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

Learning to Author Text with textual CBR

Ibrahim Adeyanju[1] and **Nirmalie Wiratunga** and **Juan A. Recio-García**[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 pre-defined 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

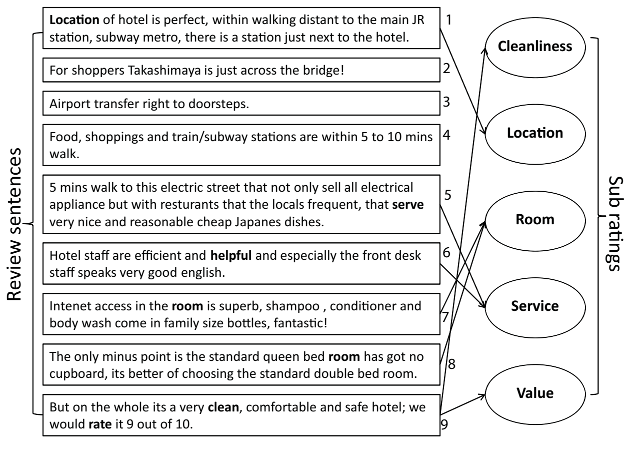

1 **Location** of hotel is perfect, within walking distant to the main JR station, subway metro, there is a station just next to the hotel.
2 For shoppers Takashimaya is just across the bridge!
3 Airport transfer right to doorsteps.
4 Food, shoppings and train/subway stations are within 5 to 10 mins walk.
5 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.
6 Hotel staff are efficient and **helpful** and especially the front desk staff speaks very good english.
7 Intenet access in the **room** is superb, shampoo , conditioner and body wash come in family size bottles, fantastic!
8 The only minus point is the standard queen bed **room** has got no cupboard, its better of choosing the standard double bed room.
9 But on the whole its a very **clean**, comfortable and safe hotel; we would **rate** it 9 out of 10.

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

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

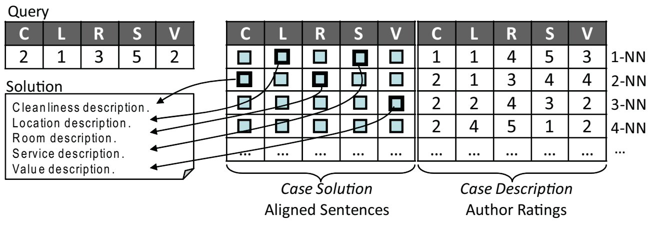


Figure 4. Transformational approach to text reuse

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
CB$_{local} = Retrieve$(CB$, Q, k$), retrieve $k$ similar cases
for each IE$_j \in Q$
    $qr = ratingType$(IE$_j$); $qv = ratingValue$(IE$_j$)
    for each C$_i \in$ CB$_{local}$, in order of decreasing similarity
        ReviewText= $getReviewText(C_i)$
        $r_j = ratingType$(IE$_j, C_i$); $v_j = ratingValue$(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.

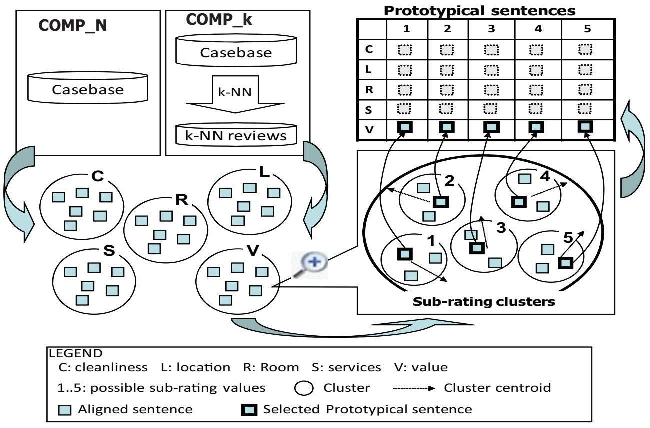


Figure 6. Clustering similar aligned sentences in review texts

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
$G = \left\{ \begin{array}{l} g_{11}, \ldots, g_{1q} \\ g_{21}, \ldots, g_{2q} \\ \cdots \\ g_{p1}, \ldots, g_{pq} \end{array} \right\}$, set of clustered similar sentences;
                                  each cluster belongs to a pair
                                  from $p$ ratings and $q$ values
    CB$_{local} = Retrieve$(CB$, Q, k$); retrieve $k$ similar cases
    Initialise $SOLN$= {},
        to contain sentences in the proposed solution text
    for each C$_i \in$ CB$_{local}$, in order of decreasing similarity
        ReviewText= $getReviewText(C_i)$
        for each IE$_j \in C_i$
            $r_j = ratingType$(IE$_j$); $v_j = ratingValue$(IE$_j$)
            $g_j = getClusteredSimilarSentences(G, r_j, v_j)$
            $S_j = selectAlignedSentences(r_j$, ReviewText)
            $addSentences(S_j, g_j)$
    for each IE$_k \in Q$
        $r_k = ratingType$(IE$_k$); $v_k = ratingValue$(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.

