Solution reuse for textual cases

Ibrahim Adeyanju[1], Nirmalie Wiratunga[1], Robert Lothian[1], Somayajulu Sripada[2], and Susan Craw[1]

[1] School of Computing
The Robert Gordon University
Aberdeen AB25 1HG, Scotland, UK
[iaa|nw|rml|smc]@comp.rgu.ac.uk

[2] Department of Computing Science,
University of Aberdeen,
Aberdeen, AB24 3UE, Scotland, UK
yaji.sripada@abdn.ac.uk

Abstract. The reuse stage in textual CBR identifies reusable textual constructs in solution content. This involves content annotation so that reusable solution text is easily identifiable from the rest. We present a generic architecture, Case Retrieval Reuse Net (CR2N), that can be used to annotate text to denote text content as reusable or not. Initial results from a weather forecast revision dataset shows up to 80% accuracy and a significantly better precision over a retrieve-only system. Although our work with CR2N is still on going, it also provides useful insight into the text revision stage.

Keyword Textual CBR, Case Reuse, Case Retrieval Net, NLG

1 Introduction

Textual Case Based Reasoning (TCBR) solves new problems by reusing previous similar problem-solving experiences documented as text. TCBR is a subfield of Case Based Reasoning (CBR) but has evolved as a specialized research area due to challenges associated with reasoning with textual attributes [1] as opposed to structured attributes consisting of numeric and symbolic values.

The reuse of retrieved similar case(s) precedes revision in TCBR's problem-solving cycle. Reuse in structured CBR typically involves using the entire solution part of a retrieved similar case since the actual solution to a new problem is similar in both content and in the number of attributes. This is not always applicable to TCBR when the solution is textual and its decomposition into sections (tokens, phrases or sentences) is viewed as attributes. The number of sections in a retrieved textual solution could differ from the actual solution; therefore, the reuse stage for TCBR requires that sections of a solution text relevant to a given problem are identified. This then would help to identify *what to revise* at the revision stage.

Natural Language Generation (NLG) is a complimentary field of research concerned with the construction of understandable texts in English (or other human languages) from some underlying non-linguistic representation of information [2]. NLG systems are usually knowledge intensive whereby text production relies on both grammatical and manually acquired rules from experts. Post-editing is a common feature of existing NLG systems (e.g. SUM-TIME METEO [3]) and involves manual revision of generated text by domain experts before presentation to final users. TCBR can be used to automate post-editing of NLG systems by reusing past editing experiences of domain experts. In this scenario, the NLG system generated text is captured as a problem experience while the edited text (by domain experts) forms the solution to that problem. We create a case base of such experiences and use it on new NLG system text by generating annotations relevant for the post-editing task.

We first present the CR2N architecture for text reuse in Section 2 while experimental setup, evaluation and discussion of results appear in Section 3. Related work and theoretical background are reviewed in Section 4, followed by conclusion and future directions in Section 5.

2 Case Retrieval Reuse Net (CR2N) for Textual reuse

Our approach to reuse involves automated annotation of retrieved solution text as relevant or not. Essentially textual units (tokens, phrases, sentences etc) annotated as relevant suggests that they can be reused without revision. In order to achieve this, we propose an extension to the CRN architecture called CR2N. The CR2N architecture consists of two CRNs: the original Case Retrieval Net (CRN) [4] which indexes the problem vocabulary and a second CRN referred to as Case Reuse Net (CReuseNet) indexes the solution vocabulary.

