Acquisition and Reuse of Reasoning Knowledge from Textual Cases for Automated Analysis

Gleb Sizov, Pinar Öztürk and Jozef Štyrák

Department of Computer Science
Norwegian University of Science and Technology
Trondheim, Norway
{sizov, pinar}@idi.ntnu.no, jozef.styrak@gmail.com

Abstract. Analysis is essential for solving complex problems such as diagnosing a patient, investigating an accident or predicting the outcome of a legal case. It is a non-trivial process even for human experts. To assist experts in this process we propose a CBR-based approach for automated problem analysis. In this approach a new problem is analysed by reusing reasoning knowledge from the analysis of a similar problem. To avoid the laborious process of manual case acquisition, the reasoning knowledge is extracted automatically from text and captured in a graph-based representation, which we dubbed *Text Reasoning Graph* (TRG), that consists of causal, entailment and paraphrase relations. The reuse procedure involves adaptation of a similar past analysis to a new problem by finding paths in TRG that connect the evidence in the new problem to conclusions of the past analysis. The objective is to generate the best explanation of how the new evidence connects to the conclusion. For evaluation, we built a system for analysing aircraft accidents based on the collection of aviation investigation reports. The evaluation results show that our reuse method increases the precision of the retrieved conclusions.

Keywords: practical reasoning knowledge, causal relation extraction, knowledge acquisition, case reuse, textual CBR, automated analysis

1 Introduction

Which is more exciting about Sherlock Holmes stories: learning who the murderer was, or following Sherlock's reasoning on the road from evidence to conclusion leading him to the murderer?

Our overarching goal is to facilitate the reuse of reasoning knowledge residing in documents written by problem solvers. Such reasoning knowledge can be found embedded in, for example, accident reports. CBR research recognized accident reports as reusable experiences, and a special workshop challenge about analysing air investigation reports was organized as a part of the 4th Workshop on Textual case-based Reasoning in 2007 [6]. Accident reports were used by CBR researchers also before this workshop [11, 10, 22, 9]. Much of these work

combines CBR with information retrieval and focuses on the retrieval stage of the CBR cycle. It may be considered close to the "weakly textual" end of the weakly-strongly textual scale defined in [24]. Weakly textual means either that the concerned documents are sufficiently structured which reduces the need for sophisticated natural language processing (NLP), or that the type of task under consideration does not need deep NLP. Weakly textual CBR is appropriate for tasks that focus on retrieval and classification. However, when moved beyond retrieval or simple reuse of classes without adaptation, and toward the reuse of reasoning knowledge, strongly textual CBR becomes more appropriate because reasoning knowledge is often buried in text making it difficult to discern or process without deep NLP techniques. Bruninghaus and Ashley [8] are pioneers in strongly textual CBR who took in use information extraction and sophisticated NLP techniques. The work presented in this paper also uses deep NLP techniques for case acquisition (i.e. mining the reasoning knowledge that in turn comprises the problem solution part of a case) and reuse purpose.

Causality is recognized as the most fundamental type of reasoning knowledge [18]. In textual CBR research, Orecchioni et al. [17] attempted to extract causal knowledge by classifying sentences in accident reports as causal or factual. In our work we also aim to extract and reuse causal knowledge, not as a set of isolated causal sentences but rather as a chain of causal relations reflecting the problem solver's line of reasoning from the evidence to the conclusion. The rationale behind the proposed CBR-based analysis approach is that computers may support human analysts in understanding a problem through reuse of previously constructed reasoning paths that show how the conclusion *explains* the evidences. Such a CBR system essentially involves adaptation of past reasoning paths. Two challenges pertinent to the textual CBR task we are targeting and the type of data we are concerned with are: (i) representation and extraction of reasoning knowledge contained in text (ii) automated adaptation of this explanatory/reasoning knowledge to a new problem.

The rest of this paper is organized as follows. Section 2 provides an overview of our approach to CBR-based problem analysis. Section 3 describes Text Reasoning Graph (TRG), a graph-based representation for capturing reasoning knowledge extracted from text. In section 4, we present the procedure for automatic adaptation of reasoning knowledge to a new problem. Section 5 explains details for extraction of TRG from text. Empirical evaluation of the approach is described in section 6 with the results and error analysis in section 7. In section 8 we look at the related work. The conclusion and the future work directions are presented in section 9.