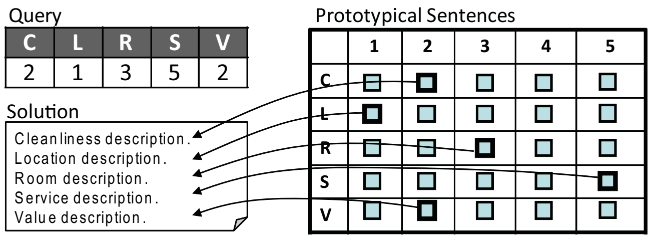


Figure 8. Compositional approach to text reuse

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

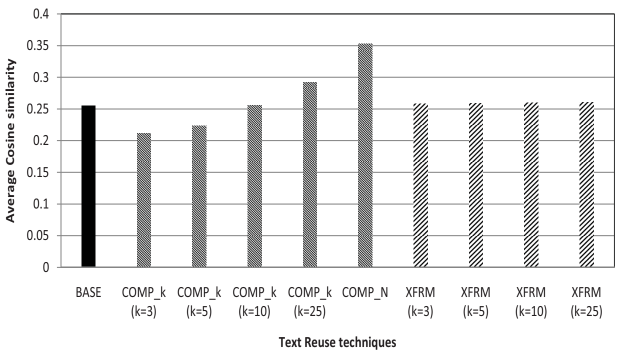
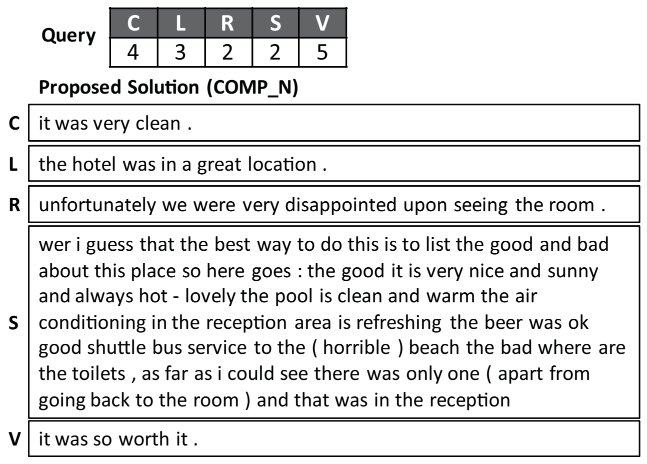


Figure 9. Graph of cosine similarity across text reuse techniques

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].

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

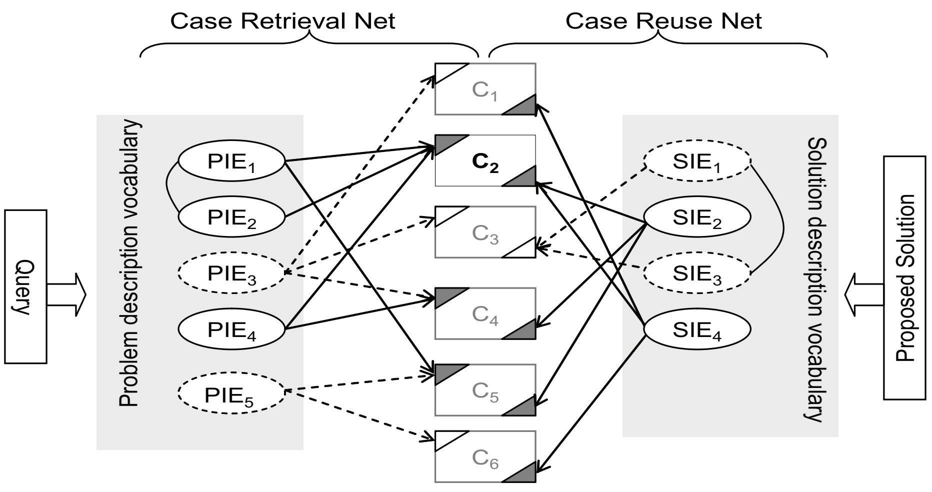


Fig. 1. The Case Retrieval Reuse Net (CR2N) architecture

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] (**BiL**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.

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

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

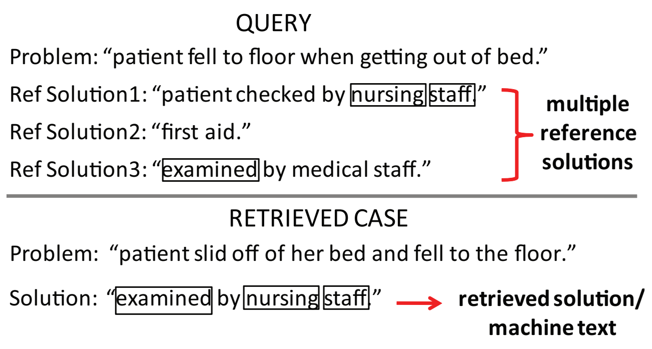
$$BLEU = BP \cdot \exp(\sum_{n=1}^{N} \frac{1}{N} \log p_n)$$

where

p_n = n-gram precision	BP = brevity penalty
l_{sys} = length of machine text	$i = 1$ 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



QUERY

Problem: "patient fell to floor when getting out of bed."

Ref Solution1: "patient checked by $\boxed{\text{nursing}}$ $\boxed{\text{staff}}$"

Ref Solution2: "first aid."

Ref Solution3: "$\boxed{\text{examined}}$ by medical staff."

multiple reference solutions

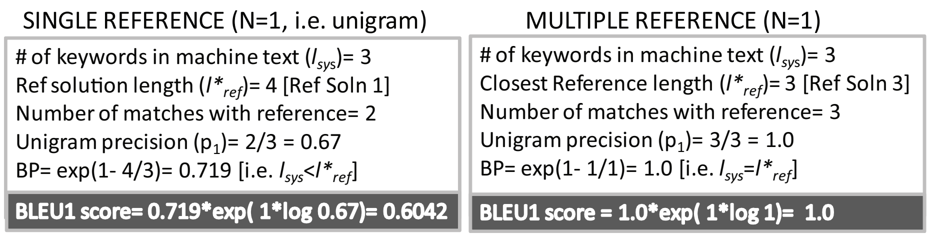
RETRIEVED CASE

Problem: "patient slid off of her bed and fell to the floor."

