techniques to aid reuse of textual solutions in Textual Case Based Reasoning (TCBR). This necessitates that we automatically evaluate written natural language (text) proposed or generated by our techniques for its syntactic and semantic correctness. The need for automated text evaluation is not peculiar to TCBR but common to several other disciplines such as Information Retrieval (IR) (Lenz 1998b, Baeza-Yates & Ribeiro-Neto 1999), Text Summarization (Neto et al. 2002, Lin & Hovy 2003),

Natural Language Generation (Sripada et al. 2005, Belz 2005) and Machine Translation (White & Connell 1994, Hovy 1999). A common feature of these research areas is that they all propose or generate textual content as output or as part of their final output.

In this chapter, we propose the use of Machine Translation (MT) evaluation measures for TCBR. Both research areas (MT and TCBR) typically propose textual solutions and rely on human reference texts for automated evaluation purposes. Section 5.1 discusses the current TCBR evaluation metrics such as precision and recall (adapted from IR) which employ a single human reference. It then explains the drawbacks of these metrics when semantically similar texts are expressed with different sets of keywords. MT metrics overcome this drawback mainly with the use of multiple human references. Two MT evaluation metrics analysed in Section 5.2 are proposed to be adapted for automated TCBR evaluation. However to reap full benefits of these MT evaluation metrics, multiple references are required as opposed to a single reference text. Here, we discuss how multiple references can be created introspectively from TCBR datasets by exploiting the CBR similarity assumption. Section 5.3 presents this method of creating multiple references and other factors that need to be considered when MT measures are utilised for TCBR evaluation. A summary of the main contributions of this chapter appears in Section 5.4.

5.1 Evaluating Textual Solutions in TCBR

TCBR deals with reusing past experience stored mainly in the form of textual documents such as incident reports, frequently asked questions (FAQ) and emails. The minimum requirement for a TCBR based application is that at least one of its components (problem or solution) must have a textual attribute. All the solution attributes in a TCBR system can be structured which means that traditional CBR evaluation metrics such as accuracy can be utilised directly during the evaluation phase. Evaluation measures for traditional CBR systems with structured attributes will not apply directly to textual solutions due to the nature of textual attributes as discussed in Section 1.1.3. Also, text with identical semantic meanings might be written differently by experts. Therefore, qualitative user evaluation is generally accepted as the best way to evaluate any piece of text. However

due to the relatively high cost of human user evaluation, automated evaluation is desirable where a human expert would be expected to provide a single reference (actual) solution. These reference texts are then used as gold standard to evaluate any solution texts proposed by TCBR systems. In this section, we examine some of the common TCBR evaluation measures and identify scenarios (from our medical health and safety incident reporting domain) where they might be misleading. These examples are discussed in the context of challenges and drawbacks with the current TCBR evaluation measures.

5.1.1 Current text evaluation measures in TCBR

Precision and Recall are the most common measures for evaluating textual solutions in TCBR (Brüninghaus & Ashley 1998a, Lamontagne & Lapalme 2004, Asiimwe 2009). Both measures are adapted from IR and measure the goodness of a proposed textual solution by comparing it with a reference solution text. The reference is typically the actual solution of a test case in experimental designs such as cross-validation, hold-out and leave-one-out. Precision, sometimes synonymous with accuracy, in its simplest form is measured as a ratio of the number of terms from the reference solution text that occur in the proposed TCBR solution to all terms in the proposed solution text. Recall (also known as coverage) on the other hand is the ratio of the number of common terms in the reference and proposed solution texts to all terms in the reference solution text.

However, weighted precision and recall can also be computed if different weights are assigned to terms based on their relative importance in the textual solution. Such weights can come from domain knowledge or be inferred from weighting schemes such as inverse document frequency, which gives more weight to terms that occur less frequently across all cases. Similarity of terms might also be used for precision/recall computation when terms represent higher levels of text granularity than keywords (e.g. phrase or sentence level). Here, each term in the proposed text is matched to its most similar term in the reference solution without replacement. The precision/recall computation can be simplified further by assuming that terms (phrases or sentences) are identical if their similarity exceeds a specified threshold.

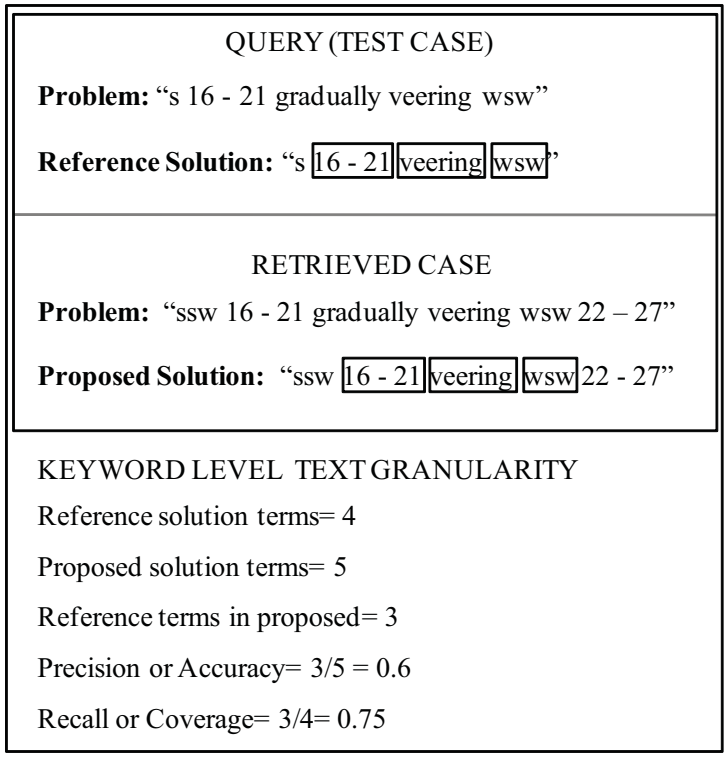


Figure 5.1: Example of TCBR precision & recall computation on a weather test case

Figure 5.1 contains snippets from our weather forecast revision dataset to illustrate how precision and recall are typically computed in TCBR. Here, the text granularity is at the keyword level, although number intervals, where two numbers have a dash between them (e.g. 16-21), are tokenized as a single term. The reference and proposed texts therefore have 4 and 5 terms respectively. The number of common terms between the proposed and reference text is 3, (i.e. terms '16-21', 'veering' and 'wsw') giving a precision of 0.6 and a recall of 0.75. It can be seen from the figure that such an evaluation though simplistic can provide good, accurate and reliable results.

5.1.2 Challenges with current TCBR text evaluation measures

The major drawback of IR type evaluation measures is that the sequence and position of terms are not taken into account, nor is the context for terms that could have several meanings. For example, 'plate' can refer to a 'food plate' or 'plate kit' both applicable in the medical incident reports. However, these measures work best on domains with

smaller solution vocabularies such as the weather forecast revision. This is because a large vocabulary size is more likely to contain semantically similar texts with fewer keywords in common. Precision/Recall evaluation will also be suitable for domains where there is a standard template that must be adhered to when writing or generating textual solutions thereby restricting grammatical variation across different authors. In other words, precision and recall results are less reliable if there is a large degree of variation in the way different authors express text with similar meanings. The effect of such grammatical variations can be reduced by using the root of keywords obtained from lemmatization and/or stemming. Synonyms and specialized/generalized equivalents of terms can also be looked up from domain glossaries, thesauri or ontologies to minimize the effect of grammatical variations. Nevertheless, precision/recall results which use lemmas, stems and/or synonyms can still be misleading in domains where grammatical variations occur at the phrase or sentence level since these methods only minimise word variation at the keyword level. Therefore, they are unable to capture variation in phrases/sentences that have similar meanings but expressed with different choice of words. Our health and safety (H&S) incident reporting domain is an example of such a domain.

Table 5.1: Sample retrievals from the H&S dataset

	Query	**Retrieved Problem**	**Simi larity**	**Retrieved Solution**	**Reference Solution**	**Preci sion**
1	nurse slipt and fell on wet floor	staff member slid on something wet and fell to the floor	0.612	examined by nursing staff	nurse given first aid	0.333
2	patient fell to the ground as nurse assisted him to bed	patient fell out of bed	0.707	examined by medical staff	patient was advised to get assistance in and out of bed	0.0
3	needlestick injury sustained	needlestick injury sustained by a member of staff	0.775	first aid, blood sample taken, visited occupational health	occupational health contacted	0.333

We illustrate this phrase/sentence level variation challenge which might affect TCBR

evaluation with examples from our H&S dataset shown in Table 5.1. Here, three incident queries along with the retrieved best match case (problem and solution), retrieval similarity value, reference solution and precision results are shown. With query 1, although the retrieved and reference solutions are similar in meaning, retrieval precision or accuracy is just 0.333. This is because one out of the three keywords ("nurse/nursing") is matched in the retrieved solution and the remaining keywords though semantically similar are lexically different. Query 3 poses a similar challenge as query 1 while query 2 highlights a slightly different problem. Here, the level of detail/abstraction in the reference solution is different from the retrieved solution thereby causing accuracy to be computed as zero (0.0). Therefore the precision results obtained in this domain can be misleading. Retrieved solutions judged as correct and applicable to the queries by a human will seem less applicable from values obtained with these basic metrics.

Capturing semantic similarity is key to a good text evaluation metric. Machine Translation (MT) evaluation measures are therefore proposed for evaluation of textual solutions in TCBR since they both produce textual contents as part of their output. However, evaluation of machine translated text must attempt to capture semantic meaning as well as differences in word choice and order (grammatical/ semantic variations) since these are more common with language translations. We expect that MT evaluation will give more reliable and accurate results especially for TCBR domains with large grammatical variations across various authors.

5.2 Machine Translation Evaluation techniques

Machine Translation (MT) is a research area that deals with techniques to enable automated translation from one language to another. There are MT techniques which are similar to those used in TCBR. For instance, Example Based Machine Translation (EBMT) (Brown 1996, Zhang et al. 2001) uses the same similarity assumption as the foundation of its methodology; this was discussed as part of the substitutional text reuse techniques in Section 2.2.3. MT techniques have also been proposed to improve the retrieval stage of TCBR (Lamontagne, Langlais & Lapalme 2003). The output of MT systems is typically

a textual translation in a specified language of a given piece of text written in another language. There is therefore a need to evaluate such machine generated translations for grammatical and semantic correctness. This is identical to the need for evaluating textual contents proposed by a TCBR system assuming the translation is from any other language to English language. The focus in this chapter is on exploring the use of MT evaluation metrics for TCBR.

Initial research in MT used human expert translators for evaluating several aspects of a translated text in terms of adequate coverage, semantic meaning and grammatical correctness (White & Connell 1994, Hovy 1999). However, more recent work (Papineni, Roukos, Ward & Zhu 2002, Doddington 2002) has reduced the demand for user-driven quality assessments by developing automated text comparison techniques with high correlation to human judgements. These automated MT evaluation techniques are quick, inexpensive, language independent and repeatable.

In this section, we look at the details of two de-facto standard MT evaluation measures, BLEU and NIST. We chose these because they are highly regarded within not just the MT community but in other related areas such as Text Summarization (Lin & Hovy 2003) and Natural Language Generation (Belz 2005, Belz & Reiter 2006). They are also the basis of other MT measures, e.g. (Babych & Hartley 2004), and evaluation measures such as ROUGE (Lin & Hovy 2003) in Text Summarization. BLEU and NIST are still generally more acceptable and more widely in use than other newer MT evaluation measures like Translation Error Rate (TER) with its Targeted Human Annotation (HTER) version (Snover, Dorr, Schwartz, Micciulla & Makhoul 2006) and METEOR (Banerjee & Lavie 2005, Lavie & Agarwal 2007, Lavie & Denkowski 2009).

5.2.1 BLEU

BLEU (Papineni et al. 2002) (**Bi**lingual **E**valuation **U**nderstudy) is an automated MT evaluation technique and was used as an understudy of skilled human judges in translation. The idea is to measure the closeness of a machine text to its human equivalent using weighted average of phrases matched with variable length (n-grams). It enables the use of

multiple reference solutions from different experts and this allows for legitimate differences in word choice and sequence. The BLEU score is a precision-based metric which can use multiple reference solutions and aggregates the precision scores from different word lengths; this concept is known as modified n-gram precision. BLEU also ensures that the machine text's length is comparable to at least one of the reference solutions using a brevity penalty.

Modified n-gram precision matches position-independent n-grams; where $n \geq 1$ and each uni-gram (1-gram) is typically a keyword but can also be a stand-alone special character or punctuation. This is similar to precision measure in TCBR. However, it is modified to ensure that n-grams can be matched across multiple reference solutions. Each n-gram is matched to the reference solution with the maximum count for the n-gram. The overall precision is a geometric average of all individual n-gram precisions from 1 to N. Using $N = 4$ has been found to give the best correlation to human judgements (Papineni et al. 2002). In comparison to the criteria used in human evaluation, uni-gram precision (i.e. $n = 1$) measures adequate coverage of a machine text while n-gram precision (when $n > 1$) shows grammatical correctness.

Brevity Penalty (BP) on the other hand ensures that the length of machine text is penalized if it is shorter than all the reference solutions. This is because a machine text of shorter length might have a very high n-gram precision if most of its keywords occur in any of the reference solutions. Therefore, modified n-gram precision alone fails to enforce proper translation length. BP focuses solely on penalizing shorter machine texts as unnecessarily long texts will have been penalized by the modified n-gram precision. Although recall has been combined with precision to overcome problems with text lengths to give measures like f-measure (Baeza-Yates & Ribeiro-Neto 1999), it cannot be used in BLEU because it employs the use of multiple references and each reference might use different word choices and order. Also, recalling all choices gives a bad translation since a good translation will only use one of the possible choices. BP is formulated as a decaying exponential function which gives a value of 1 when machine text's length is at least identical to any of the reference solutions length otherwise it gives a value less than 1. The BLEU

metric is calculated as follows.

$$BLEU = BP \cdot \exp(\frac{1}{N}\sum_{n=1}^{N} \log p_n) \tag{5.1}$$

$$p_n = \frac{\sum_i \left(\begin{array}{c} \text{\# of } n\text{-grams in segment } i \text{ of machine text} \\ \text{matched in segment } i \text{ of any of the reference solutions} \end{array} \right)}{\sum_i (\text{\# of } n\text{-grams in segment } i \text{ of machine text})}$$

$$BP = \begin{cases} 1 & \text{if } l_{sys} > l^*_{ref} \\ \exp^{(1-\frac{l^*_{ref}}{l_{sys}})} & \text{if } l_{sys} \leq l^*_{ref} \end{cases}$$

where

$p_n = n$-gram precision $\qquad BP =$ brevity penalty

$l_{sys} =$ length of machine text $\qquad i = 1$ for TCBR

$l^*_{ref} =$ nearest reference solution length to machine text

$N =$ maximum size of n-gram (i.e. $n = 1 \ldots N$)

It is important to note that the entire text is typically regarded as one segment in TCBR (i.e. $i = 1$) when calculating precision, p_n. This is because there is usually no knowledge of aligned segments between proposed and reference texts unlike MT where translations are done segment by segment. Figures 5.2 and 5.3 show a sample retrieval from our H&S dataset and its corresponding BLEU calculation. Here, we compare the proposed solution with the three reference solutions and show sample BLEU calculation for both single and multiple references as well as when N= 1 and N=2 in Figure 5.3(a)-(d).

Precision with a single reference solution (say Ref Solution1) when N=1 matches only keywords "nursing" and "staff" from the machine text resulting in a precision of 0.6 as shown in Figure 5.3(a). However, the keyword "examined" is also matched when multiple reference solutions are in use; see Figure 5.3(b). A bigger, more accurate and reliable BLEU score of 1.0 is obtained with multiple references. The expectation is that this value from the use of multiple references will correlate better with human judgements since the meaning of the proposed solution is identical to the three reference solutions. Figures 5.3(c) and (d) also repeats the BLEU calculation for N=2. Here, just one ("nursing staff") of

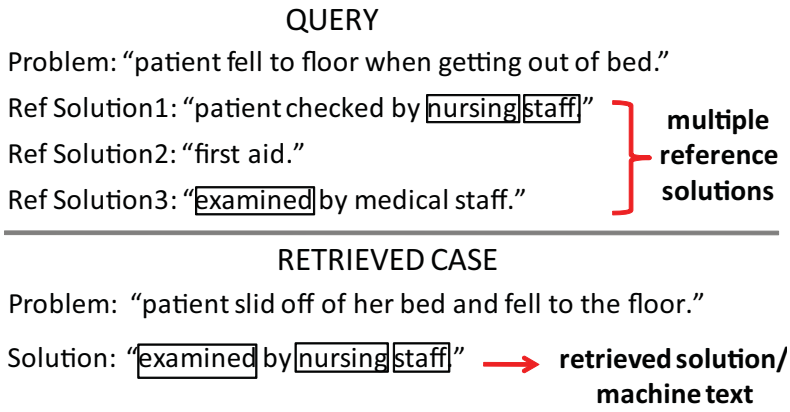


Figure 5.2: A test case with multiple reference solutions

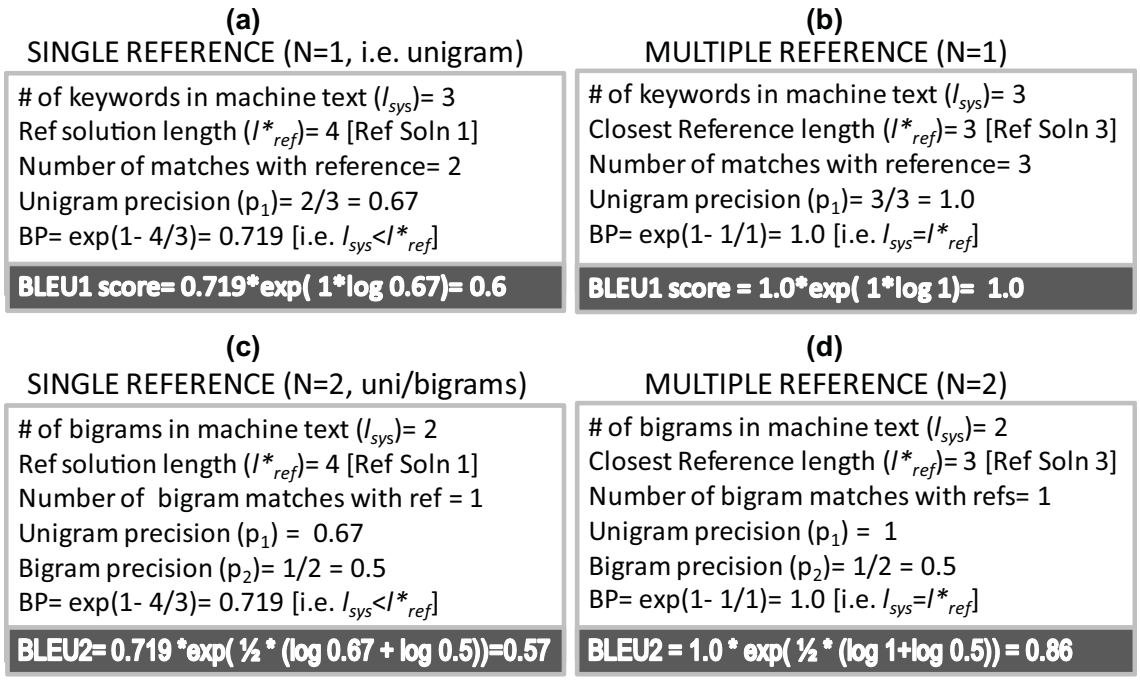


Figure 5.3: Sample BLEU calculation with H&S dataset

the two bi-grams ("examined nursing" and "nursing staff") is matched in the reference solutions (single or multiple). The bi-gram precision and previously calculated uni-gram precision are then used to calculate the BLEU2 score. Notice that the brevity penalty (BP) value remains the same since the length of the solutions (reference and proposed) are based only on counts of keywords. Also, the BLEU2 scores are less than their BLEU1 scores since BLEU uses an arithmetic average and the number of n-gram matches typically decreases with increasing n. Another important observation is that the same uni-gram

precision can be obtained if the multiple references are concatenated together. However, this will not make sense for $n > 1$ because n-grams created from the end of one reference and the beginning of another might be incorrectly matched during precision calculation.

5.2.2 NIST n-gram co-occurrence statistics

NIST n-gram co-occurrence statistics (Doddington 2002) is a more sophisticated MT evaluation measure that builds on the BLEU idea by modifying the weighting scheme for calculating precision. This is done by using information weights rather than frequency of occurrence and an arithmetic average of n-gram weights as opposed to geometric mean of n-gram precisions. Information weights are computed for n-grams such that those that occur less frequently have higher weights as they are deemed to be more informative. In addition, brevity penalty was modified to minimize the impact of small variations in the generated text's length, as they do not generally affect human judgements. A significant improvement in stability and reliability was reported with NIST when compared with BLEU (Doddington 2002). In other words, NIST is less sensitive to variation in the level of human expertise. Its correlation to human judgement is also more consistent across corpora from different languages. The NIST formula is given below.

$$NIST = BP \cdot \sum_{n=1}^{N} \left\{ \frac{\sum \forall n\text{-gram} \in \text{sys}\, \text{info}(n\text{-gram})}{\#\text{ of } n\text{-grams in machine text}} \right\} \tag{5.2}$$

$$BP = \begin{cases} 1 & \text{if } l_{sys} > \overline{l}_{ref} \\ \exp\left\{ \beta \log^2 \left[min\left(\frac{l_{sys}}{\overline{l}_{ref}}, 1 \right) \right] \right\} & \text{if } l_{sys} \leq \overline{l}_{ref} \end{cases}$$

$$\begin{aligned} \text{info}(n\text{-gram}) &= \text{info}(w_1 \ldots w_n) \\ &= \log_2 \left(\frac{\#\text{ of } w_1 \ldots w_{n-1} \text{ in reference solutions}}{\#\text{ of } w_1 \ldots w_n \text{ in reference solutions}} \right) \end{aligned}$$

where

w= a word in the machine text

info= information weight

N = maximum size of n-gram (i.e. $n = 1 \ldots N$)

$\beta = -4.3218$, chosen such that BP=0.5 when $l_{sys}/\bar{l}_{ref}$=2/3

l_{sys} = number of words in machine text (sys)

$\bar{l}_{ref}$ = average number of words in reference solutions

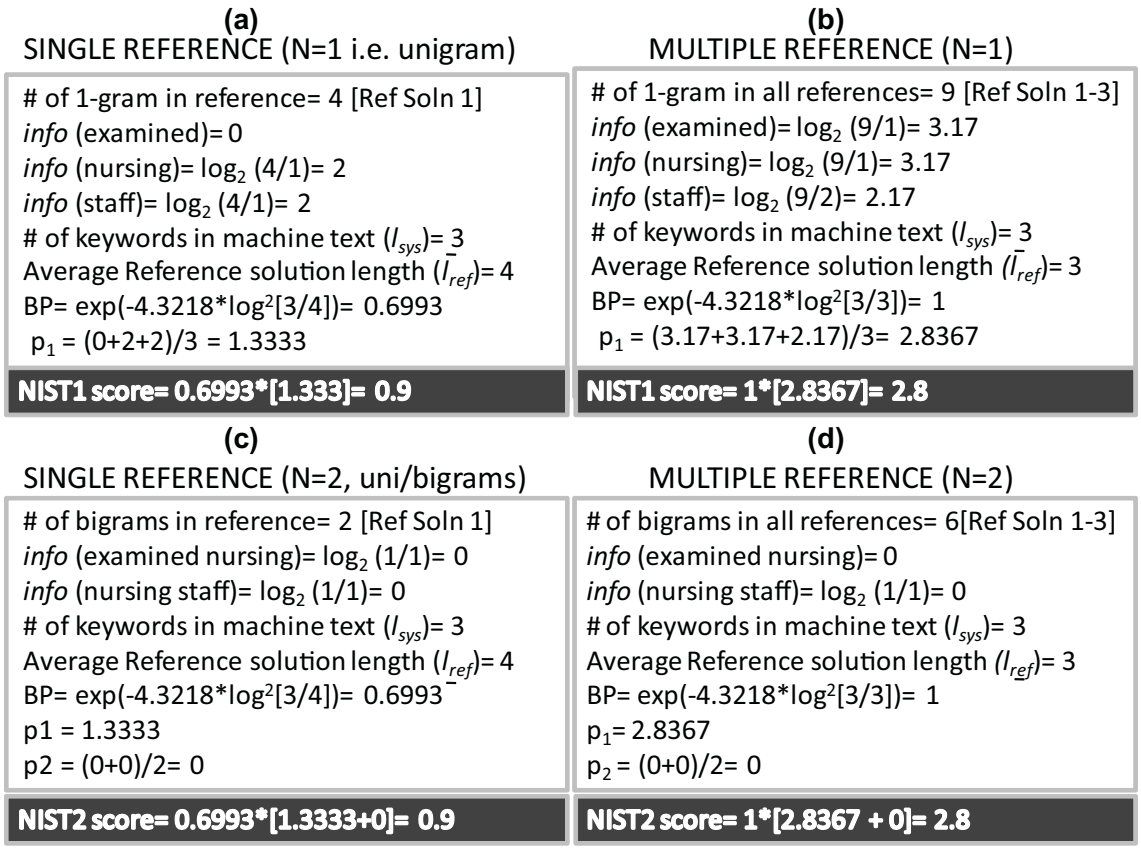


Figure 5.4: Sample NIST calculation with H&S dataset

A sample NIST calculation which also uses the example test case from our H&S dataset (see Figure 5.2) is shown in Figure 5.4. We show sample calculation for N=1 and N=2 as well as single/multiple references. NIST penalizes shorter machine text more when compared with BLEU since it uses the average reference length rather than closest reference length used in BLEU. For instance, its brevity penalty value of 0.6993 is lesser than

BLEU's 0.719 for a single reference from Figures 5.4(a) and 5.3(a).

Information weights are typically calculated using all reference solutions in the training data. We therefore assume that the reference solution or solutions (for multiple references) in Figure 5.2 are the only ones available to simplify our illustration in Figure 5.4. To calculate 1-grams' information weights, each 1-gram is assumed to be preceded by a common 1-gram (e.g. blank space). Therefore the numerator in the logarithmic function equals the total number of 1-grams in the references; see Figures 5.4(a) and (b). Consequently, the information weight for 'staff' when using multiple references is $\log_2(9/2)$ since there are nine 1-grams (with 'by' as a stop word) in the three references with 'staff' occurring twice. The use of information weights leads to NIST scores that can be greater than 1 (e.g. 2.8 in Figure 5.4(b)) as opposed to BLEU's which are always between 0 and 1. Generally, the larger the NIST (or BLEU) scores then the more accurate the proposed/generated text and the more likelihood of a higher correlation to human judgements. Therefore, the NIST value of 2.8 obtained with the use of multiple references in the sample calculation is more accurate than 0.9 from a single reference. The NIST2 values obtained in Figure 5.4(c) and (d) are identical to their corresponding NIST1 values because the bi-grams in the proposed solution have zero information weights. Again, this is based on our assumption in this illustration that the training data contains only the references shown in Figure 5.2. However, it is more likely that these bi-grams will have non-zero information weights if all reference solutions from the training data is used. NIST values should generally increase with increasing N if bigger n-grams ($n > 1$) in the proposed solution are found to have non-zero information weights. This is in contrast to BLEU which decreases with increasing N since the number of n-gram matches typically reduces as n grows bigger.

5.3 Applying MT Evaluation techniques to TCBR

The application of MT evaluation techniques to measuring the goodness of textual solutions in TCBR should be relatively easy and straight-forward as both research areas produce textual contents as a major part of their output. The main advantage of using MT measures is that TCBR evaluation results will be more accurate, reliable and

comparable to those from other text related research areas.

Nevertheless, differences exist between MT and TCBR in their methods of text generation, structure of output text and number of available reference texts used for automated evaluation. For instance, TCBR typically produces new textual output by modifying and adapting solution text from a case or cases with similar problem to the current problem. This is unlike MT which typically generates textual outputs from scratch using mathematical models. The structure of MT output texts are also well defined as translations are done on a segment by segment basis unlike TCBR. A segment usually represents a group of text that has a specific context and meaning such as a sentence. Although TCBR textual solutions can also be divided into meaningful segments similar to those used in MT, such segments cannot be matched to specific segments in a reference text as the alignment of segments are not usually unknown in TCBR domains. This differs from MT where each segment in the generated text can be directly aligned to a specific segment in the reference text. Therefore, the entire text in a TCBR scenario will be regarded as a single segment when MT measures are used for evaluation. Lastly in terms of difference between MT and TCBR, several reference texts are typically available for each segment when evaluating machine generated texts in MT. The use of multiple reference texts is a major factor in the reliability and acceptability of results from MT evaluation measures because they take care of the variability in vocabulary across different experts. Such variability can come in terms of word choice and/or sequence as a piece of text can be written with a slightly or totally different set of words without losing its meaning. This is intuitive and similar to the use of more than one human judge during a user evaluation. To the best of our knowledge, human judges have been used in CBR but the use multiple references has never been exploited in TCBR. This might be partly due to the demand on manual effort needed to generate multiple references per case.

We explore how multiple references might be created introspectively. A requirement of our proposed method is the existence of several identical or very nearly identical problems with solutions that are lexically different in the domain. Our medical health and safety incident report domain is used to illustrate the method as it meets this requirement. We

then discuss other factors that need to be considered when MT measures are used for TCBR evaluation.

5.3.1 Introspective creation of multiple reference texts in TCBR

The CBR similarity assumption states that "similar problems have similar solutions". This implies that identical problems should have solutions with identical meanings even though the solutions, when textual, might be written using a different set of words. We refer to this as the identity assumption. Although identical textual solutions with different lexical structure lead to duplicate cases in the casebase, the difference in syntax can be exploited for better text evaluation. This is the basis of our introspective approach to create multiple reference texts for experiment purposes in TCBR domains where only single reference text are available for each problem but in which the identity assumption holds. A new dataset can be created from the original dataset where test cases will have multiple solutions and cases with single solutions will be used in the training set. However, the new dataset can only be used effectively in a hold-out experimental design since the test and training cases are already separated.

A new dataset is created by designing a leave one out test where each case is used as a query to retrieve its nearest neighbours. Textual solutions from neighbours with identical problems are then selected to form multiple reference texts for each case. Here, reference solutions are kept only if they are lexically different. The notion of identical problems can be relaxed to a lower similarity threshold (e.g. 0.9 instead of 1.0) especially if the text similarity measure does not return a similarity of 1 though the texts are adjudged semantically identical by a human.

An illustration of our proposed method for creating test cases with multiple reference texts is shown in Figure 5.5. Here, a case consisting of *P1* and *S1* in the figure is left out of the original casebase and used to create a test case by retrieving cases with identical (or nearly identical) problems. Two problems (P2 & P3) are found to be identical and therefore their solutions (S2 & S3) are added to form multiple reference solution texts for problem P1. These two cases will be removed from the training set of the newly created

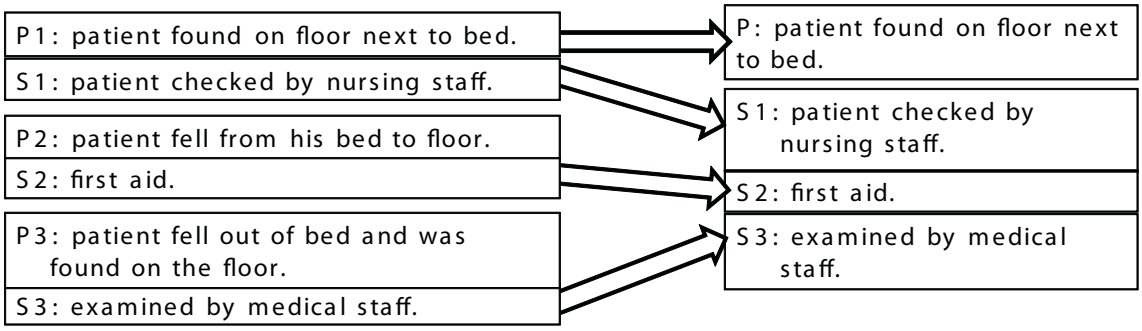


Figure 5.5: Introspective creation of multiple reference texts with H&S dataset

dataset. When investigating the effect of using multiple references as opposed to single references, these cases should also be added to the test set for single references. This ensures that the evaluation results obtained will take all three solutions into account. However, this is not required for the test set with multiple references because all reference texts in the new test case are treated as equals.

We hypothesize that datasets having multiple reference solution texts created by our proposed approach will give better, more accurate and reliable evaluation results from hold-out experiments than the original dataset with single reference texts. The main limitation of this approach is that the identity assumption must apply to the original dataset. Further discussion on the introspective approach to creation multiple reference texts can be found in Section 6.4 where experiments are designed to verify our hypothesis as well as a critical analysis of results.

5.3.2 Factors to consider when MT measures are applied to TCBR

The differences between TCBR and MT are mainly in terms of the mechanism for text generation, structure of textual output and the number of reference texts used for automated evaluation. We expect that these factors would influence how MT measures are used for TCBR evaluation. We address the difference between these two research areas as regards the number of reference text used for automated evaluation with our approach to creating multiple references. Here, we consider the effects of the other two differences (i.e. mechanism for text generation and structure of textual output) by identifying parameters in MT measures that are constant or might not be useful for TCBR.

One of such parameters is the brevity penalty (BP) captured by both BLEU and NIST. BP is used to ensure that generated texts that are relatively short compared to the reference texts do not have any undue advantage. This is because textual outputs can be generated from scratch in MT and some algorithms might decide to produce texts for only particular words during translation thereby giving them a high precision. However, TCBR textual outputs (solutions) are typically adapted versions of retrieved texts whose problem components are similar to the current problem. TCBR solution texts are therefore unlikely to differ largely from references in length if variation in word choice is properly captured with the use of multiple references.

Another factor to consider is the optimal value of N, maximum size of n-gram used during evaluation. TCBR solution texts are sometimes short (e.g. 1 sentence long) and might therefore not contain enough n-grams to allow for evaluation at the optimal values observed in MT evaluation (4 for BLEU and 5 for NIST). We expect that optimal values of N in such TCBR domains will be smaller, e.g. $N = 2$.

5.4 Chapter Summary

The use of MT evaluation metrics to evaluate quality of proposed textual solutions in TCBR is the main contribution of this chapter. Two MT metrics, BLEU and NIST, are adapted for TCBR evaluation with multiple reference solutions. We also propose a novel introspective approach to generate multiple references when they do not naturally occur in a domain. Our approach is applicable to domains where the textual solutions of identical (or nearly identical) problems are written in lexically different forms. It should also work well with structured problems that have very limited attributes values in which case the likelihood of identical problems occurring in the casebase is higher. The casebase should also be relatively large since identical cases are merged into one thereby resulting in a casebase with a smaller size than the original. Multiple references should reduce the effect of different writing styles or variations in word choice on text evaluation. They should therefore give more reliable and accurate results that correlate better with human judgements.

We then considered the effect of the intrinsic differences in the text generation process between TCBR and MT and how these might influence the application of MT measures during TCBR evaluation. We hypothesize that certain parameters such as brevity penalty might not be very important for TCBR. This is because the proposed text in TCBR is typically similar in length to the reference since it is an adapted form of the actual solution to a similar problem, unlike MT where generated texts can sometimes be significantly shorter. Also the optimal value of N (maximum size of n-gram used) in TCBR might be smaller than that used in MT (4 for BLEU and 5 for NIST) since TCBR solution texts can be just one or two sentences.

Chapter 6

Evaluation: What to Reuse?

This chapter gives details of experimental set-up, methodology and analysis of evaluation results that address our research question of what to reuse. In particular, we discuss results related to our text reuse architecture and explore the use of machine translation evaluation metrics presented in Chapters 3 and 5 respectively. Analysis of the experimental results across different domains each with its peculiar data characteristics also provide better insight into the generic nature of our proposed techniques and some of their limitations.

We discuss our evaluation methodology in Section 6.1. Experiments related to our contribution on learning what to reuse are described in Section 6.2 where we examine the capability of the Case Retrieval Reuse Nets (CR2N) to predict each term in a retrieved solution as reuse or adapt. Section 6.3 reports the evaluation of our reuse guided retrieval technique to test our hypothesis that the solution from another nearest neighbour might be easier to adapt than the best match. We then obtain evidence that applying MT evaluation measures to TCBR is appropriate in Section 6.4 before concluding the chapter with a summary in Section 6.5.

6.1 Methodology

Evaluation remains a challenge for TCBR especially where the solution is textual. This is mainly due to the high cost of user evaluation and the inadequacies of current automated TCBR evaluation metrics (Precision and Recall). Extensive analysis and discussion of

current TCBR evaluation measures were given earlier in Chapter 5. We also proposed the use of two text evaluation measures, BLEU and NIST, in TCBR. Although these were introduced for automated evaluation of machine translation (MT) text, they (or their variant) have been used for text evaluation in other research areas like text summarization and natural language generation due to their high correlation to human judgements. Here, we use the standard TCBR measures of Precision/Recall and their combination called F-measure (with equal weighting for precision/recall) as well as the MT metrics of BLEU and NIST. We compute accuracy results where appropriate and also record evaluation results with cosine coefficient similarity. Although cosine coefficient is typically employed to measure similarity between two pieces of text, we used it for empirical text evaluation as it gives a single value that represents how close a proposed solution text is to the actual text rather than precision and recall where two different values are obtained. It also provides a better estimate of the effectiveness of some techniques like the CR2N where recall is often worse for reuse than for retrieval because terms annotated as 'adapt' are currently deleted but not replaced with other terms. Therefore combining precision and recall into a metric like f-measure gives the retrieval baseline an undue advantage since it will always have a higher recall. The evaluation metrics of precision, recall, f-measure and cosine similarity are defined in Equations 6.1, 6.2, 6.3 and 6.4 respectively. The BLEU and NIST were given earlier in Chapter 5 (see Equations 5.1 and 5.2) while accuracy and reuse precision/ recall are defined in Equations 6.5, 6.6 and 6.7 shown in Section 6.2 as they are used primarily in experiments related to our reuse annotation techniques.

$$\text{Precision} = \frac{|\text{Proposed Solution} \bigcap \text{Reference Solution}|}{|\text{Reference Solution}|} \tag{6.1}$$

$$\text{Recall} = \frac{|\text{Proposed Solution} \bigcap \text{Reference Solution}|}{|\text{Proposed Solution}|} \tag{6.2}$$

$$\text{F-measure} = \frac{\text{Precision} * \text{Recall}}{\alpha * \text{Recall} + (1 - \alpha) * \text{Precision}} \tag{6.3}$$

$$\text{Cosine Similarity} = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}} \tag{6.4}$$

where,

A, B = term frequency vector representations of the proposed and reference solutions.

Detailed description of the datasets available for our experiments has been given in Section 2.4. In this chapter, we discuss their suitability for testing the different techniques introduced in the previous chapters. It should be emphasized here that BLEU and NIST results shown in Sections 6.2 and 6.3 were obtained from using a single reference solution during evaluation; however, machine translation researchers have access to multiple references or actual solutions which are typically not available in TCBR datasets. We could only generate multiple reference solutions for the medical health and safety incident reports using the method described earlier in Section 5.3.1. Hence, Section 6.4 focused mainly on comparing the results from applying MT evaluation techniques to TCBR with multiple and single reference solutions. Also, the method for generating the multiple reference cases restricts the experiment design to hold-out validation as opposed to a ten-fold cross validation used in all other experiments.

Test of significance on all our evaluation results is done at 95% confidence with a non-parametric method (Kruskal-Wallis test) since their deviation from the normal distribution is significant, that is p-value$<$ 0.05. Evaluation result values are shown in tables with bold font (apart from the column titles) indicating that a value is statistically better at 95% confidence than others in the same column. An italicized result value in a table shows that it is the highest but not significantly while an underlined value is significantly worse than others. When more than one value are shown with same font style, then there is no significant difference between them. For example, if two of five values in the same column are in bold while others are not, then, the two values are significantly better than the other three values. But this also indicates that there is no significant difference between these two values.

6.2 CR2N and other reuse strategies

We evaluate the effectiveness of the reuse strategies described in Chapter 3 primarily by measuring accuracy of their annotations; that is, 'reuse verbatim' or 'adapt'. Accuracy is measured as a ratio of retrieved keywords correctly annotated as reuse/adapt to the total number of keywords retrieved. A retrieve-only system is chosen as our baseline because our underlying hypothesis is that an effective reuse of retrieved similar cases must enhance retrieval since reuse succeeds retrieval in the CBR cycle. We measure accuracy in the same way for the retrieval system with all retrieved keywords deemed reusable since no annotation is done at retrieval. Other evaluation metrics used for comparing the retrieval baseline and reuse techniques include precision, recall, f-measure, cosine coefficient, BLEU and NIST.

$$\text{Reuse Accuracy} = \frac{|\text{true positives}| + |\text{true negatives}|}{|\text{Retrieved solution}|} \tag{6.5}$$

$$\text{Reuse Precision} = \frac{|\text{true positives}|}{|\text{true positives}| + |\text{false positives}|} \tag{6.6}$$

$$\text{Reuse Recall} = \frac{|\text{true positives}|}{|\text{Reference solution}|} \tag{6.7}$$

where,

|Retrieved solution| =

|true positives| + |true negatives|+ |false positives| + |false negatives|

|true positives|= number of terms in retrieved solution correctly annotated as reuse

|true negatives|= number of terms in retrieved solution correctly annotated as adapt

|false positives|= number of terms in retrieved solution incorrectly annotated as reuse

|false negatives|= number of terms in retrieved solution incorrectly annotated as adapt

|true positives| + |false positives| = Reuse proposed solution with adapt terms deleted

Accuracy shows predictive performance of the reuse strategies while the other metrics

quantify system performance if we were to delete terms annotated as adapt by a reuse strategy. A higher reuse cosine coefficient and precision with comparable recall over a retrieve-only system (baseline) would indicate better effectiveness for our simplified TCBR system in which only delete operations are carried out during adaptation. Here, terms annotated as adapt are assumed to be deleted but not replaced with other terms. We expect this partial proposed solution (with adapt terms deleted) to be closer to the actual solution than the retrieved solution if most of the deleted terms were truly irrelevant. However, a complete adaptation stage will also include substitute and insert edit operations and should lead to higher precision and recall. The effect of different neighbourhood sizes (k) on reuse performance is also investigated by repeating our experiments for increasing values of k up to the size of the casebase. This enables us to observe any result patterns across neighbourhoods and to determine the optimal neighbourhood size. The average evaluation results (accuracy, cosine coefficient, precision, recall, BLEU and NIST) are plotted against increasing neighbourhood sizes on the graphs used for our analysis in this section. We compared the baseline retrieval with the following four textual reuse algorithms.

1. CR2N as explained in Section 3.3.

2. CR2N$_p$, which uses the problem neighbourhood for reuse evidence computation rather than the retrieved solution's neighbourhood as used in the CR2N (specifically line 2 of Algorithm 3.1). CR2N$_p$ uses k-neighbours of the query and allows us to study the effect of using problem-side similarity to compute reuse evidence.

3. Case grouping strategy (CG), as explained in Section 2.2.3 but modified to use neighbourhoods (instead of the entire casebase) of the query to make it comparable to CR2N and CR2N$_p$.

4. Simple naive reuse strategy (NS) where the most frequent terms across the solution space are annotated as reuse in the retrieved solution while other terms are annotated as adapt. We use neighbourhood size (k) as a frequency threshold for the purpose of comparison. In other words, a term is annotated as reuse if it occurs in the solution

of at least k cases, otherwise it is annotated as adapt. This algorithm should help establish that the reuse strategies truly incorporate reasoning and should therefore outperform a simple reuse strategy based solely on document frequencies of terms.

A ten-fold cross validation experimental design is used for empirical evaluation and cosine coefficient metric is used for similarity computation at both retrieval and reuse stages. Each information entity (IE) in the CR2N represents a keyword from our domain vocabulary. We chose keywords as our textual units for annotations because the size of each retrieved solution text in the datasets (used for the experiments in this section) is small, typically 1 or 2 sentences with an average of 6 keywords. The similarity arcs in the Case Retrieval Nets from which the CR2N architecture is formed were not used in CR2N and CR2N$_p$ reuse strategies to make the retrievals comparable to the other two reuse strategies, that is Case grouping (CG) and Naive strategy (NS). Although the use of similarity arcs[1] can improve retrieval efficiency and address problems with variability of vocabulary by reducing any sparseness in a feature vector representation (Chakraborti et al. 2006), this should not make a significant difference for our domains since the size of each text is restricted to 1 or 2 sentences. The problem and solution texts are preprocessed using the GATE (Cunningham, Maynard, Bontcheva & Tablan 2002) library, available as part of the jCOLIBRI (Díaz-Agudo, González-Calero, Recio-García & Sánchez 2007) framework. These textual attributes are parsed into different layers such as sentences and keywords using the GATE Splitter. Suitable stop words are also removed and keywords stemmed to cater for morphological variations except where it is stated otherwise. Experimental results on datasets from weather forecast revision and medical health and safety incident reports are discussed in Sections 6.2.1 and 6.2.2 respectively. This is followed by a further analysis of results across the domains in Section 6.2.3.

[1]Attempts to automatically generate similarity arcs for our weather and incident reporting domains using WordNet and Latent Semantic Indexing led to worse retrieval results. This might be due to the domains having lots of specialised keywords and limited co-occurrence of similar keywords. Better context might alleviate this problem as discussed in Section 8.2.

6.2.1 Text Reuse Strategies for Weather forecast revision

We analyse the results of testing the four reuse strategies on a sample dataset from the weather forecast revision domain (see Section 2.4.1 for details). This sample consists of total of 2414 cases out of 5011 unique wind weather forecasts. The problem component of each case is the unedited text generated by a Natural Language Generation (NLG) system (SUMTIME-MOUSAM) while its human edited/revised version makes up the solution component. We restrict the average size of problem/solution text to two wind phrases (about 2 sentences) allowing us to operate at the keyword granularity level.

The GATE parser used for splitting text into different granularity such as sentences and keywords is modified for this dataset to ensure that the dash ('-') between words and numbers is not considered as a separator. Accordingly, wind magnitude ranges (e.g. '10-15') and directions (e.g. 'w-sw') will be correctly regarded as single entities. Also, separating such entities will provide misleading retrieval similarities and evaluation results by wrongly increasing the number of matched terms. An alternative parser, OpenNLP[2], which is publicly available also suffers from the same defect as dashes between words or numbers are used for splitting the text into different tokens. No stemming of words was carried out on the dataset since the vocabulary is small. Stop words are also limited to few punctuations such as fullstops and commas as many of the abbreviations in the datasets such as 's' ('south' wind direction) will be contained in a normal stop words list.