2 Automated problem analysis

In this section we introduce our approach to the automated problem analysis through reuse of reasoning knowledge extracted from textual cases. Under this approach, the analysis for a new problem is generated by retrieving a similar problem and adapting its analysis to fit the new problem. For this to work,

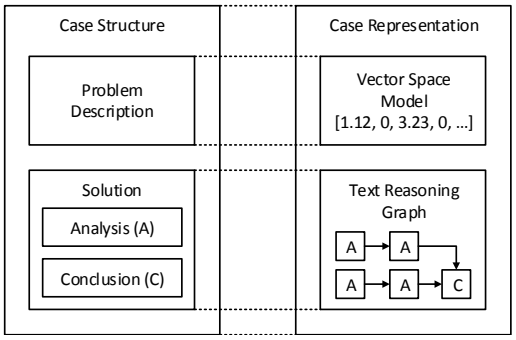

Fig. 1. Case structure and representation.

cases need to contain an analysis as part of their solution, as shown in Figure 1. The *analysis* should describe the reasoning of an expert connecting *evidences* to *conclusions* where the evidence is part of the problem description that serves as the starting point for the analysis while a conclusion refers to a decision or judgement that solves or explains the problem, e.g. the cause of an accident, a diagnosis for a patient or an outcome of a legal case.

Manual case acquisition is a laborious task. It becomes even more challenging when a case includes also an analysis part. To overcome the manual knowledge engineering problem and to take the advantages of abundant free-text analysis reports existing either in organizations or on the web, we target automated extraction of cases from such reports. First, we propose a case structure with a hybrid representation, as shown in Figure 1. In this structure, problem description is represented by a vector space model (VSM) with TFIDF weights, which is a well known representation for free text documents, often used in information retrieval (IR) and for document classification. We introduce a different representation for representation of the solution part which we dubbed *Text Reasoning Graph* (TRG), a graph-based representation with expressive power to represent the chain of reasoning underlying the analysis as well as to facilitate the adaptation of a past analysis to a new problem.

An overview of our approach is shown in Figure 2. In the *case acquisition* stage, the case base is populated by converting each free-text document to a case of which the problem description part is represented with VSM and the solution part as a TRG. Given the description of a new problem, the *retrieval process* finds a case with the most similar problem description. This is implemented by converting the new problem description to a VSM representation and then computing its *cosine* similarity with problem descriptions of the cases in the case base. Solution of the retrieved best case in the TRG form, further referred to as *CaseGraph*, is then adapted to the new problem description in the *reuse step*. The result of the reuse step is the *ReuseGraph* representing the analysis of the new problem in the TRG form - the details are described in section 4. Finally, the ReuseGraph is visualized for manual interpretation by a user.

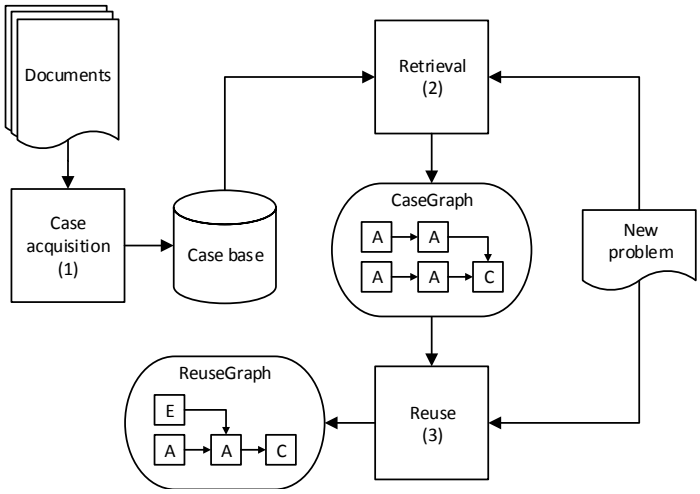

Fig. 2. System diagram

For our empirical work, we used a collection of aviation investigation reports where each report documents an investigation of an aircraft accident including a brief description of the accident, an analysis, and conclusions reached by a human expert. The reports have a consistent structure, with different sections corresponding to different parts in the case structure (Figure 1). For instance, the "Summary"/"Synopsis" section in a report is considered as the problem description, the "Analysis" section as the analysis, and several sections with titles similar to "Finding as to causes and contributing factors", "Findings as to risk" and "Other findings" as the conclusion. Hence, the conclusion consists of one or more sentences highlighting the possible causes and findings.