Solution: "$\boxed{\text{examined}}$ by $\boxed{\text{nursing}}$ $\boxed{\text{staff}}$"

retrieved solution/ machine text

Fig. 2. A test case with multiple reference solutions



SINGLE REFERENCE (N=1, i.e. unigram)	MULTIPLE REFERENCE (N=1)
# of keywords in machine text (l_{sys})= 3	# of keywords in machine text (l_{sys})= 3
Ref solution length (l^*_{ref})= 4 [Ref Soln 1]	Closest Reference length (l^*_{ref})= 3 [Ref Soln 3]
Number of matches with reference= 2	Number of matches with reference= 3
Unigram precision (p_1)= 2/3 = 0.67	Unigram precision (p_1)= 3/3 = 1.0
BP= exp(1- 4/3)= 0.719 [i.e. $l_{sys}<l^*_{ref}$]	BP= exp(1- 1/1)= 1.0 [i.e. $l_{sys}=l^*_{ref}$]
BLEU1 score= 0.719*exp(1*log 0.67)= 0.6042	**BLEU1 score = 1.0*exp(1*log 1)= 1.0**

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.

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

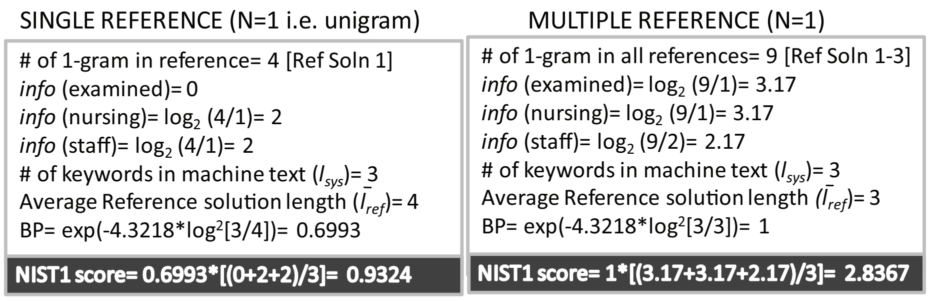
$$\begin{aligned} \text{info}(n\text{-gram}) &= \text{info}(w_1 \ldots w_n) \\ &= \log_2 \left(\frac{\#\text{ of } w_1 \ldots w_{n-1} \text{ in reference solutions}}{\#\text{ of } w_1 \ldots w_n \text{ in reference solutions}} \right) \end{aligned}$$

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

where

w= a word in the machine text
info= information weight
N = maximum size of n-gram (i.e. $n = 1 \ldots N$)
$\beta = -4.3218$, 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

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.


SINGLE REFERENCE (N=1 i.e. unigram)

# of 1-gram in reference= 4 [Ref Soln 1]
info (examined)= 0
info (nursing)= $\log_2$ (4/1)= 2
info (staff)= $\log_2$ (4/1)= 2
# of keywords in machine text (l_{sys})= 3
Average Reference solution length ($\bar{l}_{ref}$)= 4
BP= exp(-4.3218*$\log^2$[3/4])= 0.6993

NIST1 score= 0.6993*[(0+2+2)/3]= 0.9324

MULTIPLE REFERENCE (N=1)

# of 1-gram in all references= 9 [Ref Soln 1-3]
info (examined)= $\log_2$ (9/1)= 3.17
info (nursing)= $\log_2$ (9/1)= 3.17
info (staff)= $\log_2$ (9/2)= 2.17
# of keywords in machine text (l_{sys})= 3
Average Reference solution length ($\bar{l}_{ref}$)= 3
BP= exp(-4.3218*$\log^2$[3/3])= 1

NIST1 score= 1*[(3.17+3.17+2.17)/3]= 2.8367

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%

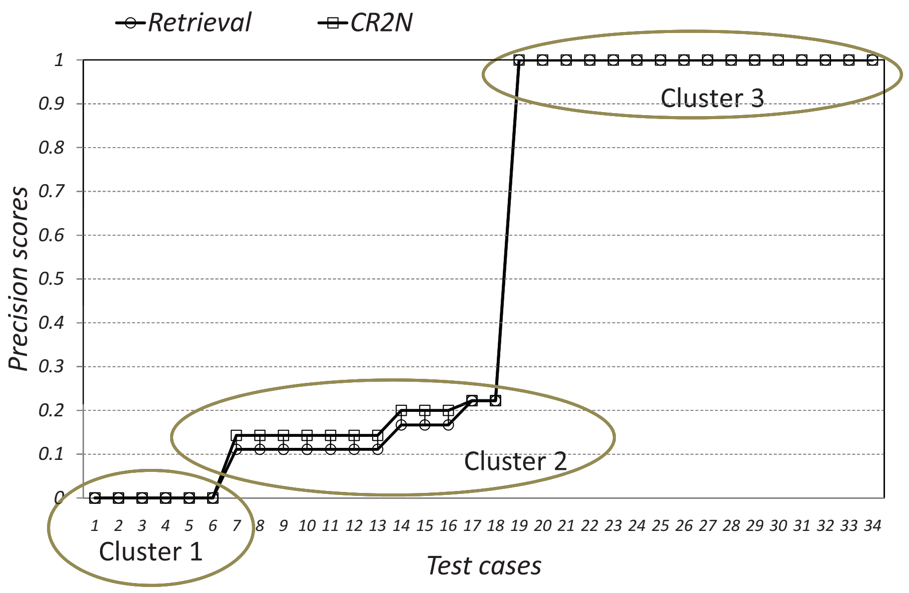


Fig. 5. Graph of precision results for the 34 test cases with multiple references

($p = 0.045 < 0.05$). 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 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.