Figure 6.1 shows the results from empirical evaluation of the four reuse strategies and the baseline retrieval (CRN) on our wind weather forecast dataset using different automated performance metrics. The average annotation accuracy of the strategies are plotted against several reuse neighbourhood sizes on graph (a) in the figure. A comparison of the actual solution and the proposed solution with cosine coefficient, precision and recall values are shown on graphs (b), (c) and (d) respectively. The average values for accuracy and cosine coefficient evaluation are also given in Table 6.1 and 6.2 respectively; values for $k = 7$, 17 and 2000 are not shown due to space limitation but do not affect our discussion as they are shown on the graph.

[2]http://incubator.apache.org/opennlp/

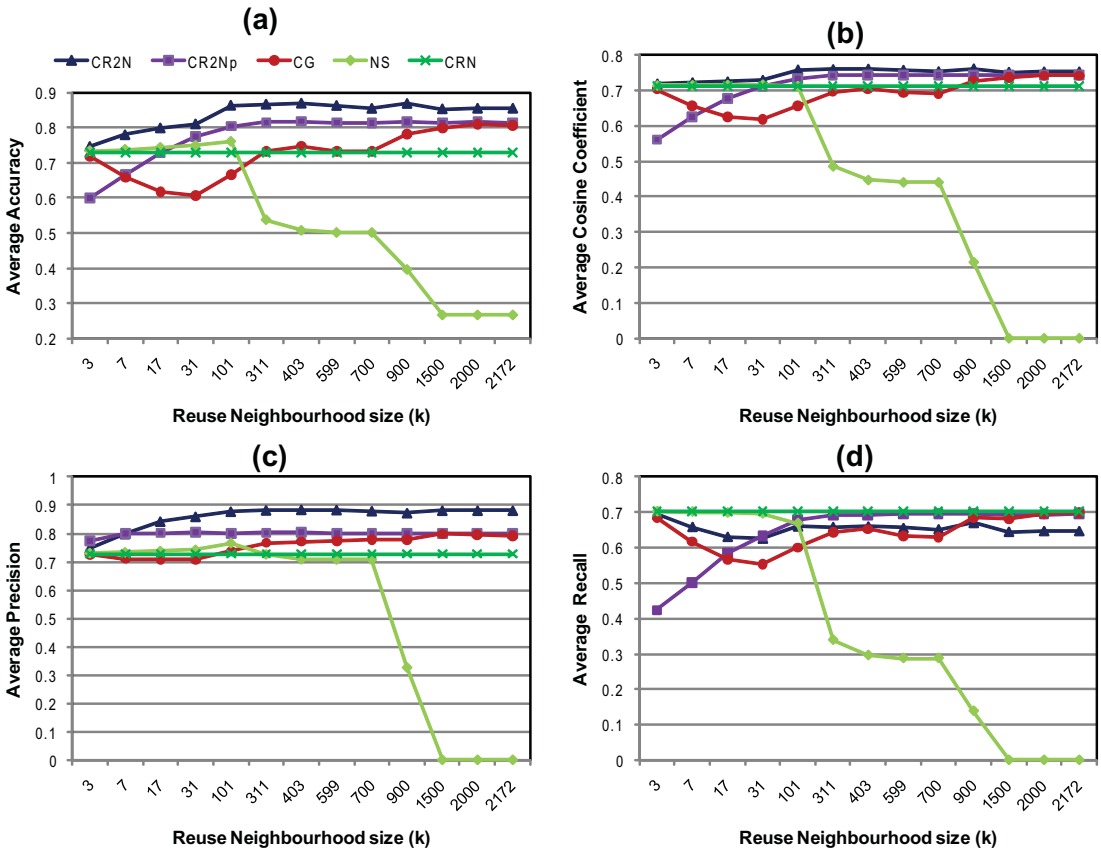


Figure 6.1: Evaluation results for reuse strategies in weather forecast revision

The average accuracy of the CR2N clearly outperforms the other reuse strategies as well as the baseline since CR2N's curve is above all other curves in the graph and its values are significantly better than most of the others across the different neighbourhood sizes as can be seen in Table 6.1. CR2N's accuracy increases with increasing k neighbourhood, attains its best value of 0.870 when $k = 403$ (about one-fifth of 2172 cases in the training set) and starts to decrease thereafter. This increase in accuracy with k can be attributed to CR2N gaining more contextual knowledge for its reuse/adapt annotation task. The decrease thereafter establishes the fact that comparison of local neighbourhoods is sufficient rather than the entire case base. $k = 101$ can be viewed as optimal based on the accuracy values as its difference from the value at $k = 403$ is minimal. The optimal local neighbourhood is relatively large in this domain because the vocabulary is small. Consequently, majority of cases have common keywords in their solution text.

Table 6.1: Comparison of Accuracy results for TCBR reuse in weather forecast revision

	3	**31**	**101**	**311**	**403**	**599**	**700**	**900**	**1500**	**2172**
CR2N	**0.748**	**0.811**	**0.864**	**0.867**	**0.870**	**0.863**	**0.855**	**0.860**	**0.854**	**0.855**
CR2N$_p$	0.560	**0.775**	**0.804**	**0.816**	**0.818**	**0.815**	**0.814**	**0.817**	**0.815**	**0.814**
CG	0.718	0.608	0.667	0.734	**0.747**	0.734	0.733	**0.781**	**0.800**	**0.808**
NS	**0.734**	**0.749**	0.761	0.538	0.507	0.501	0.501	0.396	0.267	0.267
CRN	0.729	0.729	0.729	0.729	0.729	0.729	0.729	0.729	0.729	0.729

Table 6.2: Comparison of Cosine results for TCBR reuse in weather forecast revision

	3	**31**	**101**	**311**	**403**	**599**	**700**	**900**	**1500**	**2172**
CR2N	**0.718**	**0.729**	**0.758**	**0.759**	**0.760**	**0.757**	**0.753**	**0.761**	**0.751**	**0.752**
CR2N$_p$	0.561	0.710	**0.733**	**0.742**	**0.743**	**0.743**	**0.742**	**0.743**	**0.742**	**0.743**
CG	0.703	0.617	0.657	0.696	**0.703**	**0.693**	**0.692**	**0.727**	**0.735**	**0.741**
NS	**0.714**	**0.717**	0.713	0.486	0.447	0.441	0.441	0.215	0.000	0.000
CRN	0.713	**0.713**	0.713	**0.713**	**0.713**	**0.713**	**0.713**	**0.713**	**0.713**	**0.713**

CR2N$_p$ also exhibits a similar trend to CR2N by generally having a better average accuracy than CG and NS. However, CR2N$_p$'s curve is below CR2N's indicating a slightly worse performance than CR2N. This indicates that the retrieved solution's neighbours are more informative for reuse evidence computation than the query's neighbours (i.e. problem-side neighbours). CG shows a different trend; the accuracy is initially below that of the baseline, decreases until $k = 31$ but increases subsequently outperforming the baseline after $k = 311$. The initial decrease could be attributed to any misleading and unpredictable evidence from the use of centroids especially when a smaller number of cases are used to create the clusters. Nevertheless, CG achieves similar average accuracy to CR2N and CR2N$_p$ when the entire casebase is used for reuse evidence computation but this is still less than the accuracy value for the optimal CR2N which uses neighbourhoods (i.e. $k = 101$). NS, which has the worst (lowest) accuracy especially with higher k values, can be viewed as another baseline and clearly shows that the reuse strategies are more complex and achieve better than recommending terms as reuse if they occur in several cases. The tapered end of NS's accuracy curve means that an average accuracy of about 27% can be still be achieved if all terms in each retrieved solution were annotated as adapt.

Average values from cosine coefficient and precision evaluation give similar patterns to the accuracy evaluation with CR2N surpassing the others as shown in Figure 6.1(b)

and 6.1(c). Here, the NS values tended towards zero as k increases towards the size of the casebase. This is because all terms are annotated as adapt by NS for higher values of k and there was no proposed solution text which consists of the reuse terms only to compare with the actual. The average recall of the CR2N becomes comparable to the baseline recall when $k = 101$ (0.668/0.7) as observed in graph (d) of Figure 6.1, but with a far higher precision (0.878/0.729). The same trend is observed for CR2N$_p$ and CG for higher values of k (1500, 2000 and 2172). The recall results from any of the reuse strategies cannot be greater than the baseline retrieval recall since terms annotated as suited to adapt are treated as deletes (see Section 6.2). The higher average recall values for CR2N$_p$ and CG is an indication that using the query's neighbourhood is more conservative as they tend to reuse most terms in a retrieved solution.

Table 6.3: Comparison of BLEU2 results for TCBR reuse in weather forecast revision

	3	**31**	**101**	**311**	**403**	**599**	**700**	**900**	**1500**	**2172**
CR2N	**0.538**	**0.542**	**0.591**	**0.585**	**0.587**	**0.582**	**0.572**	**0.589**	**0.596**	**0.596**
CR2N$_p$	0.295	0.522	**0.558**	**0.570**	**0.570**	**0.567**	**0.568**	**0.571**	**0.568**	**0.571**
CG	0.514	0.390	0.445	0.493	**0.505**	0.486	0.484	**0.544**	**0.565**	**0.568**
NS	0.528	**0.530**	0.529	0.274	0.237	0.228	0.228	0.114	0.000	0.000
CRN	0.527	**0.527**	0.527	**0.527**	**0.527**	**0.527**	**0.527**	**0.527**	**0.527**	**0.527**

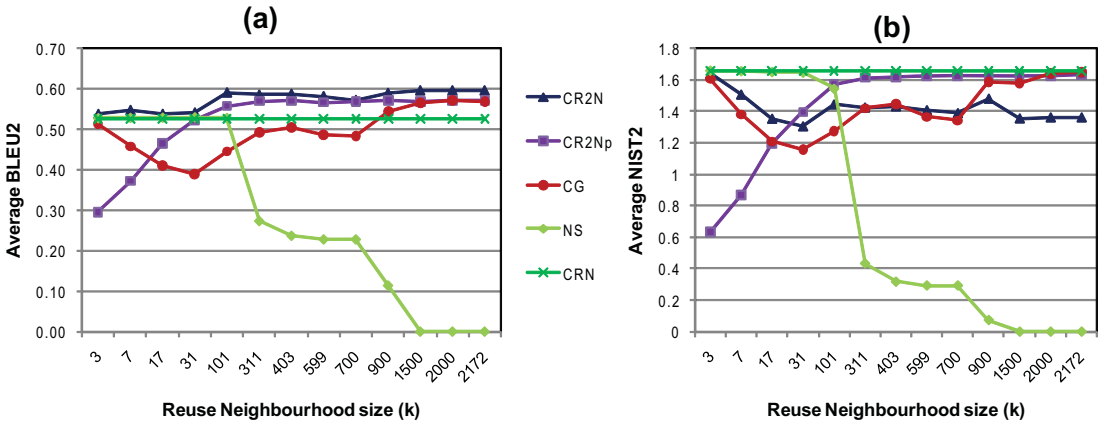


Figure 6.2: BLEU and NIST results for reuse strategies in weather forecast revision

BLEU2 and NIST2 evaluation results for weather forecast dataset are shown in Figure 6.2(a) and 6.2(b) respectively. Table 6.3 also shows the average BLEU2 scores with bold font used to indicate values that are significantly better than others across the five

reuse strategies. We limit N, the maximum number of n-grams, to 2 for these evaluation metrics rather than the standard N (which is 4 for BLEU and 5 for NIST used in Machine Translation) since we are dealing with short sentences (less than eight words) in this domain. This is even more noticeable with BLEU since the computed evaluation score reduces with increasing N tending towards zero for our short texts. For NIST, the evaluation score gets bigger with increasing N if more matching n-grams are found otherwise the score remains the same. The BLEU curves for the different techniques are quite similar to the precision and cosine coefficient curves discussed previously (in this Section) with the CR2N outperforming the rest followed closely by CR2N$_p$. The trends are also identical with CR2N and CR2N$_p$ having optimal values with small neighbourhood sizes while CG catches up when the entire casebase is used for reuse evidence computation. NS has comparable BLEU scores to the baseline CRN when $k \leq 101$ but tends towards zero afterwards. This is expected as BLEU is precision-based and its main strength lies in its use of multiple reference texts. BLEU1 and BLEU2 scores are not so different from precision values in TCBR evaluation when single references are available (as in our experiments on this domain) since the difference in length between the reference and retrieved texts will be relatively small thereby reducing the effect of the brevity penalty. On the other hand, NIST curves resemble the recall curves except that the NIST scores can be greater than 1 while recall values must be less than or equals to 1. Although NIST is also precision-based, its use of information weights and geometric mean as opposed to arithmetic mean in BLEU makes it less predictable especially for a dataset with small vocabulary. This could be a reason why the baseline's average NIST values were better than those of the reuse strategies. Overall, the empirical results from majority of evaluation metrics indicate that reuse strategies (CR2N, CR2N$_p$ and CG) outperform the baselines (CRN and NS) and is likely to aid easier adaptation of the retrieved solution.

6.2.2 Text Reuse Strategies for Health and Safety Incident reporting

The four reuse strategies are also tested on a sample dataset from the medical health and safety (H&S) incident reporting domain (see Section 2.4.2 for details). We formed

a subset of 362 cases from the original corpus of 983 unique incident reports belonging to a single care stage code (ACCID) that identifies reports relating to personal injuries and incidents during treatment or procedures. Each case has a textual description of an incident as the problem component while the solution component is the action taken by a health personnel to remedy the incident recorded in textual form. We ensured that the cases extracted for experiments have two sentences or less in the problem and solution attributes to make it comparable to the weather forecast dataset.

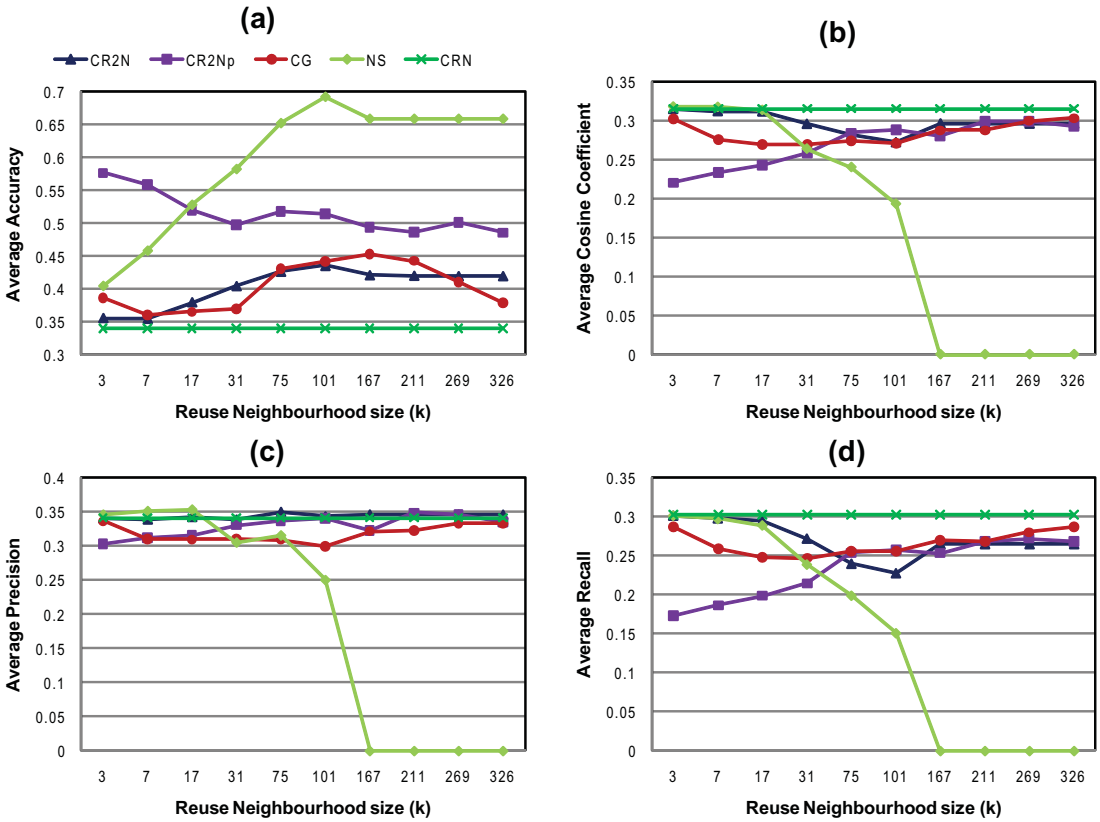


Figure 6.3: Evaluation results for reuse strategies in H&S incident reporting

Figure 6.3 shows average values from different text evaluation performance metrics (accuracy, cosine coefficient, precision and recall) comparing the reuse strategies and baseline retrieval (CRN) from our experiments with medical incident reports. Average values for accuracy and cosine coefficient evaluation are also given in Table 6.4 and 6.5 respectively. The performance of the reuse strategies exceed the baseline as shown by their accuracy plots in graph (a) of the figure. CR2N$_p$ outperforms CR2N and CG with its accuracy

curve lying above theirs for all k neighbourhood sizes; these accuracy values are significantly better for $k = 3$ to 31 as can be observed in Table 6.4. There is no clear distinction between CR2N and CG's performance but CR2N performs better when the number of neighbours used for reuse evidence computation is small, that is $k = 7$ and 31. The pattern of CR2N's effectiveness is most consistent and predictable when the accuracy curves are compared with those from the weather forecast in Figure 6.1(a). There is an initial increase in accuracy followed by a decrease which tapers into a constant value as the neighbourhood size approaches the size of the casebase. CR2N$_p$ and CG on the other hand show a less predictable accuracy trend with increasing neighbourhood sizes compared to the weather domain. Our results from experiments on the H&S incident reports confirm that CG might perform better in some domains if local neighbourhoods are used for reuse evidence computation rather than the entire casebase. This is because CG's best accuracy performance for the H&S incident dataset occurs when k=167.

Table 6.4: Comparison of Accuracy results for TCBR reuse in H&S incident reporting

	3	**7**	**17**	**31**	**75**	**101**	**167**	**211**	**269**	**326**
CR2N	0.355	0.354	0.380	0.404	0.426	0.435	0.421	0.420	0.420	0.420
CR2N$_p$	**0.576**	**0.559**	**0.520**	**0.497**	0.518	0.514	0.494	0.487	0.501	0.485
CG	0.386	0.360	0.365	0.370	0.431	0.442	0.453	0.443	0.410	0.379
NS	0.405	**0.458**	**0.528**	**0.582**	**0.652**	**0.692**	**0.659**	**0.659**	**0.659**	**0.659**
CRN	0.341	0.341	0.341	0.341	0.341	0.341	0.341	0.341	0.341	0.341

Table 6.5: Comparison of Cosine results for TCBR reuse in H&S incident reporting

	3	**7**	**17**	**31**	**75**	**101**	**167**	**211**	**269**	**326**
CR2N	0.315	0.312	0.312	0.296	0.281	0.272	**0.296**	**0.296**	**0.296**	**0.296**
CR2N$_p$	0.220	0.233	0.242	0.258	0.284	0.288	**0.280**	**0.299**	**0.299**	**0.292**
CG	0.302	0.276	0.270	0.269	0.275	0.271	**0.287**	**0.287**	**0.300**	**0.303**
NS	*0.317*	*0.317*	0.313	0.263	0.240	0.193	0.000	0.000	0.000	0.000
CRN	0.315	0.315	*0.315*	*0.315*	**0.315**	**0.315**	**0.315**	**0.315**	**0.315**	**0.315**

Graphs (b), (c) and (d) in Figure 6.3 show the cosine coefficient, precision and recall evaluation results respectively on this domain. The baseline retrieval outperforms the reuse strategies when the solution texts are evaluated with these three metrics but not significantly for most neighbourhood sizes used for reuse evidence computation; Ta-

ble 6.5 shows the results for cosine evaluation. There is no clear winner amongst the three reuse strategies (CR2N, CR2N$_p$ and CG) but NS performs worse as expected with increasing neighbourhood size. The BLEU and NIST evaluation results also give similar curves and statistical significance as can be seen in Figure 6.4 and Table 6.6. It should be re-emphasized here that BLEU and NIST results shown were obtained from using a single reference solution during evaluation; machine translation researchers have access to multiple references or actual solutions which are typically not available in TCBR datasets. We reckon that these results, though indicative, might not be a true reflection of how effective the different strategies are since they are unable to capture sentences that have similar meanings when expressed with a slightly different set of keywords.

The accuracy results for NS is surprising as it seems to perform significantly better than the other reuse strategies except CR2N$_p$ for $k = 3$ to 31 as shown in Table 6.4. It also signifies that a higher accuracy of about 66% (0.659) can be obtained by annotating all terms in the solution texts as adapt as shown by the tapered end of the NS accuracy curve. A closer look at the textual solutions of best match cases suggests that the evaluation results obtained in this domain can be misleading as our accuracy metric failed to capture semantic similarity between the proposed and actual solutions. Specifically, it was unable to capture sentences that have similar meanings when expressed with a slightly different set of keywords. We also observed that this issue of wide variation in authors' choice of words was very minimal in the problem space. Essentially, most of the problem components from our retrieved case were both lexically and semantically similar unlike the solution space where they can be semantically similar but lexically different. This was the primary motivation to apply Machine Translation evaluation measures to TCBR (explained in Chapter 5) and an analysis of these results appear in Section 6.4. Nevertheless, the results obtained from automated evaluation measures provide an insight to the complexity of this domain and annotation accuracy of the reuse strategies against a baseline retrieval.

Table 6.7 illustrates the challenge with evaluating textual solutions in this domain. This is similar to Table 5.1 but reproduced here to reinforce our result analysis. Here, two incident queries along with their retrieved cases (problem and solution), retrieval similarity

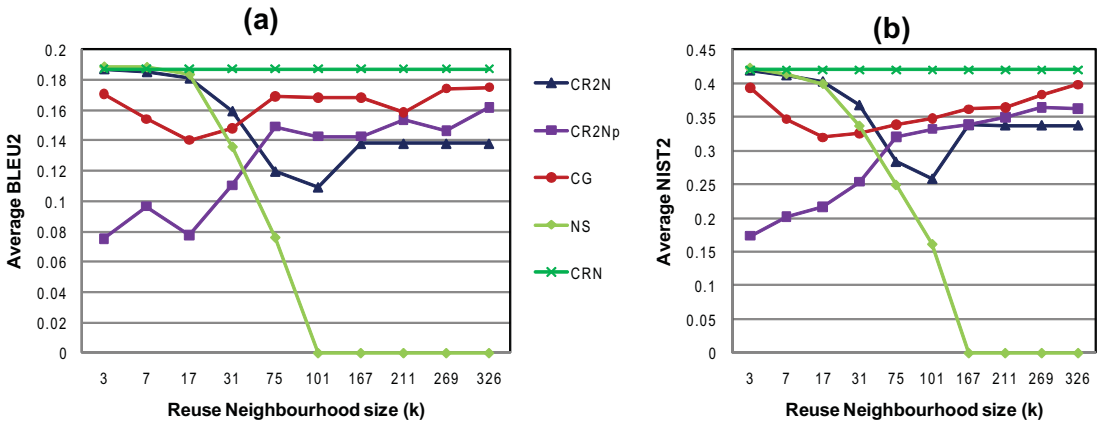


Figure 6.4: BLEU and NIST results for reuse strategies in H&S incident reporting

Table 6.6: Comparison of BLEU2 results for TCBR reuse in H&S incident reporting

	3	**7**	**17**	**31**	**75**	**101**	**167**	**211**	**269**	**326**
CR2N	0.187	0.185	0.181	0.160	0.120	0.109	0.138	0.138	0.138	0.138
CR2N$_p$	0.075	0.097	0.078	0.111	0.149	0.143	0.143	0.154	0.146	0.162
CG	0.171	0.154	0.140	0.148	0.169	0.168	0.168	0.159	0.174	**0.1751**
NS	*0.189*	*0.188*	0.184	0.136	0.076	0.000	0.000	0.000	0.000	0.000
CRN	0.187	0.187	*0.187*	*0.187*	**0.187**	**0.187**	**0.187**	**0.187**	**0.187**	**0.187**

values, reference solutions and precision results are shown. Although the retrieved and reference solutions are similar in meaning with query 1, retrieval precision or accuracy is computed as 0.333. This is because only one out of the three keywords ("nurse/nursing") is matched in the retrieved solution while other keywords though semantically similar are lexically different. Query 2 highlights a slightly different problem. Here, the level

Table 6.7: Sample retrievals from the H&S dataset

	Query	**Retrieved Problem**	**Simi larity**	**Retrieved Solution**	**Reference Solution**	**Preci sion**
1	nurse slipt and fell on wet floor	staff member slid on something wet and fell to the floor	0.612	examined by nursing staff	nurse given first aid	0.333
2	patient fell to the ground as nurse assisted him to bed	patient fell out of bed	0.707	examined by medical staff	patient was advised to get assistance in and out of bed	0.0

of detail/abstraction in the reference solution is different from retrieved solution thereby causing the accuracy to be computed as 0.0. The precision results obtained therefore are inadequate as we expect that retrieved solutions will be viewed as far more similar to the actual solutions if they were judged by domain experts.

We conducted a user study to obtain evidence that supports our observations regarding the sometimes inadequate nature of automated evaluation metrics especially for this domain. A questionnaire was generated containing ten different queries randomly selected from the 362 cases in our original casebase. This is shown in Appendix C. Each query was shown along with its actual solution and the solution retrieved during a ten-fold cross validation used in our experiments. Users are then requested to grade how similar the actual and retrieved solutions are on a scale of 1 to 5. We labelled the possible values 1, 2, 3, 4, 5 as 'Very dissimilar', 'Dissimilar', 'Unsure', 'Similar' and 'Very similar' respectively. Comments in form of free text were also allowed for each query. Thirty-one (31) users were recruited for this qualitative evaluation consisting mainly of students and academic staff all having some knowledge of CBR/data mining, text processing/analysis, similarity techniques and/or health & safety incidents. The users were also given a maximum of three days to complete the questionnaire.

The results of our user evaluation is summarised in Figure 6.5. This shows the computed precision for each query as well as the average user score normalised to a value between 0 and 1. It can be seen that for queries 1 to 4 having a precision of 0, the average user thought the retrieved solution was far more similar to the actual solution with user scores ranging between 0.4 and 0.8. The same can be said of the other six queries where the average user score was always better than what is evaluated by the precision metric. The main comments by users were that the evaluation difficulty lies in the use of varying terminology and granularity where some authors employ generic texts while others use specific terms with far more details. Based on the results of our user evaluation, we expect that the average values of the baseline retrieval (CRN) and the reuse strategies (CR2N, CR2N$_p$ & CG) to be far higher if automated evaluation is able to capture semantic similarity between texts.

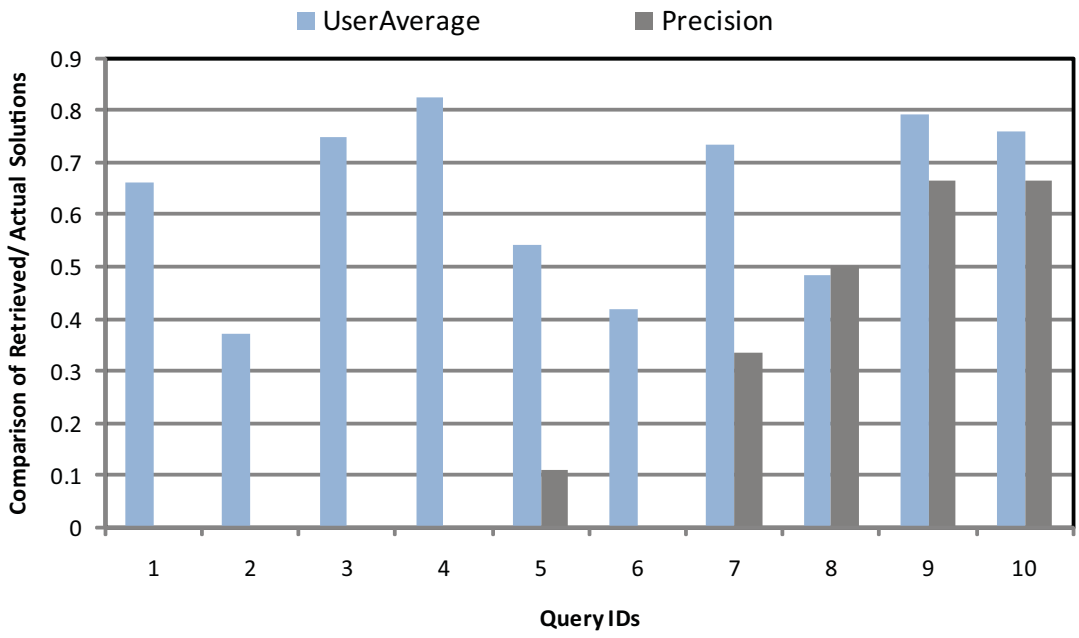


Figure 6.5: User feedback versus precision retrieval results in H&S incident reporting

6.2.3 Further Discussion on text reuse evaluation results

Evaluation results across the two domains used in our experiments provides supporting evidence that CR2N is a very useful text reuse strategy as it significantly outperforms the baseline retrieve-only system. It is also an important intermediate step during text reuse as it identifies textual constructs that need adaptation and annotates them differently from those that can be reused verbatim. The results also show that CR2N is significantly better than the simplistic Naive Strategy (NS) which annotated text constructs as reuse if their frequency across the casebase exceeds a specified threshold. The difference in performance between CR2N, which uses the solution space during reuse evidence computation, and CR2N$_p$, which uses the problem space is inconclusive from the experimental results of both domains. There is no clear winner since CR2N outperforms CR2N$_p$ in the weather forecast revision domain while the reverse is true in the medical H&S incident reporting domain. CR2N$_p$ might have performed better in the incident reporting domain because there is less variation in its problem than solution vocabulary. This means that the solution neighbourhood for each retrieved solution will be sparse since the similarity

measure (cosine coefficient) will exclude lots of semantically similar solutions written with varying lexical structure. Nevertheless, the pattern of CR2N accuracy annotation was more consistent across domains as the neighbourhood size increases towards the size of the casebase. Therefore it should be easier to predict the optimal neighbourhood size which gives the best annotation accuracy for CR2N than CR2N$_p$. When compared with CG, CR2N$_p$ outperforms CG across both domains although the pattern of accuracy results with increasing neighbourhood size for both strategies is inconsistent unlike with CR2N.

Another major observation from the results across both domains is that the optimal performance for the three reuse strategies (CR2N, CR2N$_p$ & CG) were obtained when local neighbours are used for reuse evidence computation rather than the entire case base. This is important as it reinforces the basic premise of k nearest neighbour and improves efficiency of reuse strategies since fewer cases are needed for reuse evidence computation. The use of centroids in computing reuse evidence has been proven by our experimental results from the two domains to be less effective than average similarity. This can be deduced from the accuracy results in Figures 6.1(a) and 6.3(a) where CR2N$_p$ outperforms CG in both domains even though they both use the problem space for reuse evidence computation.

Medical health and safety incident reporting is comparably a harder domain for TCBR text reuse. For instance, while annotation accuracies of up to 87% is achieved on the weather forecast dataset, only 58% is recorded the H&S incident dataset (ignoring NS results). This difference is mainly due to differences in domain vocabularies. The weather dataset has a small vocabulary size with many domain specific keywords for both problem and solution spaces. This allows for effective text comparison during retrieval and provides reliable text comparison results for evaluation. On the other hand, the incident dataset has a relatively large vocabulary size (about four times the size of weather vocabulary) consisting of mostly non-domain specific keywords. The retrieval of cases similar to a new problem is very effective as the problem texts were consistent with minimal stylistic variation across different authors. However, the evaluation results are less reliable mainly due to varying styles of writing by authors. These variations, which include the use of

different descriptions for identical problems, verbose details in some solution texts and concise solution details in others, cannot be captured by our automated evaluation metrics. Nevertheless, a user study conducted on the effectiveness of retrieved solution texts indicated that they were best; albeit, this study is not extensive and was carried with a limited/ small number of queries.

6.3 Text Reuse Guided Retrieval

Our reuse guided retrieval (RGR) technique described in Section 3.4 is evaluated for its effectiveness. There are two aspects to our evaluation of this technique. Firstly, we examine the average retrieval effectiveness of the RGR technique relative to a retrieve-only system which simply chooses the best match based solely on retrieval similarity; this baseline is denoted as CRN. Here, we compare the best match solutions proposed by RGR and CRN to an actual solution ignoring their reuse annotations. The second aspect of our evaluation is the effect of RGR on the subsequent reuse stage. In other words, does RGR improve the average annotation accuracy of the CR2N reuse strategy considering the original CR2N which always uses the best match from only retrieval similarity as a reuse baseline? During reuse evaluation, we ignore terms annotated as adapt just like we did for the reuse strategies evaluation in Section 6.2.

There are five parameters to be tuned to obtain an optimal performance according to RGR's pseudo code listed in Algorithm 3.2. These are the query's retrieval neighbourhood (ret_k), optimal reuse neighbourhood size (rs_k), similarity threshold between a retrieved solution term and other solutions (σ), and weights on retrieval similarity and reuse proportion in the reuse utility score (α and β respectively). We chose $ret\_k = 3$ based on analysis of empirical experiments on other values such as $ret\_k = 5, 7$ which showed that less than 2% of cases computed as best match by our technique were originally ranked below third position using only retrieval similarity. For instance on the weather forecast dataset, only 41 out of 2414 cases (1.7%) in a cross validation experiment was adjudged as best match by RGR (with equal weightings for α and β, $\sigma = 0$ and rs_k=3) were originally ranked below third position while 372 cases (15%) were ranked second or third. The value for

rs_k is chosen to vary increasingly up to the size of the casebase identical to what was used for the reuse strategies evaluation in the Section 6.2. This allows us to investigate if the optimal neighbourhood size for the CR2N gives the best retrieval and reuse effectiveness for RGR. σ is given a value of 0 since we used the CR2N at the keyword level for our experimental domians and any neighbour's solution containing a term annotated from the retrieved solution will have a similarity greater than zero. For the weights in the reuse utility score (α and β), we used values $\alpha = 0.25$, 0.5 & 0.75 where $\beta = 1 - \alpha$ to measure the effect of different weighting schemes. Note that $\alpha = 1$ is the same as CRN since only retrieval similarity is used for determining the best match. We therefore compared the following algorithms.

1. CRN which uses only retrieval similarity in determining the best match, as baseline.
2. RGR, explained in Section 3.4. Three versions of this algorithm were tested by varying the retrieval weights ($\alpha = 0.25$, 0.5 & 0.75) in the reuse utility score.

We used a ten-fold cross validation experimental design with cosine coefficient for similarity computation at both retrieval and reuse stages. Each information entity (IE) in the CR2N represents a keyword from our domain vocabulary because the size of each retrieved solution text in our application domain is small. The problem and solution texts are parsed into keywords and suitable stop words are also removed and keywords stemmed to cater for morphological variations except stated otherwise. We analyse evaluation results from our empirical experiments on datasets from weather forecast revision and medical health and safety incident reports in Sections 6.3.1 and 6.3.2 respectively. This is followed by further discussion of results across the domains in Section 6.3.3.

6.3.1 Reuse Guided Retrieval for Weather forecast revision

Our technique is tested on a dataset from the weather forecast revision domain described in Section 2.4.1. This is the same dataset used in experiments on the reuse strategies (see Section 6.2.1) with a casebase size of 2414.

Figure 6.6 shows the evaluation results of RGR on our weather forecast dataset using different automated performance metrics. The results for varying weights of the retrieval

Table 6.8: Cosine coefficient results for RGR in weather forecast revision

	3	**31**	**101**	**311**	**403**	**599**	**700**	**900**	**1500**	**2172**
$\alpha = 0.25$	0.709	0.699	0.717	0.717	0.719	0.718	0.715	0.721	0.721	0.721
$\alpha = 0.5$	0.709	0.703	0.721	0.720	0.720	0.719	0.717	0.723	0.723	0.723
$\alpha = 0.75$	0.710	0.707	*0.723*	*0.722*	*0.721*	*0.720*	*0.718*	*0.724*	*0.725*	*0.724*
CRN ($\alpha = 1$)	*0.713*	*0.713*	0.713	0.713	0.713	0.713	0.713	0.713	0.713	0.713

Table 6.9: F-measure results for RGR in weather forecast revision

	3	**31**	**101**	**311**	**403**	**599**	**700**	**900**	**1500**	**2172**
$\alpha = 0.25$	0.701	0.689	0.708	0.708	0.710	0.709	0.706	0.712	0.711	0.711
$\alpha = 0.5$	0.701	0.694	0.712	0.712	0.712	0.710	0.708	0.714	0.714	0.714
$\alpha = 0.75$	0.701	0.698	*0.714*	*0.713*	*0.713*	*0.712*	*0.709*	*0.715*	*0.715*	*0.715*
CRN ($\alpha = 1$)	*0.705*	**0.705**	0.705	0.705	0.705	0.705	0.705	0.705	0.705	0.705

similarity in the utility score, α, are also shown in the Figure 6.6. The average retrieval cosine coefficient are plotted against reuse neighbourhood sizes on graph (a) while f-measure, precision and recall average values are shown on graphs (b), (c) and (d) respectively. Cosine coefficient and f-measure values are also listed in Table 6.8 and 6.9 respectively; values for $k = 7$, 17 and 2000 are not shown due to space limitations but can be approximated from the graph.

The average cosine coefficient values of RGR with different weightings are better than the baseline CRN at our reuse strategy's (CR2N) optimal reuse neighbourhood size of 101 and above. In Figure 6.6(a), the cosine curves of RGR at $\alpha = 0.25$, 0.5 and 0.75 are all above the CRN curve from $k = 101$. CRN's cosine values from this point are also significantly worse than RGR's across the different neighbourhood sizes as shown in Table 6.8. With increasing reuse neighbourhood size, the average cosine values for RGR generally increases and tends to a constant value once most of the casebase is being used for reuse evidence computation. This is similar to the pattern of CR2N's accuracy/cosine curves in Figure 6.1(a) & 6.1(b). The marginal decrease on RGR's curves at rs_k = 7, 17, 31 and 700 can be due to annotation errors from CR2N which can lead to falsely computing a high reuse proportion. Nevertheless, this should not affect RGR's performance since these rs_k values are not optimal for CR2N's effectiveness.

Evaluation results for precision, f-measure and BLEU on RGR shown in Figures 6.6(c),

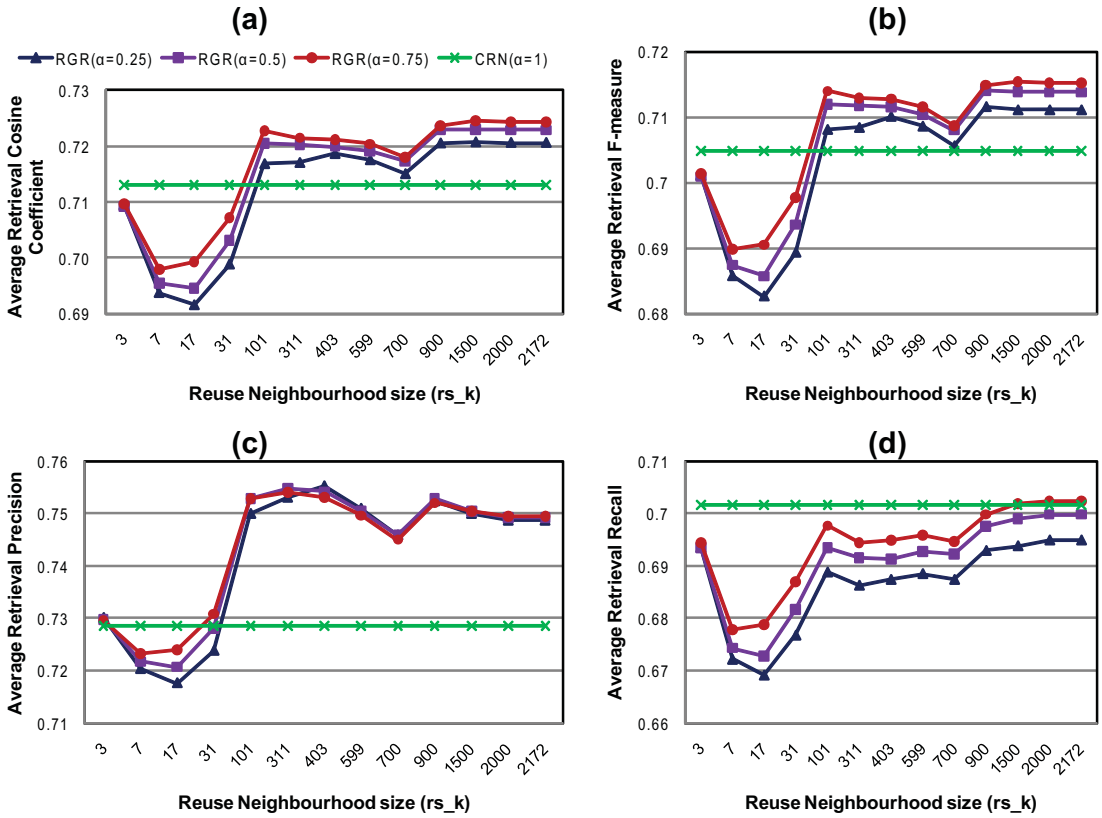


Figure 6.6: Evaluation results for RGR in weather forecast revision

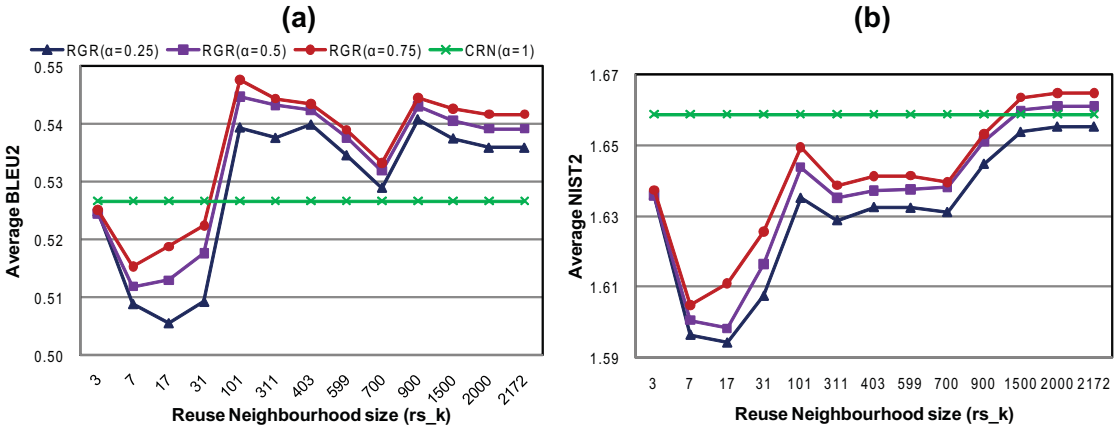


Figure 6.7: BLEU/NIST results for RGR in weather forecast revision

6.6(b) and 6.7(a) respectively also exhibit same pattern and analysis as the cosine results. But, the f-measure average values for CRN (see Table 6.9) are not significantly worse than RGR's unlike its average cosine results. This can be attributed to the average recall values of the CRN which was better than RGR's for most rs_k as shown in Figure 6.6(d).

This reinforces the view that the reuse strategy (CR2N) emphasizes more on accuracy and precision as opposed to recall which were observed in its evaluation results discussed in Section 6.2. Results from NIST in Figure 6.7(b) also project the same analysis as those from recall. This is because NIST's use of information weights and geometric mean are less relevant for small vocabularies as with the weather dataset.

Comparison of evaluation results across RGR's different weight parameters suggests that retrieval similarity should be weighted more than reuse in the utility score computation. This is because the curves for $\alpha = 0.75$ are above those for $\alpha = 0.5$ which in turn are above $\alpha = 0.25$ for the six text evaluation metrics (see Figures 6.6 and 6.7). The difference in performance between these weighting schemes are generally marginal and not statistically significant. Therefore, assigning equal weights to the retrieval similarity and reuse proportion in RGR's utility score should provide a comparable performance without the need to experiment with different values for α and β to achieve good performance.

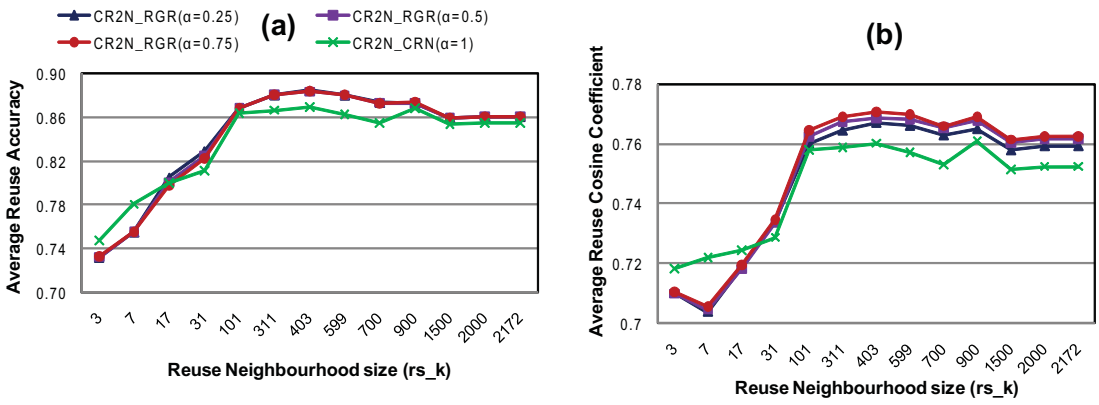


Figure 6.8: Reuse evaluation for CR2N guided retrieval in weather forecast revision

RGR also improves reuse effectiveness as shown by the curves of average reuse annotation accuracy and cosine coefficient in Figure 6.8. This is because for most neighbourhood sizes, the accuracy and cosine coefficient curves of CR2N's annotation of RGR's best match case were above that of the original CR2N which annotates best match cases obtained from using just the retrieval similarity. The neighbourhood (rs_k= 3,7,17) for which the original CR2N performed better than RGR's CR2N were those in which CR2N's performance was sub-optimal. While there is no difference in accuracy performance for CR2N with different

weights on RGR, the reuse cosine results were similar to their retrieval counterparts. In other words, the reuse cosine curves for $\alpha = 0.75$ are slightly above those for $\alpha = 0.5$ which in turn are above $\alpha = 0.25$ although the differences are marginal. Therefore our empirical results across several evaluation metrics from the weather forecast revision domain indicate that RGR improves not just retrieval but also reuse effectiveness thereby making the retrieved solution text easiest to adapt for a user during experiential reuse.

6.3.2 Reuse Guided Retrieval for Health and Safety Incident reporting

We also tested RGR on the medical health and safety (H&S) incident reporting domain described in Section 2.4.2 using the same evaluation methodology as that used for testing different reuse strategies in Section 6.2.2. CR2N was used for reuse annotation at keyword granularity level since average size of a problem/solution text is small and the similarity arcs in the CR2N are not used to make the results comparable to those obtained from our RGR experiments on the weather forecast revision dataset.