3 Representation of reasoning knowledge

For our approach, we needed a representation that is able to capture the line of reasoning embedded in text. Consider this excerpt from the analysis section of the aviation investigation report a06q0091:

> The oil that burned away did not return to the tank and, after a short time, the oil level became very low, causing the engine oil pump to cavitate and the engine oil pressure to fluctuate. Furthermore, since the oil did not return to the tank, the oil temperature did not change, or at least not significantly, and the pilot falsely deduced that the engine oil pressure gauge was displaying an incorrect indication.

The type of knowledge contained in this passage is important for understanding the accident because it describes how the human expert reasoned about the situation. The reasoning (e.g. causal) knowledge is mostly of relational nature.

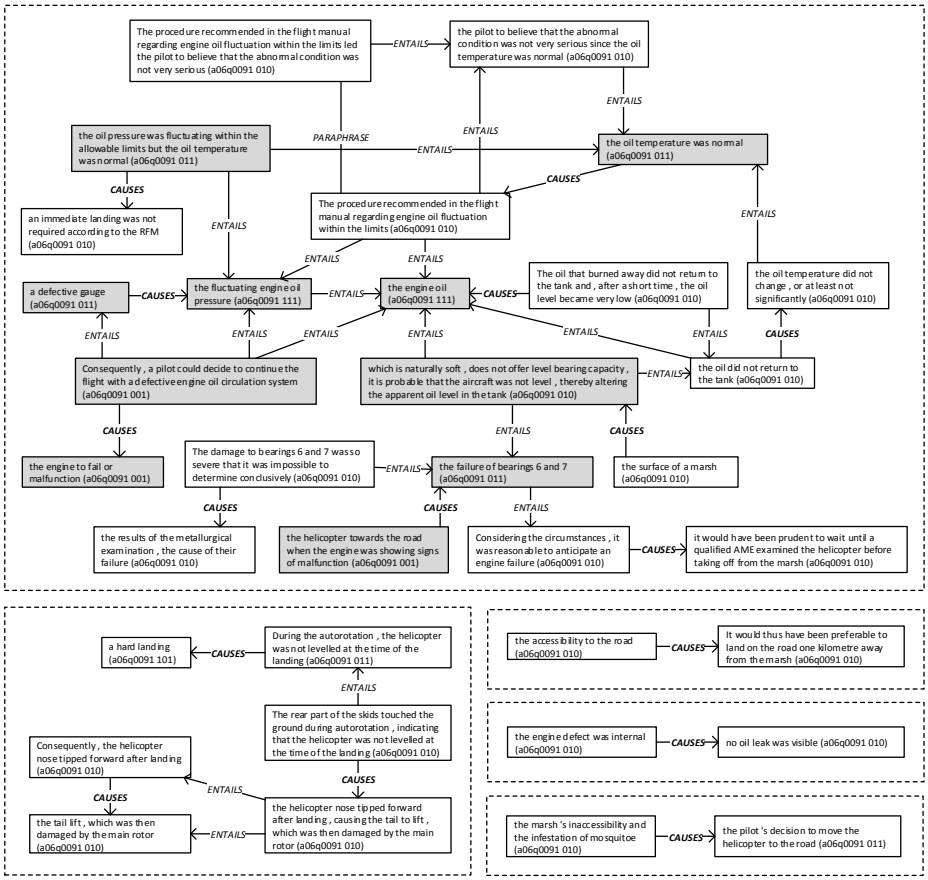

Fig. 3. Structure of the CaseGraph generated from report a06q0091. The content of the darker nodes can be seen in Figure 4.

A graph-based representation is therefore more appropriate than a frame-based representation, which is commonplace in classical CBR, because the chains/links of relations can explicitly be represented through edges in a graph. As noted in [19] causal graphs extracted from domain-specific documents provide a powerful representation of the expert's knowledge. The TRG representation comprises mainly causal relations, complemented with *textual entailment and paraphrase relations* that make the graph more connected. Automatic extraction of TRG from text relies on various NLP techniques, as described in section 5.

Figure 3 provides an overview of the CaseGraph structure generated from the report a06q0091. The graph consists of five connected components (surrounded by dashed lines) with one of them containing most nodes and relations in the graph. The content of the darker nodes is shown in Figure 4. As it can be seen from these figures, nodes in a TRG contain phrases (or sentences) that are

arguments for causal relations extracted from text. Contents of the arguments are determined through causal patterns described in section 5.1. Each node is also labelled with two codes: the report id and the code that indicates part(s) of the report the the node was extracted from, e.g. a node labelled a06q0091 010 is extracted from the analysis part (010) of the report a06q0091. Other binary codes correspond to the problem description (100) and conclusion (001). A combination of binary codes means that the concerned phrase is contained in several parts of the same report, e.g. 101 stands for the problem description and the conclusion parts together. Further in this paper, nodes extracted from the conclusion part will be referred to as *conclusion nodes*.