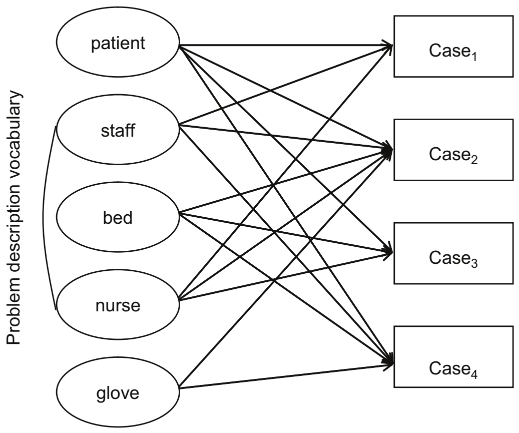
The drawbacks observed in the work reviewed above are addressed by a text reuse technique called *Case Grouping* (*CG*) [6]. The technique demonstrated on a semi-automated email response application involves reuse of previous email messages to synthesize new responses to incoming requests. A response 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.

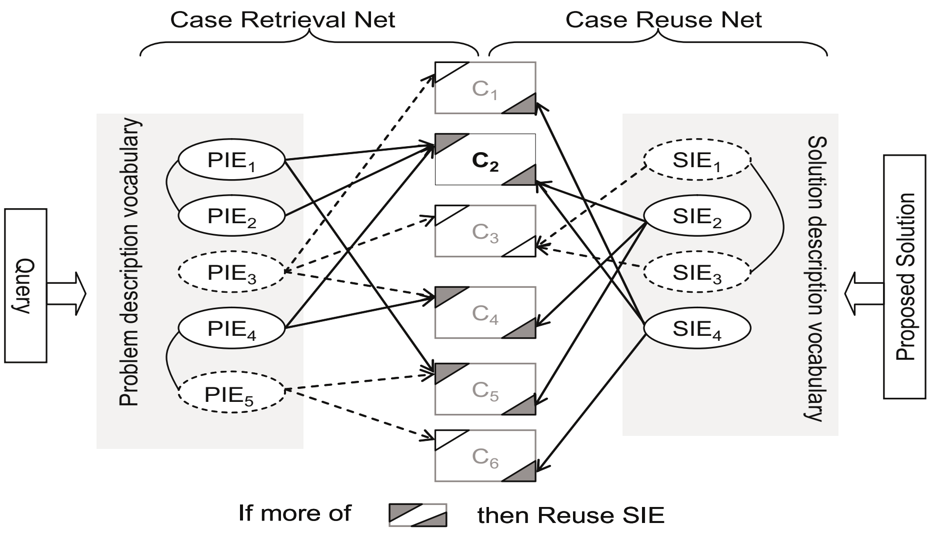


Fig. 2. The Case Retrieval Reuse Net (CR2N) architecture

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.

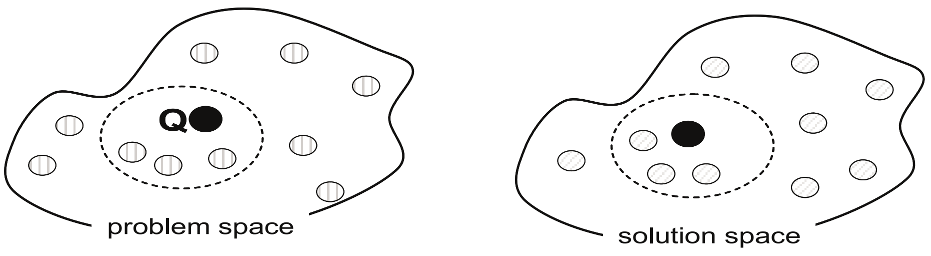


Fig. 3. Neighbourhoods in the problem and solution spaces

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.