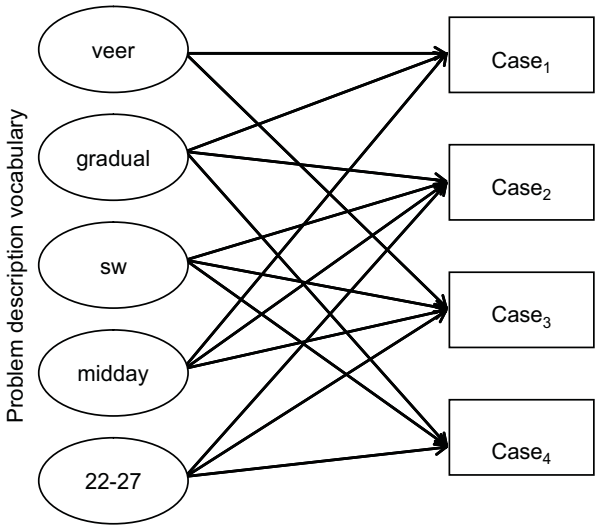
Technical details of the CRN and how the CR2N extends it for textual reuse are discussed in subsequent sections.

2.1 Case Retrieval Net(CRN)

A CRN is a memory model that can efficiently retrieve a relatively small number of relevant cases from a case base. The model in its basic form was proposed by Lenz & Burkhard [4] although several extensions to the basic CRN such as the lazy propagation CRN [5], Microfeature CRN [6] and Fast CRN [7] have been proposed. The CRN is *efficient* because it avoids exhaustive memory search and can handle partially specified queries; *complete* because it assures that every similar case in memory is found during retrieval; and *flexible* as there are no inherent restrictions concerning the circumstances under which a particular piece of knowledge can be recalled [4].

The CRN uses a net-like case memory to apply a spreading activation process for retrieval of similar cases to a query. The basic CRN consists of four components: *case nodes*, *Information Entities nodes* (IEs), *relevance arcs* and *similarity arcs* as illustrated in figure 2. An IE consists of a particular attribute-value pair and a case therefore consists of a set of IEs. A relevance arc shows the presence and strength of an IE in a case while a similarity arc indicates how similar an IE is to another. The CRN for a particular case base can be seen as a directional graph network with cases and IEs represented as nodes and the relevance arcs connecting IE nodes to their respective case nodes and similarity arcs connecting IE nodes. A case retrieval is performed by activating IEs nodes which occur in a given query, propagating this activation according to similarity through the nets of IE and aggregating activation in the associated case nodes[4]. Cases are ranked according to this aggregation and solution from the top k cases are retrieved.

When used in TCBR, each information entity (IE) node is used to represent a single textual unit (token/keyword, phrase or sentence) depending on the granularity of indexing and similarity matching. Similarity between the textual units are then captured by the similarity arcs. A CRN



	veer	gradual	sw	midday	22-27
$Case_1$	1	1	0	1	0
$Case_2$	0	1	2	1	1
$Case_3$	1	0	1	1	2
$Case_4$	0	1	2	0	1

Fig. 1. Part of the CRN for a Wind direction forecast revision dataset with matrix representation

built for the post-edit weather application is illustrated in figure 1 with its corresponding matrix

representation. A relevance arc connects an IE to a case when the token associated with the IE is contained in the case. For example the tokens "gradual", "sw", and "22-27" occur in case $Case_4$. The weight on the arc typically denotes the importance of the token in a case. Here, we use term frequency weighting and each row in the matrix relates to a case represented as a feature vector. Aggregation of activations through the network are implemented using matrix multiplication. The similarity arcs are not shown in the figure because they were not used in our experiments although they could help generalise the matrix thereby reducing any sparseness when used [7].

2.2 From CRN to CR2N

Figure 2 illustrates the components of CR2N. The Case Retrieval Net (CRN) retrieves the most similar case(s) to a query while Case Reuse Net (CReuseNet) enables text annotation on the proposed solution. CRN represents the problem vocabulary of indexed cases as a mapping between IE nodes and cases containing such IEs. Case nodes are denoted as C and the problem description IEs are denoted as PIE. Mapping of IEs onto cases are shown as relevant arcs while the similarity arcs indicate the similarity between IEs. Solution description IEs in the CReuseNet are denoted as SIE and are differentiated from the problem description IEs.