One of the advantages of the TRG representation is that it can be visualized and interpreted easily. It represents the reasoning knowledge in a more explicit way than the original text does. This makes TRG a visual summary providing a quick overview of the analysis of a case, which is useful in particular when the user needs to study several similar past cases to solve a new problem.

4 Reuse of reasoning knowledge

The flexibility of the TRG representation allows automatic adaptation of the retrieved solution to a new problem. The adaptation process generates a ReuseGraph from the retrieved CaseGraph and the description of a new problem. This process consists of the following steps:

1. Find *evidence nodes* (labelled 100 in Figure 4), i.e. nodes in the CaseGraph that are entailed by sentences in the description of the new problem.
2. Activate *reasoning paths*, i.e. one or several shortest paths (type and direction are ignored) that connect the evidence nodes (found in step 1) to conclusion nodes in the CaseGraph.
3. Construct a ReuseGraph by combining activated reasoning paths and then by adding sentences from the new problem description that entail the evidence nodes from step 1.

The first step corresponds to identifying the important pieces of evidence in the new problem description. These evidences are then used as starting points for reasoning in the second step where they are connected to the conclusions. The result of this procedure is the ReuseGraph that represents the analysis of a new problem based on the analysis of a previously solved problem. Notice that ReuseGraph may contain fewer conclusion nodes than the CaseGraph if some evidences in the new case do not link to the conclusions in the past case.

Figure 4 shows the reuse graph for the accident in report a08a0095 (new case) generated from the CaseGraph for report a06q0091 (past case) shown in Figure 3. Two nodes with bold frames on the left are sentences from report a08a0095 providing evidence that there is an engine failure or malfunction. This evidence can be explained by the defective engine oil circulation system as pointed out by the node to the right. There are several reasoning paths pointing to the failure of bearings, the oil level, the fluctuating oil pressure and, the defective gauge.

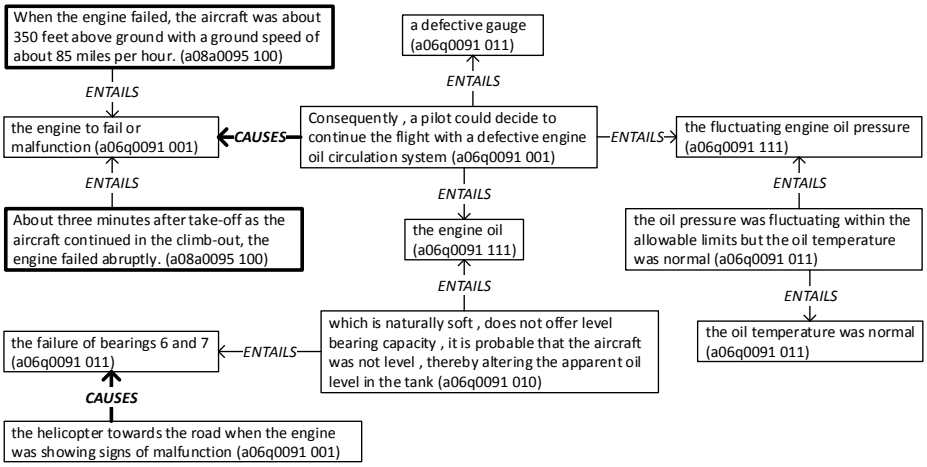

Fig. 4. ReuseGraph representing analysis of the accident a08a0095.

As it turns out, one of the correct conclusions for report a08a0095 is inadequate lubrication of linkpin bushings (type of bearing), which is close to failure of bearings and the oil level.

One major advantage of a ReuseGraph as the solution compared to a short answer, label or category, is that it provides a justification for the reached conclusions. At the same time, compared to the whole retrieved case as the output, a ReuseGraph is specifically adjusted to the new problem and eliminates all unnecessary information. A ReuseGraph tends to be much smaller than the corresponding CaseGraph, e.g. the ReuseGraph in Figure 4 contains only 12 nodes compared to 34 nodes in the corresponding CaseGraph in Figure 3.