Table 6.10: Cosine coefficient results for RGR in H&S incident reporting

	3	**7**	**17**	**31**	**75**	**101**	**167**	**211**	**269**	**326**
$\alpha = 0.25$	*0.331*	0.321	0.323	0.312	0.302	0.319	0.318	0.317	0.317	0.317
$\alpha = 0.5$	*0.331*	0.323	*0.325*	0.314	0.300	0.320	0.318	0.317	0.317	0.317
$\alpha = 0.75$	*0.331*	*0.324*	0.330	*0.320*	0.307	*0.326*	*0.323*	*0.322*	*0.322*	*0.322*
CRN ($\alpha = 1$)	0.315	0.315	0.315	0.315	*0.315*	0.315	0.315	0.315	0.315	0.315

Table 6.11: F-measure results for RGR in H&S incident reporting

	3	**7**	**17**	**31**	**75**	**101**	**167**	**211**	**269**	**326**
$\alpha = 0.25$	*0.324*	0.313	0.317	0.308	0.297	0.313	0.313	0.312	0.312	0.312
$\alpha = 0.5$	*0.324*	0.315	0.319	0.310	0.295	0.314	0.313	0.312	0.312	0.312
$\alpha = 0.75$	*0.324*	*0.316*	*0.325*	*0.316*	0.302	*0.320*	*0.317*	*0.316*	*0.316*	*0.316*
CRN ($\alpha = 1$)	0.308	0.308	0.308	0.308	*0.308*	0.308	0.308	0.308	0.308	0.308

Evaluation results of the RGR technique on our H&S incidents dataset with different performance metrics are shown in Figure 6.9. Three RGR curves are shown on each graph for varying weights of the retrieval similarity (α). The average retrieval cosine coefficient, f-measure, precision and recall values for increasing reuse neighbourhood sizes are plotted on graphs (a), (b), (c) and (d) respectively in Figure 6.9. Average cosine coefficient and

f-measure values are also given in Table 6.10 and 6.11 to indicate significant differences where applicable.

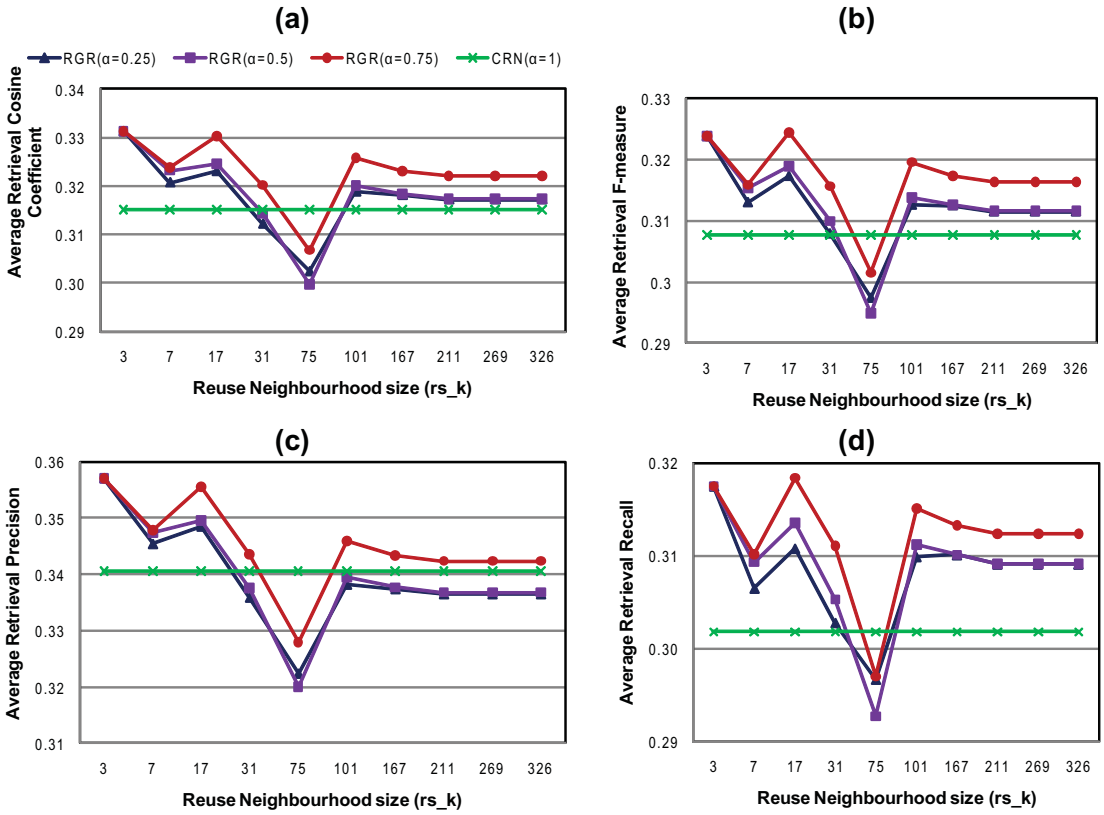


Figure 6.9: Evaluation results for RGR in H&S incident reporting

Cosine coefficient average values of RGR are better than baseline CRN for most reuse neighbourhood sizes in Figure 6.9(a). RGR's cosine curves show an unstable start with peaks and troughs but stabilises into a near-constant value once the neighbourhood size includes most of the casebase. These might be due to the fact that our evaluation measures are unable to capture sentence-level variation across texts as discussed in Section 6.2.2. The initial increase and decrease can be attributed to the reuse strategy's (CR2N) effectiveness not being optimal at these neighbourhood sizes. CR2N has the highest reuse annotation accuracy when rs_k=101 when applied to the best match from the baseline CRN as shown in Figure 6.11. This reuse neighbourhood size is also optimal for RGR. Comparison of values across the RGR cosine curves for different weightings indicates that $\alpha = 0.75$ is more effective than $\alpha = 0.5$ which is comparable to $\alpha = 0.25$. This rein-

forces our observation from our experiments with the weather dataset in Section 6.3.1. F-measure, precision and recall results from RGR experiments shown in Figures 6.9(b), (c) & (d) respectively exhibit identical patterns to the cosine evaluation with similar explanations. This also applies to the BLEU2 and NIST2 evaluation results in Figure 6.10.

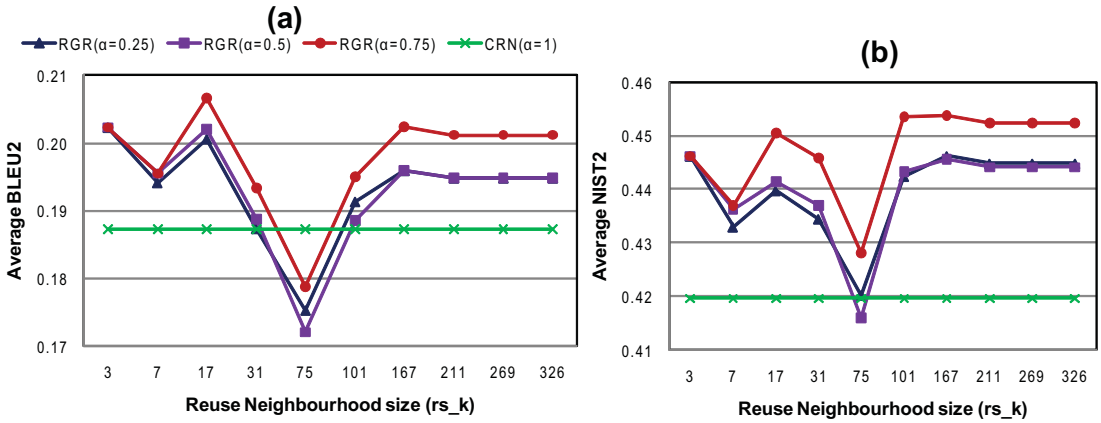


Figure 6.10: BLEU/NIST results for RGR in H&S incident reporting

Although, results across the six different evaluation measures indicate that RGR's performance is marginally better than the baseline CRN, this is not statistically significant (see Tables 6.10 and 6.11). Again, it is worth mentioning that the evaluation measures are inadequate for this domain and might have adversely affected the observed performance of RGR.

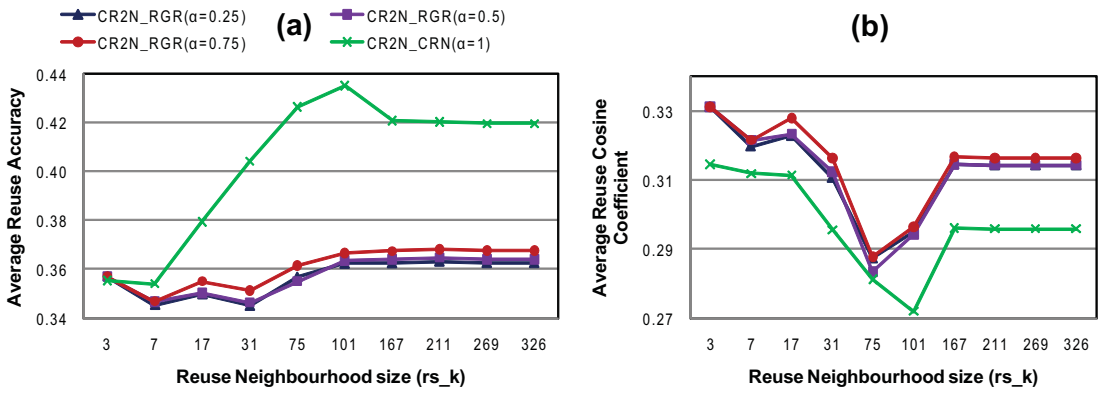


Figure 6.11: Reuse evaluation for CR2N guided retrieval in H&S incident reporting

Figure 6.11 shows RGR's impact on reuse effectiveness in this domain evaluated using the reuse annotation accuracy and cosine coefficient. It is expected that better retrieval

performance should lead to improved reuse and although this is true with cosine results, the same is not true for accuracy. Nevertheless, cosine results do indicate that on average, reuse text from RGR will be more similar to the actual text than those from the CRN even when CR2N's annotation accuracy is higher on retrieved texts from the CRN.

6.3.3 Further Discussion on CR2N Guided Retrieval Evaluation

Results from experiments on two domains indicates that RGR improves the quality of retrieval and reuse. RGR can also be viewed as a retrieval framework which can utilise different reuse strategies. This allows for choosing a reuse strategy that performs best on a particular domain for use within the RGR framework to achieve good retrieval performance. An important observation across both domains is that retrieval similarity should be weighted higher than the counter reuse as this was most effective. However, we also note that equal weights should be sufficient, when repeated experiments to determine the optimal weights is costly. This is because the differences in performance for the various RGR weight configurations used in our experiments were marginal and statistically insignificant.

Just like with experiments on the text reuse strategies, the medical health and safety incident reporting domain proved to be more difficult when compared to weather forecast revision. Even the average NIST values which were greater than 1 for the weather dataset, were in contrast, all less than 0.5 for the medical H&S incident dataset. Low result values in the medical domain were mainly due to solution texts having significant variation in terminology usage and writing styles (concise vs. verbose). This led to having several cases with identical/nearly identical problems but whose solutions though semantically similar were syntactically dissimilar. Unfortunately, the automated evaluation metrics used in our experiments were unable to capture these semantic variation.

6.4 Applying Machine Translation Techniques to TCBR

Relevance of the machine translation (MT) evaluation techniques, BLEU and NIST, for TCBR was discussed in Chapter 5. Here, we use these metrics to measure our retrieval and

reuse performance on the medical health and safety incident reports domain. Experiments discussed here are similar to those in Section 6.2.2 where the Case Retrieval Net (CRN) is used for retrieval and the results compared to those from different text reuse strategies. However, we restrict the reuse strategy to test the effectiveness of MT evaluation metrics for TCBR to the Case Retrieval Reuse Net (CR2N) as its performance was found to be superior to other reuse strategies (see Section 6.2). Another major difference is the implicit requirement for multiple reference solution texts by MT. We say implicit because we never came across any MT literature where single reference solutions are used for evaluation and therefore infer that this is crucial when employing MT metrics. But on the other hand, there is no evidence to suggest that multiple reference solutions have been used for TCBR evaluation either. We propose to create a new dataset from the original dataset to contain multiple reference solutions for each case. We use this new dataset to compare precision results with those obtained from the MT metrics. A comparison is also made between retrieval/reuse effectiveness when each test case has single and multiple reference solutions. We expect that the use of multiple reference solutions will provide more reliable evaluation results as they will be better able to capture grammatical variations in texts than single reference solutions.

6.4.1 Multiple Reference Solution Generation

Obtaining multiple reference solutions for each test case from a dataset originally containing single solutions is not trivial. Therefore a novel introspective approach to generate multiple references was proposed and described earlier in Section 5.3.1. Here, we generate multiple reference solutions for thirty four (34) test cases extracted from the original 362 H&S incident casebase. The remaining 328 cases formed our (training) case base in a hold-out experimental design. Each test case has between 2 to 4 multiple non-duplicate reference solutions. The 34 cases are actually 12 unique cases with multiple reference solutions but we added the other 22 cases whose problem components are identical/very similar to the unique cases to ensure that each unique solution is also used when evaluating single reference solutions. The 22 cases are not exact duplicates because their original

solution texts differ from the 12 unique test cases. Therefore their result with using the single reference solution will be different from the 12 unique test cases.

A hold-out validation experimental design is used for empirical evaluation and cosine coefficient is used for similarity computation at both retrieval and reuse stages. Each information entity (IE) in the CR2N represents a keyword from our domain vocabulary since each of the problem or solution text has a maximum of two sentences. GATE library is used to parse the textual cases into keywords, suitable stop words are removed and keywords stemmed to cater for morphological variations. During evaluation, synonym keywords are matched using WordNet (Fellbaum 1998) as well as keywords with the same lemma but different stems (e.g. gave/given, fallen/fell etc).

6.4.2 Results for MT Evaluation

We explore the usefulness of MT metrics, BLEU and NIST, when comparing the following two algorithms.

1. CRN, a baseline retrieve-only system
2. CR2N, a text reuse strategy

The average precision is also measured in addition to the two MT metrics using single and multiple reference solutions. The evaluation results for average precision, BLEU and NIST are shown in Tables 6.12(a), (b) and (c) respectively. It can be seen across all three tables that the use of multiple reference solutions for text (CRN or CR2N generated) evaluation always gives better results than using a single reference solution. Close examination of the 34 test cases suggests that these improvements are intuitive and better aligned with human judgement. This is because as expected multiple references reduce the effect of variability in the domain vocabulary on our evaluation metrics. This also aligns with the reason why qualitative text evaluation typically involves the use of multiple human experts to reduce bias to a certain style of writing. We therefore suggest that multiple reference solutions (when available) should be utilized for TCBR evaluation. If multiple references are not readily available in the application domain, then they might be formed introspectively from the casebase using the method explained in Section 5.3.1.

Table 6.12: Precision, BLEU and NIST evaluation of generated textual solutions

(a)

Average Precision	$N = 1$		$N = 2$	
	Single Ref	**Multiple Ref**	**Single Ref**	**Multiple Ref**
Retrieval	0.2860	0.5212	0.1471	0.3529
CR2N ($k = 9$)	0.2916	0.5307	0.1471	0.3529

(b)

Average BLEU	$N = 1$		$N = 2$	
	Single Ref	**Multiple Ref**	**Single Ref**	**Multiple Ref**
Retrieval	0.2860	0.5212	0.1516	0.4017
CR2N ($k = 9$)	0.2916	0.5307	0.1516	0.4017

(c)

Average NIST	$N = 1$		$N = 2$	
	Single Ref	**Multiple Ref**	**Single Ref**	**Multiple Ref**
Retrieval	0.4358	1.3144	0.4358	1.3438
CR2N ($k = 9$)	0.4451	1.3414	0.4451	1.3708

The result in Tables 6.12(a) & (b) also show that the precision scores are identical to BLEU when $N = 1$; this means that the length of the retrieved or CR2N generated solution texts were identical to one of the references implying that the brevity penalty (BP) had no effect. BP ensures that MT metrics penalise generated texts whose length are significantly smaller than most of the reference solutions as they are likely to have a very high precision. BP is the only thing that differentiates precision from BLEU when $N = 1$. Therefore, the average BLEU score is expected to be less than that of precision's, if it has an effect. The fact that the brevity penalty has no effect should generally be true for TCBR since proposed textual solutions are obtained from reference solutions to similar problems unlike MT where generated text can be far shorter than the reference.

We use $k = 9$ for the CR2N; this is based on empirical experiments for the optimal reuse neighbourhood size for the 34 test cases. As shown in the Table 6.12, average retrieval and CR2N results are generally comparable across all 3 metrics; precision, BLEU, and NIST. Tests of statistical significance also show no significance between each pair of retrieval/CR2N results ($p = 0.7107 > 0.05$ at 95% confidence). This suggests that the CR2N has no considerable improvement over retrieval for the 34 test cases with multiple

solutions used in our experiments. Although somewhat unexpected, this can be explained by the fact that most of the retrieved solution texts (description of the action taken) were sufficient to assist a health personnel to solve the test queries (incident descriptions) when checked manually. Over 80% (28 out of 34) of the retrieved solution texts can also be reused verbatim during documentation of incidents with very little modifications. It is important to emphasize here that CR2N captures this since it is not worse than retrieval's results according to the three metrics. Nevertheless, averages are not able to show certain patterns if the difference in average between two result sets is small but the data is skewed with a comparatively large standard deviation (SD). This was observed for the results where $N = 1$ in Table 6.12 for all evaluation metrics and $N = 2$ for NIST.

Table 6.13: Precision score clusters for the 34 cases with multiple reference solutions

Precision ($N = 1$, multiple Ref)	**Number of cases**	**Average Retrieval**	**Average CR2N**
$Score = 0$	6	0	0
$0 < Score < 1$	12	0.1435	0.1704
$Score = 1$	16	1	1

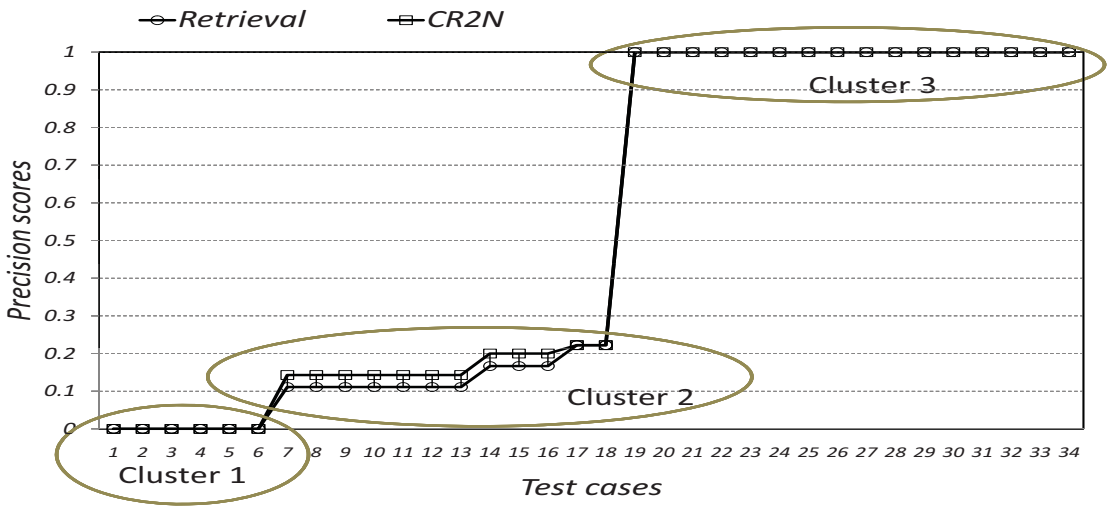


Figure 6.12: Graph of precision results for the 34 test cases with multiple references

Further investigation revealed that the standard deviation of the individual 34 results were large as compared to the average; for instance, $SD = 0.46$ against average precision $= 0.52$ for the retrieval results with multiple references. The same applies to the results in Table 6.12 where N=1 for the three evaluation metrics and N=2 for NIST. The SD results from the use of single references were generally greater than their averages. We discovered that the results where CR2N slightly improves over retrieval formed three natural clusters: $score$=0, $0 < score < 1$ and $score$= 1 as shown in Table 6.13 and Figure 6.12. The 6 cases with zero retrieval scores (cluster 1 in Figure 6.12) cannot be improved since it means that none of the retrieved keywords matched the query's reference solutions. The CR2N aptly identifies this by discarding all of these keywords during it text generation process. CR2N also uses all keywords in its generated text for the 16 cases where retrieval precision is one (cluster 3). Importantly, it is able to identify when all keywords in the retrieved solution text should be included in its generated text solution. The CR2N generated text outperforms retrieval for the 12 middle cases with retrieval scores between 0 and 1 (cluster 2) and this is significant at 95% ($p = 0.045 < 0.05$). A similar trend is observed for the BLEU and NIST results. Here, precision and MT metrics are therefore only able to show improvements in retrieval when the retrieval scores are greater than zero.

6.5 Chapter Summary

Experimental set-up to test the effectiveness of our novel contributions on TCBR-based techniques to aid text reuse and improve text evaluation with machine translation metrics have been presented in this chapter. A critical and thorough analysis of the evaluation results indicated that our techniques generally outperform standard baselines as well as comparable state-of-the-art techniques. Experiments with CR2N algorithm for text reuse on two datasets from the domains of weather forecast revision and health & safety incident reporting outperformed a comparable reuse technique, Case Grouping and a retrieve-only CRN. Our reuse guided retrieval with CR2N was also tested on these two datasets and was more effective than a baseline retrieval which always chooses its best match using only the retrieval similarity between the cases and a query. Lastly, our experiments with

machine translation evaluation techniques for TCBR on a health and safety incidents dataset gave results that should correlate better to human judgements. However, extensive user experiments is needed to quantify this correlation as discussed later in Section 8.2. We also discovered that the use of multiple reference solutions as opposed to the use of single references is critical to obtaining better and more reliable results with these MT metrics.

Chapter 7

Evaluation: How to Reuse?

This chapter discusses the evaluation of the reuse algorithms introduced in Chapter 4 which address our research question of how to reuse textual solutions. We explained how a textual solution might be generated from several similar cases. The text reuse approaches were illustrated with hotel review generation which is a typical application domain. Here, we discuss our experimental evaluation on hotel reviews generation and university coursework feedback authoring.

Section 7.1 discusses the evaluation methodology related to the experiments analysed in this chapter. Discussion of evaluation results with respect to the hotel reviews and coursework feedback authoring appear in Sections 7.2 and 7.3 respectively. We conclude the chapter with a summary in Section 7.4.

7.1 Methodology

Comparison of the proposed and actual solution texts for authoring tasks are done using cosine coefficient similarity. We manage this process by restricting the comparison to aligned sentences in these solution texts. This is to remove the adverse effect of unaligned sentences, which are typically too specific and not likely to be reused across authors. Cosine coefficient evaluation has a unique advantage over precision/recall because it allows a comparison of performance across algorithms with a single metric that also can take texts' length into account. Nevertheless, we measured precision, recall and f-measure. We

did not evaluate with BLEU and NIST metrics because previous experimental results (see Sections 6.2 and 6.3) suggest that they are comparable to precision/recall except when multiple reference solution texts are available (see Section 6.4). As in Chapter 6, test of significance for comparison of evaluation results is done at 95% confidence with a non-parametric method (Kruskal-Wallis test). Bold fonts in result tables indicate that a value is statistically better than others in the same column while italics is used to show that it is highest but not significantly. An underlined value is significantly worse than others in the same column.

Three text reuse algorithms discussed in Section 4.2 were tested across two domains of hotel reviews and university coursework feedback authoring.

1. BASE, baseline retrieval discussed in Section 4.2.1
2. XFRM, transformational text reuse (in Section 4.2.2) and
3. COMP_k, compositional text reuse; see Section 4.2.3 for more details

We are also interested in the effect of different neighbourhood sizes (k) on reuse performance for transformational (XFRM) and Compositional/Constructive (COMP_k) text reuse. Experiments for both techniques (XFRM and COMP_k) were therefore repeated using increasing values of k ($3, 5, 10$ & 25). The solution texts in both domains are parsed into different layers such as paragraphs, sentences and keywords using GATE libraries. Suitable stop words are also removed and keywords stemmed to cater for morphological variations during text evaluation.

7.2 Hotel Reviews Generation

We evaluate the three reuse algorithms by using them to generate hotel reviews using the TripAdvisor dataset detailed in Section 2.4.3. We use 641 cases with at least one sentence aligned to each of the five rating attributes; that is cleanliness, location, room, service and value. These five attributes plus the overall rating make up the problem component of each case while the solution is the review text. Although some of the other available attributes from the corpus such as 'trip type', 'date of stay' and 'recommend to friend'

might have improved retrieval if included in the problem component, their values were absent in most of the reviews in the corpus. Also, we decided not to limit our dataset to a single hotel or author because the number of reviews per hotel or author in the corpus is small (less than 50). Therefore our dataset includes reviews from several hotels across the globe as it is more intuitive to reuse similar reviews across all hotels instead of within a single hotel or a single author which would have led to a very small casebase. Review texts were normalised by substituting named entities such as person names, currencies, locations and dates with generic labels. This ensures that our proposed texts are more easily reusable across authors and that evaluation results from our automated metrics are reliable. Table 7.1 lists some of these entities extracted with GATE together with the general category label.

Table 7.1: Examples of named entities found in Hotel Reviews

Category	**Named entity examples**
person name	yang, vincent, susanne, patrick, katherine
currency	yen, pounds, francs, euros, dollars, cents
date	september 2009, mid august 08, last year,april 26th, 22nov07
time	9.30pm, 8:00 a.m., 5pm, 3:45pm, 17:45

A ten-fold cross validation is employed in our experiments for this domain. We ensure that retrieved reviews have very similar rating values to the query by using an interval similarity of 2. For example, it will be very difficult to reuse a sentence aligned to the cleanliness attribute with rating 4 (very good) from a review for a query with a cleanliness rating 2 (poor). This means that a difference of 1 between two rating attributes gives a 0.5 similarity while a difference greater than 1 gives zero similarity. A global similarity is calculated across attributes using a weighted average; 0.25 for the overall rating attribute and 0.15 for each of the other five rating attributes.

7.2.1 Evaluation Results on Hotel Reviews

Figure 7.1 shows the evaluation results of the TCBR-based text authoring algorithms applied to hotel reviews generation with different performance metrics. Average cosine coefficient similarities between proposed texts by the three text reuse techniques and actual

solutions appear in graph(a), while f-measure, precision and recall are shown in graphs (b), (c) and (d) respectively. The increasing k-neighbourhoods for compositional (COMP_k) and transformational (XFRM) approaches to text reuse are the numbers shown as suffix after the algorithm name. For instance, XFRM_k10 implies that the proposed text is assembled from a maximum of ten nearest neighbours using the transformational text reuse algorithm. The values plotted on the graphs also appear in Table 7.2 along with the number of sentences and tokens in the proposed text. All values in the table are averages of the individual scores for each test case.

Table 7.2: Comparison of evaluation results for TCBR authoring of Hotel reviews

	Cosine Coefficient	**F-measure score**	**Precision score**	**Recall score**	**Number of Sentences**	**Number of Keywords**
BASE	0.2245	**0.1517**	0.1762	**0.1710**	9.54	253.17
XFRM _k3	0.2271	**0.1535**	0.1742	**0.1757**	9.80	260.50
XFRM _k5	0.2271	**0.1535**	0.1741	**0.1757**	9.81	260.61
XFRM _k10	0.2271	**0.1535**	0.1741	**0.1757**	9.81	260.61
XFRM _k25	0.2271	**0.1535**	0.1741	**0.1757**	9.81	260.61
COMP_k3	0.1803	0.1164	0.1835	0.1101	3.33	145.31
COMP_k5	0.1952	0.1262	0.1864	0.1241	3.58	159.75
COMP_k10	0.2121	0.1366	**0.2174**	0.12889	3.85	155.43
COMP_k25	**0.2418**	0.1501	**0.2767**	0.1316	4.30	129.30
COMP_N	**0.2969**	**0.1754**	**0.3733**	0.1274	5.00	84.70

BASE which recommends a subset of sentences from the best match case by ignoring sentences unaligned to any of the five rating attributes performs quite well when compared to XFRM from the graphs in Figure 7.1. Table 7.2 also shows that there is no significant difference between BASE and XFRM for this domain across the four evaluation metrics. This is because the retrieved best match case is sufficiently similar to a new problem in the cross validation experiments. Therefore XFRM rarely substitutes or transforms sentences from the best match's solution text. It implies that XFRM does not find other nearest neighbours whose rating attribute values are more similar to the new problem than the best match's solution. The results trend with increasing neighbourhood for transformational reuse also supports this explanation. Here, we see that the results do not change at all even for a neighbourhood size of 25 which is fairly large. This means that the marginal difference between BASE and XFRM must be as a result of transforming the retrieved

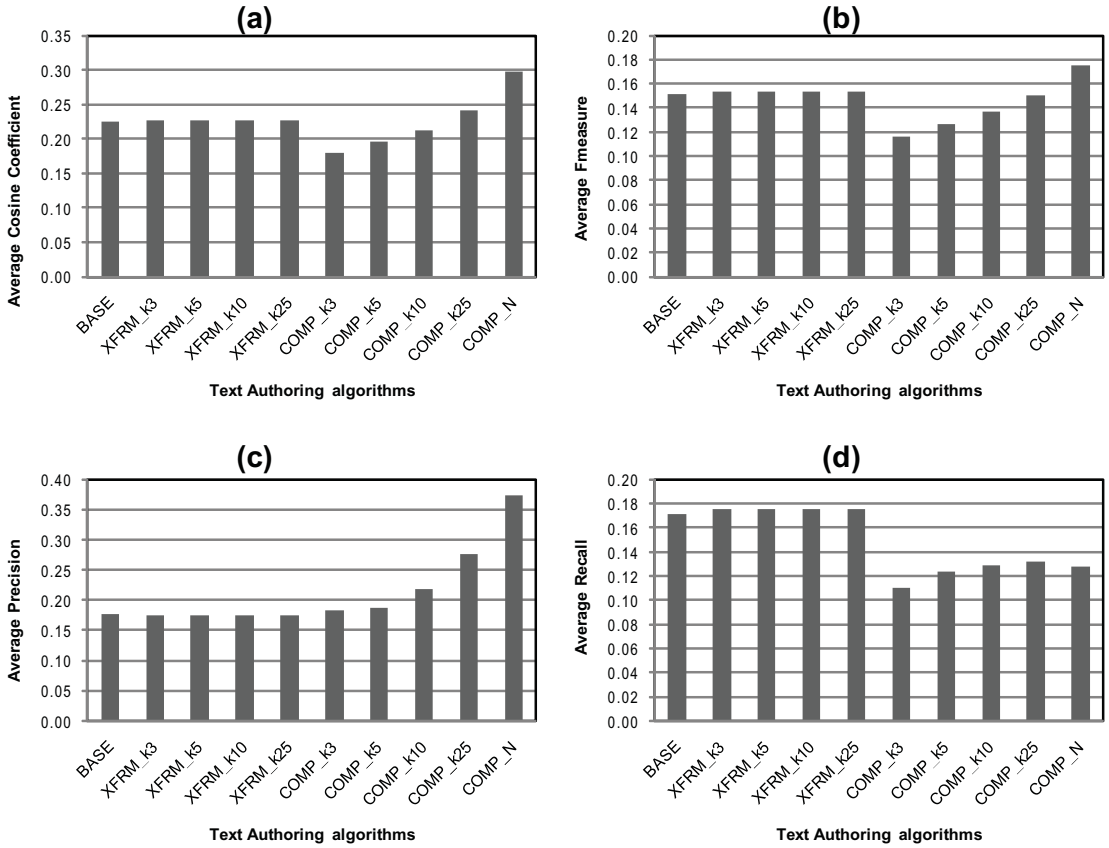


Figure 7.1: Evaluation results for TCBR authoring of Hotel Reviews

solution using the second or third nearest neighbour for one or more of the test cases.

For COMP_k where local prototypical sentences are proposed, the results from the four evaluation metrics generally improve with increasing neighbourhood size. This trend suggests that the performance will match up with COMP_N as the neighbourhood size tends toward the entire casebase. Our intuition based on these results is that local prototypical sentences tend to capture less keywords that are reusable across authors as compared to the global ones. Prototypes generated from very small neighbourhoods (i.e. 3, 5 and 10) though useful will lead to the proposed text being too specialized and not easily reusable by others. This accounts for their worse performance when compared to BASE and XFRM with the cosine and f-measure evaluation results. COMP_N which uses all cases to generate prototypical sentences for each of the five rating attributes clearly outperforms the other reuse algorithms. Its performance is statistically significant for the evaluation metrics except recall where its performance is significantly worse. BASE and

XFRM having significantly better recall values show a trade-off with precision which is typical with precision/recall evaluation. The number of sentences and keywords in the solution text proposed by these techniques are far bigger than than those from the COMP_k approach as shown in Table 7.2. The size of their proposed texts means that they might have many keywords in common with the actual solution text. However, this necessitates an author to do more editing apart from other additions to create the solution text to what he/she requires. On the other hand, algorithms like COMP_N gives more precise texts which will need little editing apart from addition of other specific texts. The f-measure results show that COMP_N is better as its average value is higher than those of BASE or XFRM. This is further reinforced by the cosine results. An advantage of the COMP_N approach is that generic sentences are likely to be more similar to the actual solution compared to a local sentence which might express the same opinion using different terms. Although at first surprising, these results compliment findings in other related studies on text reuse (Lamontagne & Lapalme 2003, Lamontagne & Lapalme 2004).

7.2.2 Further Discussion on Reviews Authoring

The evaluation results from all the metrics used in our experiment are generally low. For example the highest average cosine similarity between the proposed texts and their actual solution is 0.297 (< 0.3) while that of the average precision is slightly higher at 0.373 (< 0.4). The highest average values for recall and f-measure are much lower at 0.176 and 0.154 (both less than 0.2) respectively. However, closer examination of solution texts proposed by the reuse algorithms indicate that such low values do not necessarily mean poor solution quality. Figure 7.2 shows a sample of the proposed text generated by COMP_N. Most of the sentences seem reasonable given the query ratings, except for service rating attribute which is comparatively more verbose. Text parsers produce such long sentences when they do not find pre-defined sentence markers such as full stop or question mark. The service rating text in Figure 7.2 does not contain any sentence marker and is therefore parsed as a single sentence. Wrongly parsed sentences will contain specific details that might adversely affect the overall evaluation. Nevertheless, the results indicate

that proposed texts were similar to the actual and it might be easier to edit them and add other details than writing from scratch. Also, our proposed texts will encourage new authors to write reasons for each given rating attribute value rather than a lot of verbose but unnecessary details thereby making future reviews more useful to others.

Query	C	L	R	S	V
	4	3	2	2	5

Proposed Solution (COMP_N)

C	it was very clean .
L	the hotel was in a great location .
R	unfortunately we were very disappointed upon seeing the room .
S	wer i guess that the best way to do this is to list the good and bad about this place so here goes : the good it is very nice and sunny and always hot - lovely the pool is clean and warm the air conditioning in the reception area is refreshing the beer was ok good shuttle bus service to the (horrible) beach the bad where are the toilets , as far as i could see there was only one (apart from going back to the room) and that was in the reception
V	it was so worth it .

Figure 7.2: An example review text proposed by COMP_N

7.3 University Coursework Feedback Generation

Our reuse algorithms for text authoring are also tested on the university coursework feedback generation task discussed in Section 2.4.4. The dataset has 46 cases with at least one of the eight grading criteria having a minimum of one sentence aligned to it based on generated seeds. The seed generation process explained in Section 4.1.1 remains largely the same for this domain. However, specific representative seeds can be generated for this domain because each grading criteria value has a short description of what the student is expected to achieve in order to be given such grade value. Table 7.3 lists some of these seeds; the complete seed list used in this domain can be found in Appendix A.2. We were also able to generate seeds for the overall grade by inspecting most of the feedback text since the casebase is relatively small. The eight grading attributes plus the overall grade attribute make up the problem component for each case while the solution is the feedback

text. We did not carry out any text normalisation such as substituting named entities (person names, currencies, locations and dates) with generic labels for feedback texts in this domain because such named entities do not occur. The feedback text is also more consistent and reliable because it is authored by a single person and for a single piece of academic coursework. This is unlike the hotel reviews where review texts used for our experiment were written by more than one author and cut across several hotels.

Table 7.3: Sample seeds used for text alignment in coursework feedback authoring

Criterion descriptor	**Sample seeds list**
Task1.1	class, attribute, numeric, skewed, nominal
Task1.2	tree, generate, depth, parameter
Task2.1	parameter, experimental, design, evaluation
Task2.2	classifier, result, comparison, baseline
Task3.1	trials, effectiveness, criteria
Task3.2	efficiency, explanation, significance
Task3.3	discussion, presentation
Task4	arff, preamble, extend
Overall	submission, incomplete, solution

A leave-one-out evaluation methodology is employed in our experiments for this domain because of the small size of the casebase. The casebase size of 46 is relatively small compared to 641 cases in the hotel review dataset where we used a ten fold cross validation. We ensure that retrieved reviews have very similar ratings to the query by using an interval similarity of 3 as we expect that feedback text from a coursework with grade 6 ('A') will be difficult to reuse for a query with grade 3 ('D'). This means that a difference of 2 between the values of identical grading criterion attribute will give 0.33 similarity, difference of 1 gives 0.67 similarity while differences greater than 2 gives zero similarity. A global similarity is calculated across the grading criterion attributes using a weighted average; 0.2 for the overall grade attribute and 0.1 for each of the other eight grading criteria.

7.3.1 Evaluation Results on Coursework Feedback

Evaluation results of the TCBR-based text authoring techniques on our coursework feedback dataset with four performance metrics are shown in Figure 7.3. Average cosine coefficient, f-measure, precision and recall scores between proposed texts by the reuse

techniques and actual solutions appear in graphs (a), (b), (c) and (d) respectively. Different k-neighbourhoods for compositional (COMP_k) and transformational (XFRM) text reuse are shown on the graph as numbers after the algorithm name preceded by 'k' for XFRM. Table 7.4 also shows the same values for average cosine coefficient, f-measure, precision and recall scores as well as the average number of sentences and keywords in the proposed text by each of the reuse techniques.

Table 7.4: Comparison of results for TCBR authoring of coursework feedback

	Cosine Coefficient	**F-measure score**	**Precision score**	**Recall score**	**Number of Sentences**	**Number of Keywords**
BASE	0.4778	*0.4281*	**0.4616**	0.4551	3.34	49.60
XFRM _k3	0.4851	0.4130	**0.3520**	0.5536	4.98	90.47
XFRM _k5	0.4791	0.3821	0.3008	0.5860	6.15	110.55
XFRM _k10	*0.4948*	0.3746	0.2809	**0.6280**	7.02	125.11
XFRM _k25	0.4849	0.3487	0.2528	**0.6349**	7.91	140.77
COMP_k3	0.4246	0.3284	0.3229	0.4236	3.36	67.32
COMP_k5	0.4268	0.3173	0.2839	0.4727	4.30	87.26
COMP_k10	0.4379	0.3178	0.2577	0.5230	5.19	105.66
COMP_k25	0.4617	0.3195	0.2441	0.5610	6.34	128.62
COMP_N	0.4848	0.3409	0.2593	0.5887	6.43	126.30

Analysis of the cosine evaluation in graph(a) of Figure 7.3 indicates that XFRM outperforms BASE with the best performance at $k = 10$. XFRM's best cosine value is also better than COMP_k and COMP_N. However, Table 7.4 shows that XFRM's better performance over all others is marginal because there is no statistically significant difference between cosine results of the four techniques. But unlike the hotel review dataset where there is no difference in XFRM's results with varied neighbourhood size, there is noticeable difference in XFRM's results for this domain when different neighbourhood sizes are used to transform the best match's solution. This means that some of the retrieved best match cases are not sufficiently similar to their test cases in the leave-one-out experiment. Therefore XFRM can substitute sentences from the best match's solution text where it finds other nearest neighbours whose grading attribute values are more similar to the query than that of the best match. A possible explanation for the retrieved best match case not being sufficiently similar is the small casebase size, which means that not all valid combinations of attribute-value pairs are present in the casebase. Furthermore, the

nearest neighbour's attribute values with aligned sentences used for transformation can be similar to those of the query but not identical in this domain. This contrasts with the hotel review domain where most of the attributes values used for transformation are identical because of the casebase size.

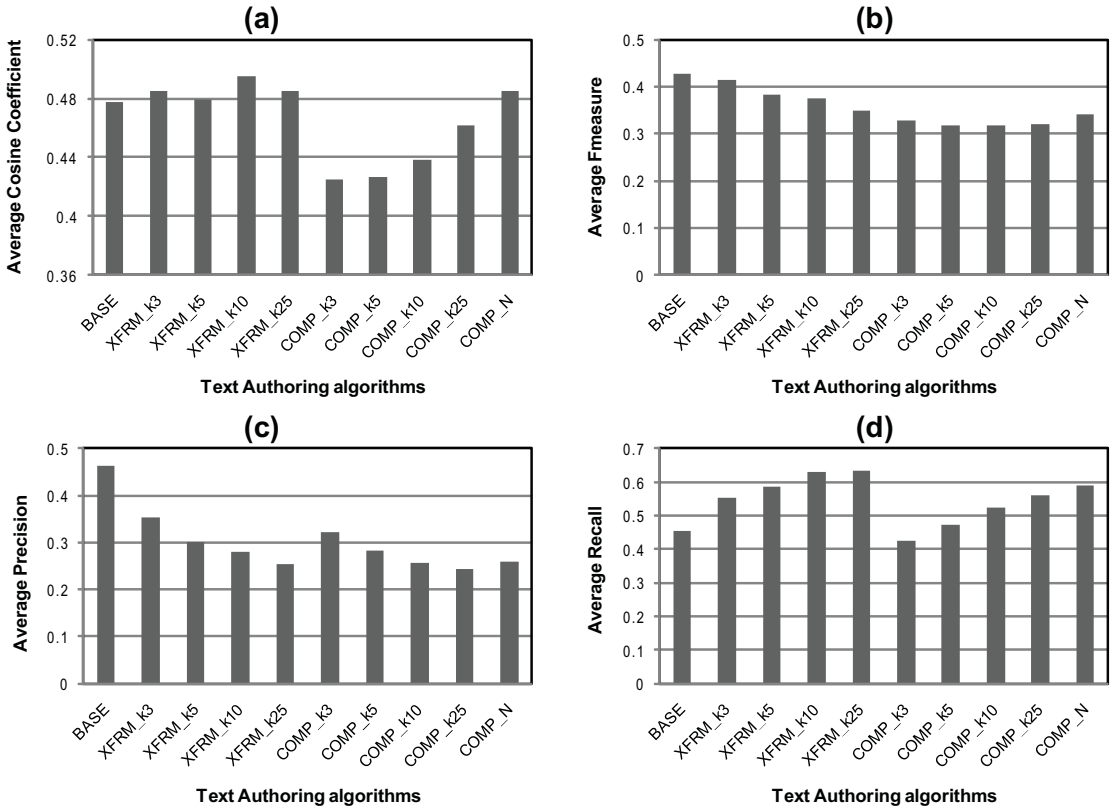


Figure 7.3: Evaluation results for TCBR authoring of Coursework Feedback

COMP_k's cosine performance with this coursework feedback dataset is consistent and identical to what was obtained in the hotel reviews domain. Its average cosine value improves with increasing neighbourhood size and tends towards matching up with COMP_N which uses the entire casebase. COMP_N's performance is slightly worse than XFRM in this domain because no sentence is generated for some attributes as can be seen from Table 7.4. The average number of sentences is about 6 even though there are 9 grading attributes. One factor that could account for this is that all cases had fewer than nine aligned sentences with seven alignments being the highest found and only for a single case while most cases (41 out of 46) had between 3-6 alignments. The inadequate alignment

might also be a result of small number of sentences in each case; only three cases have seven sentences while other have between 1-6 sentences. Another factor that could have influenced the COMP_k and COMP_N result is the small number of cases available. The 46 cases available for our experiments did not cover all six grading values for each of the 9 attributes.

On the other hand, the precision, recall and f-measure evaluation portray a different performance outlook for the reuse techniques. As expected, the precision results were inversely proportional to their recall counterparts. Therefore, BASE which has the best average precision value also has the lowest recall. The precision values for BASE and XFRM with $k = 3$ were significantly better than others as shown in Table 7.4. But while BASE's recall is significantly worse than others, XFRM _k3's recall is comparable. Cosine results for XFRM are also marginally better than BASE. The average number of sentences in BASE's proposed text is unusually low most likely due to inadequate aligned sentences and this might have given it undue advantage during the precision calculation. COMP_k up to COMP_N's performance based on precision, recall and f-measure are not impressive based on the same reasons given for the poor cosine evaluation performance, that is few cases in the casebase and few authored sentences per case. We reckon that the results will be more similar to what was obtained with the hotel review dataset if we have access to a larger casebase. Nevertheless, the current results indicates that the reuse algorithms are a useful tool for text authoring in domains where some text is written to complement or explain other more structured attributes.

7.3.2 Further Discussion on Feedback Authoring

Evaluation results from all metrics used in our experiments on this domain are also generally low but better than those obtained when we tested the TCBR-based authoring techniques on the hotel review dataset. For instance, the highest average cosine similarity between the proposed texts and their actual solution is 0.495 compared to 0.297 with hotel review authoring. The average precision, recall and f-measure values for both domains, coursework feedback/hotel review authoring, are 0.462/0.373, 0.635/0.176 and

0.428/0.154 respectively. Such low evaluation results do not necessarily constitute poor solution quality as discussed in Section 7.2. There are a number of reasons for the generally better evaluation results obtained in the coursework feedback domain as compared to hotel reviews. First, the coursework feedback dataset is written by a single author for a single coursework which makes the syntax of the text more consistent with little stylistic variation. This is unlike the hotel reviews dataset written by several authors with varying styles for many different hotels which makes the text less consistent and more difficult to evaluate with automated metrics. Another reason for bigger evaluation values from the coursework feedback domain is that the texts are generally shorter and well written with little or no long sentences that might be wrongly parsed due to lack of sentence markers.

7.4 Chapter Summary

We have discussed experimental evaluation related to our algorithms that address our research question of how to reuse in this chapter. Analysis of the results indicated that our techniques generally outperformed reasonable baselines. The three TCBR-based reuse techniques for text authoring were evaluated on datasets from hotel reviews and coursework feedback authoring domains. Empirical evidence from our evaluation results show a close similarity between texts authored by the reuse techniques and actual text authored by humans and will assist authors to write better and more useful feedback texts in both domains.

Chapter 8

Conclusion

In this thesis, we investigated the challenges with case reuse in textual case based reasoning (TCBR) particularly when the solution is textual. Novel algorithms were then developed to address some of the gaps observed in text reuse with the ultimate goal of proposing a solution text that is more useful to a user than the retrieved solution text. This chapter concludes the thesis by presenting our main contributions to knowledge and identifies desirable future extensions.