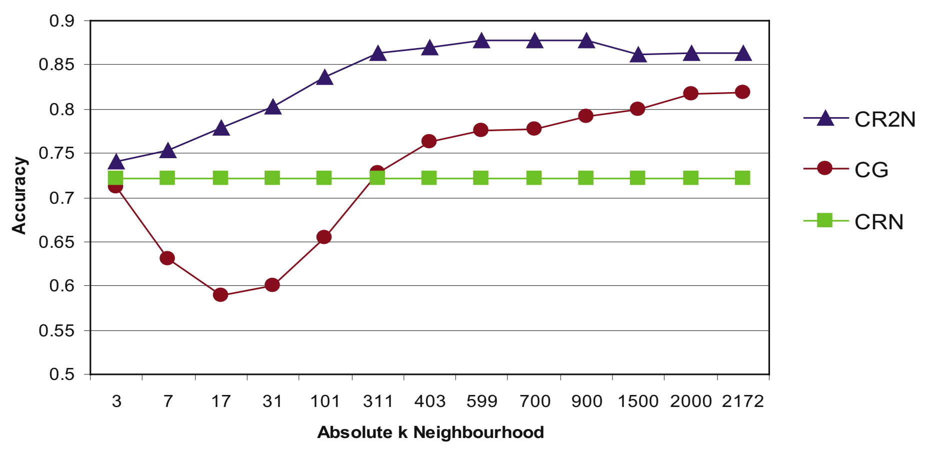


Fig. 5. Accuracy results for CR2N, CG and CRN in weather forecast revision

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

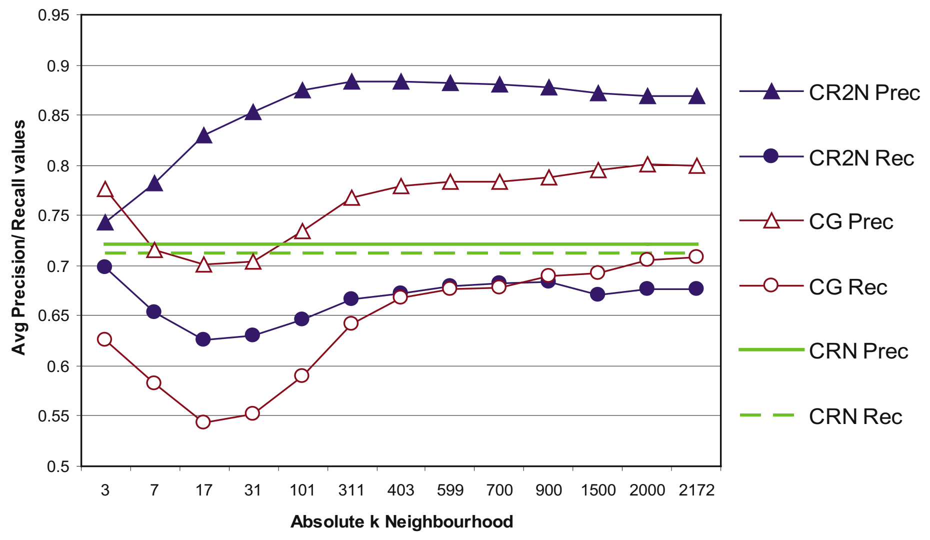


Fig. 6. Precision/Recall results for CR2N, CG and CRN in weather forecast revision

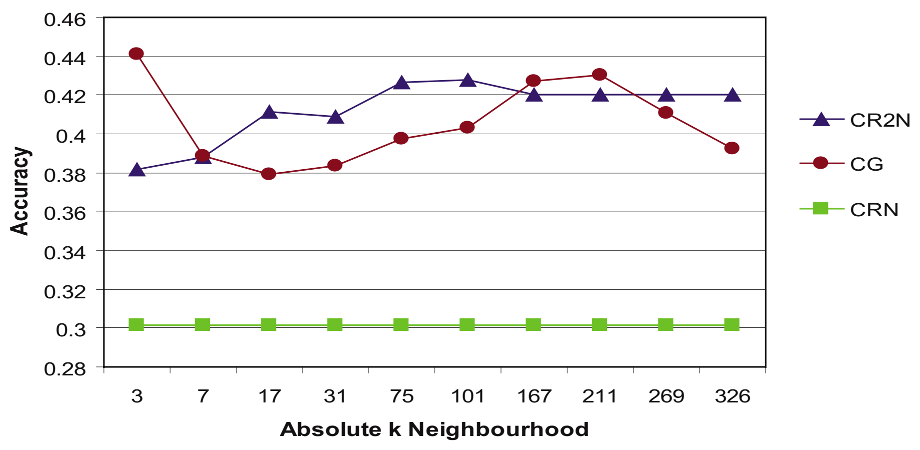


Fig. 7. Accuracy results for CR2N, CG and CRN in H&S incident reporting

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

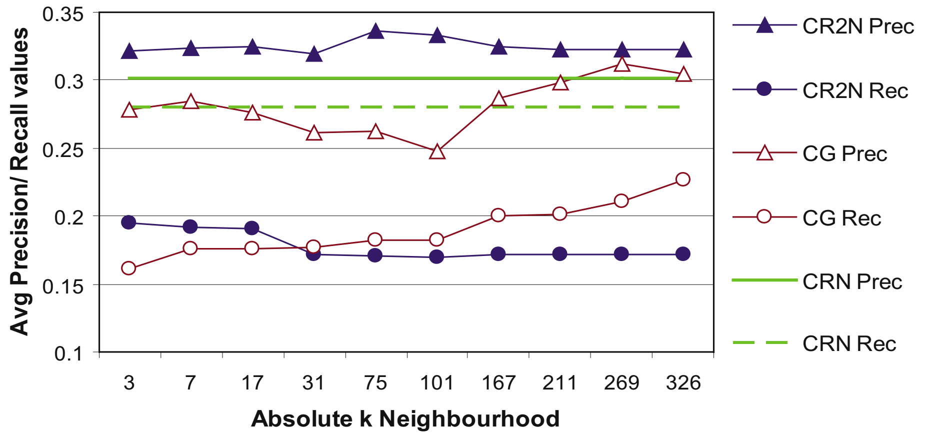


Fig. 8. Precision/Recall results for CR2N, CG and CRN in H&S incident reporting

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
All rights reserved

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.

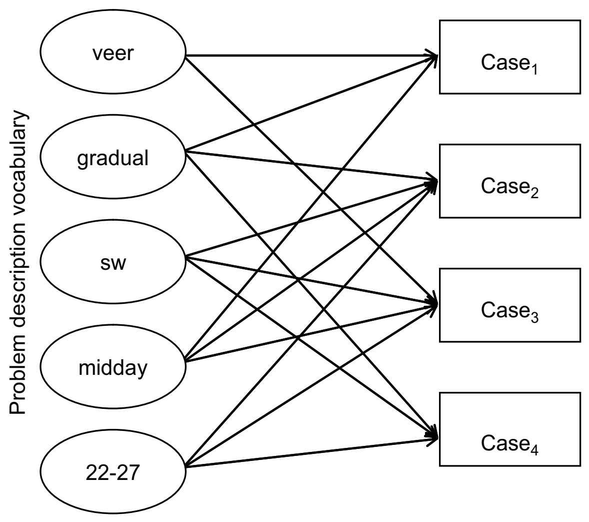
Technical details of the CRN and how the CR2N extends it for textual reuse are discussed in subsequent sections.

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.