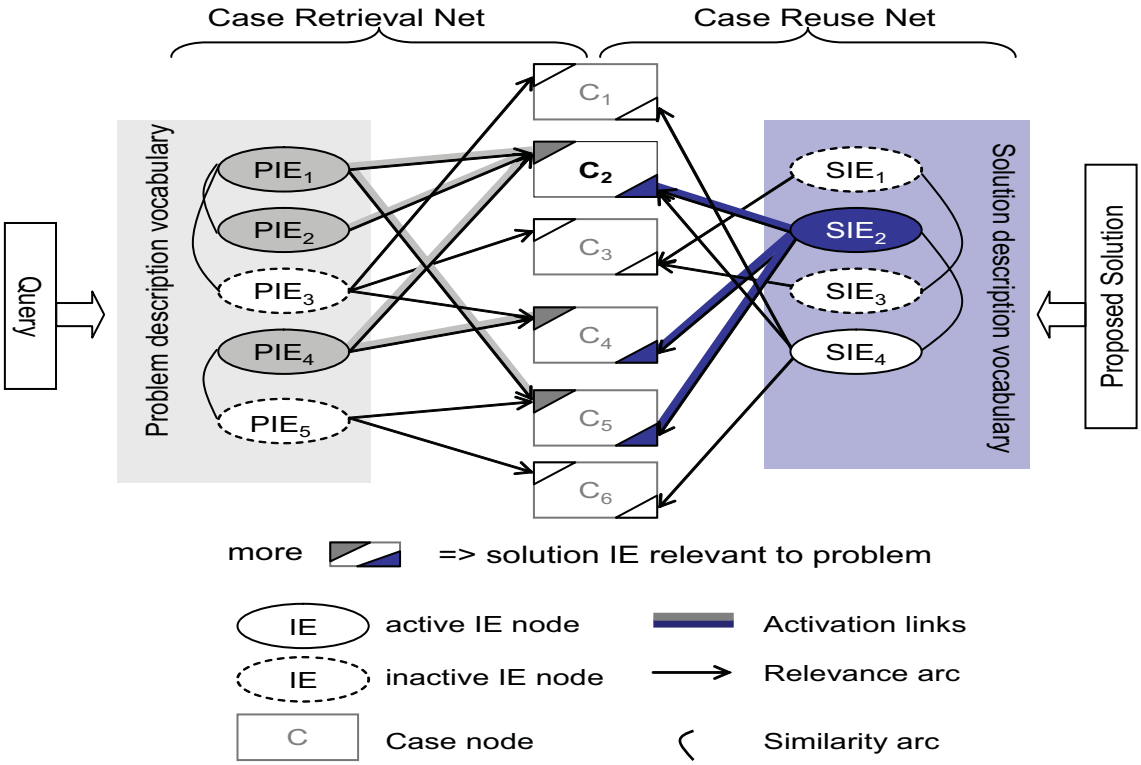


Fig. 2. The CR2N architecture

A query spreads activation in the CRN through its PIEs. The most similar case is identified as that having the highest aggregation of activations(C_2 in figure 2). Each SIE (or group of SIEs) in the most similar case then spreads activation in the CReuseNet one at a time to determine its relevance to the query. An SIE is relevant if a majority of the case nodes it activates were also activated in the CRN. For example in figure 2, C_2 (most similar to the query) contains SIE_2 & SIE_4. SIE_2 is determined to be relevant because all the cases (C_2, C_4 & C_5) activated by the query in the CRN are also activated by the SIE_2 node. SIE_4 is less relevant because it only activates one (C_2) out of the three cases activated by the query in the CRN.

2.3 Text Annotation with CR2N

Text reuse in TCBR involves the separation of reusable textual solution content from that which needs revised. This separation is suitably presented to the user as annotated text. Lamontange's reuse strategy [8] applied to email response generation annotates sentences from the proposed solution as either relevant or not. Here, a relevant annotation suggests that the sentence is directly applicable to the current problem's context and otherwise requires revision before it is applicable. We employ a similar annotation scheme but use the CR2N algorithm (see Figure 3) to automatically generate the annotations.