8.1 Contributions

We revisit our research objectives listed in Section 1.2 and examine how these were achieved in this section.

1. **Propose a method to approximate problem-solution alignment in textual cases:** Identification of specific parts of a textual solution which occur as a result of the presence of particular problem attributes or attribute values is crucial for text reuse. This is because it enables a more fine-grained retrieval where each attribute in a target problem is matched to an identical case attribute whose value is most similar. Consequently, the proposed solution might be assembled from several similar cases. The intuition here is that such assembled solution text will be more relevant and require less adaptation since best matched values for the target problem are more likely to be found in several similar cases rather than in a single

best match case. However, problem to solution alignment is not typically available in most TCBR domains. Our text alignment method automatically generates such alignments, provided that seeds for each problem attribute are in place. Here, seeds are terms (words or phrases) closely related to a problem attribute. The seed generation process is semi-automated where terms having a relationship with a problem attribute descriptor are discovered followed by manual weeding of irrelevant seeds. Relationships to an attribute descriptor that qualify terms to be selected as seeds include synonyms, antonyms, and restricted hypernyms (generalisations) and hyponyms (specializations). An alignment is established between a problem attribute and a specific part (e.g. sentence) of a solution text when any of the attribute's seeds occur in that part. Solution parts that are not aligned to any of the problem attributes are less likely to be reusable by others as they could either be too specific or extra details that are not covered by the problem description. Inspection of the alignments created by our technique on some cases showed a very high accuracy as most of the alignments were correct and reasonably intuitive.

2. **Develop an algorithm for textual solution reuse in TCBR:** Most current TCBR applications have focused on developing techniques that enhance retrieval effectiveness when problem attributes are textual. However, very little had been done to make retrieved textual solutions more suitable to the target problem. In this thesis, we proposed text reuse (the reuse stage in TCBR applications with textual solutions) as the identification of reusable terms (keywords, phrases, sentences etc.) in a retrieved solution text relevant to a target problem and more likely to be reused verbatim. The first task was to differentiate reusable terms from others that are not relevant to the target problem and therefore need adaptation during reuse. We proposed a strategy to annotate terms in a retrieved solution text as reuse or adapt depending on their relevance to the target problem. This strategy was formalized into a novel architecture called Case Retrieval Reuse Net (CR2N) which integrates the retrieval and reuse stages by extending the popular Case Retrieval Net (CRN) knowledge structure. The algorithm for our reuse technique which derives its name

from the CRN architecture is generic as it allows for the use of different neighbourhood sizes during reuse evidence computation. We also explored how these neighbourhoods can be formed from either the problem or solution space. Different levels of text granularity such as keywords, sentences and paragraphs can also be annotated during text reuse by our algorithm. Experimental studies on our text reuse technique showed that it was an effective tool to annotate retrieved textual solutions. Its overall performance was also better than a retrieval baseline and a comparable text reuse technique.

3. **Propose an algorithm for reuse guided text retrieval:** Although similar problems should have similar solutions, it is sometimes possible that the solution of the most similar case to a query is not the easiest to adapt during reuse. This has therefore led to the development of several algorithms in which the best match case is determined not just by the similarity of its problem to a query but also the ease of adaptability of its solution. However, all current adaptation guided retrieval techniques have been applied to cases whose solutions are structured. Therefore, there exists a gap in algorithms that can be used to guide retrieval when solutions are textual. We extend CR2N's utility to determine the ease of adaptability of a retrieved textual solution. Our intuition is that the easiest solution to adapt will also have the highest number of terms annotated as reuse and the least number of terms as adapt. The reuse proportion (which is a quotient of the number of terms annotated as reuse and the total number of terms) aptly captures this intuition. We propose a reuse guided retrieval (RGR) mechanism which ranks cases based on a utility score which is a weighted combination of the retrieval similarity and reuse proportion. Result from empirical experiments supports RGR as its performance surpassed a standard retrieval system. We also discovered that RGR's performance is best when retrieval similarity is weighted higher than reuse proportion by the reuse utility function.

4. **Develop techniques for transformational and compositional approaches to text reuse:** Text reuse occurs at the reuse stage of the TCBR cycle when solutions

are textual. It is at this stage that retrieved solutions are adapted into a proposed solution for a query. Although several transformational and compositional reuse approaches have been introduced for traditional CBR where solution attributes are structured, very few transformational text reuse approaches currently exist, while compositional text reuse is virtually non-existent. Transformational reuse involves the re-organisation of solution elements through add and delete operations while in compositional reuse, a solution is obtained by combining solution elements of several partially similar cases. This thesis introduced novel methods for transformational (XFRM) and compositional (COMP) text reuse when problem and solution text are aligned. An aspect of our contribution to compositional text reuse is the introduction of a novel method to combine several similar texts into a single meaningful prototype whose syntactic construct is most general and therefore most easily reusable by others. Experimental studies on real world application datasets show that solution text from XFRM was more suitable than that of a baseline retrieval where the solution is copied directly from the best match case. We also observed that its performance was hardly affected by the size of neighbourhoods used for transformation. COMP also improved over the baseline retrieval based on empirical results. Its performance improved with increasing neighbourhood sizes, with best performance recorded when all cases are used in creating the prototypes. However, the quality of solution text assembled from COMP depends on the availability of cases with identical attribute values to the target problem. This is because the prototype is created from text chunks aligned to identical attributes and having the same attribute values. Automated evaluation measures seem to favour COMP over XFRM as solution texts assembled from COMP are generic and more likely to be matched in reference (actual) solution text for a query. XFRM's solution texts, though semantically similar to the reference solutions, are less likely to be matched, because they are typically from local neighbourhoods which might use a different set of words to express the same opinion.

5. **Propose semantic-aware automated evaluation measures for TCBR:** The most common TCBR automated evaluation measures for solution texts are precision and recall (adapted from Information Retrieval research). Precision is typically computed as the ratio of the number of common terms between a proposed and reference solution to the number of terms in the proposed solution. Recall, on the other hand, is the ratio of the number of common terms to the number of terms in the reference solution. These metrics are known to be reliable when the vocabulary is relatively small or when text is written strictly in accordance with a standard template. The small size of the vocabulary or standard template ensures that there is little variation in the choice of words used in the solution text across all cases. Precision/recall evaluation results can be less reliable if it is used with a dataset with large vocabulary and containing wide variation in the choice of words. Another drawback of precision/recall metrics is that the sequence and position of terms as well as the semantic context is ignored. We addressed some of these drawbacks observed in current TCBR evaluation metrics by exploring the text evaluation measures used in other research areas. In particular, we propose and adapt two machine translation (MT) evaluation measures, BLEU and NIST, for TCBR. Due to their high correlation with human judgements, these specific measures are the de-facto standard for MT evaluation. They have also been adapted for text evaluation in other research areas like Natural Language Generation and Text Summarization. BLEU and NIST incorporate term sequence because they allow the matching of consecutive words/keywords of different sizes called n-grams. The precision scores from different n-grams (1 up to a specified N) are aggregated using an arithmetic (in BLEU) or geometric (in NIST) average. The use of n-grams of different sizes also indirectly takes term position into account. It addresses sense disambiguation since bigger n-grams ($n > 1$) matched between the proposed and actual reference texts are more likely to have the same context. The difficulty in evaluating texts in domains where there is wide variation in word choice across cases is handled by these MT metrics with their use of multiple references for each test case. This ensures that semantically similar

solutions are not unnecessarily penalised because they have a different word choice since this is more likely to be found in one of the reference solutions. The application of MT metrics for TCBR evaluation with multiple reference solutions for test cases produced better and more reliable results than with precision/recall. However MT metrics, as expected, had similar results to precision/recall when using a single reference solution. We also discovered that certain parameters in MT metrics such as brevity penalty are not required in TCBR evaluation due to intrinsic differences in text generation for TCBR and MT.

8.2 Way Forward

We have investigated the reuse stage for TCBR and proposed techniques to aid text reuse and TCBR evaluation. In this section, we highlight known limitations of our research contributions and discuss them in the light of our research scope and desirable extensions to minimise or eradicate these limitations.

Richer contextual knowledge for CR2N

Our text reuse technique employs a knowledge light approach to text representation. This means that the context of each term might not be sufficiently captured especially when working at the keyword granularity level. We expect that sense disambiguation will be important to establish context. For example 'plate' in 'plate kit' should be differentiated from 'food plate', similarly for 'bank' in 'river bank' and 'blood bank'. A consequence of this inadequate contextual knowledge in the CR2N is that multiple instances of each unique term in a retrieved solution will be given the same annotation (reuse or adapt) in relation to a query irrespective of their context. Although not a problem for the datasets used in our experimental studies as the solution text were small (typically 1-2 sentences), it is likely to be problematic in others. A simple but effective way to capture contextual knowledge at the keyword level is to include a specified number of keywords on the left and right hand side of the keyword being annotated in the Case Reuse Net. These extra keywords to the left and right should be given lesser weights than the keyword of interest

during the activation propagation in the Case Reuse Net to ensure that their influence is sufficient but minimal. Another method to address the challenge of inadequate context is to annotate the solution text at the level of larger text units such as phrases and sentences. This is done by activating the Case Reuse Net with all the keywords in the phrase or sentence of interest with equal weights. However, this could in turn lead to a sparse representation.

Learning how to adapt with the CR2N

A major limitation of CR2N as a text reuse strategy is that it does not address the question of how to adapt the terms annotated as irrelevant to a query. Although adaptation is an important task during TCBR reuse, it remains a challenging and difficult task. One reason why the CR2N is unable to learn substitutes for terms deemed to need adaptation is because its alignment of problem to solution terms is implicit. Each solution term deemed relevant to the query (reusable without adaptation) must have been aligned to one or more terms in the query. Therefore substitutes can be easily determined if the implicit alignments between terms in the problem and solution space are made explicit. The text granularity will also be important for adaptation. Extending the CR2N for explicit problem-solution alignment is highly desirable and will address the question of how to adapt. However, such alignment might also be learnt using external domain knowledge or statistical techniques based on co-occurrence, such as Mutual information (Manning & Schütze 1999) and machine translation models (Brown et al. 1993), that establish alignment between parallel texts. Nevertheless, establishing problem-solution alignment might not be sufficient where insertion of other terms is required apart from substitutions. One way of ensuring that substitutions are adequate will be to allow substitution of one 'adapt' term with more than one term during adaptation.

Introspective seed generation for text alignment

The problem with the current seeds generation process used for our text alignment method is that it relies solely on the problem attribute descriptors which are sometimes insuffi-

cient. In the coursework feedback authoring domains used in our experiments, additional seeds were extracted by manually skimming through all feedback texts. This was feasible given the dataset was small. This suggests that the feedback text in the training corpus will contain several other seeds that might be obtained apart from words having a direct relationship to the problem attribute descriptors. Therefore a useful and desirable extension to our seed generation process is the development of an introspective method which is capable of correctly identifying seeds in the training feedback text and extracting them as additional seeds to those generated by the current method. Such a method should detect more seeds by looking at other terms in the corpus having a high mutual information with the initial seeds based on statistical co-occurrence. A few regular expressions containing seeds can also be manually crafted and used to learn several other similar expressions in the training corpus from which more seeds can easily be extracted. Active learning strategies (Baram, EL-Yaniv & Luz 2004, Hu, Delany & Mac Namee 2010) might be useful to learn regular expressions as well as extraction of additional seeds.

Quantify correlation of BLEU/NIST to human judgements in TCBR

Experimental studies showed that TCBR evaluation with machine translation metrics (MT), BLEU and NIST, gave values which we expect to correlate better to human judgements than the common evaluation measures, precision/recall, currently in use. This is more obvious when the vocabulary size is large and there are wide variations in word choice and order. We also observed that the difference between these MT metrics and the common TCBR evaluation measures are mainly significant when using multiple reference solutions for each test case. Although MT researchers have reported high correlation to human expert judgement, the correlation has not been properly quantified for TCBR domains. As much as we expect a high correlation as well in TCBR evaluation, an extensive qualitative evaluation across several TCBR datasets that confirms this expectation will make the MT metrics more widely acceptable to TCBR researchers. The major challenge in carrying out such qualitative evaluation will be to obtain suitable TCBR datasets with multiple reference solutions as most current TCBR domains have single reference solu-

tions. Also the costs associated with recruiting experts for each experimental domain is another important factor to be considered as non-experts might not be suitable for some problem-solving domains.

8.3 Reflections on TCBR systems and applications

We tested our novel text reuse algorithms on datasets from four different domains as detailed in previous chapters. These are the weather forecast revision, medical health and safety incident reporting, hotel review and academic coursework feedback authoring domains. Two of these domains are very specialized (weather forecast and health & safety incidents) and greater care must be taken when automated systems are deployed on the field due to associated risks. For example, a TCBR system for the medical health and safety incident reporting will be more useful for documentation of incidents than during an emergency where immediate action must be taken. When used to assist such documentation, users might be restricted to use it for less critical incidents which are mainly used for producing statistical summaries rather than diagnosis by others. This ensures that minor but vital details are not left out by users who might be inclined to accept all suggestions by a TCBR system in order to complete the documentation task on time. The same risks might be associated with weather forecasting when used in industries such as aviation or offshore oil exploration. The cost of an incorrect weather forecast is typically very high for such an industry. The other two domains (hotel review and academic coursework feedback authoring) are less critical with minimal associated risks. We therefore expect that TCBR can be deployed quickly and immediately for practical use in these other domains.

8.4 Thesis Summary

This thesis has presented our work on text reuse using the TCBR paradigm which attempts to solve new problems by reusing similar problem solving experiences stored in textual form. Our main contribution to knowledge is the development of several techniques that

make a retrieved solution text more useful to a user.

A critical analysis of previous and current literature showed a wide gap in work related to reusing cases with textual solutions. This research has attempted to fill this observed gap with three main contributions. The first contribution is the introduction of a novel text reuse strategy (CR2N) which identifies reusable textual constructs in a retrieved solution text relevant to a given query and differentiates them from those that need adaptation. The strategy was formalized as a generic algorithm which can work with any level of text granularity (keyword, phrase, sentence or paragraph levels). We also propose the integration of the retrieval and reuse stages in TCBR into a single architecture called Case Retrieval Reuse Net (CR2N) from which our text reuse strategy derives its name. The integration enables our algorithm to leverage the renowned effectiveness and efficiency of the Case Retrieval Net. We then extended CR2N's utility to guide retrieval of similar cases whose solutions are easiest to adapt during reuse.

Second contribution of our research is the introduction of two novel methods for text alignment and prototypical text generation during text reuse. Text alignment links structured problem attributes to specific chunks of a solution while prototypical text generation abstracts similar chunks of text into a single meaningful prototype. These concepts are generally applicable in domains where cases consist of pre-defined structured attributes along with written text. Transformational and compositional approaches to text reuse were then proposed to generate solution texts in response to a query with the predefined structured attributes. Our third and last contribution is the adaptation of two machine translation (MT) evaluation techniques, BLEU and NIST, for TCBR research. A novel introspective approach to generate multiple references when they do not naturally occur in a domain was also proposed to take full advantage of the capabilities of the MT metrics.

Desirable extensions to our contributions include the incorporation of better contextual knowledge within the CR2N architecture and devising methods to learn how to adapt by determining substitutes for terms annotated as irrelevant to a query. Methods to improve the seed generation process used in our text alignment by introspective learning will also increase the effectiveness of the technique.

Bibliography

Aamodt, A. & Plaza, E. (1994). Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, *Artificial Intelligence Communications, AICom* **7**(1): 39–59.

Adeyanju, I., Craw, S., Ghose, A., Gray, A. & Wiratunga, N. (2008). RaGoÛt: An Arpeggio of Tastes, *in* M. Schaaf (ed.), *ECCBR'08 workshop Proceedings (Computer Cooking Contest)*, Tharax-Verlag, Hildesheim, pp. 229–238.

Agirre, E. & Edmonds, P. (eds) (2006). *Word Sense Disambiguation: Algorithms and Applications*, Springer.

Ahmed, M. U., Begum, S., Funk, P., Xiong, N. & Schéele, B. V. (2008). Case-based Reasoning for Diagnosis of Stress using Enhanced Cosine and Fuzzy Similarity, *Transactions on Case-Based Reasoning for Multimedia Data* **1**(1): 3–19.

Arshadi, N. & Badie, K. (2000). A Compositional Approach to Solution Adaptation in Case-Based Reasoning and its Application to Tutoring Library, *Proceedings of 8th German Workshop on Case-Based Reasoning*.

Ash, R. B. (1990). *Information Theory*, Dover Publications.

Ashley, K., Lynch, C., Pinkwart, N. & Aleven, V. (2009). Towards Modeling and Teaching Legal Case-Based Adaptation with Expert Examples, *in* L. McGinty & D. C. Wilson (eds), *ICCBR 2009*, LNAI 5650, Springer-Verlag, Berlin Heidelberg, pp. 45–59.

Asiimwe, S., Craw, S., Wiratunga, N. & Taylor, B. (2007a). Automatic Text Standardisation by Synonym Mapping, *UKCBR'07 Workshop Proceedings*, pp. 88–98.

Asiimwe, S., Craw, S., Wiratunga, N. & Taylor, B. (2007b). Automatically Acquiring Structured Case Representations: The SMART Way, *in* M. Bramer, F. Coenen & T. Allen (eds), *Proceedings of the 27th SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence (AI-2007)*, Springer.

Asiimwe, S. M. (2009). *A Knowledge Acquisition Tool to Assist Case Authoring from Texts*, PhD thesis, The Robert Gordon University, Aberdeen.

Babych, B. & Hartley, A. (2004). Extending the BLEU MT evaluation method with frequency weightings, *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, Association for Computational Linguistics.

Badra, F., Bendaoud, R., Bentebibel, R., Champin, P.-A., Cojan, J., Cordier, A., Despres, S., J-Daubias, S., Lieber, J., Meilender, T., Mille, A., Nauer, E., Napoli, A. & Toussaint, Y. (2008). TAAABLE: Text Mining, Ontology Engineering and Hierarchical classification for Textual Case-Based Cooking, *in* M. Schaaf (ed.), *ECCBR 2008 Workshop Proceedings*, Tharax Verlag, pp. 219–228.

Badra, F., Cordier, A. & Lieber, J. (2009). Opportunistic Adaptation Knowledge Discovery, *in* L. McGinty & D. C. Wilson (eds), *ICCBR 2009*, LNAI 5650, Springer-Verlag, Berlin Heidelberg, pp. 60–74.

Baeza-Yates, R. & Ribeiro-Neto, B. (eds) (1999). *Modern Information Retrieval*, Addison Wesley.

Banerjee, S. & Lavie, A. (2005). METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments, *Proceedings of Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization at the 43th Annual Meeting of the Association of Computational Linguistics (ACL-2005)*, pp. 65–72.

Baram, Y., EL-Yaniv, R. & Luz, K. (2004). Online choice of active learning algorithms, *Journal of Machine Learning Reseach* **5**: 255–291.

Belz, A. (2005). Statistical generation: Three methods compared and evaluated, *10th European Workshop on Natural Language Generation (ENLG'05)*, pp. 15–23.

Belz, A. & Reiter, E. (2006). Comparing automatic and human evaluation in NLG, *11th Conference of the European Chapter of the Association for Computational Linguistics (EACL'06)*, pp. 313–320.

Bentebibel, R. & Despres, S. (2006). Using compositional and hierarchical adaptation in the SAARA system, *ECCBR'06 Workshop Proceedings (Reasoning with Text)*, pp. 98–108.

Bergmann, R., Kolodner, J. L. & Plaza, E. (2005). Representation in case-based reasoning, *The Knowledge Engineering Review* **20**(3): 1–4.

Bridge, D. & Waugh, A. (2009). Using experience on the read/write web: The GhostWriter system, *ICCBR09 Workshop Proceedings (WebCBR)*, pp. 15–24.

Brown, P., Della Pietra, S., Della Pietra, V. & Mercer, R. (1993). The mathematics of statistical machine translation: Parameter estimation, *Computational Linguistics* **19**(2): 263–311.

Brown, P. F., Pietra, S. A. D., Pietra, V. J. D. & Mercer, R. L. (1991). Word-sense disambiguation using statistical methods, *Proceedings of the 29th annual meeting on Association for Computational Linguistics*, pp. 264–270.

Brown, R. D. (1996). Example-based machine translation in the Pangloss system, *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, pp. 169–174.

Brüninghaus, S. & Ashley, K. D. (1998a). Evaluation of textual CBR approaches, *AAAI-98 Workshop on Textual Case-Based Reasoning*, AAAI Press, pp. 30–34.

Brüninghaus, S. & Ashley, K. D. (1998b). How machine learning can be beneficial for textual case-based reasoning, *AAAI-98 / ICML-98 Workshop on Learning for Text Categorisation (AAAI Technical Report WS-98-05)*, AAAI Press, pp. 71–74.

Brüninghaus, S. & Ashley, K. D. (2001). The Role of Information Extraction for Textual

CBR, *in* D. W. Aha, I. Watson & Q. Yang (eds), *ICCBR 2001*, Springer-Verlag, Berlin Heidelberg, pp. 74–89.

Brüninghaus, S. & Ashley, K. D. (2005). Reasoning with textual cases, *in* Munoz-Avila & R. Francesco (eds), *ICCBR 2005*, LNAI 3620, Springer-Verlag, Berlin Heidelberg, pp. 137–151.

Burke, R., Hammond, K., Kulyukin, V., Lytinen, S., Tomuro, N. & Schoenberg, S. (1997). Natural Language Processing in the FAQ Finder System: Results and Prospects, *Working Notes from AAAI Spring Symposium on NLP on the WWW*, pp. 17–26.

Chakraborti, S. (2007). *Introspective Knowledge Acquisition for Case Retrieval Networks in Textual Case Base Reasoning*, PhD thesis, The Robert Gordon University, Aberdeen.

Chakraborti, S., Lothian, R., Wiratunga, N., Orecchioni, A. & Watt, S. (2006). Fast Case Retrieval Nets for Textual Data, *in* T. Roth-Berghofer, M. H. Göker & H. A. Güvenir (eds), *ECCBR 2006*, Springer-Verlag, Berlin Heidelberg, pp. 400–414.

Chakraborti, S., Wiratunga, N., Lothian, R. & Watt, S. (2007). Acquiring word similarities with higher order association mining, *in* R. O. Weber & M. M. Ritcher (eds), *ICCBR 2007*, LNAI 4626, Springer-Verlag, pp. 61–76.

Chang, C.-G., Cui, J.-J., Wang, D.-W. & Hu, K.-Y. (2004). Research on case adaptation techniques in case-based reasoning, *ICMLC'04*, pp. 2128–2133.

Cojan, J. & Lieber, J. (2008). Conservative adaptation in metric spaces, *in* K.-D. Althoff, R. Bergmann, M. Minor & A. Hanft (eds), *ECCBR 2008*, LNAI 5239, Springer-Verlag, Berlin Heidelberg, pp. 135–149.

Cordier, A., Fuchs, B., de Carvalho, L. L., Lieber, J. & Mille, A. (2008). Opportunistic acquisition of adapatation knowledge and cases- the IAKA approach, *in* K.-D. Althoff, R. Bergmann, M. Minor & A. Hanft (eds), *ECCBR 2008*, LNAI 5239, Springer-Verlag, Berlin Heidelberg, pp. 150–164.

Cowie, J. & Lehnert, W. (1996). Information extraction, *Communications of ACM* **39**(1): 80–91.

Craw, S., Wiratunga, N. & Rowe, R. C. (2006). Learning adaptation knowledge to improve case-based reasoning, *Artificial Intelligence* **170**: 1175–1192.

Cunningham, C., Weber, R., Proctor, J., Fowler, C. & Murphy, M. (2004). Investigating graphs in textual case-based reasoning, *in* P. Funk & P. A. Gonzalez-Calero (eds), *ECCBR'04*, LNAI 3155, Springer Verlag, Heidelberg-Berlin, pp. 573–586.

Cunningham, H., Maynard, D., Bontcheva, K. & Tablan, V. (2002). GATE: A framework and graphical development environment for robust NLP tools and applications, *40th Anniversary Meeting of the Association for Computational Linguistics (ACL '02)*.

Cunningham, P. (1998). CBR: Strengths and weaknesses, *in* A. P. Pobil, J. Mira & A. Moonis (eds), *11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA-98-AIE)*, LNAI 1416, Springer-Verlag, Berlin Heidelberg, pp. 517–524.

D'Aquin, M., Badra, F., Lafrogne, S., Lieber, J., Napoli, A. & Szathmary, L. (2007). Case Base Mining for Adaptation Knowledge Acquisition, *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07)*, pp. 750–755.

De Marneffe, M.-C., MacCartney, B. & Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses, *International Conference on Language Resources and Evaluation (LREC-06)*.

De Silva Garza, A. G. & Maher, M. L. (1999). An evolutionary approach to case adaptation, *in* K. D. Althoff, R. Bergmann & K. L. Branting (eds), *ICCBR '99*, LNAI 1650, Springer Verlag, Berlin Heidelberg, pp. 162–172.

Deerwester, S., Dumais, S., Furnas, G., Landuer, T. & Harshman, R. (1990). Indexing by latent semantic analysis, *Journal of the American Society of Information Science* **41**(6): 391–407.

Delany, S. J., Cunningham, P. & Coyle, L. (2005). An assessment of case-based reasoning for spam filtering, *Artificial Intelligence Review Journal* **24**(3–4): 359–378.

DeMiguel, J., Plaza, L. & Díaz-Agudo, B. (2008). ColibriCook: A CBR system for ontology-based recipe retrieval and adaptation, *in* M. Schaaf (ed.), *ECCBR 2008 Workshop Proceedings*, Tharax Verlag, pp. 199–208.

Díaz-Agudo, B., Gervás, P. & González-Calero, P. A. (2003). Adaptation guided retrieval based on formal concept analysis, *in* K. D. Ashley & D. G. Bridge (eds), *ICCBR 2003*, LNAI 2689, Springer Verlag, Berlin-Heidelberg, pp. 131–145.

Díaz-Agudo, B., González-Calero, P. A., Recio-García, J. A. & Sánchez, A. (2007). Building CBR systems with jCOLIBRI, *Special Issue on Experimental Software and Toolkits of the Journal Science of Computer Programming* **69**: 68–75.

Díaz-Agudo, B., Plaza, E., Recio-García, J. A. & Arcos, J.-L. (2008). Noticeably new: Case reuse in originality-driven tasks, *in* K.-D. Althoff, R. Bergmann, M. Minor & A. Hanft (eds), *ECCBR 2008*, LNAI 5239, Springer-Verlag, Berlin Heidelberg, pp. 165–179.

Doddington, G. (2002). Automatic evaluation of machine translation quality using n-gram co-occurence statistics, *Proceedings of the 2nd International Conference on Human Language Technology*, Morgan Kaufmann, pp. 138–145.

Domeshek, E. & Kolodner, J. L. (1991). Towards a case-based aid for conceptual design, *International Journal of Expert Systems* **4**: 201–220.

Domeshek, E. & Kolodner, J. L. (1992). A case-based design aid for architecture, *in* J. Gero (ed.), *Proceedings of the 2nd International Conference on Artificial Inteliligence in Design*, Kluwer Academic, Norwell, MA, pp. 497–516.

Domeshek, E. & Kolodner, J. L. (1993). Finding the points of large cases, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing (AIEDAM)* **7**(2): 87–96.

Dufour-Lussier, V., Lieber, J., Nauer, E. & Toussaint, Y. (2010). Text adaptation using formal concept analysis, *in* I. Bichindaritz & S. Montani (eds), *ICCBR 2010*, LNAI 6176, Springer-Verlag, Berlin Heidelberg, pp. 96–110.

Fellbaum, C. (ed.) (1998). *WordNet: An Electronic Lexical Database*, MIT press.

Ferguson, W., Bareiss, R., Birnbaum, L. & Osgood, R. (1992). ASK systems: An approach to the realization of story-based teachers, *Journal of the Learning Sciences* **2**: 95–134.

Gervás, P. (2001). Automatic Generation of Poetry using a CBR Approach, *Proceedings of the Conference of the Spanish Association for Artificial Intelligence (CAEPIA).*

Gervás, P., Díaz-Agudo, B., Peinado, F. & Hervás, R. (2004). Story plot generation based on CBR, *in* A. Macintosh, R. Ellis & T. Allen (eds), *Twelveth Conference on Applications and Innovations in Intelligent Systems*, Springer, Cambridge, UK.

Gervás, P., Hervás, R. & Recio-García, J. A. (2007). The role of natural language generation during adaptation in textual CBR, *in* D. G. Bridge, P. Gomes, N. Seco, D. C. Wilson & D. Khemani (eds), *ICCBR'07 Workshop proceedings on Textual CBR: Beyond Retrieval*, ICCBR, pp. 227–236.

González-Calero, P. A., Gómez-Albarran, M. & Díaz-Agudo, B. (1999). A substitution-based adaptation model, *ICCBR'99 workshop proceedings (Challenges for CBR).*

Goodman, M. (1989). CBR in battle planning, *Proceedings of Workshop on case-based reasoning (DARPA)*, Morgan Kaufmann, San Mateo, CA.

Goodrich, H. (1996). Understanding Rubrics, *Educational Leadership* **54**(4): 14–18.

Hanft, A., Ihle, N., Bach, K., Newo, R. & Mänz (2008). Realising a CBR-based approach for Computer Cooking Contest with e:IAS, *in* M. Schaaf (ed.), *ECCBR 2008 Workshop Proceedings*, Tharax Verlag, pp. 249–258.

Hanney, K. & Keane, M. T. (1997). The adaptation knowledge bottleneck: How to ease it by learning from cases, *ICCBR 1997*, Springer verlag, pp. 359–370.

Healy, P. & Bridge, D. (2010). The GhostWriter-2.0 System: Creating a Virtuous Circle in Web 2.0 Product Reviewing, *in* C. Marling (ed.), *ICCBR'10 Workshop Proceedings*, University of Piemonte Orientale, pp. 121–130.

Hennessy, D. & Hinkle, D. (1992). Applying case-based reasoning to autoclave loading, *IEEE Expert* **7**(5): 21–26.

Hovy, E. H. (1999). Toward finely differentiated evaluation metrics for machine translation, *Proceedings of the Eagles Workshop on Standards and Evaluation*.

Hu, R., Delany, S. J. & Mac Namee, B. (2010). EGAL: Exploration Guided Active Learning for TCBR, *in* I. Bichindaritz & S. Montani (eds), *ICCBR 2010*, LNAI 6176, Springer-Verlag, Berlin Heidelberg, pp. 156–170.

Iglezakis, I., Reinartz, T. & Roth-Berghofer, T. (2004). Maintenance memories: Beyond concepts and techniques for case base maintenance, *ECCBR '04*, LNAI 3155, Springer-Verlag, Berlin-Heidelberg, pp. 227–241.

Kamp, G., Lange, S. & Globig, C. (1998). *Case-Based Reasoning Technology- From Foundations to Applications*, LNAI 1400, Springer-Verlag, Berlin-Heidelberg, chapter 13-Related Areas, pp. 327–351.

Koehn, P. (2010). *Statistical Machine Translation*, Cambridge University Press.

Kolodner, J. L. (1993). *Case-Based Reasoning*, Morgan Kaufmann, San Mateo, CA.

Kunze, M. & Hubner, A. (1998). CBR on Semi-structured Documents: The ExperienceBook and the FAllQ Project, *in* L. Gierl & M. Lenz (eds), *Proceedings of the Sixth German Workshop on Case-Based Reasoning*, IMIB 7, University of Rostock, pp. 77–85.

Lamontagne, L. (2006). Textual CBR authoring using case cohesion, *Reasoning with Text: Proceedings of the ECCBR06 Workshops (TCBR 06)*, pp. 33–43.

Lamontagne, L., Bentebibel, R., Miry, E. & Despres, S. (2007). Finding lexical relationships for the reuse of investigation reports, *ICCBR07 Workshop Proceedings (Textual Case-Based Reasoning: Beyond Retrieval)*.

Lamontagne, L., Langlais, P. & Lapalme, G. (2003). Using statistical models for the retrieval of fully-textual cases, *in* I. Rusell & S. Haller (eds), *Proceedings of FLAIRS-2003*, AAAI press, pp. 124–128.

Lamontagne, L. & Lapalme, G. (2003). Applying case-based reasoning to email response, *ICEIS '03 Proceedings*, pp. 115–123.

Lamontagne, L. & Lapalme, G. (2004). Textual reuse for email response, *in* P. Funk & P. A. González-Calero (eds), *ECCBR 2004*, LNAI 3155, Springer, Heidelberg, pp. 234–246.

Lavie, A. & Agarwal, A. (2007). METEOR: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments, *In Proceedings of the Second ACL Workshop on Statistical Machine Translation*, pp. 228–231.

Lavie, A. & Denkowski, M. (2009). The METEOR Metric for Automatic Evaluation of Machine Translation, *Machine Translation Journal (special issue)* **1**: 1.

Leake, D. B. (ed.) (1996). *Case-Based Reasoning: Experiences, Lessons & Future Directions*, AAAI Press/ MIT press.

Leake, D. & Kendall-Morwick, J. (2009). Four heads are better than one: Combining suggestions for case adaptation, *in* L. McGinty & D. C. Wilson (eds), *ICCBR 2009*, LNAI 5650, Springer-Verlag, Berlin Heidelberg, pp. 165–179.

Leake, D., Kinley, A. & Wilson, D. (1995). Learning to improve case adaptation by introspective reasoning and CBR, *in* M. Veloso & A. Aamodt (eds), *ICCBR 1995*, Springer-Verlag, pp. 229–240.

Leake, D. & Powell, J. (2007). Mining large-scale knowledge sources for case adaptation knowledge, *in* R. O. Weber & M. M. Ritcher (eds), *ICCBR 2007*, LNAI 4626, Springer-Verlag, Berlin Heidelberg, pp. 209–223.

Leake, D. & Powell, J. (2008). Knowledge planning and learned personalization for web-based case adaptation, *in* K.-D. Althoff, R. Bergmann, M. Minor & A. Hanft (eds), *ECCBR 2008*, LNAI 5239, Springer-Verlag, Berlin Heidelberg, pp. 284–298.

Leake, D. & Powell, J. (2010). A general introspective reasoning approach to web search for case adaptation, *in* I. Bichindaritz & S. Montani (eds), *ICCBR 2010*, LNAI 6176, Springer-Verlag, Berlin Heidelberg, pp. 186–200.

Lee-Urban, S. & Munoz-Avila, H. (2009). Adaptation versus retrieval trade-off revisited: An analysis of boundary conditions, *in* L. McGinty & D. C. Wilson (eds), *ICCBR 2009*, LNAI 5650, Springer-Verlag, Berlin Heidelberg, pp. 180–194.

Lenz, M. (1998a). Knowledge sources for textual CBR applications, *in* M. Lenz & K. Ashley (eds), *Workshop notes of the AAAI-98 Workshop on Textual Case-Based Reasoning*, AAAI Press, pp. 24–29.

Lenz, M. (1998b). Textual CBR and information retrieval - a comparison, *in* L. Gierl & M. Lenz (eds), *Proceedings of the Sixth German Workshop on Case-Based Reasoning*.

Lenz, M., Bartsch-Sporl, B., Burkhard, H.-D. & Wess, S. (eds) (1998). *Case-Based Reasoning Technology- From Foundations to Applications*, Springer-Verlag, Berlin.

Lenz, M. & Burkhard, H.-D. (1996a). Case retrieval nets: Basic ideas and extensions, *in* G. Gorz & S. Holldobler (eds), *20th Annual German Conference on Artificial Intelligence (KI-96)*, LNAI 1137, Springer, Heidelberg, pp. 227–239.

Lenz, M. & Burkhard, H.-D. (1996b). Case retrieval nets: Foundations, properties, implementations and results, *Technical report*, Humboldt University, Berlin.

Lenz, M. & Burkhard, H.-D. (1997). CBR for Document Retrieval - The FALLQ Project, *in* D. B. Leake & E. Plaza (eds), *ICCBR'97*, LNAI 1266, Springer-Verlag, Berlin Heidelberg, pp. 84–93.

Lenz, M., Hubner, A. & Kunze, M. (1998). Question answering with textual CBR, *in* T. Andreasen, H. Christiansen & H. L. Larsen (eds), *International Conference on*

Flexible Query Answering Systems (FQAS 98), LNAI 1495, Springer, Heidelberg, pp. 236–247.

Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions and reversals, *Soviet Physics- Doklady, Cynernetics and Control theory* **10**(8): 707–710.

Lieber, J. (2007). Application of the revision theory to adaptation in case-based reasoning: The conservative adaptation, *in* R. O. Weber & M. M. Ritcher (eds), *ICCBR 2007*, Springer-Verlag, Heidelberg, pp. 239–253.

Lin, C.-Y. & Hovy, E. (2003). Automatic evaluation of summaries using n-gram co-occurence statistics, *In proceedings of the Human Technology Conference (HLT-NAACL 03)*, pp. 71–78.

Liu, T., Liu, S., Chen, Z. & Ma, W.-Y. (2003). An evaluation on feature selection for text clustering, *Proceedings of the Twentieth International Conference on Machine Learning (ICML'03)*, Springer- Verlag press, Berlin Heidelberg, pp. 806–820.

Lopez de Mantaras, R., McSherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., Faltings, B., Maher, M., Cox, M., Forbus, K., Keane, M., Amodt, A. & Watson, I. (2005). Retrieval, reuse, revision and retention in case-based reasoning, *Knowledge Engineering Review* **20**: 215–240.

Manning, C. D. & Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*, The MIT Press, Cambridge, Massachusetts.

Mansilla, V., Duraisingh, E., Wolfe, C. R. & Haynes, C. (2009). Targeted assessment rubric: An empirically grounded rubric for interdisciplinary writing, *Journal of Higher Education* **80**(3): 334–353.

Massie, S. (2006). *Complexity Modelling for Case Knowledge Maintenance in Case-Based Reasoning*, PhD thesis, The Robert Gordon University, Aberdeen.

Massie, S., Wiratunga, N., Craw, S., Donati, A. & Vicari, E. (2007). From anomaly

reports to cases, *in* K. D. Ashley & D. G. Bridge (eds), *ICCBR'07*, Springer-Verlag, Berlin-Heidelberg, pp. 359–373.

Minor, M., Bergmann, R., Grg, S. & Walter, K. (2010). Towards case-based adaptation of workflows, *in* I. Bichindaritz & S. Montani (eds), *ICCBR 2010*, LNAI 6176, Springer-Verlag, Berlin Heidelberg, pp. 421–435.

Mitra, M., Singhal, A. & Buckley, C. (1997). Automatic text summarization by paragraph extraction, *Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization*.

Muñoz Avila, H. & Cox, M. T. (2008). Case-based plan adaptation: An analysis and review, *IEEE Intelligent Systems* **23**(4): 75–81.

Mudambi-Ananthasayanam, R., Chakraborti, S. & Khemani, D. (2009). Robust measures of complexity in TCBR, *in* L. McGinty & D. Wilson (eds), *ICCBR 2009*, LNAI 5650, Springer-Verlag, Berlin Heidelberg, pp. 270–284.

Mudambi-Ananthasayanam, R., Wiratunga, N., Chakraborti, S., Massie, S. & Khemani, D. (2008). Evaluation measures for TCBR systems, *in* K.-D. Althoff, R. Bergmann, M. Minor & A. Hanft (eds), *ECCBR 2008*, LNAI 5239, Springer-Verlag, Berlin Heidelberg, pp. 444–458.

Nagao, M. (1984). A framework of a mechanical translation between japanese and english by analogy principle, *Proceedings of the international NATO symposium on Artificial and human intelligence*, pp. 173–180.

Navigli, R. (2009). Word sense disambiguation: A survey, *ACM Computing Surveys (CSUR)* **41**(2): 1–69.

Nenadic, G., Spasic, I. & Ananiadou, S. (2004). Automatic discovery of term similarities using pattern mining, *International Journal of Terminology* **10**: 55–80.

Neto, J. L., Freitas, A. A. & Kaestner, C. A. A. (2002). Automatic text summarization using a machine learning approach, *in* G. Bittencourt & G. L. Ramalho (eds), *16th*

Brazilian Symposium on Artificial Intelligence (SBIA 2002), LNAI 2507, Springer-verlag, Berlin Heidelberg, pp. 205–215.

Nouaouria, N. & Boukadoum, M. (2010). Case retrieval with combined adaptability and similarity criteria: Application to case retrieval nets, *in* I. Bichindaritz & S. Montani (eds), *ICCBR 2010*, LNAI 6176, Springer-Verlag, Berlin Heidelberg, pp. 242–256.

O'Mahony, M. P. & Smyth, B. (2009). A classification-based review recommender, *in* M. Bramer, R. Ellis & M. Petridis (eds), *Twenty-ninth SGAI International Conference on Artificial Intelligence (AI-2009)*, Springer, London, pp. 49–62.

Ontanón, S. & Plaza, E. (2010). Amalgams: A formal approach for combining multiple case solutions, *in* I. Bichindaritz & S. Montani (eds), *ICCBR 2010*, LNAI 6176, Springer-Verlag, Berlin Heidelberg, pp. 257–271.

Orecchioni, A., Wiratunga, N., Massie, S., Chakraborti, S. & Mukras, R. (2007). Learning incident causes, *ICCBR'07 Workshop Proceedings*, pp. 247–251.

Orecchioni, A., Wiratunga, N., Massie, S. & Craw, S. (2008). k-nn aggregation with a stacked email representation, *in* K. D. Althoff, R. Bergmann, M. Minor & A. Hanft (eds), *ECCBR'08 Proceedings*, LNAI 5239, Springer Verlag, Berlin-Heidelberg, pp. 415–429.

Palshikar, G. K. (2007). Keyword extraction from single document using centrality measures, *in* A. Ghosh, R. De & S. Pal (eds), *Second International Conference on Pattern Recognition and Machine Intelligence (PReMI 2007)*, LNAI 4815, Springer-Verlag, Berlin Heidelberg, pp. 503–510.

Papineni, K., Roukos, S., Ward, T. & Zhu, W.-J. (2002). BLEU: A method for automatic evaluation of machine translation, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 311–318.

Patterson, D., Anand, S., Dubitzky, D. & Hughes, J. (2000). A knowledge light approach to similarity maintenance for improving case-based competence, *Workshop on Flexible*

Strategies for Maintaining Knowledge Containers at 14th European Conference on Artificial Intelligence, pp. 65–77.

Patterson, D., Rooney, N., Galushka, M. & Anand, S. (2002). Towards dynamic maintenance of retrieval knowledge in CBR, *in* S. M. Haller & G. Simmons (eds), *Fifteenth International Florida Artificial Intelligence Research Society Conference*, AAAI Press, pp. 126–131.

Plaza, E. (2008). Semantics and experience in the future web, *in* K. D. Althoff, R. Bergmann, M. Minor & A. Hanft (eds), *ECCBR'08 Proceedings*, LNAI 5239, Springer Verlag, Berlin-Heidelberg, pp. 44–58.

Plaza, E. & Arcos, J.-L. (2002). Constructive adaptation, *in* S. Craw & A. D. Preece (eds), *ECCBR'02 Proceedings*, LNAI 2416, Springer Verlag, Berlin-Heidelberg, pp. 306–320.

Purvis, L. & Pu, P. (1995). Adaptation using constraint satisfaction techniques, *ICCBR'95*, Springer-Verlag, London, UK, pp. 289–300.

Reategui, E., Campbell, J. & Leaob, B. (1996). Combining a neural network with case-based reasoning in a diagnostic system, *Artificial Intelligence in Medicine* **9**: 5–27.

Recio-García, J. A., Díaz-Agudo, B. & González-Calero, P. A. (2007). Textual CBR in jCOLIBRI: From retrieval to reuse, *ICCBR'07 workshop proceedings (Textual CBR)*, pp. 217–226.

Recio-García, J. A., Díaz-Agudo, B., Sánchez-Ruiz, A. & González-Calero, P. A. (2006). Lessons Learnt in the Development of a CBR Framework, *in* M. Petridis (ed.), *UK CBR Workshop Proceedings*, CMS Press,University of Greenwich, pp. 60–71.

Recio-García, J. A. & Wiratunga, N. (2010). Taxonomic semantic indexing, *in* I. Bichindaritz & S. Montani (eds), *ICCBR 2010*, LNAI 6176, Springer, Berlin.

Reiter, E. & Dale, R. (1995). Building applied natural language generation systems, *Natural Language Engineering* **1**: 1–32.

Richter, M. M. (1998). Introduction, *Case-Based Reasoning Technology: From Foundations to Applications*, LNAI 1400, Springer, Berlin/ Heidelberg, pp. 1–15.

Rubin, J. & Watson, I. (2010). Similarity-based retrieval and solution re-use policies in the game of texas hold'em, *in* I. Bichindaritz & S. Montani (eds), *ICCBR 2010*, LNAI 6176, Springer-Verlag, Berlin Heidelberg, pp. 465–479.

Sánchez-Ruiz, A. A., Gómez-Martín, P. P., Díaz-Agudo, B. & González-Calero, P. A. (2008). Adaptation through planning in knowledge intensive CBR, *in* K.-D. Althoff, R. Bergmann, M. Minor & A. Hanft (eds), *ECCBR 2008*, LNAI 5239, Springer-Verlag, Berlin Heidelberg, pp. 503–517.

Selvamani, R. B. & Khemani, D. (2005). Decision tree induction with CBR, *in* S. K. Pal, S. Bandyopadhyay & S. Biswas (eds), *Pattern Recognition and Machine Intelligence (PReMI 05)*, LNAI 3776, Springer-Verlag, Berlin Heidelberg, pp. 786–791.

Shannon, C. E. (1948). A mathematical theory of communication, *Bell System Technical Journal* **27**: 379–423, 623–656.

Simoudis, E. (1991). Knowledge acquisition in validated retrieval, *International Journal of Expert Systems: Research and Applications* **4**(3): 299–315.

Simoudis, E. (1992). Using case-based retrieval for customer technical support, *IEEE Expert* **7**(5): 7–13.

Smyth, B. & Cunningham, P. (1993). Complexity of adaptation in real-world case-based reasoning systems, *Proceedings of the Sixth Irish conference on Artificial Intelligence & Cognitive Science*, Queens University Press, Belfast, pp. 228–240.

Smyth, B. & Keane, M. (1994). Retrieving adaptable cases: The role of adaptation knowledge in case retrieval, *Topics in Case-Based Reasoning*, Springer-Verlag.

Smyth, B. & Keane, M. (1996). Adaptation-guided retrieval: Using adaptation knowledge to guide the retrieval of adaptable cases, *UK CBR Workshop Proceedings*, pp. 2–15.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L. & Makhoul, J. (2006). A study of translation edit rate with targeted human annotation, *Proceedings of Association for Machine Translation in the Americas*.