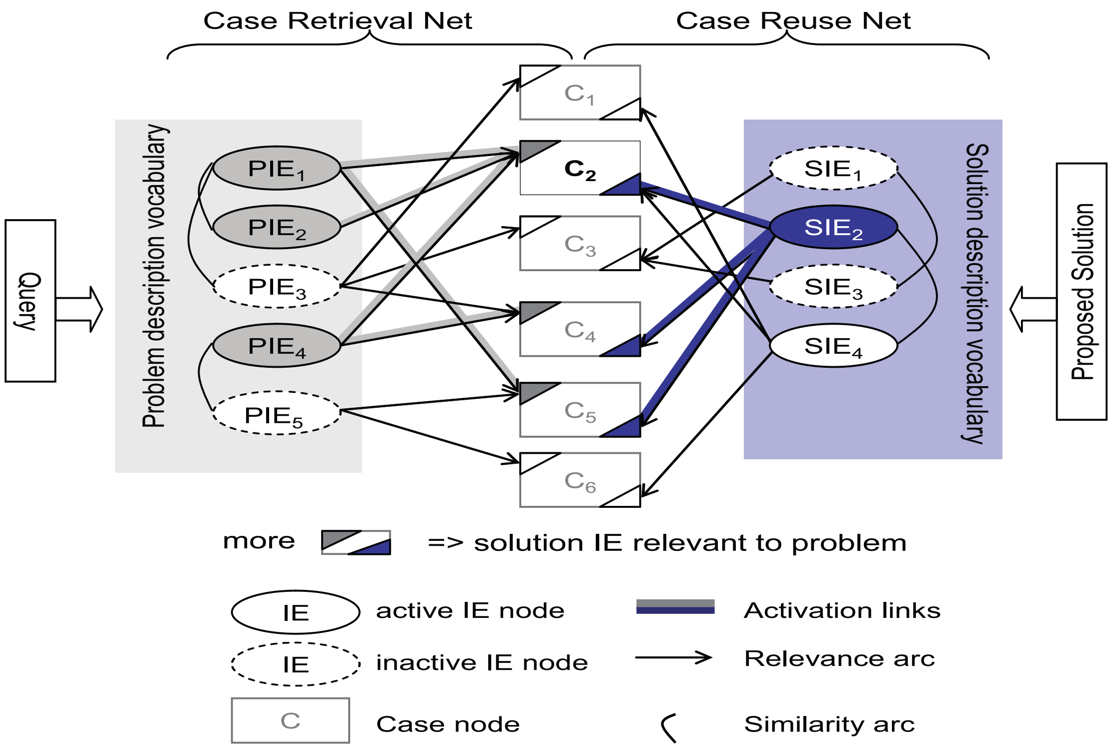


Fig. 2. The CR2N architecture

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($V_s$, $\{sie_i\}), \sigma$)
        AS= $RS_1 \cap RS_2$
        BS= $RS_1 \setminus RS_2$
        $\overline{S}_A$= $\frac{1}{|AS|} \sum_{a \in AS} Sim(a, Q)$
        $\overline{S}_B$= $\frac{1}{|BS|} \sum_{b \in BS} Sim(b, Q)$
        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.

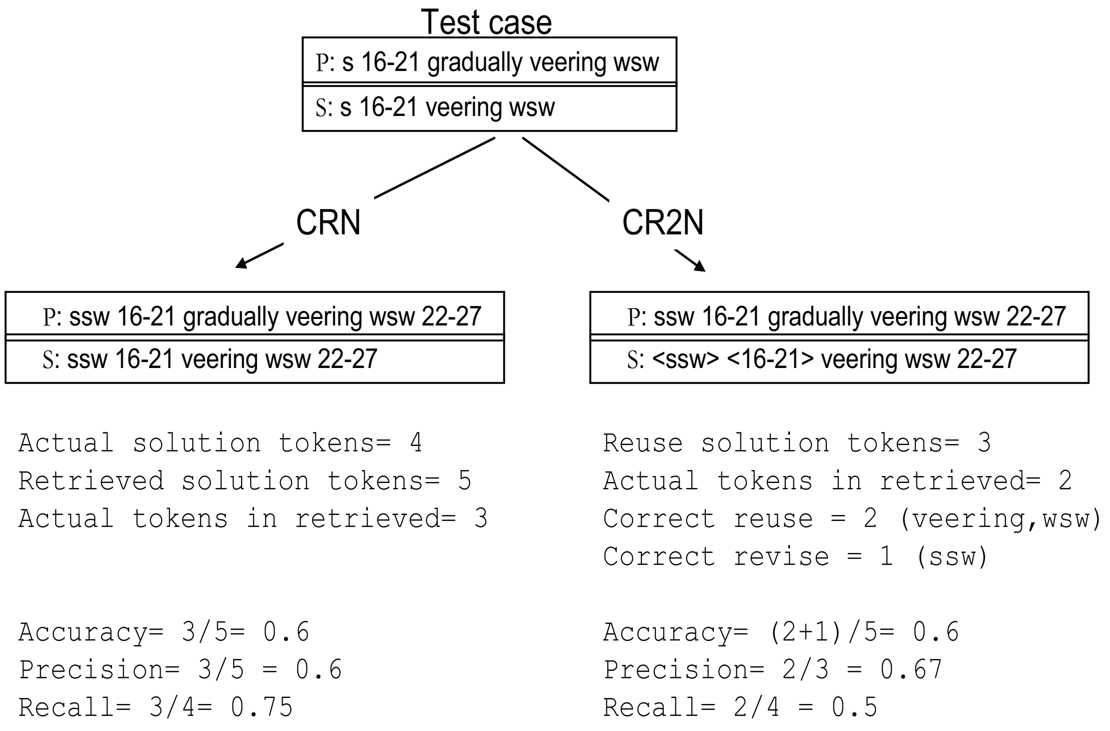

Test case
P: s 16-21 gradually veering wsw
S: s 16-21 veering wsw
CRN
CR2N
P: ssw 16-21 gradually veering wsw 22-27
S: ssw 16-21 veering wsw 22-27
P: ssw 16-21 gradually veering wsw 22-27
S: <ssw> <16-21> veering wsw 22-27
Actual solution tokens= 4
Retrieved solution tokens= 5
Actual tokens in retrieved= 3
Accuracy= 3/5= 0.6
Precision= 3/5 = 0.6
Recall= 3/4= 0.75
Reuse solution tokens= 3
Actual tokens in retrieved= 2
Correct reuse = 2 (veering,wsw)
Correct revise = 1 (ssw)
Accuracy= (2+1)/5= 0.6
Precision= 2/3 = 0.67
Recall= 2/4 = 0.5

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.