CB= $\{C_1,\ldots,C_n\}$, set of cases in the case base
V_p= $\{pie_1,\ldots,pie_m\}$, set of problem IEs in CB
V_s= $\{sie_1,\ldots,sie_l\}$, set of solution IEs in CB
C= $\{P, S\}, where(C \in CB) \wedge (P \subset V_p) \wedge (S \subset V_s)$
Q= a query, where Q $\subset V_p$
k= local neighbourhood used for relevance calculation, where k<= n

C_{best}= SelectK(CRN(V_p, Q),1)
RS_1= SelectK(CRN(V_s, C_{best}), k)
for each $\{sie_i\} \in C_{best}$
  RS_2= SelectT(CRN(V_s, $\{sie_i\}$), σ)
  AS= $RS_1 \cap RS_2$
  BS= $RS_1 \backslash RS_2$
  $\overline{S}_A = \frac{1}{|AS|} \sum_{a \in AS} Sim(a, Q)$
  $\overline{S}_B = \frac{1}{|BS|} \sum_{b \in BS} Sim(b, Q)$
  if $\overline{S}_A > \overline{S}_B$
  then
    REUSE $\{sie_i\}$ (relevant to the query)
  else
    REVISE $\{sie_i\}$ (irrelevant to query)

Fig. 3. The CR2N Algorithm

CR2N uses a generic CRN function to retrieve cases given a partial case description and an indexing vocabulary. There are two CRN function calls, with the first retrieving over the problem vocabulary, V_p, and the second over the solution vocabulary, V_s. The retrieval sets returned by the CRNs are qualified by two further Select functions: SelectK returns the top k cases, and SelectT returns all cases with similarity above a specified threshold.

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

A solution IE is relevant to the query if cases containing it are similar to the query. In other words we want to establish if cases with similar problem descriptions to the query also contain the solution IE of interest, $\{sie_i\}$. For this purpose the retrieval sets RS_1 and RS_2 are compared. The intersection of these sets contain cases (AS) that have similar solution to the retrieved solution and also contain the sie_i, whilst the set difference identifies cases (BS) that are similar to the retrieved solution but not containing $\{sie_i\}$. The annotation is conditioned on the average similarity of the query to cases in the intersection versus that of the set differences.

The SelectK(CRN(V_s, C_{best}), k) function retrieves k-cases similar to the retrieved solution. The function thereby allows the retrieved solution's overall context to be taken into account even when IEs are used for activation one at a time. The use of a specified k-neighbourhood increases the efficiency of the algorithm since a smaller number of cases are used for relevance computation. Small values of k would ensure that a local neighbourhood is used for relevance computation and removes the influence of cases with little similarity to the retrieved. This is important since cases with little similarity to the retrieved case could negatively affect the relevance computation because they reduce average similarity of AS.

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

3 Evaluation

We evaluate the effectiveness of CR2N by measuring the accuracy of its annotations. We chose a retrieve-only system as our baseline since reuse succeeds retrieval and its use can only be justified if it improves the retrieval results. We are also interested in the effect of different neighbourhood sizes (k) on CR2N performance, we therefore repeated our experiments for increasing values of k. We compared the baseline with two textual reuse algorithms.

1. CR2N as explained in section 2.3.
2. CR2N$_p$, a variation of CR2N by replacing SelectK(CRN(V_s, C_{best}),k) with SelectK(CRN(V_p, Q), k) in figure 3. CR2N$_p$ uses k-neighbours of the query and allows us to measure the effect of ignoring the context of the retrieved solution during relevance computation.

3.1 Dataset Preparation

The evaluation uses the wind dataset extracted from a post-edit corpus [3] of an NLG weather forecast system called SUMTIME-MOUSAM (SM). The dataset consists of weather forecast text generated from numerical data by SM and its edited form after revision by domain experts. A case in our experiments therefore consists of the NLG system generated text (Unedited Text) as problem and its revised form by domain experts (Edited text) as solution.