5 Acquisition of reasoning knowledge

TRG contains three types of relations: causal, entailment and paraphrase relations. These relations are extracted automatically from text and combined in one graph. The process of generating a TRG is as follows:

1. Preprocessing text documents.
2. Extracting the causal relations.
3. Determining the informativeness of arguments in causal relations.
4. Combining relations where both arguments are informative into one graph.
5. Adding entailment and paraphrase relations between arguments.

Steps involved in the preprocessing stage are HTML scraping, tokenization, POS tagging, syntactic parsing, lemmatization and stop word removal. HTML scraping is implemented using custom component specifically designed to obtain the text and the structure of an investigation report from Transportation Board of Canada website. The rest of the preprocessing is accomplished using Stanford CoreNLP pipeline. Further in this section we describe steps 2, 3, 5 in more detail.

5.1 Causal relation extraction

The causal relation extraction component implements the approach described by Khoo [12] which uses manually constructed lexico-syntactic patterns. The extraction algorithm applies the set of patterns to a given sentence or a pair of neighbour sentences. If matching succeeds, the cause and the effect phrases are extracted according to the applied pattern. The patterns contain elements such as part-of-speech tags, phrase types and specific tokens. For example, given a pattern "[effect] because [cause]" and a sentence "The rotor blade failed because its structural integrity was compromised by a manufacturing defect" as the input, the system will extract "The rotor blade failed" as the effect and "its structural integrity was compromised by a manufacturing defect" as the cause. The patterns are built around causal triggers, words that indicate a causal relation between phrases. Based on Altenberg's [3] *typology of causal linkage*. Khoo defines a list of 651 causal patterns, 382 sub-patterns and 2115 causal verbs in his PhD thesis [12] where he describes also a pattern matching algorithm. We were unable to obtain the original implementation of the system and reimplemented the system using CoreNLP pipeline. We have updated Khoo's list of patterns, sub-patterns and causal triggers to improve the performance for our task.

5.2 Node informativeness

The quality of a TRG depends on the quality of the constituent nodes and relations. The quality of a node is determined by its informativeness. Nodes that do not carry a concrete piece of information are considered uninformative. Nodes in a TRG are the arguments of causal relations extracted from text, which are phrases of different size. Smaller phrases tend to be less informative than longer ones. Even if a word is relevant to the domain, e.g. "pilot", "flight", "procedure", "altitude", it often does not carry specific enough information to be used in a reasoning process. For phrases that contain more than one word we need to measure the informativeness of each word first. For this purpose we use inverse document frequency (IDF), a statistical measure commonly used in IR. Words with IDF values above 1.0 are considered informative. This threshold was manually determined by finding an IDF value that would filter out general terms for our dataset. For a phrase to be considered informative it should contain at least two non-stop words where at least one is informative.

5.3 Textual entailment and paraphrase relations

Causal relations are typically scattered in text and rarely form connected graphs with more than two or three relations in one graph. In order to link these graphs to form larger and more connected ones, we rely on entailment and paraphrase relations between the arguments of the causal relations. For this purpose, we employ methods for textual entailment and paraphrase detection.

Paraphrase detection is the task of recognizing text fragments with approximately the same meaning. For example, "the engine failure during take off"

and "in the climb-out, the engine failed abruptly" are paraphrases. In textual entailment the task is to determine whether one text fragment, called text (T), infers another text fragment, called hypothesis (H), i.e. if T is true H is also true. Unlike paraphrase, entailment is not symmetrical, e.g. "the engine failure during take off" entails "the engine failed" but "the engine failed" does not entail "the engine failure during take off" because part of the information, i.e. "during take off", is missing. A nice overview of the existing methods for detection of paraphrases and entailments can be found in the survey paper by [4].

In our system, paraphrase detection and textual entailment are based on the same text similarity measure. This measure assigns words in one text fragment to words in another fragment through a word similarity measure. LCH [14] is used for word similarity measure, which is based on a shortest path between the corresponding senses in WordNet[16]. The text similarity is computed by solving the assignment problem, where each word in one text fragment is assigned to a word in another text fragment so that no two words are assigned to the same word and the sum of similarities between assigned words is maximized. The final value is obtained by normalizing this sum by the average number of words in both fragments.