Sparck-Jones, K. (1999). Automatic summarizing: factors and directions, *in* I. Mani & M. Maybury (eds), *Advances in Automatic Text Summarization*, pp. 1–12.

Sripada, S. G., Reiter, E., Hunter, J. & Yu, J. (2002). SUMTIME-METEO: Parallel Corpus of Naturally Occurring Forecast Texts and Weather Data, *Technical Report AUCS/TR0201*, Department of Computer Science, University of Aberdeen.

Sripada, S., Reiter, E. & Davy, I. (2003). SumTime-Mousam: Configurable marine weather forecast generator, *Expert Update* **6**(3): 4–10.

Sripada, S., Reiter, E. & Hawizy, L. (2005). Evaluation of an NLG system using post-edit data: Lessons learnt, *Proceedings of European Natural Language Generation Workshop*, pp. 133–139.

Sugandh, N., Ontanón, S. & Ram, A. (2008). Real-time plan adaptation for case-based planning in real-time strategy games, *in* K.-D. Althoff, R. Bergmann, M. Minor & A. Hanft (eds), *ECCBR 2008*, LNAI 5239, Springer-Verlag, Berlin Heidelberg, pp. 533–547.

Sumita, E., Lida, H. & Kohyama, H. (1990). Translating with examples: A new approach to machine translation, *Proceedings of the 3rd International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages*.

Sycara, E. (1987). Resolving adversarial conflicts: An approach to integrating case-based and analytic methods, *Technical Report GIT-ICS-87/26*, Georgia Institute of Technology, School of Information and Computer Science.

Sycara, K. (1988). Patching up old plans, *Proceedings of the Tenth Annual Conference of the Cognitive Science Society*, Erlbaum, Northvale, NJ.

Watson, I. D. (1997). *Applying Case-Based Reasoning: Techniques for Enterprise Systems*, Morgan Kaufmann, San Francisco, CA.

Weber, R., Martin, A. & Barcia, R. (1998). On legal texts and cases, *in* M. Lenz & K. D. Ashley (eds), *Textual Case-Based Reasoning: Papers from the AAAI-98 Workshop*, AAAI Press, pp. 40–50.

Weber, R. O., Ashley, K. D. & Bruninghaus, S. (2006). Textual case-based reasoning, *Knowledge Engineering Review* **20**(3): 255–260.

Wess, S., dieter Althoff, K. & Derwand, G. (1994). Using k-d trees to improve the retrieval step in case-based reasoning, *in* S. Wess, K.-D. Althoff & M. M. Richter (eds), *Topic in Case Based Reasoning*, LNAI 837, Springer-Verlag, Berlin Heidelberg, pp. 167–181.

White, J. & Connell, T. (1994). The ARPA MT evaluation methodologies: evolution, lessons and future approaches, *Proceedings of the First Conference of the Association for Machine Translation in the Americas*, pp. 193–205.

Wilke, W. & Bergmann, R. (1998). Techniques and knowledge used for adaptation during case-based problem solving, *in* A. P. Pobil, J. Mira & A. Moonis (eds), *IEA-AIE'98 Proceedings*, LNAI 1416, Springer-Verlag, Berlin Heidelberg.

Wille, R. (1992). Conceptual lattices and conceptual knowledge systems, *Computers and Mathematics with Apps* **23**: 6–9.

Wilson, D. (2000). CBR Textuality, *Expert Update* **3**(1): 28–37.

Wilson, D. C., Carthy, J., Abbey, K., Sheppard, J., Wang, R., Dunnion, J. & Drummond, A. (2003). Textual CBR for Incident Report Retrieval, *Computational Science and Its Applications (ICCSA 2003)*, LNCS 2667, Springer, Heidelberg, pp. 358–367.

Wilson, D. R. & Martinez, T. R. (1997). Improved heterogeneous distance functions, *Journal of Artificial Intelligence Research* **6**: 1–34.

Wiratunga, N., Koychev, I. & Massie, S. (2004). Feature selection and generalisation

for retrieval of textual cases, *in* P. Funk & P. Gonzalez Calero (eds), *ECCBR 2004*, Springer Verlag, pp. 806–820.

Wiratunga, N., Lothian, R., Chakraborti, S. & Koychev, I. (2005). A propositional approach to textual case indexing, *Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery and Data Mining (PKDD)*, Springer, pp. 380–391.

Wiratunga, N., Lothian, R. & Massie, S. (2006). Unsupervised feature selection for text data, *in* T. Roth-Berghofer, M. H. Göker & H. A. Güvenir (eds), *ECCBR 2006*, Springer-Verlag, Berlin-Heidelberg, pp. 340–354.

Yang, Y. & Pedersen, J. O. (1997). A comparative study of feature selection in text categorisation, *in* D. A. Fisher (ed.), *Proceedings of the 14th International Conference on Machine Learning ICML-97*, Morgan Kaufmann, San Francisco, US, pp. 412–420.

Yarowsky, D. (2000). Word-sense disambiguation using statistical models of roget's categories trained on large corpora, *in* R. Dale, H. Moisl & H. Somers (eds), *Handbook of Natural Language Processing*, Marcel Dekker, pp. 629–654.

Zhang, Q., Hu, R., Namee, B. M. & Delany, S. J. (2008). Back to the future: Knowledge light case base cookery, *in* M. Schaaf (ed.), *ECCBR 2008 Workshop Proceedings*, Tharax Verlag, pp. 239–248.

Zhang, Y., Brown, R. D. & Frederking, R. E. (2001). Adapting an example-based translation system to chinese, *In Proceedings of HLT 2001: First International Conference on Human Language Technology Research*, pp. 7–10.

Zhang, Y., Louvieris, P. & Petrou, M. (2007). Case-based reasoning adaptation for high dimensional solution space, *in* R. O. Weber & M. M. Ritcher (eds), *ICCBR 2007*, Springer-Verlag, Berlin Heidelberg, pp. 149–163.

Zhao, W., Chellapa, R. & Krishnaswamy, A. (1998). Discriminant analysis of principal components for face recognition, *Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition*, pp. 336–341.

Published Papers

- Ibrahim Adeyanju, Nirmalie Wiratunga, Juan Recio-García, Robert Lothian (2010). Learning to Author text with textual CBR. *in* H. Coelho, R. Studer and M. Wooldridge (eds), *Proceedings of ECAI 2010*. IOS Press, Amsterdam, pp. 777-782.

- Ibrahim Adeyanju, Nirmalie Wiratunga, Robert Lothian, Susan Craw (2010). Applying Machine Translation Evaluation Techniques to Textual CBR. *in* I. Bichindaritz and S. Montani (eds), *Proceedings of ICCBR 2010*, LNAI 6176. Springer-Verlag, Berlin Heidelberg, pp. 21-35.

- Ibrahim Adeyanju, Nirmalie Wiratunga, Robert Lothian, Somayajulu Sripada, Luc Lamontagne (2009). Case Retrieval Reuse Net (CR2N): An Architecture for Reuse of Textual Solutions, *in* L. McGinty and D.C. Wilson (eds), *Proceedings of ICCBR 2009*, LNAI 5650, Springer-Verlag, Berlin Heidelberg, pp. 14-28.

- Ibrahim Adeyanju, Nirmalie Wiratunga, Robert Lothian, Somayajulu Sripada, Susan Craw (2008). Solution reuse for textual cases, *in* M. Petridis (ed.), *Proceedings of UKCBR Workshop 2008*, CMS Press, Greenwich, pp. 54-62.

- Ibrahim Adeyanju, Susan Craw, Abhishek Ghose, Allyson Gray, Nirmalie Wiratunga (2008). RaGoUt: An Arpeggio of Tastes, *in* M. Schaaf (ed.), *ECCBR'08 Workshop Proceedings (Computer Cooking Contest)*, Tharax-Verlag, Hildesheim, pp. 229-238.

Appendix A

Complete seeds list generated for text authoring

This relates to our work on the transformational and compositional approaches to text reuse discussed in Chapter 4. Here, we provide the complete list of seeds used experiments as detailed in Chapter 7.

A.1 Hotel reviews

1. **Cleanliness rating:** cleanliness, clean, clean-up, cleanse, clear, contaminate, contaminating, decontaminate, dirt, dirty, disinfect, dry-clean, dust, filth, filthy, fresh, hoover, houseclean, hygiene, hygienise, hygienize, impure, kempt, neat, neaten, neatness, sanitise, sanitize, soil, soiled, sweep, tidy, unclean, unclouded, uncontaminating, uninfected, unobjectionable, tidy,tidiness, uncleanliness, unclouded, uncontaminating, uninfected, unobjectionable, vacuum, wash, wash-up

2. **Location rating:** emplacement, home, location, locality, position, place, region

3. **Room rating:** accommodation, bath, bathroom, bed, bedchamber, bedroom, ceiling, chair, chamber, cloakroom, closet, coatroom, compartment, dinette, dining room, door, dressing room, dwell, floor, flooring, hall, kitchen, lav, lavatory, library, living room, lobby, locker room, lounge, parlor, parlour, reading room, seating, seats,

shower, sitting room, sleeping, study, table, television, toilet, tv room, waiting area

4. **Service rating:** activity, aid, assist, assistance, avail, bringing, care, delivery, dinner, disservice, facility, fault, help, helpdesk, helpfulness, ill service, inspection, maintenance, parking, repair, reception, serve, service, servicing, serving, star, standard, standardise, standardize, upkeep, utility

5. **Value rating:** admire, amount, appraise, appreciate, assess, censor, cost, disesteem, disrespect, esteem,evaluate, grade,invaluable, overvalue, price, priceless, rate, undervalue, valuable, valuate, value, worth, worthless

A.2 Academic Coursework Feedback

1. **Task1.1:** 1.1, class, attribute, numeric, distribution, skewed, nominal

2. **Task1.2:** 1.2, tree, generate, depth, parameter, attribute

3. **Task2.1:** 2.1, parameter, experimental, design, evaluation, methodology, cross, validation, folds, comparison, criteria, percent, correct, parameter, classifier, trials, repetitions, significance, level, dataset

4. **Task2.2:** 2.2, classifier, result, comparison, baseline, significance

5. **Task3.1:** 3.1, boost, classifier, evaluation, methodology, trials, effectiveness, criteria, efficiency

6. **Task3.2:** 3.2, effectiveness, comparison, result, explanation, efficiency, explanation, significance

7. **Task3.3:** 3.3, discussion, presentation

8. **Task4:** 4, arff, preamble, extend, new, value

9. **Overall:** overall, submission, incomplete, solution

Appendix B

Approval by TripAdvisor for use of their web contents for Research

We provide evidence here that permission was sought and obtained before the hotel review dataset used for our experiments (details in Sections 2.4.3 and 7.2) was crawled from the TripAdvisor website. The email correspondence during the approval process are shown verbatim below.

From: cmieth@tripadvisor.com on behalf of
copyrightissues@tripadvisor.com
Sent: 13 August 2009 15:27
To: IBRAHIM ADEYANJU (0612836)
Subject: Re: PERMISSION TO UTILIZE REVIEWS ON YOUR WEBSITE FOR RESEARCH PURPOSE [ref:00D8LC7N.50086k9vf:ref]

Dear Ibrahim,

Thank you for submitting your permission request.

Unfortunately, there was a delay in my responding to your request, because these requests are handled through

copyrightissues@tripadvisor.com (as is stated at the
bottom of the permissions request form.)

We do allow the use of our content for educational purposes
as is outlined in your request form.

If you have any further questions, please contact me directly
at this email address.

Best Regards,

Corie Mieth
Paralegal
TripAdvisor LLC
www.tripadvisor.com

Get the truth. Then Go.
* Confidentiality Notice: This e-mail and any associated files are
intended solely for the individual or entity to whom they are
addressed. Please do not copy it or use it for any purposes, or disclose
its contents to any other person. Further, this e-mail and any associated
files may be confidential and further may be legally privileged.
If you have received this email in error, please notify us by reply
email and then delete the message from your system.
Thanks for your cooperation.

Begin forwarded message:
From: <i.adeyanju@rgu.ac.uk>
Date: August 13, 2009 5:01:05 AM PDT

To: <partners@tripadvisor.co.uk>, <support@tripadvisor.com>

Subject: FW: PERMISSION TO UTILIZE REVIEWS ON YOUR WEBSITE FOR RESEARCH PURPOSE [ref:00D8LC7N.50086k9vf:ref]

Hi,

I am yet to get any response/ acknowledgement from you as regards my request. Please, I will greatly appreciate your acknowledgement of this mail.

Find chain of earlier mails exchanged below.

Thanks.

Regards,

Ibrahim Adeyanju

Research Student,

School of Computing,

The Robert Gordon University,

St. Andrews Street,

Aberdeen, AB25 1HG,

United Kingdom.

Tel: +44(0) 1224-262577

-----Original Message-----

From: IBRAHIM ADEYANJU (0612836)

Sent: 20 July 2009 11:17

To: 'TripAdvisor Support'

Subject: RE: PERMISSION TO UTILIZE REVIEWS ON
YOUR WEBSITE FOR RESEARCH PURPOSE [ref:00D8LC7N.50086k9vf:ref]

Hi,

I tried to fax the completed form to the number given on it
(+16176706301) as well as +17814441146 found on your website
but the fax failed in both cases.

Please, can you provide me with the correct fax number
or is this email (with the completed form attached) sufficient?

Thanks.
Ibrahim

NB: See original email and response from a member of
your support team below.

-----Original Message-----
From: no-reply@salesforce.com [mailto:no-reply@salesforce.com]
On Behalf Of TripAdvisor Support
Sent: 17 July 2009 16:34
To: IBRAHIM ADEYANJU (0612836)
Subject: RE: PERMISSION TO UTILIZE REVIEWS ON
YOUR WEBSITE FOR RESEARCH PURPOSE [ref:00D8LC7N.50086k9vf:ref]

Hi there,

Thanks for contacting TripAdvisor. Please fill out the

attached form and use the instructions to send it back to
the proper email address.
Someone from our legal team will then contact you directly.

Kind Regards and good luck with your research,

Jeanine
TripAdvisor Support Team

---------- Original Message ----------
From: [i.adeyanju@rgu.ac.uk]
Sent: 7/17/2009 8:00 AM
To: partners@tripadvisor.co.uk
Subject: PERMISSION TO UTILIZE REVIEWS ON
YOUR WEBSITE FOR RESEARCH PURPOSE

The Director, Partnerships and Business Development
TripAdvisor LLC
141 Needham Street
Newton, MA 02464, USA

Dear Sir/Ma,

PERMISSION TO UTILIZE HOTEL REVIEWS ON YOUR WEBSITE FOR RESEARCH PURPOSE

I write to request permission to download and utilise some hotel reviews
from your website for research purposes. I am a second year PhD student
at the Robert Gordon University, Aberdeen working on an area of research

called Textual Case Based Reasoning. The research area is an AI paradigm
that helps to solve new problems by reusing
past experiences documented as text.

The objective of my research in relation to hotel
reviews is to assist users when writing new reviews
by suggesting useful text based on previous similar
structured attributes chosen by the user . Such
structured attributes could include ratings for
value, rooms, location, cleanliness etc. The user can
then edit the suggested text rather than writing a
review from the scratch thereby reducing the time
spent on writing a new review.

I promise to protect any confidential/ private contents
of reviews downloaded in any material published as a result
of research experiments. I am also willing to adhere to any
other restrictions that might apply if the permission is granted.

Thanks and I look forward to your reply.

Regards,

Ibrahim Adeyanju
Research Student,
School of Computing,
The Robert Gordon University,
Aberdeen, UK
(http://www.comp.rgu.ac.uk/staff/iaa/)

Appendix C

User questionnaire for text evaluation in H&S incident reporting

As explained in Section 6.2.2, a user study was conducted to obtain evidence to support our observations that the automated evaluation metrics (accuracy, cosine coefficient, precision, recall) are inadequate for domains where there is phrase or sentence level variation across authors. This applies to our domain of medical health and safety incidents reporting. The complete questionnaire is shown below. Ten different queries were randomly selected from our casebase of 362 cases. Each query is shown along with its actual solution and the solution retrieved during a ten-fold cross validation used in our automated experiments. Users are then requested to grade how similar the actual and retrieved solutions are on a scale of 1 to 5. We labelled the possible values 1, 2, 3, 4, 5 as 'Very dissimilar', 'Dissimilar', 'Unsure', 'Similar' and 'Very similar' respectively. Comments in form of free text were also allowed for each query.

USER EVALUATION (PILOT) OF THE RETRIEVAL/ REUSE STAGES OF A TCBR SYSTEM ON HEALTH & SAFETY INCIDENT REPORTS

1. INTRODUCTION

Textual Case Based Reasoning (TCBR) solves new problems by reusing previous similar problem-solving experiences documented as text. The basic principle is that similar problems have similar solutions and it is therefore easier to modify a past solution in solving a new problem than solving the problem from scratch. We developed a TCBR system which aids the reuse of health and safety incident reports. Each report consists of a textual description of an incident (as problem) and the action taken (as solution) by the staff on duty to resolve this incident.

The purpose of this user experiment is to measure qualitatively the effectiveness of our system using sample retrievals from a ten-fold cross validation experiment.

2. ENTRY QUESTIONNAIRE

Please answer these initial questions as regards your **familiarity** with the following topics by writing the appropriate number using the scale 0-3 as defined below

3- Expert *2- Average* *1- Little* *0- None*

Case Based Reasoning/ Data mining	
Text processing/ analysis	
Similarity techniques	
Health & Safety incidents	

3. TASK QUESTIONNAIRE

With reference to resolving the incidents shown in the **query** column, how similar is the **retrieved solution** when compared to the **actual solution**? Please tick the appropriate box on the scale as defined below

1- Very Dissimiar *2- Dissimilar* *3- Unsure* *4- Similar* *5- Very similar*

	Query	Actual Soln	Retrieved Soln	1	2	3	4	5	Comments
1	hot cup of tea spilled over a patient.	first aid.	examined by nursing staff.						
2	patient fell off of the commode and was found on the floor.	examined by medical staff.	assisted back to bed						
3	patient sustained a skin flap whilst being hoisted.	wound cleaned and dressed.	first aid.						
4	open medicine glass vial and sustained a cut to left index finger.	washed cut and applied plaster	wound cleaned and dressed						
5	patient fell over a portable heater.	examined by nursing staff.	helped onto wheelchair by medical, nursing staff, put into couch routine obs taken						
6	while securing a suture the surgeon was splashed in the left eye with blood.	eye wiped with with wet cloth only, at surgeons' request.	first aid, a and e contacted for advice.						
7	needlestick injury sustained.	occupational health contacted.	first aid, blood sample taken, visited occupational health.						
8	patient came out of toilet,and lost his balance and fell on floor	helped him up and wheeled him back to bedside	helped back up and sat on the toilet						
9	patient stood up unaided and fell to the floor.	examined by medical staff.	examined by nursing staff.						
10	patient fell to floor when getting out of bed.	patient checked by nursing staff.	patient checked for injury.						

4. DEBRIEFING

Thanks for taking time to fill this questionnaire.
Please, feel free to write any other comments below.

Appears in H. Coelho, R. Studer and M. Wooldridge (eds), *Proceedings of ECAI 2010*. IOS Press, Amsterdam, pp. 777-782.

Learning to Author Text with textual CBR

Ibrahim Adeyanju[1] and **Nirmalie Wiratunga** and **Juan A. Recio-Garcıa**[2] and **Robert Lothian**

Abstract. Textual reuse is an integral part of textual case-based reasoning (TCBR) which deals with solving new problems by reusing previous similar problem-solving experiences documented as text. We investigate the role of text reuse for text authoring applications that involve feedback or review generation. Generally providing feedback in the form of assigning a rating from a likert scale is far easier compared to articulating explanatory feedback as text. When previous feedback generated about the same or similar objects are maintained as cases, there is opportunity for knowledge reuse. In this paper, we show how compositional and transformational adaptation techniques can be applied once sentences in a given case are aligned to relevant structured attribute values. Three text reuse algorithms are introduced and evaluated on a dataset gathered from online Hotel reviews from TripAdvisor. Here cases consists of both structured sub-rating attributes together with textual feedback. Generally, aligned sentences linked to similar sub-rating values are clustered together and prototypical sentences are then extracted to enable reuse across similar authors. Experiments show a close similarity between our proposed texts and actual human edited review text. We also found that problems with variability in vocabulary are best addressed when prototypes are formulated from larger sets of similar sentences in contrast to smaller sets from local neighbourhoods.

1 Introduction

The task of authoring documents that include pre-defined attributes along with some textual content is common to several domains. Such documents include reviews, student feedback, medical notes and incident reports. Review of products and services is one of such web applications where authoring is increasingly being encouraged by e-commerce websites. This is very useful for both the manufacturers/service providers to improve their products/services and the customer to make informed choices. Review typically consist of predefined attributes which authors can rate on a likert scale. For example, a customer reviewing a hotel visited recently might be asked to rate the cleanliness and service enjoyed on a scale of 1 to 5 where 1 is *terrible* and 5 is *excellent*. Another component of such reviews is a free text section where authors can explain their ratings and elaborate further. However, they are sometimes reluctant to write free text especially a comprehensive one since it takes more time to put their thoughts into writing.

Textual case base reasoning (TCBR) [15] is a research area that deals with solving new problems by reusing previous similar experiences documented as text. Text reuse is an integral part of TCBR and is not only helpful in solving a new similar problem but can assist in authoring new experiences. TCBR is particularly suited to support authoring of textual contents because it can propose useful initial text from previous reviews that are similarly rated. Reusing previous textual contents is challenging as it is difficult to know sections in the text that are associated with structured attributes corresponding to the set of ratings. It is also important to avoid irrelevant verbose details that are not easily reusable across several authors.

Our focus in this paper is to assist authors to write better and more comprehensive reviews by proposing useful text which they can easily edit to taste. We propose two novel mechanisms to align rated attributes to review sentences and abstract a group of similar sentences into a prototypical sentence. These mechanisms led to the development of three text reuse techniques which differ mainly in terms of how similar case(s) are retrieved, what sentence(s) are used from these cases and whether such sentences are global or local prototypes. Our hypothesis is that each of these techniques will generate useful initial text but one of them might significantly outperform the others. We evaluate these algorithms on hotel reviews dataset and our results show a close similarity between the proposed and actual review texts. Algorithms presented in this paper have the advantage of being domain-independent and so are applicable in domains containing cases with both pre-defined structured attributes and complementary textual content.

An overview of our domain of application is given in Section 2 followed by details of our alignment approach and text reuse techniques in Sections 3 and 4. Experimental setup and discussion of results appear in Section 5 with related work in text reuse in Section 6. We conclude with the contributions of this work in Section 7.

2 Hotel Reviews Domain

User generated experiential content is readily available on the world wide web in the form of blogs, forum posts, reviews and other social applications. This provides an opportunity to reuse these experiences [12] for similar web related tasks such as search and browse, review generation and other forms of problem solving. However, reuse will only make sense if there are several experiences authored about similar/same objects (or problems). Hotel reviews are particularly useful in this regard because several reviews are available for the same or indeed similar hotels. Each review typically has some attributes rated on a likert scale and a complementary text. Hotel reviews are generally suitable for text reuse as the assumption is that authors with similar ratings will use similar explanatory feedback text. However, such review texts are prone to grammatical errors since authors rarely use spell checkers. They also contain a lot of verbose details that might not be related to hotels since unedited reviews are uploaded.

We downloaded several reviews from a hotel recommender website [3] where each review is written by an author who visited a hotel

[1] School of Computing,Robert Gordon University, Aberdeen, Scotland, UK, email: [iaa|nw|rml]@comp.rgu.ac.uk
[2] Department of Software Engineering and Artificial Intelligence, Universidad Complutense de Madrid, Madrid, Spain, email: jareciog@fdi.ucm.es

[3] www.tripadvisor.co.uk

1. Hotel name	2. Hotel town	3. Hotel country (or US state)
4. Overall rating	5. Review Title	6. Author **rating**(up to 5 stars)
7. Author ID	8. Author location	9. Trip type (solo, couple etc)
10. **Review text**	11. Date of stay	12. Recommend to friend(y/n)
13. **Sub-ratings** for value, room, location, cleanliness & service		

Table 1. Complete list of possible attributes extracted for each hotel review

and presents her opinion of the place. The 13 attributes shown in Table 1 were extracted for each review; however, some of these attributes were absent in some reviews. 39,870 reviews were extracted from our downloads cutting across 6,564 hotels in 104 different countries (or states in USA). Based on an analysis of the corpus, we discovered that the downloaded corpus contained a small number of reviews (< 50) per hotel or author. The overall rating of a hotel is also an average of authors' ratings and not those given by regulators such as ISO (International Organization for Standardization). It is therefore more intuitive to reuse similar reviews across all hotels.

```
<Review>
 <RSN>10</RSN>
 <HotelName>Sunroute Plaza Shinjuku Hotel</HotelName>
 <ReviewTitle>Perfect for the first timer to tokyo</ReviewTitle>
 <HotelTownLocation>Shibuya</HotelTownLocation>
 <HotelStateLocation>Japan</HotelStateLocation>
 <OverallRating>4.5</OverallRating>
 <Rating>5</Rating>
 <ReviewersName> REVIEWER-ID </ReviewersName>
 <ReviewersLocation>singapore</ReviewersLocation>
 <TripType>Couples</TripType>
 <ReviewText>
Location of hotel is perfect, within walking distant to the main JR station,
subway metro, there is a station just next to the hotel. For shoppers
Takashimaya is just across the bridge! Airport transfer right to
doorsteps.Food, shoppings and train/ subway stations are within 5 to 10
mins walk. 5 mins walk to this electric street that not only sell all
electrical appliance but with resturants that the locals frequent, that
serve very nice and reasonable cheap Japanes dishes.Hotel staff are
efficient and helpful and especially the front desk staff speaks very good
english.Intenet access in the room is superb, shampoo , conditioner and
body wash come in family size bottles, fantastic! The only minus point is
the standard queen bed room has got no cupboard, its better of
choosing the standard double bed room. But on the whole its a very
clean, comfortable and safe hotel; we would rate it 9 out of 10. From
REVIEWER-ID, Singapore
 </ReviewText>
 <RatingList>
  <ValueRating>5</ValueRating>
  <RoomsRating>4</RoomsRating>
  <LocationRating>5</LocationRating>
  <CleanlinessRating>5</CleanlinessRating>
  <ServiceRating>5</ServiceRating>
 </RatingList>
 <DateOfStay>July 2009</DateOfStay>
 <RecommendToFriend>Yes</RecommendToFriend>
</Review>
```

Figure 1. Example of a hotel review from tripadvisor.co.uk

Another finding from the corpus analysis is that the rating and sub-rating attributes have the greatest effect on the contents of a review text because most authors enter values for these attributes. We therefore limit our structured attributes to rating for the hotel and sub-ratings for *cleanliness*, *location*, *rooms*, *service* and *value* so that review texts can be reused across a wider range of authors. These attributes are completed on a likert scale of terrible (1), poor (2), average (3), very good (4) and excellent (5). An example review is shown in Figure 1. The author's ID is anonymised due to privacy issues and highlighted portions relate to attributes used in our experiments. The review text shown is typical where authors write not just content in relation to their ratings but also elaborate on associated concepts that contributed towards the overall experience such as local restaurants.

3 Text Alignment

A CBR case typically consists of two components: a problem and its solution. When both have multiple attributes, each solution attribute might depend on a specific combination of problem attributes rather than all. Knowledge of such problem-solution attribute alignment allows for better retrieval accuracy. This is because the best values for solution attributes can be retrieved from different cases with aligned problem attributes most similar to the query. However, learning such relationships or alignment between problem and solution attributes remains a challenge when they are not explicitly expressed in the domain. This applies to TCBR where it is difficult to predict which section of a text (e.g. sentence) aligns to specific problem attributes.

We propose a method that aligns sub-rating attributes to specific sentence(s) in the text of a review. This enables the reuse of sentences from different authors with similar sub-ratings to a query. The basic idea in our text alignment method is to bridge the vocabulary in the problem and solution spaces. This is done by compiling a list of seedwords related to each sub-rating; these seedwords were obtained from WordNet [8] by checking for synonyms of sub-rating descriptors and manually refining the list. A sample list of example seedwords extracted for the five sub-ratings is given in Table 2. Although our list of seedwords is non-exhaustive, it is a good foundation to test our text alignment hypothesis.

Sub-rating name	**Seedwords (sample)**
cleanliness	clean, neat, kempt, tidy, cleanse
location	location, position, place
room	room, bedroom
service	help, serve, service, reception, star
value	esteem, rate, valuate, value, worth

Table 2. Seedwords used for text alignment between sub-ratings and review sentences

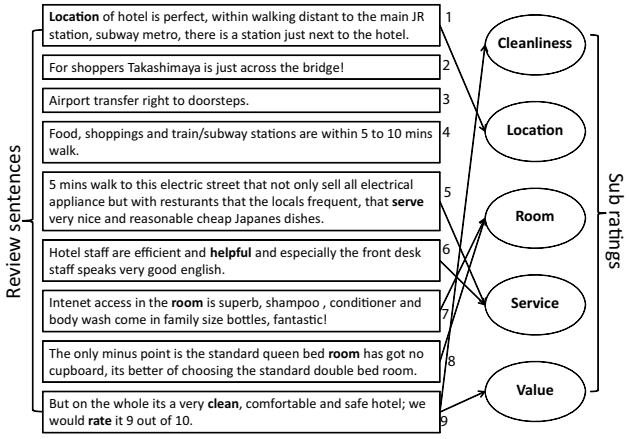


Figure 2. Example of review sentences' alignment to sub-ratings

Each review text is parsed into sentences using the GATE [6] libraries available as part of the jColibri [7] framework. Every sentence in the text is then categorised as belonging to a sub-rating if it contains any of its seedwords. Figure 2 illustrates alignment between review sentences and sub-ratings using the review text in Figure 1; here, seedwords are in bold. The text has 9 sentences of which only 6

are aligned to sub-ratings. It can be observed that most of the aligned sentences are intuitively reasonable; for example, sentence 1 is about the proximity of the hotel to rail station and is correctly aligned to *location* sub-rating. However, sentence 5 is better aligned to *location* than *service* sub-rating since it highlights the hotel's proximity to restaurants and local shops. The unaligned sentences (i.e. 2, 3 & 4) are related to *location* but were not linked because they contain none of the seedwords. This highlights the need for a representative set of relevant seedwords.

Overall, the text alignment process approximates the relationship between sub-ratings and review sentences. The alignment link is a many-to-many relationship as a sentence can belong to more than one sub-rating and vice versa. This is illustrated in Figure 2 where sentences 7 & 8 are linked to *room* sub-rating while sentence 9 is linked to *cleanliness* and *value* sub-ratings. Unaligned review sentences are regarded as verbose details (sometimes unrelated to the hotel) that cannot be easily reused across authors. For example, not every hotel will be in a town with an airport; therefore in Figure 2, sentence 3 cannot be reused by such authors. In our experiments, only aligned sentences are used since this allows the system to propose generic texts while insertion of contextual details is left for authors.

4 Text Reuse Algorithms

We propose three different techniques to assist with reusing textual contents. These techniques make use of text alignment between sub-ratings and review sentences as explained in Section 3. The differences between them are in terms of what neighbourhood of a query is used in generating a proposed solution and how sentences are combined from similar cases. Here, techniques similar to CBR substitutional, transformational and compositional adaptation are applied to textual cases in relation to sentence aggregation from different neighbours of a query.

4.1 Baseline retrieval

Given a query, Q, consisting of a set of rating and sub-ratings, baseline (BASE) retrieves the nearest neighbour and reuses its review text. In Figure 3, *Retrieve* returns the most similar case (C_{best}). Here, sub-ratings are termed ratings since they are graded on the same likert scale. Sentences in the review text aligned to the five sub-ratings are then identified and concatenated to form the proposed solution, $SOLN$. Identification of aligned sentences is achieved with the *selectAlignedSentences* method for each rating in C_{best}. The use of $SOLN$ as a set ensures that duplicate sentences in the proposed solution are removed because each sentence can be aligned to more than one rating. Our baseline technique is essentially a retrieve-only system except for the removal of unaligned sentences. BASE generates five or more sentences in a proposed solution text since there can be multiple sentences aligned to each sub-rating.

4.2 Transformational approach to text reuse

This approach denoted as XFRM uses multiple nearest neighbours of a query to propose a review solution text rather than the nearest neighbour used in baseline retrieval discussed in Section 4.1. Given a query of rating and sub-ratings attributes, specified $k-$nearest neighbours are retrieved. To reuse review texts from these neighbours, we propose and progressively transform aligned sentences from the best match solution. This takes place only if there are mismatches between the query and best match's sub-ratings. Sentences aligned to

CB= $\{C_1, \ldots, C_n\}$, set of cases in the case base
$R = \{r_1, \ldots, r_p\}$, set of rating types i.e. pre-defined attributes
$V = \{v_1, \ldots, v_q\}$, set of possible values for each rating type
IE= information entity consisting of a rating with distinct value,
where $(ratingType(IE) \in R) \wedge (ratingValue(IE) \in V)$
$C_i = \{\text{IE}_{i1}, \ldots, \text{IE}_{ip}, \text{ReviewText}_i\}$, where $(i \in \{1 \ldots n\})$
i.e. a case consists of p rating values and a review text
$Q = \{\text{IE}_1, \ldots, \text{IE}_p\}$, a query with p rating values

$C_{best} = Retrieve(\text{CB}, Q, 1)$, retrieve most similar case
Initialise $SOLN$= {},
to contain sentences in the proposed solution text
ReviewText= $getReviewText(C_{best})$
for each $\text{IE}_j \in C_{best}$
$r_j = ratingType(\text{IE}_j)$
$S_j = selectAlignedSentences(r_j, \text{ReviewText})$
$addSentences(S_j, SOLN)$
Concatenate sentences in $SOLN$ for reuse

Figure 3. Baseline text reuse algorithm (BASE)

mismatched sub-ratings are removed if they are not aligned to any other sub-ratings and replaced with aligned sentences from nearest neighbours matching the query's sub-rating. This approach is similar to CBR transformational adaptation [5] where solution elements are re-organised through add and delete operations. However it is also similar to substitutional adaptation [16, 9, 1] if seen as successive replacement of aligned sentences in baseline text (see Section 4.1).

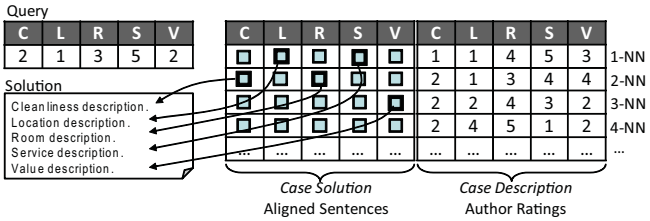


Figure 4. Transformational approach to text reuse

Figure 4 illustrates this approach with a query and four nearest neighbours with their sub-rating values and aligned sentences. Here, sentences for *location* and *service* are chosen from the first neighbour. Any mis-matched values are resolved by extracting aligned sentences from the neighbourood (2NN & 3NN). Note that if a query sub-rating value is not matched in any of the nearest neigbours, no sentence is generated for such sub-rating and number of sentences can be less than 5. However in reality, there are multiple sentences per rating resulting in a reuse solution with five or more sentences.

The transformational approach is also formalised as an algorithm (see Figure 5). Here, we compare each sub-rating (IE_j) in the query with similar sub-ratings of neighbouring cases (CB_{local}). Functions $ratingType$ and $ratingValue$ returns the sub-rating type (e.g. location) and values (e.g. 4) respectively. The conditional statement SOL_j=$null$ ensures that aligned sentences are chosen from the first similar case whose sub-rating values matches the query.

4.3 Text generation with sentence clustering

Here, a proposed text is generated in response to a query by combining sentences from several similar cases. Hence it is called compo-

Initialise $SOLN = \{SOL_1, \ldots, SOL_p\}$,
set of proposed sentences for each rating
$\text{CB}_{local} = Retrieve(\text{CB}, Q, k)$, retrieve k similar cases
for each $\text{IE}_j \in Q$
$qr = ratingType(\text{IE}_j)$; $qv = ratingValue(\text{IE}_j)$
for each $\text{C}_i \in \text{CB}_{local}$, in order of decreasing similarity
ReviewText= $getReviewText(C_i)$
$r_j = ratingType(\text{IE}_j, C_i)$; $v_j = ratingValue(\text{IE}_j, C_i)$
if ($qr = r_j$ AND $qv = v_j$ AND $SOL_j = null$)
$S_j = selectAlignedSentences(r_j$, ReviewText)
$addSentences(S_j, SOL_j)$
Concatenate all sentences in $SOLN$ for reuse

Figure 5. Transformational text reuse algorithm (XFRM)

sitional (COMP) text reuse because of its similarity to CBR's compositional [5, 3] or constructive [13] adaptation where a solution is obtained by combining solution elements of several partially similar cases. Sentences are considered to be contextually similar when they are aligned to an identical sub-rating value. For example, all sentences aligned to a cleanliness sub-rating of 3 can be regarded as similar. Aggregating several pieces of similar sentences into a single meaningful prototype is not trivial. Concatenation is inappropriate since it leads to tautology and summarisation methods might not work because sentences are semantically similar yet lexically different. Thus, we introduce a mechanism that combines several similar sentences into a single meaningful text called the *prototypical* sentence. For prototypes, a term frequency vector is first created for each sentence. Each vector length is the size of unique keywords in all similar sentences for which a prototype is being determined. A centroid is calculated for these vectors as the average term frequency across each unique keyword. Accordingly, a prototypical sentence is a sentence whose vector is most similar to the centroid vector. Intuitively, our prototype will contain common keywords used across sentences. This is because values of such keywords in the prototype vector will be closer to the average.

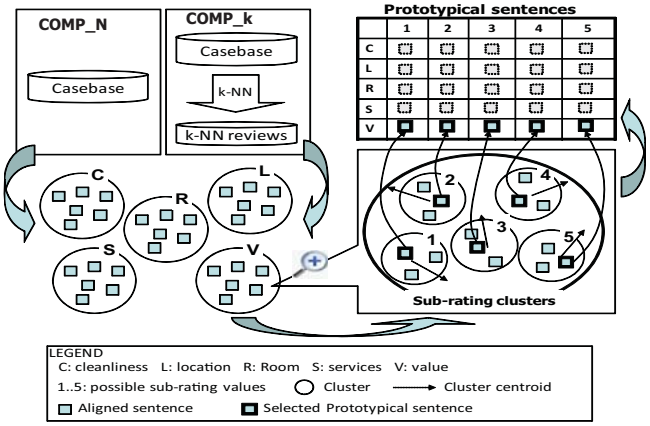


Figure 6. Clustering similar aligned sentences in review texts

The generation of prototypical sentences is illustrated in Figure 6. These prototypes can be generated from either k nearest neighbours to the query (COMP_k) or all reviews in the casebase (COMP_N). Aligned sentences across the specified reviews (local or global) are clustered according to the class they belong to given the five sub-ratings. Each cluster is then further re-clustered into five groups using their rating value (i.e. 1 to 5). The smaller group of clusters shown for the *value* sub-rating also applies to the other four sub-ratings. The outcome of this clustering process is 25 small clusters and a prototypical sentence per cluster.

Initialise

$$G = \left\{ \begin{array}{l} g_{11}, \ldots, g_{1q} \\ g_{21}, \ldots, g_{2q} \\ \ldots \\ g_{p1}, \ldots, g_{pq} \end{array} \right\},$$ set of clustered similar sentences; each cluster belongs to a pair from p ratings and q values

$\text{CB}_{local} = Retrieve(\text{CB}, Q, k)$; retrieve k similar cases
Initialise $SOLN$= {},
to contain sentences in the proposed solution text
for each $\text{C}_i \in \text{CB}_{local}$, in order of decreasing similarity
ReviewText= $getReviewText(C_i)$
for each $\text{IE}_j \in C_i$
$r_j = ratingType(\text{IE}_j)$; $v_j = ratingValue(\text{IE}_j)$
$g_j = getClusteredSimilarSentences(G, r_j, v_j)$
$S_j = selectAlignedSentences(r_j$, ReviewText)
$addSentences(S_j, g_j)$
for each $\text{IE}_k \in Q$
$r_k = ratingType(\text{IE}_k)$; $v_k = ratingValue(\text{IE}_k)$
$g_k = getClusteredSimilarSentences(G, r_k, v_k)$
$ps_k = getPrototypicalSentence(g_k)$
$addSentences(ps_k, SOLN)$
Concatenate sentences in $SOLN$ for reuse

Figure 7. Compostional text reuse algorithm (COMP_k)

The COMP_k algorithm shown in Figure 7 is illustrated in Figure 8 where the query consists of 5 (p in algorithm) ratings of $2, 1, 3, 5, 2$ for *cleanliness*, *location*, *room*, *service* and *value* respectively. Five sentences are then obtained from the prototypical sentences with identical sub-rating values to the query and concatenated as proposed text ($SOLN$). In this algorithm, each prototypical sentence is generated from an element in the matrix of sentence clusters (G) having $p \times q$ elements. A major difference between COMP_k that use reviews from neighbours and COMP_N that uses all reviews is that it might generate less than five sentences since a small neighbourhood may not contain all sub-rating values required by a query.

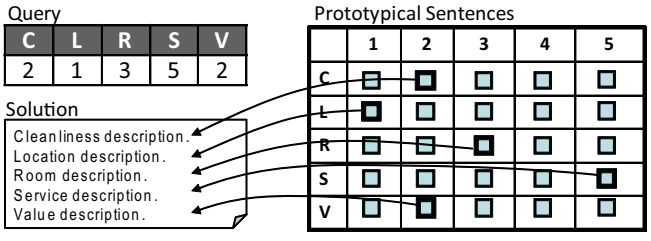


Figure 8. Compositional approach to text reuse

5 Experimental Setup

We compare three text reuse techniques.

1. Baseline retrieval (BASE) in Section 4.1
2. Transformational approach to text reuse (XFRM) in Section 4.2
3. Text generation with sentence clustering (COMP_k & COMP_N) in Section 4.3

A ten-fold cross validation is employed in our experiments. We want to ensure that retrieved reviews have very similar ratings to the query. For example, it will be very difficult to reuse text from a review with rating 4 (very good) for a query with rating 2 (poor). Therefore, similar cases are retrieved with an interval of 2 between rating (or sub-rating) attributes. This means that a difference of 1 between two ratings gives a 0.5 similarity while a difference greater than 1 gives zero similarity. Similarities across attributes are aggregated using a weighted average; 0.25 for rating and 0.15 for each sub-rating.

The effectiveness of the text reuse techniques is measured using cosine coefficient similarity between aligned sentences in our actual solutions and the proposed text. Cosine similarity is employed as opposed to precision/recall because it allows us to compare the performance of the reuse techniques with a single metric which also takes the texts' length into account. We are also interested in the effect of different neighbourhood sizes (k) on reuse performance for COMP_k and XFRM. Experiments for the two techniques were therefore repeated using increasing values of k ($k = 3, 5, 10$ & 25).

5.1 Dataset

A sample dataset from the hotel reviews (Section 2) was created by selecting reviews with sentences aligned to each sub-rating; 641 of such reviews were found. Review texts were normalised by substituting named entities such as person names, currencies, locations and dates with generic labels. Table 3 lists some of these entities extracted with GATE [6] together with the general category label.

Category	Named entity examples
person name	yang, vincent, susanne, patrick, katherine
currency	yen, pounds, francs, euros, dollars, cents
date	september 2009, mid august 08, last year,april 26th
time	9.30pm, 8:00 a.m., 5pm, 3:45pm, 17:45

Table 3. Examples of named entities found in Hotel Reviews

5.2 Discussion of results

Figure 9 shows the average cosine similarities between proposed text and actual solution across the three text reuse techniques. The different k-neighbourhoods are shown in brackets for compositional (COMP_k) and transformational (XFRM) approaches. The baseline (BASE) which recommends a subset of sentences from the best match case by ignoring sentences unaligned to any sub-rating does well as compared to COMP_k and XFRM. This is because they use similar neighbouring case(s) unlike COMP_N which uses all cases.

COMP_k where local prototypical sentences are proposed improves with increasing neighbourhood size. This trend suggests that the performance will match up with COMP_N as the neighbourhood size tends toward the entire casebase. This shows that local prototypical sentences tend to capture less keywords that are reusable across authors as compared to the global prototypes. COMP_N which uses all cases to generate prototypical sentences for each sub-rating clearly outperforms the rest. An advantage of this approach is that these generic sentences are likely to be more similar to the actual solution compared to a local sentence which might express the same

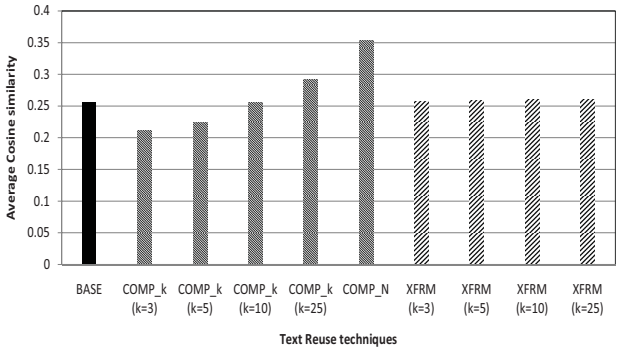


Figure 9. Graph of cosine similarity across text reuse techniques

opinion using different terms. Although at first surprising, this result compliments findings in other related studies [10, 11] in text reuse. On the other hand, there is very little improvement in performance as we increase the neighbourhood size for XFRM. This means that most of the query sub-ratings are easily matched in the smaller neighbourhoods (i.e $k = 3, 5$). However, aligned sentences generated from such neighbourhoods are not as good as prototypical sentences from larger neighbourhoods.

Query

C	L	R	S	V
4	3	2	2	5

Proposed Solution (COMP_N)

C	it was very clean .
L	the hotel was in a great location .
R	unfortunately we were very disappointed upon seeing the room .
S	wer i guess that the best way to do this is to list the good and bad about this place so here goes : the good it is very nice and sunny and always hot - lovely the pool is clean and warm the air conditioning in the reception area is refreshing the beer was ok good shuttle bus service to the (horrible) beach the bad where are the toilets , as far as i could see there was only one (apart from going back to the room) and that was in the reception
V	it was so worth it .

Figure 10. An example of the proposed text from COMP_N

Generally, a low cosine similarity (less than 0.5) is seen between the proposed texts and their actual solution. Closer examination of proposed text suggests that low similarity values does not necessarily mean poor solution quality. Figure 10 shows a sample of the proposed text generated by COMP_N technique. Most of the sentences seem reasonable to the given query ratings except for *service* which is verbose. Such long sentences contain specific details that will adversely affect the cosine similarity. Nevertheless, the results indicate that proposed text were similar to the actual and it might be easier to edit them than writing from scratch. Also, our proposed texts will encourage new authors to write reasons for each sub-rating value rather than a lot of verbose but unnecessary details thereby making future reviews more useful to others.