The SM weather corpus has the following peculiar properties:

- The problem text is more similar to its solution text in a single case than to any problem text from other cases. This means that the problem & solution vocabularies are identical unless forecasters introduce new terminology. Although this is unlike most TCBR applications where the problem & solution have very few vocabulary in common (e.g. ESA incident report dataset [9], NHS dataset [10]), we expect that similar edit operations are applicable on generated texts that are similar.
- The indexing vocabulary is very small (e.g. about 75 tokens for the wind dataset from all problem text without stemming or removal of stop words).
- The problem (Unedited text) is very consistent because it is generated by an NLG system with abstracted rules but the solution(Edited text) is not as consistent and may contain typing errors (e.g. middnight, acking, deceasing, lessbecoming).

The extracted wind forecast dataset initially contained 14,690 cases with duplicates. A total of 5011 cases were left for experiments after removing duplicate cases. The textual attributes (unedited/edited text) of cases are preprocessed using the GATE library, available as part of the jCOLIBRI [11] framework. These attributes are organised into paragraphs, sentences and tokens using the GATE Splitter. The only stop words removed are punctuation marks because the text contains normal stops as either a wind direction in the short form (e.g *s* - south) or common adverbs (e.g. gradually) which are used to indicate the trend from a wind period to another. All tokens are then stemmed to cater for morphological variations (e.g. gusts/gusting).

3.2 Methodology

We use ten-fold cross validation with k-Nearest Neighbour (k-NN) in our experiments. Cosine was used for similarity computation at both retrieval and reuse stages of the architecture and the experiment was repeated for increasing values of k at the reuse stage. Each IE in the CR2N represents a token from our domain vocabulary. We chose tokens as our textual units to be annotated because the size of each retrieved solution text in our application domain is small (typically 1 sentence).

We evaluate effectiveness of the CR2N using average precision, recall and accuracy. Our underlying hypothesis is that an effective reuse of retrieved similar cases would enhance revision and should perform better than the retrieve-only baseline. Precision is measured as a ratio of the number of tokens from the actual solution present in the proposed solution to all tokens in proposed solution. Recall is a ratio of the number of tokens from the actual solution present in the proposed solution to all tokens in actual solution. These measures (borrowed from information retrieval) are commonly used to evaluate TCBR systems [12]. We also measured accuracy of the CR2N annotation as a ratio of retrieved tokens correctly annotated as reuse/revise to the total number of tokens retrieved. The retrieval precision is used as baseline accuracy since all retrieved tokens are deemed reusable if no annotation is done. Figure 4 gives snippets from our dataset to illustrate our precision, recall and accuracy calculation.

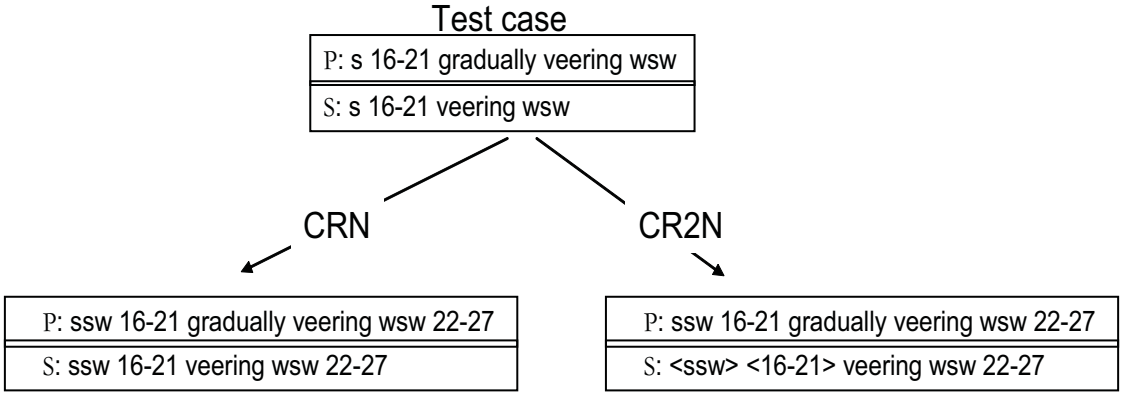


Fig. 4. Accuracy, precision & recall calculation on weather test case

Accuracy shows predictive performance of the CR2N while precision/ recall indicates its overall performance if tokens annotated as revise are ignored. A higher precision with comparable recall for the CR2N over a retrieve-only system would indicate better effectiveness.