For paraphrase detection we compute the text similarity between two arguments. If the resulting value is equal to or above a threshold value, a paraphrase relation between these arguments is added to the graph. For entailment, if H is smaller than T we extract all substrings of T with the same number of words as in H + 1. Then the text similarity is computed between T and each substring of H. If the maximum of the obtained similarities is equal to or above the threshold value, the entailment relation from T to H is added to the graph. The similarity threshold value is set to 0.6, which was manually determined by considering the number of correctly identified entailment and paraphrase relations in TRG graphs constructed with different thresholds.

6 Experimental evaluation

For evaluation, we implemented a system that, given a short textual description of an accident and a collection of investigation reports for previous accidents, automatically generates an analysis of the new accident following the approach described in section 2. As the dataset, we use a collection of 494 aviation investigation reports from Transportation Board of Canada for years 1994-2008.

An analysis generated by our system is represented by a ReuseGraph. Evaluation of a ReuseGraph directly is problematic because of the lack of standard evaluation measures or baseline systems. To overcome this problem, we evaluate only conclusions in a ReuseGraph. Our assumption is that since reasoning paths in a ReuseGraph link the evidences in the new problem with conclusions in a similar past accident analysis, validity of the conclusions will reflect the quality of the reasoning paths and thus of the whole ReuseGraph. Validity of conclusions is determined by the similarity between the actual conclusions in the test case

report and the ones generated by the system. For the baseline we use conclusions in the retrieved report, i.e. immediately after retrieval, without adaptation.

6.1 Evaluation procedure

Every investigation report in our dataset contains one or two sections that correspond to what we refer to as conclusions. These sections contain sentences enumerating findings, causes and contributing factors for an accident. The following is from report a06q0091:

1. The area adjacent to bearings 6 and 7 had exceeded a temperature of 900 °C. The bearings were destroyed for undetermined reasons, causing an engine failure.
2. Moving the helicopter towards the road when the engine was showing signs of malfunction contributed to the failure of bearings 6 and 7.
3. During the auto-rotation, the helicopter was not levelled at the time of the landing, which resulted in a hard landing.

Conclusion nodes in a ReuseGraph usually do not contain full sentences from the conclusion part but rather phrases extracted from these sentences or entailed by them. Evaluation procedure outlined in the algorithm 1 contains separate functions for evaluation of the retrieval (i.e., baseline) and reuse steps. In both of them conclusion sentences are compared with the actual conclusions in the test case. The difference between them is that the first one takes all the conclusion sentences in the retrieved case while the latter makes use of the ReuseGraph to select the subset of the conclusion sentences.

In the retrieval evaluation, a case most similar to the problem description of the test (i.e, new) case is retrieved from the case base (line 2). Conclusion sentences are obtained (line 3) from this case and compared with reference conclusion sentences, i.e. the conclusion in the test case (line 4 and 5). The result of this comparison is the retrieval score which is based on the similarity between the retrieved and the reference conclusion sentences.

Reuse is evaluated in a similar way. The difference is that after the retrieval step (line 9), the reuse procedure is applied to generate a ReuseGraph based on the problem description of the test case and the CaseGraph of the retrieved case. A subset of conclusion sentences from the retrieved case is obtained through the conclusion nodes in the ReuseGraph (line 11). Unlike evaluation of the retrieval step where all the conclusion sentences are taken, only sentences that entail or are the source of one or more conclusion nodes in the ReuseGraph are retained. This results in fewer conclusion sentences than in the retrieval evaluation because the ReuseGraph contains only the conclusion nodes that are connected to evidence in the problem description of the test case, discarding all other nodes. The retained conclusion sentences are then compared with reference conclusion sentences in the same way as for the retrieval evaluation (line 13).

For both reuse and retrieval evaluation we use the same evaluation measure outlined in lines 16-22. This measure is based on the similarity between

Algorithm 1 Evaluation procedure

```
1: function EvaluateRetrieval(TestCase, CaseBase)
2:     RetrievedCase = Retrieve(TestCase.Problem, CaseBase)
3:     RetrievedCS = RetrievedCase.Conclusion.Sentences
4:     ReferenceCS = TestCase.Conclusion.Sentences
5:     RetrievalScore = Compare(RetrievedCS, ReferenceCS)
6:     return RetrievalScore
7: end function
8: function EvaluateReuse(TestCase, CaseBase)
9:     RetrievedCase = Retrieve(TestCase.Problem, CaseBase)
10:    ReuseGraph = Reuse(TestCase.Problem, RetrievedCase.CaseGraph)
11:    ReusedCS = ReuseGraph.Conclusion.Nodes.Sentences
12:    ReferenceCS = TestCase.Conclusion.Sentences
13:    ReuseScore = Compare(ReusedCS, ReferenceCS)
14:    return ReuseScore
15: end function
16: function Compare(CandidateCS, ReferenceCS)
17:    Cost = Assignment(CandidateCS, ReferenceCS, TextSimilarity)
18:    Precision = Cost / |CandidateCS|
19:    Recall = Cost / |ReferenceCS|
20:    F-score = 2 · Precision · Recall / (Precision + Recall)
21:    return (Precision, Recall, F-score)
22: end function
```