6 Related Work

Automated reuse of text remains a challenge especially when they are available in the unstructured form. There are few studies [14, 11, 2, 4]

available in this area due to difficulties with mapping such text to a structured representation, measuring semantic similarity and automated evaluation . A restricted form of textual reuse is presented for report writing applied to the air travel incident domain [14]. Here, textual cases consist of incident reports with one or more paragraphs grouped under a specific heading as a section. The most similar document to a query is retrieved and textual reuse is facilitated for each section of the retrieved report. This is done by presenting a cluster of other documents containing similar text under the same heading. This technique ignores the context of each section within the entire report which might lead to unuseful clusters and is restrictive as it cannot be used where common section headings are absent. Therefore, this approach is not applicable directly to domains such as hotel reviews authoring with no sectional headings. The approach is similar to one of our reuse techniques because similar sections (or sentences in our work) are grouped together. However ours differ in that we propose prototypical sentences generated from sentence clusters.

The drawbacks observed in the work reviewed above are addressed by a text reuse technique called *Case Grouping* (*CG*) [11]. The technique demonstrated on a semi-automated email response application involves reuse of previous email messages to synthesize new responses to incoming requests. A response is a sequence of statements satisfying the content of a given request and requires some personalization and adjustment of specific information to be reused in a new context. The reuse technique annotates sentences as reuse if there is sufficient evidence that similar past problems contain this sentence. The case base is divided into two clusters that contain similar sentence and those that don't to quantify this evidence. Query similarity to a centroid case formed for each cluster determines whether or not to reuse. The centroid case has the average value for each feature across all cases in a cluster. Our mechanism of prototypical sentence (see Section 4.3) is also based on a centroid vector. However, we form a single feature vector for each similar sentence rather than entire text (usually several sentences) in CG. This reduces the effect of aggregating the same features across unrelated sentences.

An approach to text reuse is proposed in [4] where users are given suggestions to support the authoring process applied to a waste exchange service that links people over the web to enable transfer of unwanted items to those who can use such items. Suggestions are generated from previous successful item descriptions; these are descriptions where users have been able to complete transfer of items to others using the service. The approach extracts feature-value pairs from all previous successful descriptions using regular expressions that are manually defined. The most similar successful description is retrieved during authoring of a new item description. This is done iteratively as the author adds a specified amount of text (e.g. a sentence). Features from the similar case are then compared to those extracted from the new partial description. Top k common values of features from the retrieved case whose features are absent in the new description are ranked from top similar cases and shown to the user as suggestions. Such suggestions support the authoring process by assisting a user to write an item description that can lead to the item being transfered successfully. A major drawback is that repeated suggestions are distractive to users and can lead to more time being spent on authoring. The aim in this work is similar to ours and their use of extracted features is similar to our structured attributes. However, we suggest whole texts rather than in bits which removes unnecessary distraction to the author. Also, their technique cannot be integrated into an existing authoring system without modification to the user interface but our techniques can be integrated directly.

7 Conclusion

This work introduced two novel concepts in relation to text reuse: text alignment and sentence aggregation. Text alignment links rated attributes to specific sentences in a review text while sentence aggregation abstracts similar sentences into a single meaningful prototype. These concepts are generally applicable in domains where cases consists of pre-defined attributes along with written text. These mechanisms led to the development of three text reuse techniques that generate proposed texts related to the pre-defined attributes' ratings. Our results show that proposed texts were similar to the actual and will assist authors to write better and more useful reviews. We also obtained better results with global than local prototypical sentences meaning that higher level abstractions are more reusable across authors.

We intend to improve the choice of seedwords by learning introspectively from our corpus as opposed to using a external ontology like WordNet. This might be done by searching for sentences containing defined patterns and limiting our seedwords to specific parts of speech. We plan to introduce alternative evaluation measures such as edit distance and experiment with other related domains.

Acknowledgements This work is funded by the Northern Research Partnership and UK-India Education & Research Initiative.

REFERENCES

[1] Ibrahim Adeyanju, Susan Craw, Abhishek Ghose, Allyson Gray, and Nirmalie Wiratunga, 'RaGoÛt: An arpeggio of tastes', in *ECCBR 2008 workshop Proceedings (Cooking contest)*, pp. 229–238. Tharax, (2008).

[2] Ibrahim Adeyanju, Nirmalie Wiratunga, Rob Lothian, Somayajulu Sripada, and Luc Lamontagne, 'Case Retrieval Reuse Net: Architecture for reuse of textual solutions', in *Proc. of ICCBR'09*, pp. 14–28, (2009).

[3] Rim Bentebibel and Sylvie Despres, 'Using compositional and hierarchical adaptation in the SAARA system', in *ECCBR'06 workshop proceedings (Reasoning with Text)*, pp. 98–108, (2006).

[4] Derek Bridge and Aidan Waugh, 'Using experience on the read/write web: The GhostWriter system', in *ICCBR09 workshop proceedings (WebCBR)*, pp. 15–24, (2009).

[5] Chun-Guang Chang, Jian-Jiang Cui, Ding-Wei Wang, and Kun-Yuan Hu, 'Research on case adaptation techniques in case-based reasoning', in *Proc. of ICMLC'04*, pp. 2128–2133, (2004).

[6] Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan, 'Gate: A framework and graphical development environment for robust NLP tools and applications', in *Proc. of ACL '02*, (2002).

[7] Belén Díaz-Agudo, Pedro A. González-Calero, Juan A. Recio-García, and Antonio Sánchez, 'Building CBR systems with jCOLIBRI', *Special Issue on Experimental Software and Toolkits of the Journal Science of Computer Programming*, **69**, 68–75, (2007).

[8] *WordNet: An Electronic Lexical Database*, ed., Christiane Fellbaum, MIT press, 1998.

[9] Pedro A. González-Calero, Mercedes Gómez-Albarran, and Belén Díaz-Agudo, 'A substitution-based adaptation model', in *ICCBR'99 workshop proceedings (Challenges for CBR)*, (1999).

[10] Luc Lamontagne and Guy Lapalme, 'Applying case-based reasoning to email response', in *Proc. of ICEIS-03*, pp. 115–123, (2003).

[11] Luc Lamontagne and Guy Lapalme, 'Textual reuse for email response', in *Proc. of ECCBR'04*, pp. 234–246, (2004).

[12] Enric Plaza, 'Semantics and experience in the future web', in *Proc. of ECCBR 08*, pp. 44–58, (2008).

[13] Enric Plaza and Josep-Lluís Arcos, 'Constructive adaptation', in *Proc. of ECCBR'02*, pp. 306–320, (2002).

[14] Juan A. Recio-García, Belén Díaz-Agudo, and Pedro A. González-Calero, 'Textual CBR in jCOLIBRI: From retrieval to reuse', in *ICCBR'07 workshop proceedings (Textual CBR)*, pp. 217–226, (2007).

[15] Rosina Weber, Kevin Ashley, and Stefanie Bruninghaus, 'Textual case-based reasoning', *Knowledge Engineering Review*, **20**(3), 255–260, (2006).

[16] Wolfgang Wilke and Ralph Bergmann, 'Techniques and knowledge used for adaptation during case-based problem solving', in *Proc. of IEA-AIE'98*, (1998).

Applying Machine Translation Evaluation Techniques to Textual CBR

Ibrahim Adeyanju, Nirmalie Wiratunga, Robert Lothian, and Susan Craw

School of Computing
Robert Gordon University, Aberdeen, Scotland, UK
{iaa,nw,rml,smc}@comp.rgu.ac.uk

Abstract. The need for automated text evaluation is common to several AI disciplines. In this work, we explore the use of Machine Translation (MT) evaluation metrics for Textual Case Based Reasoning (TCBR). MT and TCBR typically propose textual solutions and both rely on human reference texts for evaluation purposes. Current TCBR evaluation metrics such as precision and recall employ a single human reference but these metrics are misleading when semantically similar texts are expressed with different sets of keywords. MT metrics overcome this challenge with the use of multiple human references. Here, we explore the use of multiple references as opposed to a single reference applied to incident reports from the medical domain. These references are created introspectively from the original dataset using the CBR similarity assumption. Results indicate that TCBR systems evaluated with these new metrics are closer to human judgements. The generated text in TCBR is typically similar in length to the reference since it is a revised form of an actual solution to a similar problem, unlike MT where generated texts can sometimes be significantly shorter. We therefore discovered that some parameters in the MT evaluation measures are not useful for TCBR due to the intrinsic difference in the text generation process.

1 Introduction

Textual Case Based Reasoning (TCBR) deals with reusing past experience stored in the form of text such as reports, frequently asked question (faqs) and emails. However, there is the need to evaluate textual solutions proposed by a TCBR system. User evaluation is generally accepted as the best form of text evaluation but it is expensive and the aggregation of results from repeated experiments is likely to be difficult due to subjective user judgements. This is different and far more demanding than automated evaluation where experts provide reference texts only once. Therefore automated evaluation techniques that lead to metrics such as precision and recall (also known as accuracy and coverage) obtained by comparing proposed texts with reference solutions are preferred [6,13,1]. Although there have been reports to show good and reliable results in some domains [13,1], these simple metrics have also been reported to be insufficient to capture grammatical and semantic variations in texts that occur in other domains [2].

I. Bichindaritz and S. Montani (Eds.): ICCBR 2010, LNAI 6176, pp. 21–35, 2010.

Machine Translation (MT) on the other hand deals with producing an equivalent text from one language to another. Evaluation of machine translated text must therefore attempt to capture semantic meaning as well as differences in word choice and order (grammatical/ semantic variations). More sophisticated metrics than precision and recall have therefore been developed and used for text evaluation in MT research, since semantic meaning is crucial for successful translations. These metrics such as BLEU [17] and NIST [9] have also been reported to correlate highly with human judgements.

This paper presents the evaluation challenges for TCBR and how MT metrics can be employed to address them. We present the similarities and differences in MT and TCBR evaluation requirements and accordingly propose strategies to adapt MT metrics for TCBR. MT evaluation techniques are adaptable for use in TCBR because the common goal is to quantify the goodness of a piece of text suggested by text generation systems. We experiment with datasets from a health and safety incident reporting domain and compare results from applying MT evaluation with using the simple metrics of precision and recall. Analysis of our results show that MT metrics are generally better in capturing grammatical and semantic variations due to their use of multiple human references.

Other sections in this paper are as follows. Related works are reviewed in Section 2, while the text evaluation challenge and MT evaluation metrics are discussed in Sections 3 and 4 respectively. Experimental setup, evaluation and discussion of our results appear in Section 5, before conclusion in Section 6.

2 Related Work

The need to evaluate natural language texts is common to several research areas in computer science. These areas include (but not limited to) Information Retrieval (IR) [14,3], TCBR [7,19], Natural Language Generation [18,4] and MT [20,11]. Generally, we can group text evaluation techniques into two broad categories: qualitative and quantitative.

Qualitative techniques involve the use of humans (experts and non-experts) to determine the quality of some text produced by a machine. The results from several humans are then aggregated using statistical methods to judge the average quality of such texts. The major disadvantages are that these techniques are very expensive especially when expert knowledge is required and results are not easily reproducible as human judgement is subjective. Nevertheless, qualitative techniques have been used for evaluation across many application domains involving natural language processing and generation (e.g. [18,5]).

On the other hand, quantitative techniques involve the comparison of machine texts to one or more gold standards written by humans (usually experts). Here quality of the method is gauged according to similarity at the syntactic or semantic level. Quantitative techniques are typically less reliable as most of them depend on finding matching string patterns between the machine-produced texts and human gold standard(s). However, such techniques can be automated, are less expensive and are easily reproducible. This also allows for easy comparison across several algorithms that are designed for the same purpose.

Precision and Recall are two basic quantitative metrics [6,13] widely used for text evaluation across several disciplines especially IR and TCBR. The basic idea is to regard a piece of text as a bag of (key)words and to count common words between the machine and human texts. Proportions of these common words to the machine and human texts give a metric of precision and recall respectively. A major drawback is that the sequence of words in a piece of text is ignored and this can adversely affect the grammatical and semantic meaning. In other words, a machine text with high precision and recall might not necessarily be grammatically and/or semantically correct.

The edit distance (also called Levenshtein distance [16]) has also been used for text evaluation (e.g. [4]). This technique takes the sequence of words into account and is calculated in the simplest form as the number of delete, insert and substitute operations required to change the machine text into its human solution equivalent. Typically, different costs are associated with each of these edit operations. Nevertheless, the edit distance can give misleading values as well because the same piece of text can be written in several ways without loss of meaning. In particular, machine texts with a longer length will be unfavourably penalized by this technique.

The link between MT and TCBR has been previously employed to enhance retrieval [12] . MT models are used to predict links between each keyword in the problem to one or more solution keywords in the vocabulary. Such alignments were used to generate a pseudo-solution for a new query using the statistically best solution keywords linked to keywords in the query. The pseudo-solution and original query texts are used to retrieve similar cases rather than the query text alone. This led to improvements in retrieval accuracy. Our focus is different from this; we apply MT evaluation techniques rather than MT models to TCBR.

3 Challenges with Evaluating Textual Solutions

This section provides an overview of a textual reuse approach called Case Retrieval Reuse Net (CR2N) which helps to identify relevant sections in a retrieved solution text. Detailed discussion of the technique can be found in previous work [1,2]. The focus here is to highlight the challenges faced during experimental evaluation on a health and safety incident reporting domain. We present the domain of application and our task of generating textual solutions before discussing the related evaluation challenges.

3.1 Health and Safety Incident Reports

Our corpus consists of health and safety incident reports (H&S dataset) provided by the National Health Service in Grampian. A report consists of a textual description of the incident and the action taken by the health personnel on duty. Each record is also labelled with 1 of 17 care stage codes which identifies a group of records such as accidents that result in personal injuries, incidents during treatment or procedures etc. Our intention is to build a TCBR system

that assists less experienced health personnel to generate reports when resolving/recording incidents by using previous similar experiences. Therefore, the incident description serves as our problem while the solution text is the record of actions taken to resolve the incident for each case in our experiments.

3.2 Textual Solution Generation with CR2N

In previous work, we introduced the CR2N architecture for text reuse [1,2]. Here we discuss how this architecture is used to generate textual solutions and briefly outline the key steps. The CR2N architecture consists of two Case Retrieval Nets CRNs [15]: one to index the problem space and the other referred to as the Case Reuse Net (CReuseNet) for the solution space. Figure 1 illustrates the CR2N approach to annotating a retrieved solution text on a simple case base of six cases. There are five terms from the problem vocabulary (i.e. Problem Information Entities, PIEs) and four terms from the solution vocabulary (i.e. SIEs) respectively. Given a query, the best case (C_2 in figure 1) is retrieved by activating all relevant PIEs to the query which consists of PIE_1, PIE_2, PIE_4. Generally the more activations the more relevant a case is to the query. The activations are shown as solid arrows as opposed to dotted arrows for inactive links between information entities and the cases.

Generation of a solution text begins with the activation of SIEs from the most similar case. An SIE is a textual unit such as a keyword, phrase or sentence. When an SIE activates similar cases to those activated by the query within a specified k-neighbourhood of the retrieved solution, it is considered relevant to the query. Such a relevant solution term (SIE) becomes part of the solution text generated by the CR2N, otherwise it is discarded. The optimal k-value is determined empirically but has been found to be about one-third (or less) of the

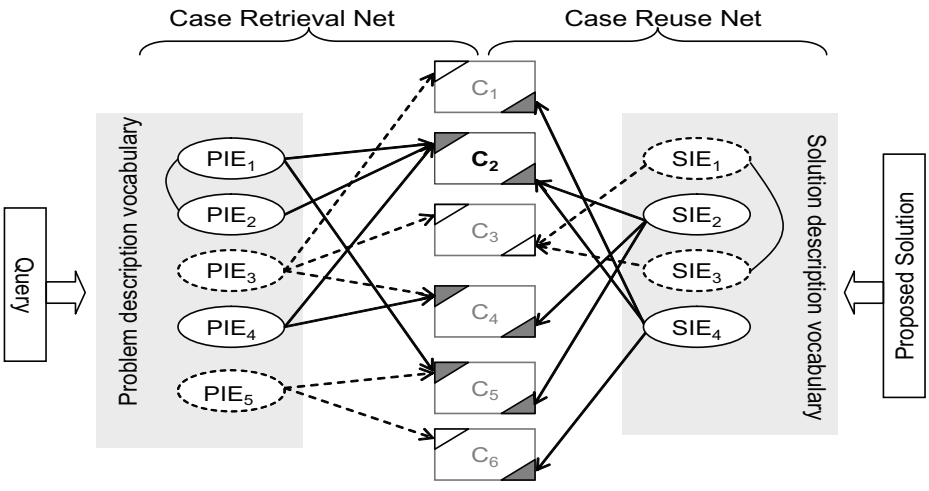


Fig. 1. The Case Retrieval Reuse Net (CR2N) architecture

size of the casebase [2]. Notice that this approach to solution text generation is different from directly proposing the solution from the best match case. Instead CR2N's approach is more akin to solution reuse, whereby the best solution is analyzed and only relevant parts of the solution are utilized to generate the proposed solution for a query. The proposed solution generated by CR2N consists of a list of reusable textual units. A complete solution is obtainable when other relevant textual units absent in the retrieved solution are added and all textual units are then put together to form a contextually coherent piece of text during revision.

3.3 Challenges with Evaluation

Quality of generated solution text from CR2N measured with precision, recall and accuracy metrics is reported in [2] using two domains: H&S incident reporting and weather forecast text generation. Both domains have the problem and solution in textual form. However, they also exhibit different textual characteristics such as vocabulary size, problem and solution vocabulary overlap and the use of synonyms. These characteristics influence the evaluation results; for instance, a large vocabulary size could mean that semantically similar texts will have few keywords in common. We compared our CR2N results with a baseline retrieve-only system and it showed a significantly better performance in both domains. However, we observed that the precision, recall and accuracy scores were comparatively lower (less than 0.5) with the H&S dataset compared to the weather forecast corpus (greater than 0.7).

Further investigation showed that these values were misleading in that proposed solutions judged relevant by a human would be judged otherwise by these basic metrics. This is because our evaluation measures (precision, recall & accuracy) only count matching keywords using their stems, lemma or synonyms.

Table 1. Sample retrievals from the H&S dataset

	Query	**Retrieved Problem (PIEs)**	**Similarity**	**Retrieved Solution (SIEs)**	**Reference Solution**	**Precision**
1	nurse slipt and fell on wet floor	staff member slid on something wet and fell to the floor	0.6124	examined by nursing staff	nurse given first aid	0.333
2	patient fell to the ground as nurse assisted him to bed	patient fell out of bed	0.7071	examined by medical staff	patient was advised to get assistance in and out of bed	0.0
3	needlestick injury sustained	needlestick injury sustained by a member of staff	0.7746	first aid, blood sample taken, visited occupational health	occupational health contacted	0.333

Therefore, they are unable to capture variation in phrases/sentences that have similar meanings but expressed by a completely different set of keywords. Poor accuracy results were also reported when retrieved solutions are more verbose than the reference solution.

Table 1 shows three incident queries as well as the retrieved case, similarity value and retrieval accuracies. With query 1, although the retrieved and reference solutions are similar in meaning, retrieval accuracy is just 0.333. This is because 1 out of 3 keywords ("nurse/nursing") is matched in the retrieved solution and the remaining keywords though semantically similar are lexically different. Query 3 poses a similar challenge while query 2 highlights a slightly different problem. Here, the level of detail/abstraction in the reference solution is different from retrieved solution thereby causing the accuracy to be calculated as 0.0.

Our hypothesis is that the use of multiple references in MT evaluation techniques will better capture the inherent variability in vocabulary as observed in the H&S dataset. The use of multiple references might also be able to reduce the problem associated with different levels of abstraction.

4 MT Evaluation Techniques

Machine Translation (MT) is a research area that deals with techniques to enable automated translation from one language to another. There is therefore a need to evaluate such machine generated translations (usually in textual form) for grammatical and semantic correctness. Initial research in MT used human expert translators for evaluating several aspects of a translated text in terms of adequate coverage, semantic meaning and grammatical correctness [20,11]. However, more recent work [17,9] has reduced the demand for user-driven quality assessments by developing automated text comparison techniques with high correlation to human judgements. As a result, automated MT evaluation techniques are quick, inexpensive, language independent and repeatable.

4.1 BLEU

BLEU [17] (**Bi**-**L**ingual **E**valuation **U**nderstudy) is an automated MT evaluation technique and it was used as an understudy of skilled human judges in translation. The idea is to measure the closeness of a machine text to its human equivalent using weighted average of phrases matched with variable length (n-grams). It enables the use of multiple reference solutions from different experts and this allows for legitimate differences in word choice and order. The BLEU score is a precision-based metric which can use multiple reference solutions and aggregates the precision scores from different word lengths; this concept is known as *modified n-gram precision*. BLEU also ensures that the machine text's length is comparable to the reference solutions' using *brevity penalty*.

Modified n-gram precision matches position independent n-grams; where $n \geq 1$ and grams are typically keywords but can include stand-alone special characters and punctuations. This is similar to precision measure in Information

Retrieval (IR). However, it is modified to ensure that n-grams can be matched across multiple reference solutions. Each n-gram is matched in only one of the reference solutions with the maximum count for such n-gram. The overall n-gram precision is a geometric average of all individual precisions from 1 to n. Using $n = 4$ has been found to give the best correlation to human judgements [17]. In comparison to the criteria used in human evaluation, uni-gram precision (i.e. $n = 1$) measures adequate coverage of a machine text while n-gram precision (when $n > 1$) shows grammatical correctness.

Brevity Penalty (BP) on the other hand ensures that the length of machine text is penalized if it is shorter than all the reference solutions. This is because a text of shorter length might have a very high n-gram precision if most of its keywords occur in any of the reference solutions. Therefore, modified n-gram precision alone fails to enforce proper translation length. BP focuses mainly on penalizing shorter machine texts as unnecessarily long texts will have been penalized by the modified n-gram precision. Although recall has been combined with precision to overcome problems with text lengths in some areas like IR, it cannot be used in BLEU because it employs the use of multiple references and each reference might use different word choices and order. Also, recalling all choices is bad since a good translation will only use one of the possible choices. BP is formulated as a decaying exponential function which gives a value of 1 when machine text's length is greater than or identical to any of the reference solutions length otherwise $BP < 1$. The BLEU metric is calculated as follows.

$$p_n = \frac{\sum_i \left(\begin{array}{c} \#\text{ of } n\text{-grams in segment } i \text{ of machine text} \\ \text{matched in segment } i \text{ of any of the reference solutions} \end{array} \right)}{\sum_i (\#\text{ of } n\text{-grams in segment } i \text{ of machine text})}$$

$$BP = \begin{cases} 1 & \text{if } l_{sys} > l^*_{ref} \\ \exp^{(1-\frac{l^*_{ref}}{l_{sys}})} & \text{if } l_{sys} \leq l^*_{ref} \end{cases}$$

$$BLEU = BP \cdot \exp(\sum_{n=1}^{N} \frac{1}{N} \log p_n)$$

where

$p_n = n$-gram precision $\quad BP =$ brevity penalty
$l_{sys} =$ length of machine text $\quad i = 1$ for TCBR
$l^*_{ref} =$ nearest reference solution length to machine text
$N =$ maximum size of n-gram (i.e. $n = 1 \ldots N$)

It is important to note that the entire text is typically regarded as one segment in TCBR (i.e. $i = 1$) when calculating p_n. This is because there is usually no knowledge of aligned segments between proposed and reference texts unlike MT where translations are done segment by segment. Figure 2 shows an example from our H&S dataset with multiple references and is used in the sample BLEU calculation shown in figure 3. Here, we compare the generated solution with

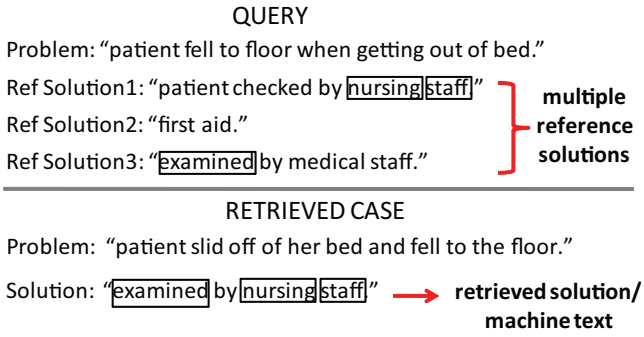


Fig. 2. A test case with multiple reference solutions

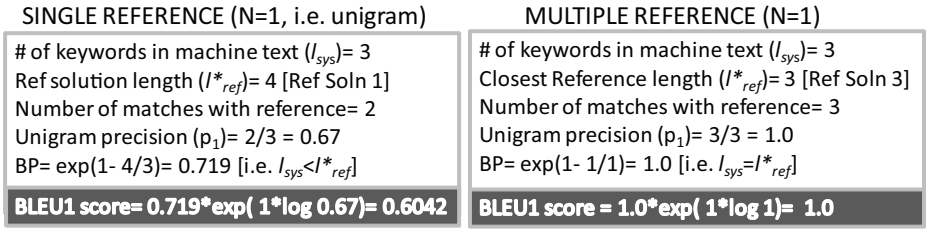


Fig. 3. A sample BLEU calculation with H&S dataset

the three reference solutions. Precision with a single reference solution (say Ref Solution1) matches only keywords "nursing" and "staff" from the machine text. However, keyword "examined" is also matched when multiple reference solutions are in use. A larger BLEU score is therefore obtained with multiple references.

4.2 NIST n-gram Co-occurrence Statistics

NIST n-gram co-occurrence statistics [9] is a more sophisticated MT evaluation technique. It was designed while experimenting with BLEU for stability and ability to reliably predict human quality assessments. NIST builds on the BLEU idea by modifying the weighting scheme for calculating precision. This is done by using information weights rather than frequency of occurrence and an arithmetic average of n-gram weights as opposed to geometric mean of n-gram precisions. Information weights are computed for n-grams such that those that occur less frequently have more weights as they are deemed to be more informative. In addition, brevity penalty was modified to minimize the impact of small variations in the generated text's length as they do not generally affect human judgements.

A significant improvement in stability and reliability was reported with NIST when compared with BLEU from experiments across several copora[9]. In other words, NIST is less sensitive to variation in the level of human expertise. Its correlation to human judgement is also more consistent across corpora from different languages. The NIST formula is given below.

$$BP = \begin{cases} 1 & \text{if } l_{sys} > \overline{l}_{ref} \\ \exp\left\{\beta \log^2\left[min\left(\frac{l_{sys}}{\overline{l}_{ref}}, 1\right)\right]\right\} & \text{if } l_{sys} \leq \overline{l}_{ref} \end{cases}$$

$$\begin{aligned} \text{info}(n\text{-gram}) &= \text{info}(w_1 \dots w_n) \\ &= \log_2\left(\frac{\#\text{ of } w_1 \dots w_{n-1} \text{ in reference solutions}}{\#\text{ of } w_1 \dots w_n \text{ in reference solutions}}\right) \end{aligned}$$

$$NIST = BP \cdot \sum_{n=1}^{N} \left\{ \frac{\sum \forall n\text{-gram} \in \text{sys}\, \text{info}(n\text{-gram})}{\#\text{ of } n\text{-grams in machine text}} \right\}$$

where

w= a word in the machine text
info= information weight
N = maximum size of n-gram (i.e. $n = 1 \dots N$)
$\beta = -4.3218$, chosen such that BP=0.5 when $l_{sys}/\overline{l}_{ref}$=2/3
l_{sys} = number of words in machine text (sys)
$\overline{l}_{ref}$ = average number of words in reference solutions

A sample NIST calculation which also uses the example test case from our H&S dataset (see figure 2) is shown in figure 4. NIST penalizes shorter machine text more as shown by the smaller BP score as compare to BLEU's for a single reference. As expected, the NIST values obtained are larger than BLEU's due to the use of information weights. NIST values can also be greater than 1 as opposed to BLEU values which are always between 0 and 1. Larger NIST (or BLEU) scores indicate better machine text's correlation to human judgement.

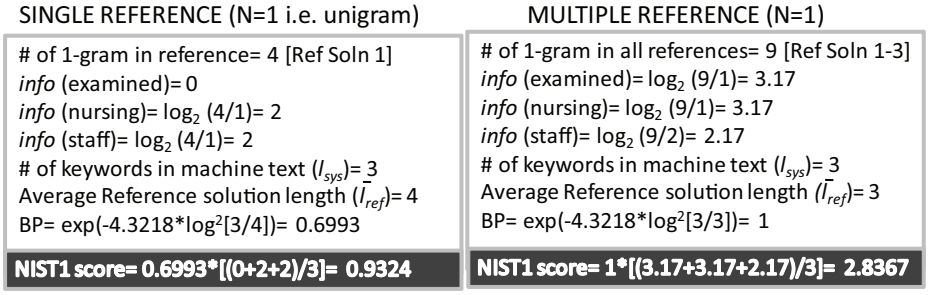


Fig. 4. A sample NIST calculation with H&S dataset

5 Experimental Setup

We evaluate the quality of the text generated by CR2N on the H&S dataset using the MT evaluation metrics, BLEU and NIST, discussed in sections 4.1 and 4.2 respectively after creating a new dataset with multiple references. Our

new dataset is also evaluated using the previous metric of precision and compare results with those obtained from the MT metrics. Multiple references give better evaluation results as they are better able to capture grammatical variations in texts but obtaining multiple references is not trivial. Therefore a novel introspective approach is employed to generate these references for our evaluations.

5.1 Generation of Dataset with Multiple Human References

Our original H&S incidents dataset consist of 362 cases belonging to the same care stage code. Each case has just 1 sentence in both the problem and solution texts since our evaluation metrics work at keyword granularity and alignment of sentences across cases are unknown. A new dataset with multiple reference solutions is needed to test our hypothesis that multiple references capture variability in word choice/order during evaluation. However, such multiple references were absent in the original H&S dataset. The CBR assumption that *similar problems have similar solutions* implies that identical problems should have same solutions. We therefore exploited this similarity assumption to create a new dataset from the original dataset with multiple references which was hitherto absent. This is done in a leave one out experiment design where each case is used as a query to retrieve the nearest neighbours. Solutions from neighbours with a similarity of 1 are then selected to form multiple reference solutions for each case while ignoring identical solutions. Here, a similarity of 1 does not necessarily mean that the problem texts are identical. This is because our similarity metric uses a bag of word representation in which stop words are removed and keywords stemmed. This process led to the extraction of 34 cases generally with 2 to 4 multiple non-duplicated reference solutions. An example of such a test case with three solutions is shown in Table 2. These 34 cases were used as test cases while the remaining 328 cases formed our case base.

Table 2. A sample test case with multiple solutions created from the previous dataset

Problem: “patient fell to floor when getting out of bed.”
Solution1: “patient checked by nursing staff.”
Solution2: “first aid.”’
Solution3: “examined by medical staff.”

The problem and solution texts are preprocessed using the GATE library [8] where texts are split into keywords. Stop words are removed and keywords stemmed to cater for morphological variations. During evaluation, synonym keywords are matched using WordNet [10] as well as keywords with the same lemma but different stems (e.g. gave/ given, fallen/ fell etc).

5.2 Evaluation and Discussion

We explore the usefulness of MT metrics, BLEU and NIST, when comparing two text reuse techniques.

1. Baseline retrieve-only system
2. Textual solution generation with CR2N

The average precision is also measured in addition to the two MT metrics using single and multiple reference solutions. The evaluation results for average precision, BLEU and NIST are shown in Tables 3A, B & C respectively. It can be seen across all three tables that the use of multiple reference solutions for text (retrieved or CR2N generated) evaluation always gives better results than using a single reference solution. Close examination of the 34 test cases suggests that these improvements are intuitive and better aligned with human judgement. This is because multiple references reduce the effect of variability in the domain vocabulary on our evaluation metrics thereby giving higher values that correlate better with human judgements. This also aligns with the reason why qualitative text evaluation typically involves the use of multiple human experts to reduce bias to a certain style of writing. We therefore suggest that multiple reference solutions (when available) should be utilized for TCBR evaluation but they can also be learnt introspectively from the casebase as explained in Section 5.1.

The result in Tables 3A & B also show that the precision scores are identical to BLEU when $N = 1$; this means that the length of most retrieved or CR2N generated solution texts were identical to one of the references implying that the brevity penalty has no effect. The brevity penalty is the only thing that differentiates precision from BLEU when $N = 1$. Therefore, the average BLEU score is expected to be less than precision's if it has an effect. This effect is illustrated in figure 3 when a single reference is used; precision is 0.67 while BLEU score is 0.6042 due to a brevity penalty of 0.719. The fact that the brevity penalty has no effect is generally true for TCBR since generated textual solutions are obtained from reference solutions to similar problems unlike MT where generated text can be shorter.

We use $k = 9$ for the CR2N after conducting an empirical study on the neighbourhood size. As shown in the Table 3, average retrieval and CR2N results are generally comparable across all 3 metrics; precision, BLEU, and NIST. Tests of statistical significance also showed no significance between each pair of retrieval/CR2N results ($p = 0.7107 > 0.05$ at 95% confidence). This shows that the CR2N has no considerable improvement over retrieval for the 34 test cases with multiple solutions used in our experiments. This can be explained by the fact that most of the retrieved solution texts (description of the action taken) were sufficient to assist a health personnel to solve the test queries (incident descriptions) when checked manually. Over 80% (28 out of 34) of the retrieved solution texts can also be reused verbatim during documentation of incidents with very little modifications. It is important to emphasize here that CR2N captures this since it is not worse than retrieval's results according to the three metrics. Nevertheless, averages are not able to show certain patterns if the difference in average between two result sets is small but the data is skewed with a comparatively large standard deviation (SD).

Further investigation revealed that the standard deviation of the individual 34 results were large as compared to the average; for instance, $SD = 0.46$ against

Table 3. Evaluation of textual solution generation quality in H&S incident reporting (A)Average precision (B) Average BLEU scores (C) Average NIST scores

(A)

Average Precision	$N = 1$		$N = 2$	
	Single Ref	**Multiple Ref**	**Single Ref**	**Multiple Ref**
Retrieval	0.28595	0.52124	0.14706	0.35294
CR2N ($k = 9$)	0.29160	0.53072	0.14706	0.35294

(B)

Average BLEU	$N = 1$		$N = 2$	
	Single Ref	**Multiple Ref**	**Single Ref**	**Multiple Ref**
Retrieval	0.28595	0.52124	0.15161	0.40167
CR2N ($k = 9$)	0.29160	0.53072	0.15161	0.40167

(C)

Average NIST	$N = 1$		$N = 2$	
	Single Ref	**Multiple Ref**	**Single Ref**	**Multiple Ref**
Retrieval	0.43575	1.31441	0.43575	1.34382
CR2N ($k = 9$)	0.44511	1.34139	0.44511	1.37081

Table 4. Clusters of precision results indicating where CR2N improves significantly over retrieval for the 34 cases with multiple reference solutions

Precision ($N = 1$, multiple Ref)	**Number of cases**	**Average Retrieval**	**Average CR2N**
$Score = 0$	6	0	0
$0 < Score < 1$	12	0.1435	0.1704
$Score = 1$	16	1	1

average precision = 0.52 for the retrieval results with multiple references. The same phenomenon applies to the results in Table 3 where N=1 for the three evaluation metrics and N=2 for NIST. The SD for results from the use of single references was generally greater than their averages. We discovered that the results where CR2N slightly improves over retrieval formed three natural clusters: $score$=0, $0 < score < 1$ and $score$= 1 as shown in Table 4 and Figure 5. The 6 cases with zero retrieval scores (cluster 1 in Figure 5) cannot be improved since it means that none of the retrieved keywords matches the query's reference solutions. The CR2N aptly identifies this by discarding all of these keywords during it text generation process. However, this cannot be captured by the precision measure as well as the MT metrics since they do not take true negatives into account. CR2N also uses all keywords in its generated text for the 16 cases where retrieval precision is one (cluster 3). Importantly, it is able to identify when all keywords in the retrieved solution text should be included in its generated text solution. The CR2N generated text outperforms retrieval for the 12 middle cases with retrieval scores between 0 and 1 (cluster 2) and this is significant at 95%

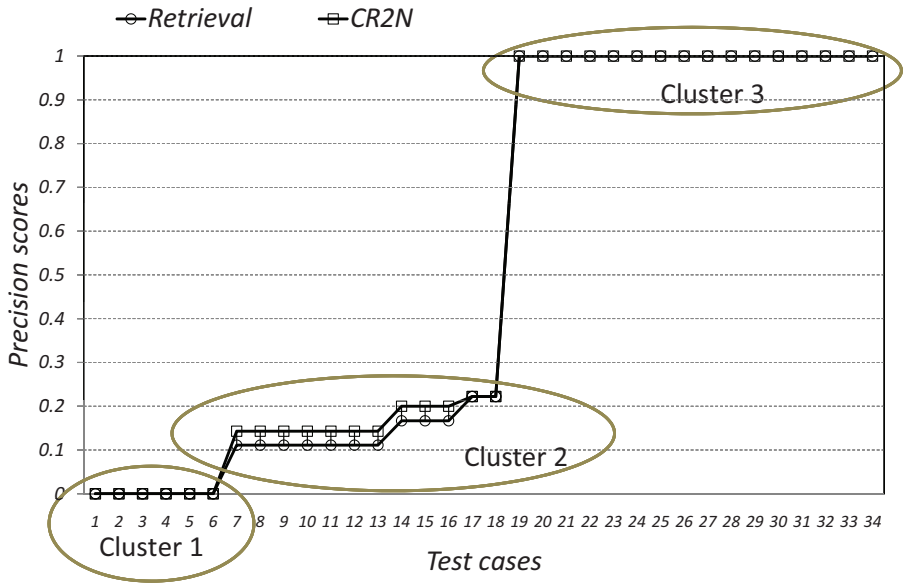


Fig. 5. Graph of precision results for the 34 test cases with multiple references

($p = 0.045 < 0.05$). A similar trend is observed for the BLEU and NIST results. Here, precision and the MT metrics are therefore only able to show improvements in retrieval when the retrieval scores are greater than zero.

6 Conclusion

The use of MT evaluation metrics to evaluate quality of generated textual solutions for TCBR is the main contribution of this paper. Two MT metrics, BLEU and NIST are adapted for TCBR evaluation with multiple reference solutions. We also propose a novel introspective method to generate multiple references when they do not naturally occur in a domain. Multiple references reduce the effect of different writing styles or variations in word choice on text evaluation. They therefore give more reliable and accurate results that correlate better with human judgements. Experimental results on a health and safety incidents dataset gave better results that were closer to human judgements with multiple reference solutions as opposed to the use of single references. We intend to carry out an extensive user evaluation to quantify the correlation of these MT metrics with human judgements for this dataset.

We also discovered that parameters like brevity penalty are not very important for TCBR because the generated texts are usually not significantly different from the reference solutions in length. We intend to verify this further by applying the MT metrics to other TCBR domains where multiple references are available or can be created introspectively.

Acknowledgements. This research work is funded by the Northern Research Partnership (NRP) and the UK-India Education and Research Initiative (UKIERI).

References

1. Adeyanju, I., Wiratunga, N., Lothian, R., Sripada, S., Craw, S.: Solution reuse for textual cases. In: Petridis, M. (ed.) Proceeding of 13th UK Workshop on Case-Based Reasoning, pp. 54–62. CMS Press, University of Greenwich (2008)
2. Adeyanju, I., Wiratunga, N., Lothian, R., Sripada, S., Lamontagne, L.: Case retrieval reuse net (CR2N): Architecture for reuse of textual solutions. In: McGinty, L., Wilson, D.C. (eds.) ICCBR 2009. LNCS, vol. 5650, pp. 14–28. Springer, Heidelberg (2009)
3. Baeza-Yates, R., Ribeiro-Neto, B., Bertino, E., Brown, E., Catania, B., Faloutsos, C., Ferrari, E., Fox, E., Hearst, M., Navarro, G., Rasmussen, E., Sornil, O., Ziviani, N.: Modern Information Retrieval. Addison-Wesley, Reading (1999)
4. Belz, A.: Statistical generation: Three methods compared and evaluated. In: Proceedings of the 10th European Workshop on Natural Language Generation (ENLG 2005), pp. 15–23 (2005)
5. Belz, A., Reiter, E.: Comparing automatic and human evaluation in NLG. In: Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2006), pp. 313–320 (2006)
6. Brüninghaus, S., Ashley, K.D.: Evaluation of textual CBR approaches. In: Proceedings of the AAAI 1998 Workshop on Textual Case-Based Reasoning, pp. 30–34. AAAI Press, Menlo Park (1998)
7. Brüninghaus, S., Ashley, K.D.: Reasoning with textual cases. In: Muñoz-Ávila, H., Ricci, F. (eds.) ICCBR 2005. LNCS (LNAI), vol. 3620, pp. 137–151. Springer, Heidelberg (2005)
8. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A framework and graphical development environment for robust NLP tools and applications. In: Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics, ACL 2002 (2002)
9. Doddington, G.: Automatic evaluation of machine translation quality using n-gram co-occurence statistics. In: Proceedings of the 2nd International Conference on Human Language Technology, pp. 138–145. Morgan Kaufmann, San Francisco (2002)
10. Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database. MIT Press, Cambridge (1998), `http://wordnet.princeton.edu`
11. Hovy, E.H.: Toward finely differentiated evaluation metrics for machine translation. In: Proceedings of the Eagles Workshop on Standards and Evaluation (1999)
12. Lamontagne, L., Langlais, P., Lapalme, G.: Using statistical models for the retrieval of fully-textual cases. In: Rusell, I., Haller, S. (eds.) Proceedings of FLAIRS 2003, pp. 124–128. AAAI Press, Menlo Park (2003)
13. Lamontagne, L., Lapalme, G.: Textual reuse for email response. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 234–246. Springer, Heidelberg (2004)
14. Lenz, M.: Textual CBR and information retrieval - a comparison. In: Gierl, L., Lenz, M. (eds.) Proceedings of the Sixth German Workshop on Case-Based Reasoning (1998)
15. Lenz, M., Burkhard, H.D.: Case retrieval nets: Basic ideas and extensions. In: Görz, G., Hölldobler, S. (eds.) KI 1996. LNCS, vol. 1137, pp. 227–239. Springer, Heidelberg (1996)

16. Levenshtein, V.: Binary codes capable of correcting deletions, insertions and reversals. Soviet Physics- Doklady, Cynernetics and Control theory 10(8), 707–710 (1966)
17. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: A method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 311–318 (2002)
18. Sripada, S., Reiter, E., Hawizy, L.: Evaluation of an NLG system using post-edit data: Lessons learnt. In: Proceedings of European Natural Language Generation Workshop, pp. 133–139 (2005)
19. Weber, R.O., Ashley, K.D., Bruninghaus, S.: Textual case-based reasoning. Knowledge Engineering Review 20(3), 255–260 (2006)
20. White, J., Connell, T.: The ARPA MT evaluation methodologies: evolution, lessons and future approaches. In: Proceedings of the First Conference of the Association for Machine Translation in the Americas, pp. 193–205 (1994)

Case Retrieval Reuse Net (CR2N): An Architecture for Reuse of Textual Solutions

Ibrahim Adeyanju[1], Nirmalie Wiratunga[1], Robert Lothian[1], Somayajulu Sripada[2], and Luc Lamontagne[3]

[1] School of Computing
Robert Gordon University, Aberdeen, Scotland, UK
{iaa,nw,rml}@comp.rgu.ac.uk
[2] Department of Computing Science,
University of Aberdeen, Aberdeen, Scotland, UK
yaji.sripada@abdn.ac.uk
[3] Department of Computer Science and Software Engineering,
Université Laval, Québec (Québec), Canada
luc.lamontagne@ift.ulaval.ca

Abstract. This paper proposes textual reuse as the identification of reusable textual constructs in a retrieved solution text. This is done by annotating a solution text so that reusable sections are identifiable from those that need revision. We present a novel and generic architecture, Case Retrieval Reuse Net (CR2N), that can be used to generate these annotations to denote text content as reusable or not. Obtaining evidence for and against reuse is crucial for annotation accuracy, therefore a comparative evaluation of different evidence gathering techniques is presented. Evaluation on two domains of weather forecast revision and health & safety incident reporting shows significantly better accuracy over a retrieve-only system and a comparable reuse technique. This also provides useful insight into the text revision stage.

1 Introduction

Textual Case Based Reasoning (TCBR) solves new problems by reusing previous similar problem-solving experiences documented as text. TCBR is a subfield of Case Based Reasoning (CBR) but has evolved as a specialized research area due to challenges associated with reasoning with textual attributes as opposed to structured attributes consisting of numeric and symbolic values [1].

In structured CBR, a case is typically described using a fixed number of attributes; therefore, the reuse stage will propose a solution containing values for these fixed attributes. Although a solution is also proposed for reuse in TCBR, number of attributes differ when the solution is textual and its decomposition into sections (keywords, phrases or sentences) is viewed as attributes. The number of sections in a retrieved textual solution is also likely to differ from the actual solution. Therefore, the reuse stage for TCBR must identify sections of

L. McGinty and D.C. Wilson (Eds.): ICCBR 2009, LNAI 5650, pp. 14–28, 2009.

a solution text that are relevant (reusable) to a given problem. The rest are candidates for revision which may take the form of deletion.

In this paper, we present a novel architecture for textual reuse which identifies sections of a retrieved text as reusable or alternatively needing revision. Our architecture extends the Case Retrieval Net (CRN) and establishes evidence in support of either reuse or revision by analysing the retrieval neighbourhoods. We design an algorithm to formalise our architecture and evaluate it on two application domains: post-editing of weather forecast texts and health and safety incident reporting. Common to both domains is that the problem and solution are in textual form. However, the domains also exhibit different textual characteristics such as in vocabulary size, problem and solution vocabulary overlap and the use of synonyms. Such differences allows us to evaluate the transferability of our technique across domains.

Section 2 discusses related work in CBR reuse and distinguishes textual from structured reuse. We then explain details of our novel architecture for textual reuse in Section 3 and compare it with an existing technique in Section 4. This is followed by experimental setup and discussion of results in Section 5 with conclusions in Section 6.

2 Related Work

The concept of CBR reuse was introduced in [2] to cover different ways in which knowledge is processed from retrieved cases prior to revision. In broad terms, this consists of generative and transformational reuse. Generative reuse (also called replay) involves a trace of the retrieved solution in the context of the new problem. A search-based approach to generative reuse was proposed for configuration tasks in [3]. Solutions in this domain consist of a complex structure of elements which are captured as *states*; a domain-specific representation of a partially specified solution. A solution to a given problem is therefore generated by searching the space of solutions guided by the retrieved solution. This technique is not directly applicable to textual content because of the difficulty in capturing a partially specified text content without losing its contextual meaning within the entire solution.