3.3 Results

Figure 5 shows an accuracy graph comparing the retrieved similar solution (CRN), CR2N and $CR2N_p$. The accuracies of the CR2N and $CR2N_p$ increase as the neighbourhood of the query or retrieved solution is being expanded with the k parameter and outperform the baseline (precision

of the retrieved solution) when k=311. This increase in accuracy becomes marginal after k=1500 (about one-third of 4510 cases in the training set) and starts to decrease after k=2500. This increase in accuracy with increasing k can be attributed to the CR2N (or CR2N$_p$) having more contextual knowledge to predict the relevance/irrelevance of a token better. The marginal increase after, k=1500, establishes the fact that comparison of local neighbourhoods is sufficient rather than the entire case base. The efficiency of the algorithm is also improved if a fraction (k) of the case base (rather than all cases) is employed for reuse computation. CR2N also performs better than CR2N$_p$ which uses the query ranking. This shows the importance of using the context of the retrieved solution when determining relevance of a single token.

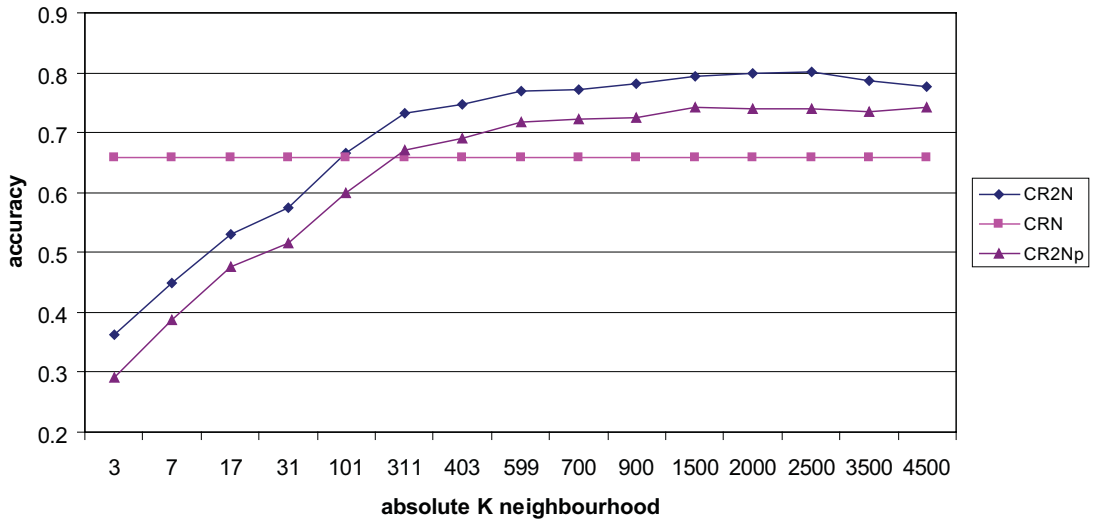


Fig. 5. Accuracy graph for the retrieved solution, CR2N & CR2N$_p$ at different k-neighbourhoods

The precision-recall curve in figure 6 shows a similar pattern in effectiveness. The average recall of the CR2N (0.6138) becomes significantly comparable to the average retrieval recall (0.6624) when k=1500 but with a much higher precision. The recall of CR2N (or CR2Np) cannot be greater than the retrieval recall as tokens can only be deleted and not inserted. The precision-recall curve of CR2N is also above that of CR2N$_p$ on the graph. This also emphasizes the significance of using the retrieved solution's context.