two sets of sentences, candidate conclusion sentences (CandidateCS) and the reference conclusion sentences (ReferenceCS). CandidateCS correspond to conclusions under evaluation, i.e. either RetrievedCS or ReusedCS. ReferenceCS are the correct conclusion sentences, i.e., the ones in the test case. Higher similarity between a CandidateCS and the ReferenceCS results in a higher evaluation score. The similarity is computed by using the notion of the *assignment problem* where each sentence in CandidateCS is assigned to a sentence in ReferenceCS so that no two sentences are assigned to the same sentence and the sum of text similarities between assigned sentences is maximized. This sum is referred to as Cost (line 17) and is used to compute precision, recall and F-score in lines 18-20. For the text similarity we use the same measure as described in section 5.3.

Our evaluation procedure follows leave-one-out cross-validation strategy, where one case is selected as the test case while the rest are considered as the case base. The evaluation scores are obtained for each test case to compute mean and standard deviation. Paired difference tests are carried out to compare the scores for retrieval and reuse steps.

7 Results and error analysis

Our system was able to generate ReuseGraphs for 118 of 494 reports that were used as the test cases. For the remaining cases, no connection was found between the evidence in the problem description of the test case and the conclusions of

Step	Precision	Recall	F-score
retrieval	19.71 ± 7.11	20.97 ±7.40	18.70 ± 5.18
reuse	25.03 ± 8.90	11.35 ± 7.99	13.74 ± 6.99
reuse - retrieval	5.32	-9.62	-4.96

Table 1. Evaluation results, mean ± standard deviation in % values.

the most similar case from the case base. This is mostly because the "Summary" sections used as the problem descriptions did not provide enough evidence for reuse.

Table 1 summarizes the results. Retrieval scores indicate the correctness of conclusions in a retrieved case before the reuse procedure, and the reuse results those of after the reuse procedure. Since the reuse can't add any new conclusions but merely removes conclusions that can't be connected to the evidence in the new/test case, the recall score after reuse is expected to be equal to or lower than that of before reuse. At the same time, reuse is expected to increase the precision score because conclusions are selected based on the reasoning paths originating from the new evidence. In contrast, a randomly selected subset of retrieved conclusions should not change the precision.

Evaluation results confirm our hypothesis about an increase in the precision and a decrease in the recall scores after reuse. The F-score decreased as well. Paired t-test and Wilcoxon signed-rank test show that the difference in scores is statistically significant with p-value < 0.0001. Increased precision indicates that the analysis generated by the system is able to connect the evidences in the test case to the correct conclusions, at least to some degree. It can also be argued that for our task precision is more important than recall because it reduces information overload for the user of the system by eliminating conclusions that the system is not able link to the new evidences. If necessary, the user can always see the solution before the reuse.

Manual inspection of the generated CaseGraphs and ReuseGraph indicates that although a lot of relations and paths through these graphs are coherent, there are many errors as well. These errors are mostly attributed to automatic generation of TRG from text. The used dataset is quite complex having long sentences and domain-specific terminology. Components in our NLP pipeline that rely on supervised machine learning are mostly trained on news articles and other corpora unrelated to the aviation domain which results in suboptimal performance. These errors propagate further in the system and decrease the accuracy of causal extraction, entailment and paraphrase identification components, which introduce errors of their own. We believe it is possible to improve each of these components by combining multiple approaches together and adjusting them specifically to the target domain.