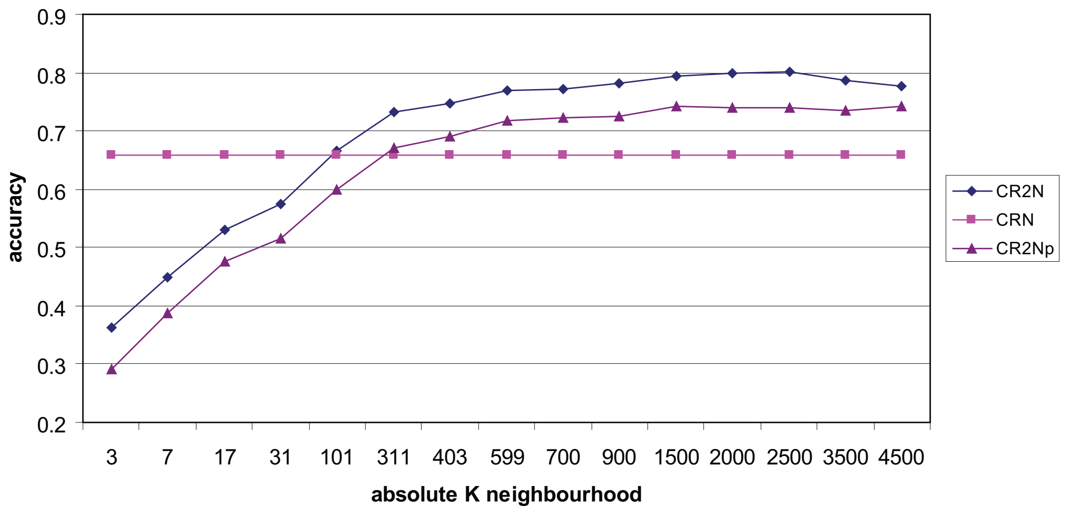


Fig. 5. Accuracy graph for the retrieved solution, CR2N & CR2N$_p$ at different k-neighbourhoods

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.

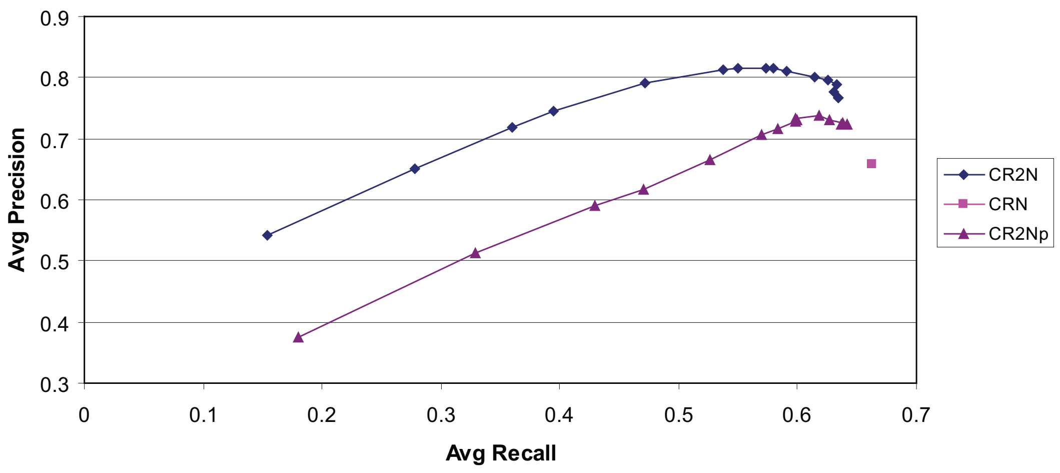


Fig. 6. Precision-Recall curve for the retrieved solution, CR2N & CR2N$_p$ at different k-neighbourhoods

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

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 × term matrix **R** where each of the r rows corresponds to the feature vector for that recipe,

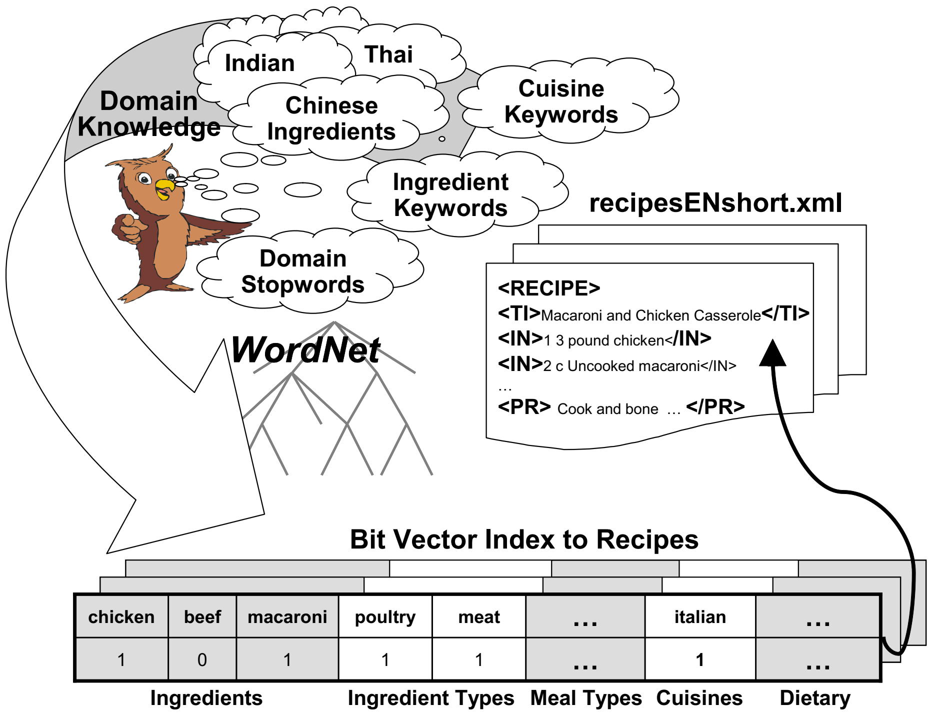


Fig. 3. RaGoÛt Case Acquisition

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.