4 Related Work

Gervás etal [13] exploited a relationship between NLG & CBR for automatic story generation. They use CBR to obtain a plot structure by reusing stories from a case base of tales and an ontology of explicitly declared relevant knowledge. NLG is then used to describe the story plot in natural language. Although the story generated is a sketch of a plot, it assists screen writers in fast prototyping of story plots which can easily be developed into a story. The CBR approach employed is knowledge intensive and use of a domain specific ontology limits its applicability.

A supervised approach to textual reuse is proposed in [14]. Here, the most similar document to a query is retrieved using an information retrieval search engine (Lucene) and textual reuse is aided by presenting clusters containing similar documents for sections of the document. Each section is identified by a distinct heading common to all documents in the application domain (air travel incident reports). The major drawback of the approach is that it cannot be used when documents are unstructured. This means that common headings cannot be identified across documents for clustering to take place.

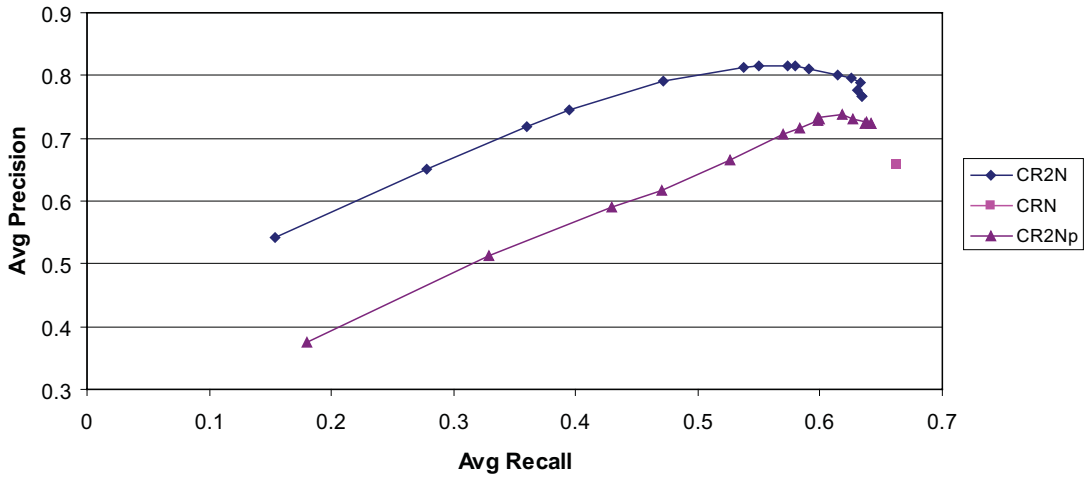


Fig. 6. Precision-Recall curve for the retrieved solution, CR2N & CR2N$_p$ at different k-neighbourhoods

Reuse of retrieved textual cases has been demonstrated on a semi-automated email response application [8]. The technique involves reuse of previous email messages to synthesize new responses to incoming requests. A response is a sequence of statements satisfying the content of a given request and requires some personalization and adjustment of specific information to be reused in a new context. Like CR2N, the reuse technique annotates sentences of the proposed solution. A sentence is annotated as reuse if there is sufficient evidence that similar past problems contain this sentence. The evidence is quantified by dividing the case base into 2 clusters that contains a similar sentence those that don't. A centroid case is formed for each cluster and compared with the query. Unlike CR2N's use of localised neighbourhood knowledge, here centroids can result in misleading evidence because two clusters of cases would have the same centroid if distance ratio between their cases are equal. Also, use of the entire case base to form clusters is inefficient for a large case base as the process has to be repeated for each sentence in a retrieved response.

5 Conclusions and Future Work

Three issues of *when*, *what* and *how* to revise need to be addressed when revising a piece of text. CR2N addresses the issue of *what* to be revised at the reuse stage by automatically annotating components of a solution text as reuse or revise. Experiments with CR2N on an NLG post-edit dataset shows up to 80% accuracy. It also has a higher precision and comparable recall to a retrieve-only system when tokens annotated as revise are ignored.