Transformational reuse on the other hand is one in which the contents (all attributes and values) of a retrieved solution are copied verbatim or aggregated by consensus of retrieved solutions. This technique was exploited for automatic story plot generation [4]. A plot structure is obtained by reusing stories from a case base of tales and an ontology of explicitly declared relevant knowledge. The ontology enables measuring of semantic distance between words/structures in the query and previous problems while the solution in each case consist of fairy tale texts analysed and annotated according to Propp's morphology. Natural Language Generation (NLG) techniques are then used to describe the story plot in natural language. Although the story generated is a complete sketch of the plot, it assists screen writers in fast prototyping of story plots which can easily be developed into a story. The approach is knowledge intensive and use of a

domain specific ontology limits its transferability. However, it paves the way to exploit other interesting synergies between NLG and CBR.

A restricted form of textual reuse is presented for report writing applied to the air travel incident domain [5]. Here, textual cases consist of incident reports with one or more paragraphs grouped under a specific heading as a section. The most similar document to a query is retrieved and textual reuse is facilitated for each section of the retrieved report. This is done by presenting a cluster of other documents containing similar text under the same heading. This technique ignores the context of each section within the entire report which leads to unuseful clusters. The approach is restrictive since it cannot be used in the absence of common section headings across the set of documents.

The drawbacks observed in the work reviewed above are addressed by a text reuse technique called *Case Grouping* (*CG*) [6]. The technique demonstrated on a semi-automated email response application involves reuse of previous email messages to synthesize new responses to incoming requests. A response is a sequence of statements satisfying the content of a given request and requires some personalization and adjustment of specific information to be reused in a new context. The reuse technique annotates sentences as reuse if there is sufficient evidence that similar past problems contain this sentence. The evidence is quantified by dividing the case base into two clusters that contain similar sentence and those that don't. Query similarity to a centroid case formed for each cluster determines whether or not to reuse. The centroid case has the average value for each feature across all cases in a cluster. The use of similarity knowledge to guide reuse/revision is novel; however, use of centroids to achieve this is less desirable because two clusters could have the same centroid if the spread of cases result in similar intra-cluster distance ratios. Also, use of the entire casebase to form clusters implies that the computation is influenced by cases which have no similarity to the query nor to the retrieved case.

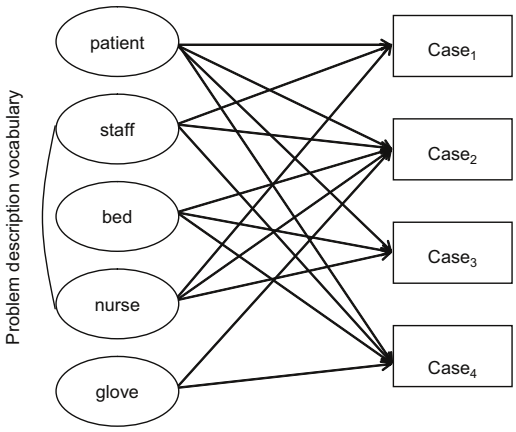
3 Case Retrieval Reuse Net (CR2N)

Our approach to reuse involves automated annotation of retrieved solution text as relevant or not. Essentially, textual units (keywords, phrases, sentences etc) annotated as relevant suggests that they can be reused without revision. In order to achieve this, we propose an extension to the CRN architecture called CR2N. The CR2N architecture consists of two CRNs: the original Case Retrieval Net (CRN) [7] which indexes the problem vocabulary and a second CRN referred to as Case Reuse Net (CReuseNet) which indexes the solution vocabulary.

3.1 Case Retrieval Net (CRN)

A CRN is a memory model that can efficiently retrieve a relatively small number of relevant cases from a case base. The model in its basic form was proposed by Lenz & Burkhard [7] although several extensions to the basic CRN such as the lazy propagation CRN [8], Microfeature CRN [8] and Fast CRN [9] have been

proposed. The CRN is efficient because it avoids exhaustive memory search and can handle partially specified queries; complete because it assures that every similar case in memory is found during retrieval[7]. It is also flexible as there are no inherent restrictions concerning the circumstances under which a particular piece of knowledge can be recalled and this is particularly useful for text.



	patient	staff	bed	nurse	glove
$Case_1$	1	2	0	2	0
$Case_2$	2	1	2	2	2
$Case_3$	1	0	1	1	0
$Case_4$	2	1	2	0	1

Fig. 1. Partial CRN for the health & safety dataset with matrix representation

The CRN uses a net-like case memory to spread activation for retrieval of similar cases to a query. It consists of four components: case nodes, Information Entities nodes (IEs), relevance arcs and similarity arcs. An IE consists of an attribute-value pair and a case consists of a set of IEs. A relevance arc connects IEs to cases and shows the presence and strength of an IE in a case while a similarity arc connects two IEs and indicates how similar an IE is to another. A case retrieval is performed by activating IE nodes which occur in a given query, propagating this activation according to similarity through the net of IEs and aggregating activation in the associated case nodes. Cases are ranked according to this aggregation and solution from the top k cases are retrieved.

When used in TCBR, each IE node is used to represent a single textual unit (keyword, phrase or sentence) depending on the granularity of indexing and similarity matching. Similarities between the textual units are then captured by the similarity arcs. A trivial CRN built for our incident reporting application is illustrated in figure 1 with its corresponding matrix representation. The figure shows how health & safety keywords relate to incident cases. A relevance arc connects an IE to a case when the keyword associated with the IE is contained in the case. For example the keywords “patient”, “staff”, “bed”, and “glove” occur in case $Case_4$. The weight on the arc typically denotes the importance of the keyword in a case. Here, we use term frequency weighting and each row in the matrix relates to a case represented as a feature vector. The similarity arc between “staff” and “nurse” indicates that the two keywords are similar

and could be learnt using word co-occurrences or from an ontology. Aggregation network activations are implemented using matrix multiplication [9].

3.2 From CRN to CR2N

The trivial example used to illustrate the components of the CR2N in figure 2 has a case base of six cases and five/four keywords in the problem/solution vocabulary respectively. The CRN retrieves the most similar case(s) to a query while the Case Reuse Net (CReuseNet) generates text annotation on the proposed solution. CRN represents the problem vocabulary of indexed cases as a mapping between IE nodes and cases containing such IEs. Case nodes are denoted as C and the problem description IEs are denoted as PIE. Mapping of IEs onto cases are shown as relevant arcs while the similarity arcs indicate the similarity between IEs. Solution description IEs in the CReuseNet are denoted as SIE to differentiate these from problem description IEs.

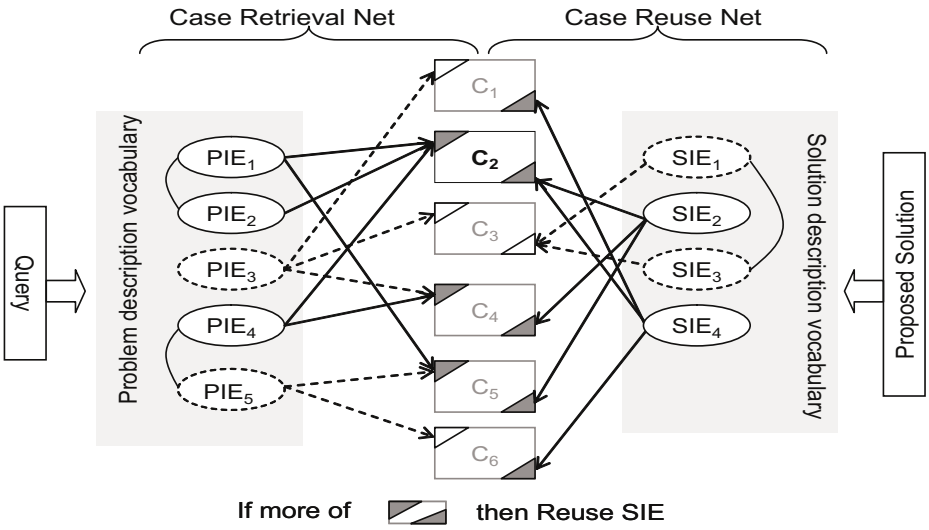


Fig. 2. The Case Retrieval Reuse Net (CR2N) architecture

A query spreads activation in the CRN through its PIEs. The most similar case is identified as that having the highest aggregation of activations (C_2 in figure 2). Each SIE from the most similar case then spreads activation in the CReuseNet to determine its reusability to the query. We decide the reusability of the SIE by comparing two retrieval sets: RS_1, the set of cases activated in the CRN by a query and RS_2, the set of cases activated by an SIE in the CReuseNet. A large intersection between RS_1 and RS_2 implies reuse of SIE otherwise revise. In other words, an SIE is reusable if a majority of the cases it activates in CReuseNet have already been activated in the CRN. For example in figure 2,

C_2 (most similar to the query) contains SIE_2 & SIE_4. SIE_2 is determined to be reusable because all cases (C_2, C_4 & C_5) activated by the query in the CRN are also activated by the SIE_2 node. On the other hand, SIE_4 is likely to need revision because it only activates one (C_2) out of the three cases activated by the query in the CRN.

4 Evidence for Annotations: Neighbouring vs. All Cases

We illustrated a simple view of our reuse architecture in figure 2 with six cases being considered in the creation of the sets RS_1 and RS_2. However, a casebase will contain many more cases. Since these sets provide evidence for our annotation judgements, we need to establish how these sets are formed for a larger casebase size. Clearly, it is sensible to use local neighbourhoods for evidence in our reuse computation rather than the entire casebase.

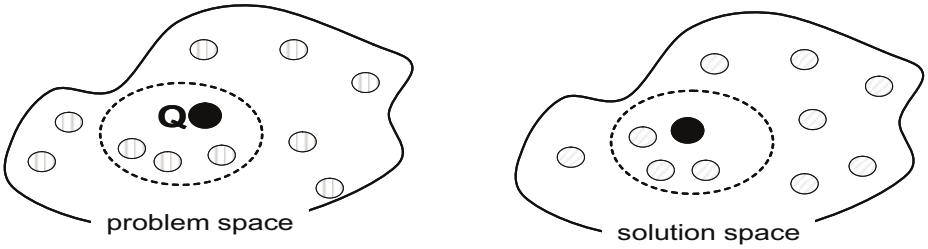


Fig. 3. Neighbourhoods in the problem and solution spaces

It is natural to use the problem space neighbourhood (i.e. query's neighbours) since similar problems should have similar solutions. This implies that evidence for reuse is computed using the query's neighbouring cases. For instance in figure 3, Q appearing in the problem space represents a query and the filled-in circle represents the best match case. The evidence for reuse/revise can be obtained from four cases nearest to the query as indicated by the outline around Q in the problem space. Alternatively, we could focus on the solution space neighbourhood consisting of the retrieved solution's neighbours. The use of this neighbourhood allows each retrieved SIE to be put in context of the entire solution during reuse computation. Such contextualisation for example enables the solution keyword *"plate"* in "food plate" to be disambiguated from "plate kit" used in surgery when applied to our health and safety incident reports.

4.1 Text Reuse with Case Grouping

Case Grouping (*CG*) [6] is a strategy which obtains evidence for textual reuse from the entire case base in its original form. An SIE in a proposed solution is annotated as reusable if there is sufficient evidence that similar past problems

also contained it in their solution. The key idea is the use of all cases in the casebase to gather evidence for or against reusing a proposed solution SIE. For each SIE in the retrieved solution, the case base is divided into 2 clusters: support and reject. The support cluster consists of cases that contain a similar SIE in their solution while the reject cluster contains cases that don't. A centroid case is then formed for each cluster by combining problem vectors of every case in the cluster. An SIE is annotated as reusable if the support centroid case is more similar to the query than the reject centroid; otherwise, the SIE is annotated as revise. Notice here that evidence for annotation is based on the whole casebase.

4.2 Text Reuse with CR2N

CR2N emphasizes the use of local neighbourhoods as opposed to CG's use a global view of the whole casebase. We formalise our CR2N architecture as an algorithm (see Figure 4) to automatically generate our textual reuse annotations (i.e. reuse/revise). The algorithm uses a generic CRN function to retrieve cases given a partial case description and an indexing vocabulary. There are two CRN function calls, with the first retrieving over the problem vocabulary, V_p, and the second over the solution vocabulary, V_s. The retrieval sets returned by the CRNs are qualified by two further Select functions: SelectK returns the top k cases, and SelectT returns all cases with similarity above a specified threshold. Although other retrieval mechanism (e.g. feature based matching) can be used, we employed CRN because of its efficiency on larger applications and its similarity arcs allows for more semantic retrieval.

The best match case C_{best}, is identified by retrieving over V_p in response to a query Q. Here Q is a case consisting of just the problem description and RS_1 is the resultant retrieval set by retrieving over V_s with the retrieved solution from C_{best}. The reuse stage involves iterating over the proposed textual solution content (i.e. C_{best}'s solution) to identify and annotate relevant parts. Like the second CRN call, the third CRN retrieves cases over the solution vocabulary given some partial solution text, which is formally denoted as a set of solution IEs or $\{sie_i\}$ in figure 4. The resultant retrieval set is RS_2. It should be noted that $\{sie_i\}$ must be a subset of C_{best}'s solution.

A solution IE is reusable by the query if cases containing it are similar to the query. In other words we want to establish if cases with similar problem descriptions to the query also contain the solution IE of interest, $\{sie_i\}$. For this purpose the retrieval sets RS_1 and RS_2 are compared. The intersection of these sets contain cases (AS) that have similar solution to the retrieved solution and also contain the sie_i, whilst the set difference identifies cases (BS) that are similar to the retrieved solution but not containing $\{sie_i\}$. The annotation is conditioned on the average similarity of the query to cases in the intersection ($\overline{S}_A$) versus that of the set differences ($\overline{S}_B$). The solution is determined to be reusable if $\overline{S}_A$ is greater than $\overline{S}_B$ else it needs revision.

The SelectK(CRN(V_s, C_{best}), k) function retrieves k cases similar to the retrieved solution. The function thereby allows the retrieved solution's overall context to be taken into account even when IEs are used for activation one at a time.

CB= $\{C_1,\ldots,C_n\}$, set of cases in the case base
V_p= $\{pie_1,\ldots,pie_m\}$, set of problem IEs in CB
V_s= $\{sie_1,\ldots,sie_l\}$, set of solution IEs in CB
C= $\{P, S\}, where(C \in CB) \wedge (P \subset V_p) \wedge (S \subset V_s)$
Q= a query, where Q $\subset V_p$
k= local neighbourhood used for reuse calculation, where k<= n

C_{best}= SelectK(CRN(V_p, Q),1)
RS_1= SelectK(CRN(V_s, C_{best}), k)
for each $\{sie_i\} \in C_{best}$
  RS_2= SelectT(CRN(V_s, $\{sie_i\}$)$, \sigma$)
  AS= $RS_1 \cap RS_2$
  BS= $RS_1 \backslash RS_2$
  $\overline{S}_A = \frac{1}{|AS|} \sum_{a \in AS} Sim(a, Q)$
  $\overline{S}_B = \frac{1}{|BS|} \sum_{b \in BS} Sim(b, Q)$
  if $\overline{S}_A > \overline{S}_B$
  then
    REUSE $\{sie_i\}$ (relevant to the query)
  else
    REVISE $\{sie_i\}$ (irrelevant to query)

Fig. 4. The CR2N Algorithm

Alternatively, neighbours of the query could have been used but our previous experiments reported in [10] showed that using neighbourhoods from solution space perform better than the problem space. The use of a specified k-neighbourhood increases the efficiency of the algorithm since a smaller number of cases are used for reuse computation. Small values of k ensure that a local neighbourhood is used for reuse computation and remove the influence of cases with little similarity to the retrieved. This is important since these cases could negatively affect the reuse computation because they reduce average similarity of AS.

The CR2N algorithm is generic because IEs can represent any form of textual units (keywords, phrases, sentences etc). Also the algorithm could still be used if each IE represents a keyword and we want to annotate larger textual units like sentences or paragraphs. This is done by using all keywords in the textual unit as a set for activation in the function SelectT(CRN($V_s, \{sie_i\}$)$, \sigma$). The best values for parameters k and σ on a given domain must be established empirically.

4.3 Distinguishing CR2N from CG

CR2N is similar to CG (see section 4.1) in that both exploit the indirect relation between a query and each textual unit in a retrieved solution by forming two sets of cases (AS/support & BS/reject). However, CR2N addresses drawbacks identified in CG as follows.

1. CR2N uses knowledge from a specified local neighbourhood to determine reusability of a solution's textual unit instead of an entire case base used in CG. This removes the influence of cases that are dissimilar to the retrieved case during reuse computation.
2. Average similarity of cases in each group to the query is employed for reuse/revise evidence in CR2N rather than centroid vectors used in CG. This is more intuitive since it takes into account similarity to the query of each case individually rather than as a group of cases.
3. Unlike CG, retrieval and reuse are integrated into a single architecture.

5 Evaluation Methodology

We evaluate the effectiveness of the reuse strategies by measuring accuracy of their annotations. We chose a retrieve-only system as our baseline since reuse succeeds retrieval and its use should improve upon retrieval results. We are also interested in the effect of different neighbourhood sizes (k) on reuse performance, we therefore repeated our experiments for increasing values of k. We compared the baseline with two textual reuse algorithms.

1. CR2N as explained in section 4.2
2. CG, as reviewed in section 4.1 but modified to use neighbourhoods (instead of the entire casebase) of the query to make it comparable to CR2N

We use a ten-fold cross validation and cosine similarity computation at both retrieval and reuse stages. Each IE in the CR2N represents a keyword from our domain vocabulary. We chose keywords as our textual units to be annotated because the size of each retrieved solution text in our application domains is small (typically 1 sentence with an average of 7 keywords).

We evaluate effectiveness of the CR2N using average accuracy, precision and recall. Our underlying hypothesis is that an effective reuse of retrieved similar cases would enhance revision and should perform better than the retrieve-only baseline. Accuracy of the CR2N is measured as a ratio of retrieved keywords correctly annotated as reuse/revise to the total number of keywords retrieved. We measure precision as a ratio of the number of keywords from the actual solution present in the proposed solution to all keywords in proposed solution. Recall is a ratio of keywords from actual solution present in the proposed solution to all keywords in actual solution. These measures are commonly used to evaluate TCBR systems [11] but have practical limitations as they are surface measures devoid of most semantics in the context of a sentence. Accuracy shows predictive performance of the CR2N and the retrieval precision is used as baseline accuracy since all retrieved keywords are deemed reusable if no annotation is done. On the hand, precision/recall indicates overall performance of our TCBR system when keywords annotated as revise by CR2N are deleted. A higher reuse precision with comparable recall over a retrieve-only system would indicate better effectiveness for a simplified TCBR system in which only delete operations are carried out during revision. However, a complete revision stage will also include substitute and insert edit operations; we intend to tackle this in our future work.

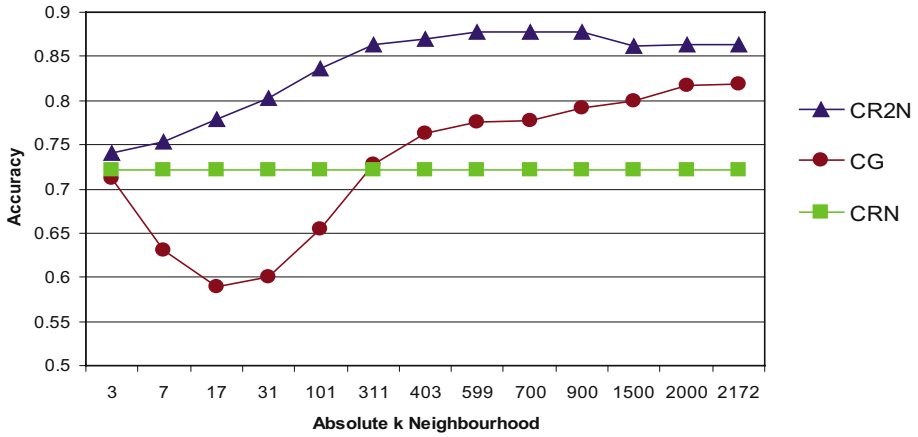


Fig. 5. Accuracy results for CR2N, CG and CRN in weather forecast revision

The problem and solution texts are preprocessed using the GATE library, available as part of the jCOLIBRI [12] framework. These attributes are divided into keywords using the GATE Splitter. Suitable stop words are also removed and keywords stemmed to cater for morphological variations.

5.1 Weather Forecast Revision

The wind dataset was extracted from a post-edit corpus [13] of an NLG weather forecast system called Sumtime Mousam (SM). The dataset consists of weather forecast text generated from numerical data by SM and its edited form after revision by domain experts. A case in our experiments therefore consists of the NLG system generated text (Unedited Text) as problem and its revised form by domain experts (Edited text) as solution.

The SM weather corpus has the following peculiar properties:

- The problem text is more similar to its solution text in a single case than to any problem text from other cases. This means that the problem & solution vocabularies are identical unless forecasters introduce new terminology. Although this is unlike most TCBR applications where the problem & solution have very few vocabulary in common (e.g. incident report datasets [14,15]), we expect that similar edit operations are applicable on solution texts.
- The indexing vocabulary is small i.e. 71/ 140 keywords in problem/ solution vocabulary respectively.
- The problem (Unedited text) is very consistent because it is generated by an NLG system with abstracted rules but the solution is not as consistent and may contain typing errors (e.g. middnight, lessbecoming).

A total of 2414 cases (from 14690) were extracted for experiments and we ensured that the average size of problem/solution text is about 1 sentence since the

reuse techniques were tested at keyword granularity. Figure 5 shows an accuracy graph comparing the retrieved similar solution (CRN), CR2N and CG from our experiments with the wind forecast dataset. The average accuracy of the CR2N clearly outperforms the baseline (precision of the retrieved solution) and CG as its curve is above them. Also, CR2N's accuracy increases with k neighbourhood of the retrieved solution, attains its best value when k=700 (about one-third of 2172 cases in the training set) and starts to decrease thereafter. This increase in accuracy with k can be attributed to the CR2N having more contextual knowledge to predict the reuse/revise of a keyword better. The decrease thereafter establishes the fact that comparison of local neighbourhoods is sufficient rather than the entire case base. The local neighbourhood is large because the vocabulary is small, therefore, majority of cases have common keywords in their solution text. CG shows a different trend; the accuracy is initially below that of the baseline (until k=17) but increases subsequently outperforming the baseline (after k=311). The initial decrease could be attributed to the misleading and unpredictable evidence from the use of centroids even when a smaller number of cases are used to create the clusters.

The average precision/recall values plotted against the absolute neighbourhood values is shown in figure 6. These curves show a similar pattern in effectiveness with the CR2N surpassing the others. The average recall of the CR2N becomes comparable to the average retrieval recall when k=900 but with higher precision. The recall of CR2N cannot be greater than the retrieval recall as keywords annotated as revise are currently treated as deletes. The CR2N's performance is generally above that of CG on the graph except when k=3 and

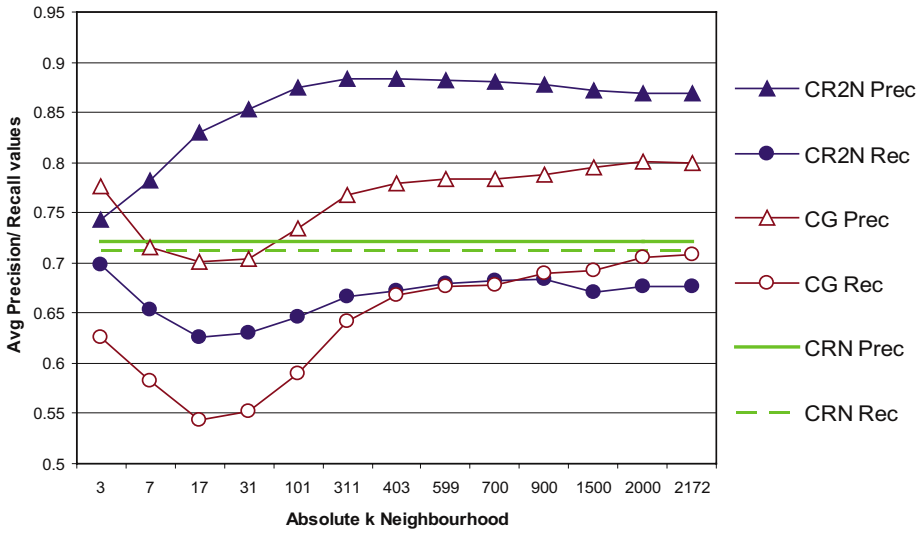


Fig. 6. Precision/Recall results for CR2N, CG and CRN in weather forecast revision

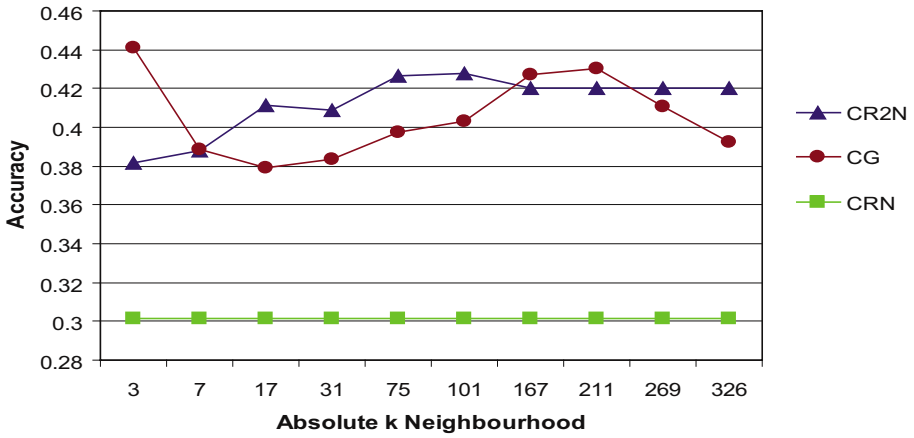


Fig. 7. Accuracy results for CR2N, CG and CRN in H&S incident reporting

k >700 for average precision and recall respectively. The higher average recall values show that CG is more conservative because it tends to reuse most keywords in a retrieved solution.

5.2 Health and Safety Incident Reporting

We also evaluated our technique on health and safety (H&S) incident reports from hospitals provided (by NHS Grampian). A report consists of a textual description of the incident and the action taken by the health personnel on duty. Each record is also labelled with 1 of 17 care stage codes which identifies a group of records such as accidents that result in personal injuries, incidents during treatment or procedures etc. Our intention is to build a TCBR system that assists less experienced health personnels when resolving/recording incidents by using previous similar experiences. Therefore, the incident description serves as our problem while the solution is the action taken to resolve the incident for each case in our experiments.

Unlike the weather forecast revision domain, health and safety incident reporting is a typical TCBR application where problem and solution vocabulary share little in common and indexing vocabulary is large (e.g. 732 keywords in solution vocabulary). Also, both problem and solution texts may contain typing errors since they are manually recorded by humans. We extracted a total of 362 cases that were grouped under a similar care stage code and having just 1 sentence in both the problem and solution texts. This allows us not only to evaluate our reuse technique at keyword granularity but makes it comparable to results from the weather domain. During evaluation, synonym keywords were matched using WordNet [16] as well as keywords with the same lemma but different stems (e.g gave and given).

Figure 7 shows an average accuracy graph comparing the baseline (CRN), CR2N and CG from our experiments with the H&S incident reports. The performance of the reuse techniques exceed the baseline as shown by their accuracy

plots. There is no clear distinction between CR2N and CG's performance but CR2N is marginally better with 6 wins out of the ten neighbourhood sizes evaluated. Overall, CR2N is most consistent with an initial increase in accuracy followed by a decrease that tappers as the neighbourhood size increases. This indicates an optimal performance when neighbourhoods are used for reuse computation as opposed to the entire case base. CG on the other hand shows a less predictable pattern with increasing neighbourhood size. In particular, the initial high accuracy is surprising. A closer look at this point (CG at $k = 3$) shows that one of the two clusters used for centroid creation was always absent leading to random behaviour that was advantageous in this instance.

CR2N's precision outperforms those of CG and CRN (see figure 8). However, the average recall of CG is better than that of CR2N emphasizing that CG is more conservative and tends to reuse most retrieved keywords. After an initial dip, CR2N's recall results remain mostly constant. The initial decline in CR2N's recall is attributed to similar problems in the dataset not sharing the same solution keywords though their solutions might have the similar meaning.

Overall, the retrieval accuracy, precision and recall results obtained are comparatively low in this domain (values are less than 0.5). A closer look suggests that values are misleading as regards the actual effectiveness of the TCBR system. This is because quantitative measures used in our evaluation only count matching keywords using their stems, lemma or synonyms. Therefore, they are unable to capture sentences that have similar meanings when expressed by a slightly different set of keywords. Poor accuracy results are also reported when the retrieved solutions are more detailed than the actual. Table 1 shows three incident queries as well as the retrieved case, similarity value and retrieval accuracies. With query 1, although the retrieved and actual solutions are similar in meaning, retrieval accuracy is calculated as just 0.333. This is because 1 out 0f 3 keywords ("nurse/nursing") is matched in the retrieved solution. Query 3

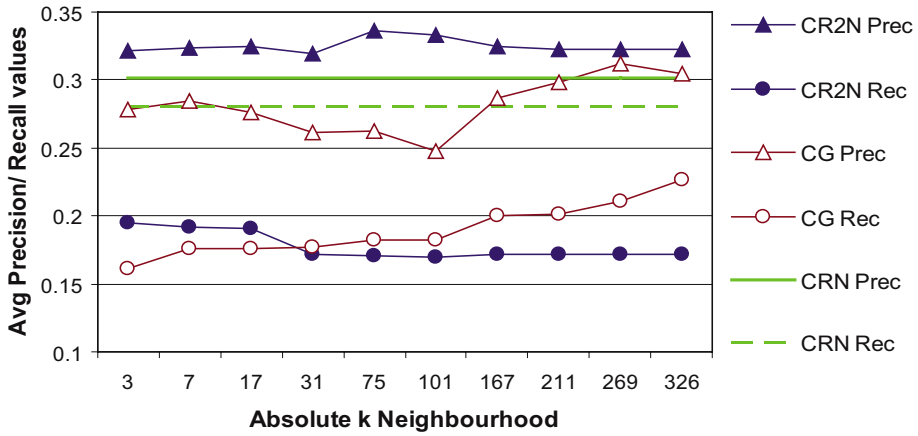


Fig. 8. Precision/Recall results for CR2N, CG and CRN in H&S incident reporting

Table 1. Sample retrievals from the Health & Safety incident reports

	Query	Retrieved Problem	Similarity	Retrieved Solution	Actual Solution	Retrieval Accuracy
1	nurse slipt and fell on wet floor	staff member slid on something wet and fell to the floor	0.61237	examined by nursing staff	nurse given first aid	0.333
2	patient fell to the ground as Nurse assisted him to bed.	patient fell out of bed.	0.7071	examined by medical staff.	Patient was advised to get assistance in and out of bed.	0.0
3	Needlestick injury sustained.	needlestick injury sustained by a member of staff.	0.7746	first aid, blood sample taken, visited occupational health.	occupational health contacted.	0.333

poses a similar challenge whilst query 2 highlights a slightly different problem. Here, the omission of information (the fact that the patient would have been examined first) caused the accuracy to be calculated as 0.0. These examples demonstrate the challenges posed by variability in vocabulary and the need for semantics-aware evaluation metrics for TCBR.

6 Conclusions and Future Work

The contribution of this work is two fold. Firstly, it proposes the reuse stage in TCBR as identification of reusable textual constructs in a retrieved solution text; the similarity assumption is used to determine reusable constructs. This is then followed by the revision of constructs that have been deemed to be non-reusable. Secondly, it provides an integration of the retrieval and reuse stages in TCBR into a single architecture called CR2N.

Three issues of *when*, *what* and *how* to revise need to be addressed when revising a piece of text. CR2N introduced in this paper addresses the issue of *what* to revise at the reuse stage by automatically annotating components of a solution text as reuse or revise. This is done by extending the CRN architecture and obtaining evidence for reuse/revise from neighbouring cases in the solution space. Experiments with CR2N on two datasets from the domains of weather forecast revision and health & safety incident reporting show better accuracy over a comparable reuse technique (CG) and a retrieve-only system (baseline).

We intend to improve CR2N by capturing context (e.g. influence of left and right adjacent keywords) for each keyword in the CReuseNet and to experiment with other levels of text granularity such as phrases and sentences. A qualitative evaluation (human validation) of our technique is needed to address problems encountered with quantitative evaluation on the health and safety incident report. We also intend to experiment with compositional text reuse where k-nearest cases of a query are combined after identifying reusable keywords by our technique.

Acknowledgements. This research work is funded by the Northern Research Partnership (NRP) and the UK-India Education and Research Initiative (UKIERI).

References

1. Brüninghaus, S., Ashley, K.D.: Reasoning with textual cases. In: Muñoz-Ávila, H., Ricci, F. (eds.) ICCBR 2005. LNCS, vol. 3620, pp. 137–151. Springer, Heidelberg (2005)
2. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AICom 7, 39–59 (1994)
3. Plaza, E., Arcos, J.L.: Constructive adaptation. In: Craw, S., Preece, A.D. (eds.) ECCBR 2002. LNCS, vol. 2416, pp. 306–320. Springer, Heidelberg (2002)
4. Gervás, P., Díaz-Agudo, B., Peinado, F., Hervás, R.: Story plot generation based on CBR. In: Twelveth Conference on Applications and Innovations in Intelligent Systems. Springer, Heidelberg (2004)
5. Recio-García, J.A., Díaz-Agudo, B., González-Calero, P.A.: Textual cbr in jcolibri: From retrieval to reuse. In: Proceedings of the ICCBR 2007 Workshop on Textual CBR: Beyond Retrieval, pp. 217–226 (2007)
6. Lamontagne, L., Lapalme, G.: Textual reuse for email response. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS, vol. 3155, pp. 234–246. Springer, Heidelberg (2004)
7. Lenz, M., Hans-Dieter, B.: Case retrieval nets: Basic ideas and extensions. In: Proceedings of 20th Annual German Conference on AI, pp. 227–239. Springer, Heidelberg (1996)
8. Lenz, M., Burkhard, H.D.: Case retrieval nets: Foundations, properties, implementations and results. Technical report, Humboldt University, Berlin (1996)
9. Chakraborti, S., Lothian, R., Wiratunga, N., Orecchioni, A., Watt, S.: Fast case retrieval nets for textual data. In: Roth-Berghofer, T.R., Göker, M.H., Güvenir, H.A. (eds.) ECCBR 2006. LNCS, vol. 4106, pp. 400–414. Springer, Heidelberg (2006)
10. Adeyanju, I., Wiratunga, N., Lothian, R., Sripada, S., Craw, S.: Solution reuse for textual cases. In: 13th UK Workshop on Case-Based Reasoning, pp. 54–62. CMS Press, University of Greenwich (2008)
11. Brüninghaus, S., Ashley, K.D.: Evaluation of textual cbr approaches. In: Proceedings of the AAAI 1998 Workshop on Textual CBR, pp. 30–34. AAAI Press, Menlo Park (1998)
12. Díaz-Agudo, B., González-Calero, P.A., Recio-García, J.A., Sánchez, A.: Building cbr systems with jcolibri. Special Issue on Experimental Software and Toolkits of the Journal Science of Computer Programming 69, 68–75 (2007)
13. Sripada, S.G., Reiter, E., Hunter, J., Yu, J.: Sumtime-meteo: Parallel corpus of naturally occurring forecast texts and weather data. Technical Report AUCS/TR0201, Department of Computer Science, University of Aberdeen (2002)
14. Massie, S., Wiratunga, N., Craw, S., Donati, A., Vicari, E.: From anomaly reports to cases. In: Weber, R.O., Richter, M.M. (eds.) ICCBR 2007. LNCS, vol. 4626, pp. 359–373. Springer, Heidelberg (2007)
15. Mudambi-Ananthasayanam, R., Wiratunga, N., Chakraborti, S., Massie, S., Khemani, D.: Evaluation measures for TCBR systems. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS, vol. 5239, pp. 444–458. Springer, Heidelberg (2008)
16. Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database. MIT Press, Cambridge (1998), `http://wordnet.princeton.edu`

Appears in M. Petridis (ed.), *Proceedings of UKCBR Workshop 2008*, CMS Press, Greenwich, pp. 54-62

Solution reuse for textual cases

Ibrahim Adeyanju[1], Nirmalie Wiratunga[1], Robert Lothian[1], Somayajulu Sripada[2], and Susan Craw[1]

[1] School of Computing
The Robert Gordon University
Aberdeen AB25 1HG, Scotland, UK
[iaa|nw|rml|smc]@comp.rgu.ac.uk

[2] Department of Computing Science,
University of Aberdeen,
Aberdeen, AB24 3UE, Scotland, UK
yaji.sripada@abdn.ac.uk

Abstract. The reuse stage in textual CBR identifies reusable textual constructs in solution content. This involves content annotation so that reusable solution text is easily identifiable from the rest. We present a generic architecture, Case Retrieval Reuse Net (CR2N), that can be used to annotate text to denote text content as reusable or not. Initial results from a weather forecast revision dataset shows up to 80% accuracy and a significantly better precision over a retrieve-only system. Although our work with CR2N is still on going, it also provides useful insight into the text revision stage.

Keyword Textual CBR, Case Reuse, Case Retrieval Net, NLG

1 Introduction

Textual Case Based Reasoning (TCBR) solves new problems by reusing previous similar problem-solving experiences documented as text. TCBR is a subfield of Case Based Reasoning (CBR) but has evolved as a specialized research area due to challenges associated with reasoning with textual attributes [1] as opposed to structured attributes consisting of numeric and symbolic values.

The reuse of retrieved similar case(s) precedes revision in TCBR's problem-solving cycle. Reuse in structured CBR typically involves using the entire solution part of a retrieved similar case since the actual solution to a new problem is similar in both content and in the number of attributes. This is not always applicable to TCBR when the solution is textual and its decomposition into sections (tokens, phrases or sentences) is viewed as attributes. The number of sections in a retrieved textual solution could differ from the actual solution; therefore, the reuse stage for TCBR requires that sections of a solution text relevant to a given problem are identified. This then would help to identify *what to revise* at the revision stage.

Natural Language Generation (NLG) is a complimentary field of research concerned with the construction of understandable texts in English (or other human languages) from some underlying non-linguistic representation of information [2]. NLG systems are usually knowledge intensive whereby text production relies on both grammatical and manually acquired rules from experts. Post-editing is a common feature of existing NLG systems (e.g. SUM-TIME METEO [3]) and involves manual revision of generated text by domain experts before presentation to final users. TCBR can be used to automate post-editing of NLG systems by reusing past editing experiences of domain experts. In this scenario, the NLG system generated text is captured as a problem experience while the edited text (by domain experts) forms the solution to that problem. We create a case base of such experiences and use it on new NLG system text by generating annotations relevant for the post-editing task.

We first present the CR2N architecture for text reuse in Section 2 while experimental setup, evaluation and discussion of results appear in Section 3. Related work and theoretical background are reviewed in Section 4, followed by conclusion and future directions in Section 5.

2 Case Retrieval Reuse Net (CR2N) for Textual reuse

Our approach to reuse involves automated annotation of retrieved solution text as relevant or not. Essentially textual units (tokens, phrases, sentences etc) annotated as relevant suggests that they can be reused without revision. In order to achieve this, we propose an extension to the CRN architecture called CR2N. The CR2N architecture consists of two CRNs: the original Case Retrieval Net (CRN) [4] which indexes the problem vocabulary and a second CRN referred to as Case Reuse Net (CReuseNet) indexes the solution vocabulary.

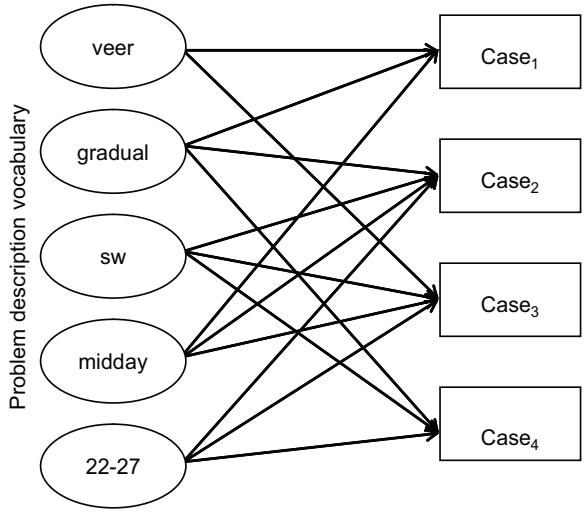
Technical details of the CRN and how the CR2N extends it for textual reuse are discussed in subsequent sections.

2.1 Case Retrieval Net(CRN)

A CRN is a memory model that can efficiently retrieve a relatively small number of relevant cases from a case base. The model in its basic form was proposed by Lenz & Burkhard [4] although several extensions to the basic CRN such as the lazy propagation CRN [5], Microfeature CRN [6] and Fast CRN [7] have been proposed. The CRN is *efficient* because it avoids exhaustive memory search and can handle partially specified queries; *complete* because it assures that every similar case in memory is found during retrieval; and *flexible* as there are no inherent restrictions concerning the circumstances under which a particular piece of knowledge can be recalled [4].

The CRN uses a net-like case memory to apply a spreading activation process for retrieval of similar cases to a query. The basic CRN consists of four components: *case nodes*, *Information Entities nodes* (IEs), *relevance arcs* and *similarity arcs* as illustrated in figure 2. An IE consists of a particular attribute-value pair and a case therefore consists of a set of IEs. A relevance arc shows the presence and strength of an IE in a case while a similarity arc indicates how similar an IE is to another. The CRN for a particular case base can be seen as a directional graph network with cases and IEs represented as nodes and the relevance arcs connecting IE nodes to their respective case nodes and similarity arcs connecting IE nodes. A case retrieval is performed by activating IEs nodes which occur in a given query, propagating this activation according to similarity through the nets of IE and aggregating activation in the associated case nodes[4]. Cases are ranked according to this aggregation and solution from the top k cases are retrieved.

When used in TCBR, each information entity (IE) node is used to represent a single textual unit (token/keyword, phrase or sentence) depending on the granularity of indexing and similarity matching. Similarity between the textual units are then captured by the similarity arcs. A CRN



	veer	gradual	sw	midday	22-27
$Case_1$	1	1	0	1	0
$Case_2$	0	1	2	1	1
$Case_3$	1	0	1	1	2
$Case_4$	0	1	2	0	1

Fig. 1. Part of the CRN for a Wind direction forecast revision dataset with matrix representation

built for the post-edit weather application is illustrated in figure 1 with its corresponding matrix

representation. A relevance arc connects an IE to a case when the token associated with the IE is contained in the case. For example the tokens "gradual", "sw", and "22-27" occur in case $Case_4$. The weight on the arc typically denotes the importance of the token in a case. Here, we use term frequency weighting and each row in the matrix relates to a case represented as a feature vector. Aggregation of activations through the network are implemented using matrix multiplication. The similarity arcs are not shown in the figure because they were not used in our experiments although they could help generalise the matrix thereby reducing any sparseness when used [7].

2.2 From CRN to CR2N

Figure 2 illustrates the components of CR2N. The Case Retrieval Net (CRN) retrieves the most similar case(s) to a query while Case Reuse Net (CReuseNet) enables text annotation on the proposed solution. CRN represents the problem vocabulary of indexed cases as a mapping between IE nodes and cases containing such IEs. Case nodes are denoted as C and the problem description IEs are denoted as PIE. Mapping of IEs onto cases are shown as relevant arcs while the similarity arcs indicate the similarity between IEs. Solution description IEs in the CReuseNet are denoted as SIE and are differentiated from the problem description IEs.

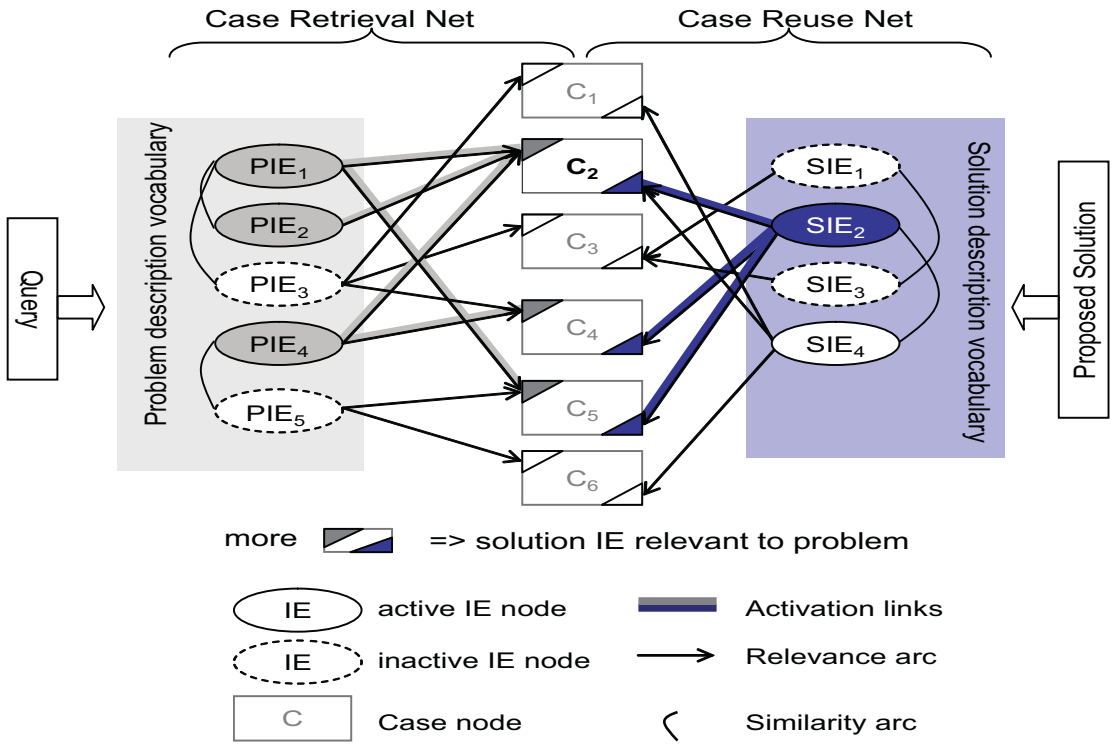


Fig. 2. The CR2N architecture

A query spreads activation in the CRN through its PIEs. The most similar case is identified as that having the highest aggregation of activations(C_2 in figure 2). Each SIE (or group of SIEs) in the most similar case then spreads activation in the CReuseNet one at a time to determine its relevance to the query. An SIE is relevant if a majority of the case nodes it activates were also activated in the CRN. For example in figure 2, C_2 (most similar to the query) contains SIE_2 & SIE_4. SIE_2 is determined to be relevant because all the cases (C_2, C_4 & C_5) activated by the query in the CRN are also activated by the SIE_2 node. SIE_4 is less relevant because it only activates one (C_2) out of the three cases activated by the query in the CRN.