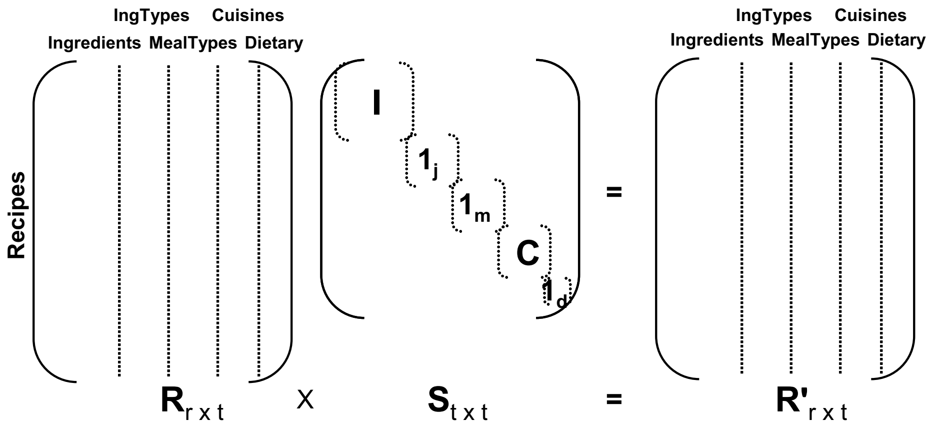


Fig. 5. Recipes as Relevance Case Base Index

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.

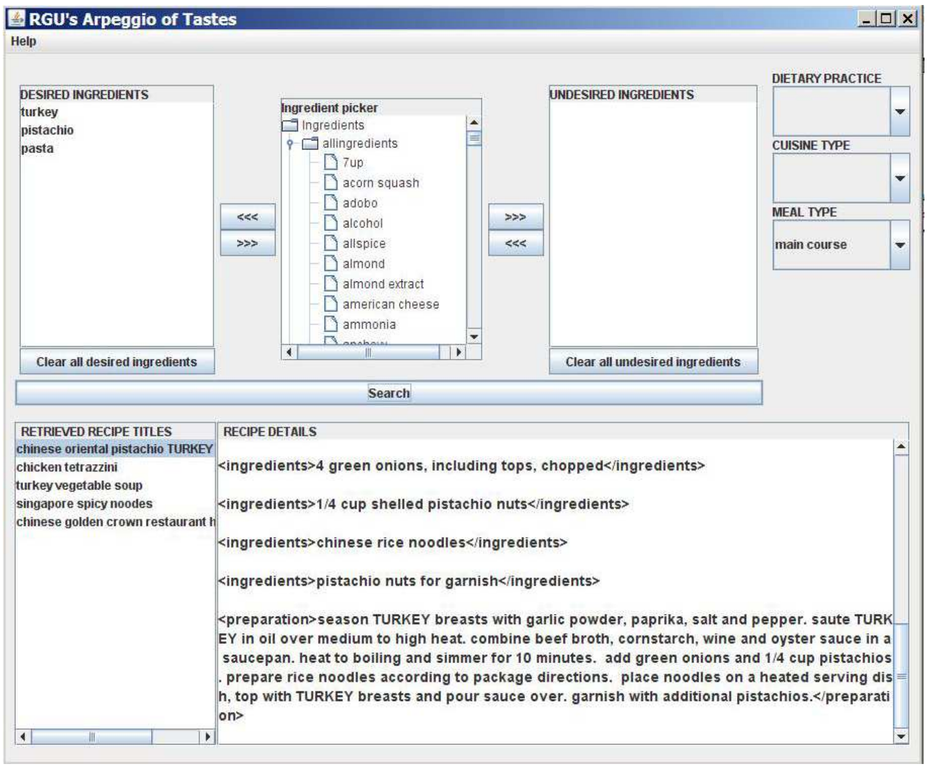
4.4 Three-Course Menu

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



DESIRED INGREDIENTS
turkey
pistachio
pasta

UNDESIRED INGREDIENTS

DIETARY PRACTICE

CUISINE TYPE

MEAL TYPE
main course

RETRIEVED RECIPE TITLES
chinese oriental pistachio TURKEY
chicken tetrazzini
turkey vegetable soup
singapore spicy noodes
chinese golden crown restaurant h

RECIPE DETAILS

```
<ingredients>4 green onions, including tops, chopped</ingredients>

<ingredients>1/4 cup shelled pistachio nuts</ingredients>

<ingredients>chinese rice noodles</ingredients>

<ingredients>pistachio nuts for garnish</ingredients>

<preparation>season TURKEY breasts with garlic powder, paprika, salt and pepper. saute TURK
EY in oil over medium to high heat. combine beef broth, cornstarch, wine and oyster sauce in a
 saucepan. heat to boiling and simmer for 10 minutes.  add green onions and 1/4 cup pistachios
. prepare rice noodles according to package directions.  place noodles on a heated serving dis
h, top with TURKEY breasts and pour sauce over. garnish with additional pistachios.</preparati
on>
```

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