We intend to apply the technique on other textual datasets with varying vocabularies and to improve CR2N by capturing context (e.g. influence of left and right adjacent tokens) for each token in the CReuseNet. Our research also aims to develop methods that can help revise a retrieved solution text during problem solving.

Acknowledgements This research work is funded by the Northern Research Partnership (NRP) and the UK-India Education and Research Initiative (UKIERI).

References

1. Brüninghaus, S., Ashley, K.D.: Reasoning with textual cases. In Munoz-Avila, H., Francesco, R., eds.: Proceedings of the 6th International Conference on Case-Based Reasoning, ICCBR 2005, Springer Verlag (2005) 137–151

2. Reiter, E., Dale, R.: Building applied natural language generation systems. Natural Language Engineering **1** (1995) 1–32
3. Sripada, S.G., Reiter, E., Hunter, J., Yu, J.: Sumtime-meteo: Parallel corpus of naturally occurring forecast texts and weather data. Technical Report AUCS/TR0201, Department of Computer Science, University of Aberdeen (2002)
4. Lenz, M., Hans-Dieter, B.: Case retrieval nets: Basic ideas and extensions. In: Proceedings of the 20th Annual German Conference on Artificial Intelligence: Advances in Artificial Intelligence, London, Springer- Verlag press (1996) 227–239
5. Lenz, M., Burkhard, H.D.: Lazy propagation in case retrieval nets. In Wahlster, W., ed.: Proceedings of European Conference on Artificial Intelligence: ECAI-96, Los Angeles, John WIley and Sons (1996) 127–131
6. Lenz, M., Burkhard, H.D.: Case retrieval nets: Foundations, properties, implementations and results. Technical report, Humboldt University, Berlin (1996)
7. Chakraborti, S., Lothian, R., Wiratunga, N., Orecchioni, A., Watt, S.: Fast case retrieval nets for textual data. In Roth-Berghofer, T., Göker, M.H., Güvenir, H.A., eds.: Proceedings of the 8th European Conference on Case-Based Reasoning (ECCBR-06), Springer (2006) 400–414
8. Lamontagne, L., Lapalme, G.: Textual reuse for email response. In: Advances in Case-Based Reasoning, London, Springer-Verlag (2004) 234–246
9. Massie, S., Wiratunga, N., Craw, S., Donati, A., Vicari, E.: From anomaly reports to cases. In Ashley, K.D., Bridge, D.G., eds.: Case-Based Reasoning Research and Development. Springer, Berlin / Heidelberg (2007) 359–373
10. M.A., R., Wiratunga, N., Chakraborti, S., Massie, S., Khemani, D.: Evaluation measures for TCBR systems. In: Proceedings of the 9th European Conference on Case Based Reasoning, Springer press (2008) 444–458
11. Díaz-Agudo, B., González-Calero, P.A., Recio-García, J.A., Sánchez, A.: Building cbr systems with jcolibri. Special Issue on Experimental Software and Toolkits of the Journal Science of Computer Programming **69** (2007) 68–75
12. Brüninghaus, S., Ashley, K.D.: Evaluation of textual cbr approaches. In: Proceedings of the AAAI-98 Workshop on Textual Case-Based Reasoning (AAAI Technical Report WS-98-12), AAAI Press (1998) 30–34
13. Gervás, P., Díaz-Agudo, B., Peinado, F., Hervás, R.: Story plot generation based on CBR. In Macintosh, A., Ellis, R., Allen, T., eds.: Twelveth Conference on Applications and Innovations in Intelligent Systems, Cambridge, UK, Springer (2004)
14. Recio-García, J.A., Díaz-Agudo, B., González-Calero, P.A.: Textual cbr in jcolibri: From retrieval to reuse. In Wilson, D.C., Khemani, D., eds.: Proceedings of the ICCBR 2007 Workshop on Textual Case-Based Reasoning: Beyond Retrieval. (2007) 217–226