2.3 Text Annotation with CR2N

Text reuse in TCBR involves the separation of reusable textual solution content from that which needs revised. This separation is suitably presented to the user as annotated text. Lamontange's reuse strategy [8] applied to email response generation annotates sentences from the proposed solution as either relevant or not. Here, a relevant annotation suggests that the sentence is directly applicable to the current problem's context and otherwise requires revision before it is applicable. We employ a similar annotation scheme but use the CR2N algorithm (see Figure 3) to automatically generate the annotations.

CB= $\{C_1,\ldots,C_n\}$, set of cases in the case base
V_p= $\{pie_1,\ldots,pie_m\}$, set of problem IEs in CB
V_s= $\{sie_1,\ldots,sie_l\}$, set of solution IEs in CB
C= $\{P, S\}, where (C \in CB) \wedge (P \subset V_p) \wedge (S \subset V_s)$
Q= a query, where Q $\subset V_p$
k= local neighbourhood used for relevance calculation, where k<= n

C_{best}= SelectK(CRN(V_p, Q),1)
RS_1= SelectK(CRN(V_s, C_{best}), k)
for each $\{sie_i\} \in C_{best}$
  RS_2= SelectT(CRN(V_s, $\{sie_i\}$), σ)
  AS= $RS_1 \cap RS_2$
  BS= $RS_1 \backslash RS_2$
  $\overline{S}_A = \frac{1}{|AS|} \sum_{a \in AS} Sim(a, Q)$
  $\overline{S}_B = \frac{1}{|BS|} \sum_{b \in BS} Sim(b, Q)$
  if $\overline{S}_A > \overline{S}_B$
  then
    REUSE $\{sie_i\}$ (relevant to the query)
  else
    REVISE $\{sie_i\}$ (irrelevant to query)

Fig. 3. The CR2N Algorithm

CR2N uses a generic CRN function to retrieve cases given a partial case description and an indexing vocabulary. There are two CRN function calls, with the first retrieving over the problem vocabulary, V_p, and the second over the solution vocabulary, V_s. The retrieval sets returned by the CRNs are qualified by two further Select functions: SelectK returns the top k cases, and SelectT returns all cases with similarity above a specified threshold.

The best match case C_{best}, is identified by retrieving over V_p in response to a problem / query Q. Here Q is simply a case consisting of just the problem description and RS_1 is the resultant retrieval set by retrieving over V_s with the retrieved solution from C_{best}. The reuse stage involves iterating over the proposed textual solution content (i.e. C_{best}'s solution) to identify and annotate relevant parts. Like the second CRN call, the third CRN retrieves cases over the solution vocabulary given some partial solution text, which is formally denoted as a set of solution IEs or $\{sie_i\}$ in figure 3. The resultant retrieval set is RS_2. It should be noted that $\{sie_i\}$ must be a subset of C_{best}'s solution.

A solution IE is relevant to the query if cases containing it are similar to the query. In other words we want to establish if cases with similar problem descriptions to the query also contain the solution IE of interest, $\{sie_i\}$. For this purpose the retrieval sets RS_1 and RS_2 are compared. The intersection of these sets contain cases (AS) that have similar solution to the retrieved solution and also contain the sie_i, whilst the set difference identifies cases (BS) that are similar to the retrieved solution but not containing $\{sie_i\}$. The annotation is conditioned on the average similarity of the query to cases in the intersection versus that of the set differences.

The SelectK(CRN(V_s, C_{best}), k) function retrieves k-cases similar to the retrieved solution. The function thereby allows the retrieved solution's overall context to be taken into account even when IEs are used for activation one at a time. The use of a specified k-neighbourhood increases the efficiency of the algorithm since a smaller number of cases are used for relevance computation. Small values of k would ensure that a local neighbourhood is used for relevance computation and removes the influence of cases with little similarity to the retrieved. This is important since cases with little similarity to the retrieved case could negatively affect the relevance computation because they reduce average similarity of AS.

The CR2N algorithm is generic because IEs can represent any form of textual units (tokens, phrases, sentences etc). Also the algorithm could still be used if each IE represents a token and we want to annotate larger textual units like sentences or paragraphs. This is done by using all tokens in the textual unit as a set for activation in the function SelectT(CRN($V_s, \{sie_i\}), \sigma$). The best values for parameters k and σ on a given textual domain must be established empirically.

3 Evaluation

We evaluate the effectiveness of CR2N by measuring the accuracy of its annotations. We chose a retrieve-only system as our baseline since reuse succeeds retrieval and its use can only be justified if it improves the retrieval results. We are also interested in the effect of different neighbourhood sizes (k) on CR2N performance, we therefore repeated our experiments for increasing values of k. We compared the baseline with two textual reuse algorithms.

1. CR2N as explained in section 2.3.
2. CR2N$_p$, a variation of CR2N by replacing SelectK(CRN(V_s, C_{best}),k) with SelectK(CRN(V_p, Q), k) in figure 3. CR2N$_p$ uses k-neighbours of the query and allows us to measure the effect of ignoring the context of the retrieved solution during relevance computation.

3.1 Dataset Preparation

The evaluation uses the wind dataset extracted from a post-edit corpus [3] of an NLG weather forecast system called SUMTIME-MOUSAM (SM). The dataset consists of weather forecast text generated from numerical data by SM and its edited form after revision by domain experts. A case in our experiments therefore consists of the NLG system generated text (Unedited Text) as problem and its revised form by domain experts (Edited text) as solution.

The SM weather corpus has the following peculiar properties:

- The problem text is more similar to its solution text in a single case than to any problem text from other cases. This means that the problem & solution vocabularies are identical unless forecasters introduce new terminology. Although this is unlike most TCBR applications where the problem & solution have very few vocabulary in common (e.g. ESA incident report dataset [9], NHS dataset [10]), we expect that similar edit operations are applicable on generated texts that are similar.
- The indexing vocabulary is very small (e.g. about 75 tokens for the wind dataset from all problem text without stemming or removal of stop words).
- The problem (Unedited text) is very consistent because it is generated by an NLG system with abstracted rules but the solution(Edited text) is not as consistent and may contain typing errors (e.g. middnight, acking, deceasing, lessbecoming).

The extracted wind forecast dataset initially contained 14,690 cases with duplicates. A total of 5011 cases were left for experiments after removing duplicate cases. The textual attributes (unedited/edited text) of cases are preprocessed using the GATE library, available as part of the jCOLIBRI [11] framework. These attributes are organised into paragraphs, sentences and tokens using the GATE Splitter. The only stop words removed are punctuation marks because the text contains normal stops as either a wind direction in the short form (e.g *s* - south) or common adverbs (e.g. gradually) which are used to indicate the trend from a wind period to another. All tokens are then stemmed to cater for morphological variations (e.g. gusts/gusting).

3.2 Methodology

We use ten-fold cross validation with k-Nearest Neighbour (k-NN) in our experiments. Cosine was used for similarity computation at both retrieval and reuse stages of the architecture and the experiment was repeated for increasing values of k at the reuse stage. Each IE in the CR2N represents a token from our domain vocabulary. We chose tokens as our textual units to be annotated because the size of each retrieved solution text in our application domain is small (typically 1 sentence).

We evaluate effectiveness of the CR2N using average precision, recall and accuracy. Our underlying hypothesis is that an effective reuse of retrieved similar cases would enhance revision and should perform better than the retrieve-only baseline. Precision is measured as a ratio of the number of tokens from the actual solution present in the proposed solution to all tokens in proposed solution. Recall is a ratio of the number of tokens from the actual solution present in the proposed solution to all tokens in actual solution. These measures (borrowed from information retrieval) are commonly used to evaluate TCBR systems [12]. We also measured accuracy of the CR2N annotation as a ratio of retrieved tokens correctly annotated as reuse/revise to the total number of tokens retrieved. The retrieval precision is used as baseline accuracy since all retrieved tokens are deemed reusable if no annotation is done. Figure 4 gives snippets from our dataset to illustrate our precision, recall and accuracy calculation.

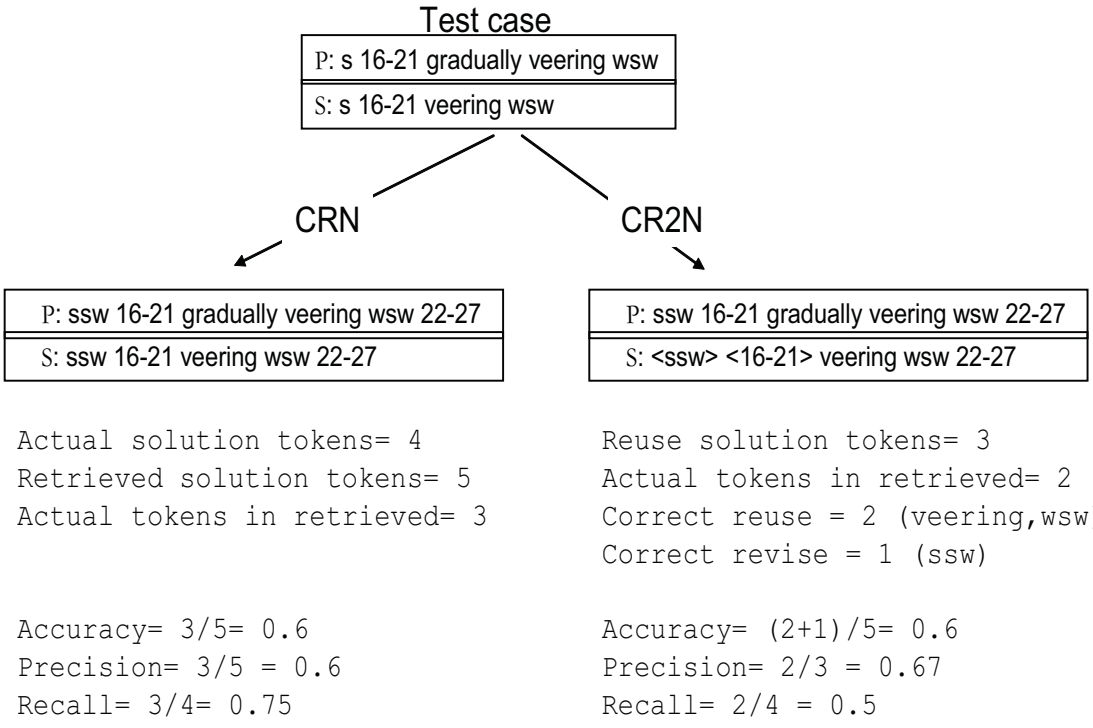


Fig. 4. Accuracy, precision & recall calculation on weather test case

Accuracy shows predictive performance of the CR2N while precision/ recall indicates its overall performance if tokens annotated as revise are ignored. A higher precision with comparable recall for the CR2N over a retrieve-only system would indicate better effectiveness.

3.3 Results

Figure 5 shows an accuracy graph comparing the retrieved similar solution (CRN), CR2N and CR2N$_p$. The accuracies of the CR2N and CR2N$_p$ increase as the neighbourhood of the query or retrieved solution is being expanded with the k parameter and outperform the baseline (precision

of the retrieved solution) when k=311. This increase in accuracy becomes marginal after k=1500 (about one-third of 4510 cases in the training set) and starts to decrease after k=2500. This increase in accuracy with increasing k can be attributed to the CR2N (or CR2N$_p$) having more contextual knowledge to predict the relevance/irrelevance of a token better. The marginal increase after, k=1500, establishes the fact that comparison of local neighbourhoods is sufficient rather than the entire case base. The efficiency of the algorithm is also improved if a fraction (k) of the case base (rather than all cases) is employed for reuse computation. CR2N also performs better than CR2N$_p$ which uses the query ranking. This shows the importance of using the context of the retrieved solution when determining relevance of a single token.

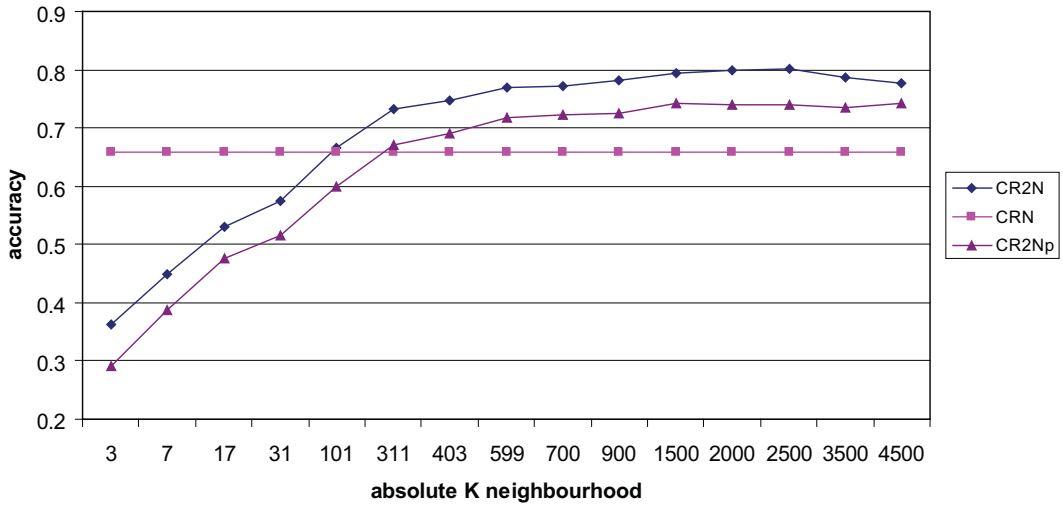


Fig. 5. Accuracy graph for the retrieved solution, CR2N & CR2N$_p$ at different k-neighbourhoods

The precision-recall curve in figure 6 shows a similar pattern in effectiveness. The average recall of the CR2N (0.6138) becomes significantly comparable to the average retrieval recall (0.6624) when k=1500 but with a much higher precision. The recall of CR2N (or CR2Np) cannot be greater than the retrieval recall as tokens can only be deleted and not inserted. The precision-recall curve of CR2N is also above that of CR2N$_p$ on the graph. This also emphasizes the significance of using the retrieved solution's context.

4 Related Work

Gervás etal [13] exploited a relationship between NLG & CBR for automatic story generation. They use CBR to obtain a plot structure by reusing stories from a case base of tales and an ontology of explicitly declared relevant knowledge. NLG is then used to describe the story plot in natural language. Although the story generated is a sketch of a plot, it assists screen writers in fast prototyping of story plots which can easily be developed into a story. The CBR approach employed is knowledge intensive and use of a domain specific ontology limits its applicability.

A supervised approach to textual reuse is proposed in [14]. Here, the most similar document to a query is retrieved using an information retrieval search engine (Lucene) and textual reuse is aided by presenting clusters containing similar documents for sections of the document. Each section is identified by a distinct heading common to all documents in the application domain (air travel incident reports). The major drawback of the approach is that it cannot be used when documents are unstructured. This means that common headings cannot be identified across documents for clustering to take place.

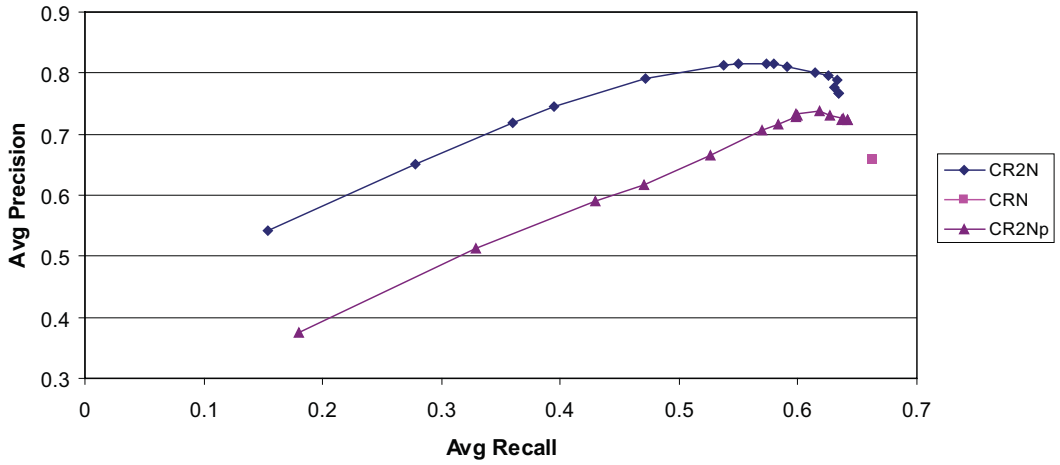


Fig. 6. Precision-Recall curve for the retrieved solution, CR2N & CR2N$_p$ at different k-neighbourhoods

Reuse of retrieved textual cases has been demonstrated on a semi-automated email response application [8]. The technique involves reuse of previous email messages to synthesize new responses to incoming requests. A response is a sequence of statements satisfying the content of a given request and requires some personalization and adjustment of specific information to be reused in a new context. Like CR2N, the reuse technique annotates sentences of the proposed solution. A sentence is annotated as reuse if there is sufficient evidence that similar past problems contain this sentence. The evidence is quantified by dividing the case base into 2 clusters that contains a similar sentence those that don't. A centroid case is formed for each cluster and compared with the query. Unlike CR2N's use of localised neighbourhood knowledge, here centroids can result in misleading evidence because two clusters of cases would have the same centroid if distance ratio between their cases are equal. Also, use of the entire case base to form clusters is inefficient for a large case base as the process has to be repeated for each sentence in a retrieved response.

5 Conclusions and Future Work

Three issues of *when*, *what* and *how* to revise need to be addressed when revising a piece of text. CR2N addresses the issue of *what* to be revised at the reuse stage by automatically annotating components of a solution text as reuse or revise. Experiments with CR2N on an NLG post-edit dataset shows up to 80% accuracy. It also has a higher precision and comparable recall to a retrieve-only system when tokens annotated as revise are ignored.

We intend to apply the technique on other textual datasets with varying vocabularies and to improve CR2N by capturing context (e.g. influence of left and right adjacent tokens) for each token in the CReuseNet. Our research also aims to develop methods that can help revise a retrieved solution text during problem solving.

Acknowledgements This research work is funded by the Northern Research Partnership (NRP) and the UK-India Education and Research Initiative (UKIERI).

References

1. Brüninghaus, S., Ashley, K.D.: Reasoning with textual cases. In Munoz-Avila, H., Francesco, R., eds.: Proceedings of the 6th International Conference on Case-Based Reasoning, ICCBR 2005, Springer Verlag (2005) 137–151

2. Reiter, E., Dale, R.: Building applied natural language generation systems. Natural Language Engineering **1** (1995) 1–32
3. Sripada, S.G., Reiter, E., Hunter, J., Yu, J.: Sumtime-meteo: Parallel corpus of naturally occurring forecast texts and weather data. Technical Report AUCS/TR0201, Department of Computer Science, University of Aberdeen (2002)
4. Lenz, M., Hans-Dieter, B.: Case retrieval nets: Basic ideas and extensions. In: Proceedings of the 20th Annual German Conference on Artificial Intelligence: Advances in Artificial Intelligence, London, Springer- Verlag press (1996) 227–239
5. Lenz, M., Burkhard, H.D.: Lazy propagation in case retrieval nets. In Wahlster, W., ed.: Proceedings of European Conference on Artificial Intelligence: ECAI-96, Los Angeles, John WIley and Sons (1996) 127–131
6. Lenz, M., Burkhard, H.D.: Case retrieval nets: Foundations, properties, implementations and results. Technical report, Humboldt University, Berlin (1996)
7. Chakraborti, S., Lothian, R., Wiratunga, N., Orecchioni, A., Watt, S.: Fast case retrieval nets for textual data. In Roth-Berghofer, T., Göker, M.H., Güvenir, H.A., eds.: Proceedings of the 8th European Conference on Case-Based Reasoning (ECCBR-06), Springer (2006) 400–414
8. Lamontagne, L., Lapalme, G.: Textual reuse for email response. In: Advances in Case-Based Reasoning, London, Springer-Verlag (2004) 234–246
9. Massie, S., Wiratunga, N., Craw, S., Donati, A., Vicari, E.: From anomaly reports to cases. In Ashley, K.D., Bridge, D.G., eds.: Case-Based Reasoning Research and Development. Springer, Berlin / Heidelberg (2007) 359–373
10. M.A., R., Wiratunga, N., Chakraborti, S., Massie, S., Khemani, D.: Evaluation measures for TCBR systems. In: Proceedings of the 9th European Conference on Case Based Reasoning, Springer press (2008) 444–458
11. Díaz-Agudo, B., González-Calero, P.A., Recio-García, J.A., Sánchez, A.: Building cbr systems with jcolibri. Special Issue on Experimental Software and Toolkits of the Journal Science of Computer Programming **69** (2007) 68–75
12. Brüninghaus, S., Ashley, K.D.: Evaluation of textual cbr approaches. In: Proceedings of the AAAI-98 Workshop on Textual Case-Based Reasoning (AAAI Technical Report WS-98-12), AAAI Press (1998) 30–34
13. Gervás, P., Díaz-Agudo, B., Peinado, F., Hervás, R.: Story plot generation based on CBR. In Macintosh, A., Ellis, R., Allen, T., eds.: Twelveth Conference on Applications and Innovations in Intelligent Systems, Cambridge, UK, Springer (2004)
14. Recio-García, J.A., Díaz-Agudo, B., González-Calero, P.A.: Textual cbr in jcolibri: From retrieval to reuse. In Wilson, D.C., Khemani, D., eds.: Proceedings of the ICCBR 2007 Workshop on Textual Case-Based Reasoning: Beyond Retrieval. (2007) 217–226

Appears in M. Schaaf (ed.), *ECCBR'08 Workshop Proceedings (Computer Cooking Contest)*, Tharax-Verlag, Hildesheim, pp. 229-238.

RaGoÛt: An Arpeggio of Tastes*

Ibrahim Adeyanju[1], Susan Craw[1], Abhishek Ghose[2],
Allyson Gray[1], and Nirmalie Wiratunga[1]

[1] School of Computing, The Robert Gordon University, Aberdeen, Scotland
[2] Department of Computer Science and Engineering, IIT Madras, India

Abstract. This paper presents the RaGoÛt system developed in response to the ECCBR'08 Computer Cooking Contest call. It proposes a case authoring approach that combines domain-specific and independent knowledge sources to create a feature vector representation of recipes. Case retrieval uses semantic similarity knowledge acquired from WordNet. This is combined with exact matching to enforce adaptation-aware case ranking, followed by substitutional adaptation of mismatched query ingredients. RaGoÛt addresses all outlined CCC tasks: compulsory, negation and menu challenges. Preliminary results from the compulsory and negation tasks are very favourable for both retrieval and adaptation. Although RaGoÛt can also retrieve three-course menus in response to the third menu task, its adaptation strategy for menus is being improved.

1 Background

The RaGoÛt system builds on RGU's Case-Based Reasoning (CBR) research on feature extraction from text [1], case retrieval nets for text [2] and adaptation [3], and benefits from IIT Madras' research on the extraction of word cohesion relations [4]. RaGoÛt was designed and implemented initially as an honours project and completed as a research project. This effort continues our participation in the Textual CBR workshop's air traffic incident report challenge [5].

Extraction of ingredients and relevant indexing terms from text content is discussed next in Section 2. Case representation and its generalisation using similarity knowledge appears in Section 3. This is followed by our retrieval and reuse strategy in Section 4. A functional overview of the RaGoÛt system appears in Section 5 with initial retrieval results and conclusions in Section 6.

2 Case Representation

The provided recipe file is a semi-structured textual document where each recipe has a recipe title labeled TI, a list of ingredients each labeled IN, and a single preparation part labeled PR. Our case representation extracts terms from the text for a recipe so that these terms can be used as an index to the recipes.

* This joint research is supported by a UKIERI standard award to the Robert Gordon University and IIT Madras.

The main content of a query is often specific ingredients. The challenge here is that words in the recipes may be redundant for recipe matching, such as teaspoon, but they may also be too specialised, such as halibut rather than fish, and adaptation by ingredient substitution must apply relationships between ingredients. The non-ingredient query terms are cuisine types, dietary practices and meal types. These may be explicitly found in any part of the recipe: TI(tle), IN(gredients) and PR(eparation). When these non-ingredient terms fail to be explicitly mentioned in the recipes, associations between recipes and terms are inferred by using the web as a source of background knowledge.

2.1 Pre-Processing of Text

The first step is to remove the stopwords that provide little meaning for the recipe retrieval task. In addition to the normal stopwords of English text such as punctuation, "a", "the", "and", ..., there are recipe-specific stopwords that are not useful for selecting a similar recipe. Common cooking units of measure such as gram, pint and cup are also removed as stopwords, as are numbers specifying any quantities. Similarly the state of an ingredient, such as chopped, grated or sliced, is regarded as not meaningful for the retrieval task. As an example, the IN(gredient) *1 cup packed brown sugar* would be reduced to *brown sugar* during pre-processing. As a final pre-processing step a very simple stemming is applied that turns plurals into their singular form.

2.2 Ingredient Extraction from Recipe Documents

The TI(tle), IN(gredient) and PR(eparation) parts of a recipe are scanned to identify unigrams and bigrams that may be ingredient terms. The example above would generate *brown*, *brown sugar*, and *sugar* as potential ingredient terms.

We use WordNet [6] as the first knowledge source to identify a term as an ingredient. The hypernym trees for the term are retrieved from WordNet and checked to see if the keyword *food* is a hypernym. Figure 1 shows the approach for *chicken*.

However, not all valid ingredients have food as a hypernym; e.g. Figure 2 shows the relevant hypernym tree for *nut*. To accommodate this, three other food-related keywords were identified: *fruit, leaven* (e.g. baking powder, yeast), and *substance* (e.g. baking soda). These keywords are used in the same way as *food* when checking the hypernym trees.

2.3 Extraction of Other Features

The cuisine types are identified in the recipe bigrams and unigrams by scanning their WordNet hypernym trees for key words related to cuisine. Using *Chinese* as a model we discovered some useful keywords: *person*, *nation*, *land*, *country* and *natural language*. Thus we identified 47 cuisine type and 19 meal type terms in the recipes in the same way as we identified ingredient terms. These extracted ingredient types and meal types are explicitly listed as domain knowledge together

chicken, poulet, volaille -- (the flesh of a chicken used for food)
=> poultry -- (flesh of chickens or turkeys or ducks or geese raised for food)
=> bird, fowl -- (the flesh of a bird or fowl (wild or domestic) used as food)
=> meat -- (the flesh of animals (including fishes and birds and snails) used as food)
=> **food**, solid food -- (any solid substance (as opposed to liquid) that is used as a source of nourishment; "food and drink")
...
=> entity -- (that which is perceived or known or inferred to have its own distinct existence (living or nonliving))

Fig. 1. WordNet Hypernym Tree for Chicken

nut -- (usually large hard-shelled seed)
=> **seed** -- (a small hard fruit)
=> **fruit** -- (the ripened reproductive body of a seed plant)
=> reproductive structure -- (the parts of a plant involved in its reproduction)
=> plant organ -- (a functional and structural unit of a plant or fungus)
=> plant part, plant structure -- (any part of a plant or fungus)
=> natural object -- (an object occurring naturally; not made by man)
...
=> entity -- (that which is perceived or known or inferred to have its own distinct existence (living or nonliving))

Fig. 2. WordNet Hypernym Tree for Nut

with manually identified dietary practices and ingredient types The ingredient type terms include *fruit, grain, meat, nut, oil, pasta, poultry, seafood, seed, vegetable*. The meal type terms include *appetizer, cake, dessert, soup, salad*. The dietary practice terms include *vegetarian* but many are ingredients that must not be included such as *nut free* or *non alcoholic*. These negative requirements are handled by the query processing (see 4.3).

2.4 Web as Background Knowledge

We found that with a majority of recipes, both the cuisine and meal type terms were not explicitly included: 669 recipes have no explicit cuisine and 453 have no explicit meal type. In order to establish and label recipes with the most likely cuisine or meal type we used web search hits to determine the cohesion between a given recipe title and each of the candidate cuisines and meal types. For example, given a recipe *cinnamon rolls*, we first note the number of hits returned with a restricted Google search query. Here the restriction is enforced so that search is confined to documents related to cooking within Google's cooking directory [7]. A

second query is constructed by combining both the recipe name and a candidate cuisine (e.g. *cinnamon rolls* + *Chinese*). In order to establish a recipe's cohesion with either a cuisine or meal type term, we combine the number of hits returned from both these queries as a ratio:

$$cohesion(recipe, term) = \frac{hits(recipe\&term)}{hits(recipe)}$$

We evaluated this web-based recipe-term association approach on a sample of recipes containing explicit cuisine and meal type terms. The top 5 highest cohesion terms associated the recipe with the correct term 80% of the time. As a result of this initial experiment, any recipe that had no explicit cuisine (or meal type) mentioned was associated with the top 6 cohesion terms. The degree of association is a function of the cohesion measure and its rank.

3 From Recipe to Case

The recipe vocabulary consists of the ingredient and cuisine type terms extracted from the recipes, together with the ingredient type, meal type and non-negative dietary practice terms provided as domain knowledge.

3.1 Bag-of-Words Representation

As a first step, each recipe is transformed into a bag-of-words representation of terms explicitly mentioned in the recipe. An additional generalisation step extends the bag-of-words by activating the ingredient type terms that were used to identify any ingredients explicitly mentioned in the recipe. The ingredient type acts as a generalised term and multiple ingredient types may be identified for an ingredient. For example Figure 1 enables a recipe that explicitly mentions the ingredient *chicken* to also activate the terms *poultry* and *meat*.

Figure 3 illustrates the background knowledge used to transform the recipe documents into a set of bags-of-words. The so far unused domain knowledge containing Chinese, Indian, Thai and other cuisine type ingredients lists ingredients distinctive of the particular cuisine. This knowledge further extends the bag-of-words by activating the cuisine type term for any ingredient found in its ingredient list that is explicitly listed in the recipe. For example an ingredient macaroni would activate the cuisine type *italian* if macaroni were listed as an Italian ingredient. These activations are further boosted with recipe-term cohesion values obtained from Web querying.

The highlighted recipe in Figure 3 is the Macaroni and Chicken Casserole recipe listed in Figure 4. Its bag-of-words is {chicken, macaroni, milk, mushroom soup, cream cheese, onion rings, poultry, meat, vegetable, italian}. Finally, the bag-of-words is represented as a feature vector of ingredient, ingredient type, cuisine type, dietary practice and meal type terms in the vocabulary.

The feature vectors for each recipe are assembled into a recipe $\times$ term matrix $\mathbf{R}$ where each of the r rows corresponds to the feature vector for that recipe,

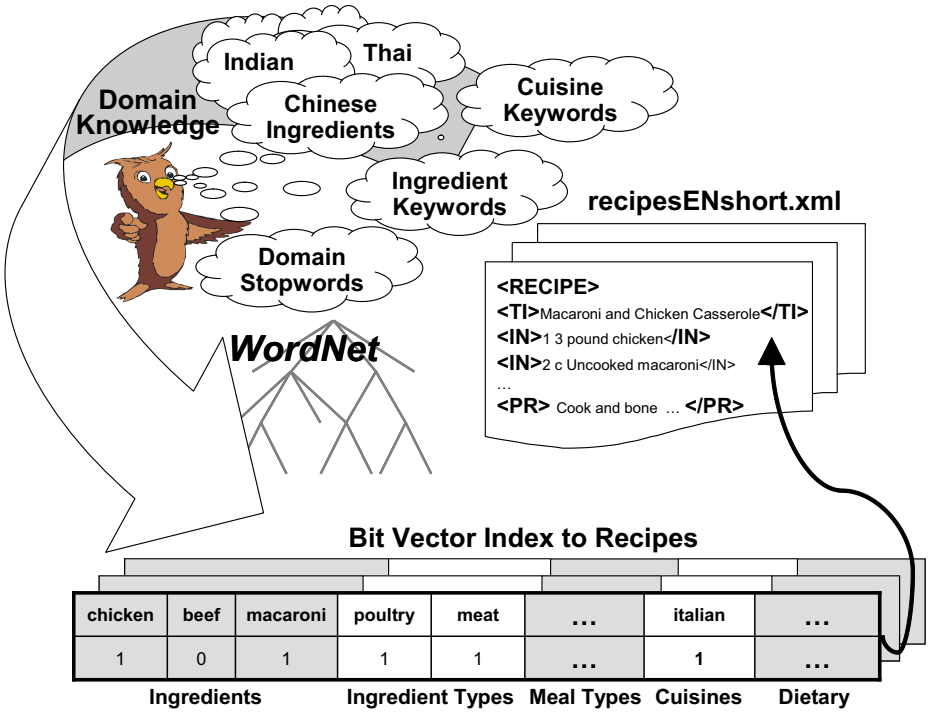


Fig. 3. RaGoÛt Case Acquisition

and the entries in each of the t columns corresponds to whether that term is relevant or not for each of the recipes. A binary 1 typically indicates that the term was explicitly present in the recipe, whilst a value less than 1 denotes semantic relevance derived from WordNet or the web background knowledge.

3.2 Semantic Relevance through WordNet

The matrix **R** identifies whether terms are relevant or not for a recipe based on the terms explicitly mentioned in the recipe. We need to extend this representation to indicate the degree of relevance of a term for a recipe even in the absence of that term. For example a recipe that uses beef as an ingredient may be worth retrieving, and adapting, when a chicken dish is sought. Thus we wish to increase the zero entry for chicken in this recipe vector to capture the similarity of beef and chicken as ingredients.

It will be convenient to note that the vocabulary of t terms comprises i ingredient terms, j ingredient type terms, m meal type terms, c cuisine terms and d dietary terms.

A term x term similarity matrix **I** is constructed for pairs of the i ingredient terms. This matrix is symmetric and its leading diagonal contains ones. The off-diagonal entries contain the similarities between the pairs of terms. The

<RECIPE> <TI>Macaroni and Chicken Casserole</TI>
<IN>1 3 pound chicken, cooked and boned</IN>
<IN>2 c Uncooked macaroni</IN>
<IN>2 1/2 c Milk</IN>
<IN>2 cn Cream of mushroom soup</IN>
<IN>1 8 oz. Philadelphia cream cheese</IN>
<IN>1 cn French fried onion rings</IN>
<PR>Cook and bone chicken. Place chicken in bottom of 13 x 9 inch pan. Pour uncooked macaroni over chicken. Pour milk over macaroni. Spread soup over mixture. Cut cream cheese into small pieces and lay over soup. Cover and place in refrigerator overnight or at least 8 to 10 hours. One hour before baking, take out of refrigerator and let set. Heat oven to 350 degrees and bake for 50 minutes uncovered. Cover with onion rings and bake for 5 to 10 minutes more.</PR>
</RECIPE>

Fig. 4. Recipe for Macaroni and Chicken Casserole

similarity between a pair of ingredient terms T_1 and T_2 is calculated using Wu & Palmer's similarity [8]:

$$\mathbf{I}_{ij} = \frac{2 * d(LCA)}{d(T_i)d(T_j)}$$

where $d(t)$ is the depth of term t from the *entity* root in WordNet and LCA is the least common ancestor of the two ingredient terms. A similarity threshold of 0.7 is applied to limit the extent of the breadth first search of WordNet.

A term x term similarity matrix for cuisines $\mathbf{C}$ is constructed in a similar fashion for the c cuisines. A vocabulary-wide similarity matrix $\mathbf{S}$ is constructed from these WordNet-populated similarity matrices $\mathbf{I}$ and $\mathbf{C}$, and the $\mathbf{1}_j$, $\mathbf{1}_m$, and $\mathbf{1}_d$ identity matrices of size j, m and d, as shown in Figure 5. The Relevance Recipe matrix $\mathbf{R}'$ is generated from the binary Recipe matrix $\mathbf{R}$ and the Similarity matrix $\mathbf{S}$ as follows. We use a *Max* operator that replaces the sum in standard matrix multiplication. Thus

$$\mathbf{R}'_{ij} = Max_{k=1}^{t} \mathbf{R}_{ik} * \mathbf{S}_{kj}$$

Whereas $\mathbf{R}$ is the incidence matrix of terms in the recipes, $\mathbf{R}'$ captures the relative relevance of terms in recipes and is much less sparse than $\mathbf{R}$. This generalisation forms the basis for fast case retrieval nets presented in [2].

4 Recipe Retrieval and Reuse

We shall focus first on queries without negation. These use the same vocabulary and binary feature vector representation as $\mathbf{R}$. However the user interface offers only a subset of the vocabulary to allow a limited but useful choice of query terms. This is not a restriction of the approach but instead simplifies the choice of the user.

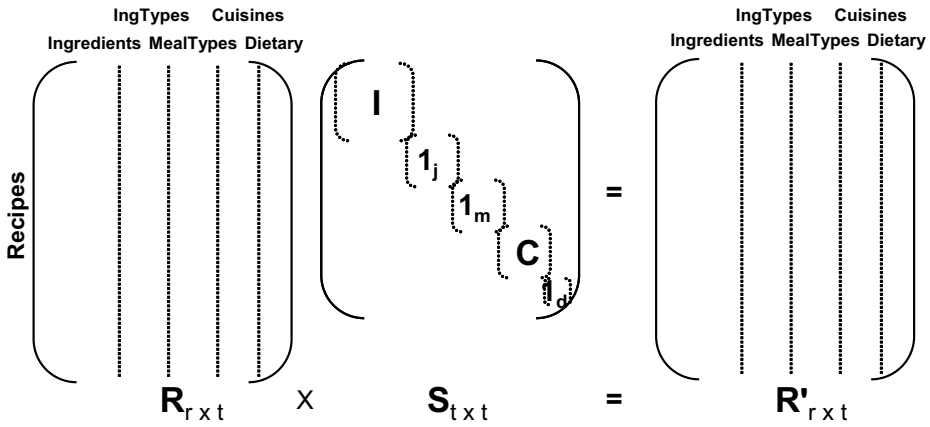


Fig. 5. Recipes as Relevance Case Base Index

4.1 Retrieval

The matrices $\mathbf{R}$ and $\mathbf{R}'$ represent the case base of recipes without and with relative relevance knowledge. Two similarities sim_R and $sim_{R'}$ are calculated between the query and the recipes represented in $\mathbf{R}$ and $\mathbf{R}'$ respectively. Each is computed using the dot product which emphasises the query terms that are matched, unlike the cosine similarity where length normalisation can adversely penalise exact matches when recipes are longer.

sim_R retrieval over $\mathbf{R}$ favours the explicit presence of query terms in the recipe. In addition, $sim_{R'}$ retrieval over $\mathbf{R}'$ can favour the presence of recipe terms that are similar to the query terms and so ensures that recipes with ingredients that are similar to those requested are considered, even if the exact ingredients are absent. We arrive at a final recipe similarity by a weighted combination of sim_R and $sim_{R'}$:

$$sim(recipe, query) = w * sim_R(recipe, query) + (1 - w) * sim_{R'}(recipe, query)$$

The recipes are ranked according to this weighted similarity value. We have found that higher values for w give best results for the CCC queries and so we use 0.8 as the weight.

4.2 Reuse

The 5 top ranked recipes are retrieved and each is considered for reuse. If the similarity indicates an exact match with the requirements in the query then the recipe is reused unchanged. Otherwise adaptation may be necessary.

Substitution adaptation is applied by finding where query mismatches occur and which recipe terms do not occur in the query. For each mismatch, the similarity matrix $\mathbf{S}$ is used to identify the non-query term in the recipe that is most

similar to each absent query term. The recipe text to be reused is rewritten with the new ingredient replacing every occurrence of the old one. The replacement is done for only ingredients and not for cuisines, ingredient types nor meal types.

4.3 Queries with Negation

These queries occur in two different ways: (1) as an ingredient that is undesirable; or (2) as a dietary practice (e.g. nut-free). For (1) the query has 1 for desired ingredients, -1 for undesirable ingredients, and 0 otherwise. The substitution adaptation described above is repeated but this time ingredients that correspond to any -1s in the query are substituted.

Negative dietary practices are handled differently. The query vector must be created to represent the practice; e.g. setting the nut ingredient entry to -1. The query can now be handled in the same way as queries of type (1). This technique is also used to incorporate ad hoc rules which improve the retrieval for meal types such as *beverage, cake, candy, dessert, ice cream* by setting the meat and fish ingredient entries to -1.

4.4 Three-Course Menu

A query for a three-course menu is handled by explicitly specifying the meal type for the first (appetizer, salad or soup) and third course (cake, candy, dessert, ice cream, snack and sweet) in addition to the ingredients. A three-step retrieval is carried out by generating three queries. Each query contains the list of desired ingredients and 2 meal types penalised. For example consider a query for *soup*, *dessert*, with desired ingredients *tomato*, *garlic*, *potato* and *orange* Here the system would negate terms *main course* and *dessert* when querying for the first course; likewise negate *soup* and *dessert* when querying for main course; and negate *soup* and *main course* when querying for *dessert*. The final system recommendation is simply a combination of the best matches from each of the three retrievals.

5 Using RaGoÛt

The software is delivered as ragout.jar. The ragout.zip[3] contains the executable jar and a readme text file. Figure 6 shows RaGoÛt's user interface. The upper section contains the requirements entry panels, and the lower section displays retrieved recipes. The user transfers ingredients from the ingredient picker to the desired or undesired lists, or may remove them from the selected lists. The user can also select from available dietary practices, cuisine types and/or meal types on the right. The results pane is displayed in two columns. The TI(tles) of the sorted list of retrieved recipes appear at the left, from which the user can select a particular recipe to display on the right.

[3] http://www.comp.rgu.ac.uk/staff/iaa/ragout.zip

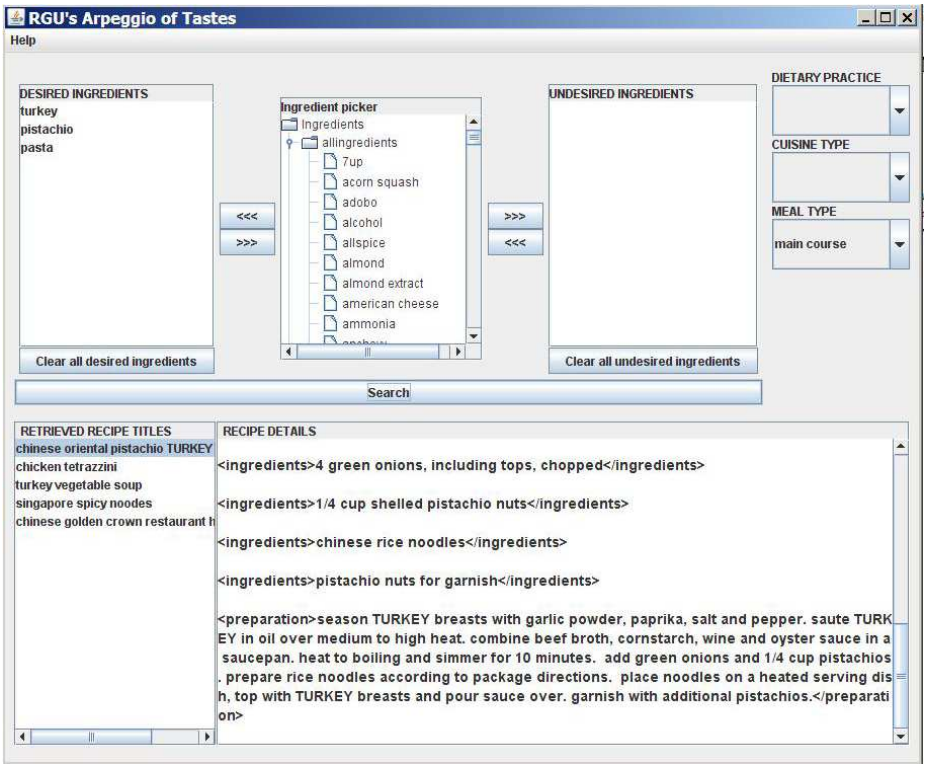


Fig. 6. RaGoÛt's User Interface

6 Initial Experiments and Conclusions

RaGoÛt addresses all outlined CCC tasks: compulsory, negation and menu challenges. Retrieval is achieved by combining similarity computations from exact matching with a WordNet-based semantic similarity matching. This combination ensures any mismatches are well placed for substitution adaptation.

On average 4 out of RaGoÛt's 5 top recommendations for each of the compulsory tasks are very relevant. In particular with query 4 (turkey, pistachio and pasta) results in the retrieval of a recipe with chicken, pistachio and rice noodles with appropriate substitutions for chicken. Here noodles are deemed similar to pasta due to their relative closeness in the WordNet hierarchy. Similarly a generalised term such as meat in query 1, resulted in the retrieval of recipes containing either turkey, beef or chicken. However some substitutions are questionable such as when cauliflower is substituted for onions in query 1. When queried for a Chinese dessert with fruits (in query 3), RaGoÛt's recommendation included a *lychee sherbet* which is a typical Chinese fruit dessert. This is possibly due to the association between lychee and Chinese cuisine.

Results from the single negation query are also very satisfactory, with the top 3 being very relevant. However the 4th recipe *cheese and macaroni* is less relevant because, although it does not contain *garlic* or *cucumber*, it is not a salad. RaGoÛt substituted *garlic* in the 5th recipe with *clove* making it relevant as it already contains *tomato* and *salad*.

RaGoÛt's approach to the three-course meal challenge is very much at an initial stage and needs further development. With this task, substitution in particular, remains difficult. We plan to address this by extending RaGoÛt's representation such that ingredients are differentiated from non-ingredients and weights are incorporated in the retrieval to allow the reuse of multiple recipes.

The cooking contest has allowed us to integrate our TCBR expertise and adaptation knowledge learning methods in the RaGoÛt system. In particular, we have proposed a similarity-aware case authoring mechanism with minimal domain-specific knowledge. This knowledge-rich representation has allowed simple retrieval and similarity-focused substitution adaptation. The weighted combination of case similarity scores enables the system to address the trade-off that exists between cases containing matching ingredients with those that have similar alternative.

References

1. Wiratunga, N., Koychev, I., Massie, S.: Feature selection and generalisation for text retrieval. In: Proceedings 7th European Conference on Case-Based Reasoning, Springer (2004) 423–437
2. Chakraborti, S., Lothian, R., Wiratunga, N., Orecchioni, A., Watt, S.: Fast case retrieval nets for textual data. In: Proceedings 8th European Conference on Case Based Reasoning, Springer (2006) 400–414
3. Craw, S., Wiratunga, N., Rowe, R.C.: Learning adaptation knowledge to improve case-based reasoning. Artificial Intelligence **170**(16-17) (2006) 1175–1192
4. Jayaran, D., Deodhare, D., B., R.: Lexical chains as document features. In: Proceedings 3rd International Joint Conference on NLP. (2008) 111–117
5. Orecchioni, A., Wiratunga, N., Massie, S., Chakraborti, S., Mukras, R.: Learning incident causes. In: Proceedings 4th Textual Case-Based Reasoning Workshop. (2007) 247–251
6. WordNet. http://wordnet.princeton.edu
7. Google Cooking Directory. http://www.google.com/Top/Home/Cooking/
8. Wu, Z., Palmer, M.: Verb semantics and lexical selection. In: 32nd Annual Meeting of the Association for Computational Linguistics. (1994) 133 –138