8 Related Work

Adeyanju's PhD work [1] extends Case Retrieval Network (CRN) [15] to Case Retrieval Reuse Network (CR2N) to enable case reuse. Nodes in CR2N represent cases and keywords from solutions of these cases, and the edges connect cases to keywords or keywords to each other. While both CR2N and our approach use graph-based representations and have a focus on reuse, the content of the representation and the reuse process in the two approaches are substantially different. First of all, CR2N is a more general approach, not aiming to capture the reasoning knowledge. In the reuse process CR2N identifies reusable keywords in the retrieved solution,while our system translates the retrieved solution to a CaseGraph and, in turn, modifies it to a ReuseGraph. This modification involves addition and deletion of solution elements, which can be considered a structural form of reuse in terms of adaptation models described by Wilke et al. [23]. Although CR2N is also considered as a transformational form of reuse, our approach does not only identify what can be reused but generates the actual solution to be reused.

There are several other textual CBR systems that do structural reuse but without transforming a textual case to a more structured representation similar to TRG. Most of these are aimed at assisting users in text authoring. For example Lamontagne et al. [13] system adapts email responses by determining a subset of sentences relevant to the new request and replacing some specific information in these sentences such as individuals, locations and addresses. A somewhat similar approach has been applied by Swanson et al. [21]. The authors developed "Say Anything" CBR system for interactive storytelling. Given the story so far, the system proposes next sentences. The adaptation involves modification of the retrieved sentences by replacing proper nouns and pronouns with corresponding frequent substitutes from the already written part of the story. GhostWriter-2.0 developed by Bridge et al. [7] assists users in writing product reviews. The system suggests phrases extracted from previous reviews similar to what the user has already written. In the work by Recio-Garcíia et al. [20], the authors experimented with the same aviation accident report dataset as we do. The user is supposed to rewrite the solution of a retrieved similar report using text spans retrieved from other reports. The system assists the user by finding and clustering sections from the past aviation investigation reports that are relevant to the user's query. One interesting aspect of these text authoring systems is that they are focusing on providing assistance to the user rather than doing adaptation automatically. This assistance involves a graphical user interface to support human interaction, which often makes systems more practical for real-life applications. A branch of CBR research called conversational CBR[2] studies such systems. Currently, we don't have a conversational component in our system but it is highly relevant for future work.

The importance of reasoning knowledge, such as contained in the analysis section of aviation accident reports, is underrated in textual CBR research. Our work captures this knowledge in TRG predominantly through causal relations. A representation containing causal relations similar to TRG has been previously

proposed by Ashley et al. [5]. For their LARGO system, the authors developed a diagrammatic representation capturing arguments in a legal case. Their diagrams very much resembles TRG but with stronger semantics, which allows more sophisticated types of domain-specific adaptation. However, the diagrams were constructed manually by human experts, which is a very laborious process compared to automatic generation of TRG from text. In general, reasoning knowledge is crucial for legal cases and TRG representation seems like a good fit for this domain.

9 Conclusion and future work

We have presented a novel CBR approach that opens up for reuse of reasoning knowledge in textual cases. Our approach relies on a textual graph-based representation TRG, which captures reasoning/explanatory knowledge through causal, entailment and paraphrase relations automatically extracted from text. We also developed a reuse procedure to adapt the explanatory path used for a previous problem to a new problem. The approach was evaluated on a collection of aviation investigation reports, demonstrating the ability to capture and adapt the analysis of a previous accident to a new accident.

The novelty of the approach suggests many directions for future work. First, automatic extraction of TRG from text is a challenging tasks, with subtasks such as causal relation extraction, entailment and paraphrase identification leaving a lot of room for improvement. Second, conversational elements can be introduced to allow users to guide the analysis. Third, the reuse procedure can be enhanced by upgrading the shortest path with a more sophisticated heuristic that takes type and directionality of relations into consideration. Forth, structured domain knowledge can be integrated in the approach to improve TRG graphs. Fifth, it makes sense to use TRG in the retrieval step as well as in the reuse. Finally, extrinsic evaluation by human experts is required to confirm the validity of the approach in practical settings.

References

1. Adeyanju, I.: Case reuse in textual case-based reasoning. Ph.D. thesis, Robert Gordon University (2011)
2. Aha, D.W., Breslow, L.A., Muñoz-Avila, H.: Conversational case-based reasoning. Applied Intelligence 14(1), 9–32 (2001)
3. Altenberg, B.: Causal linking in spoken and written english. Studia linguistica 38(1), 20–69 (1984)
4. Androutsopoulos, I., Malakasiotis, P.: A survey of paraphrasing and textual entailment methods. Journal of Artificial Intelligence Research 38, 135–187 (2010)
5. Ashley, K., Lynch, C., Pinkwart, N., Aleven, V.: Toward modeling and teaching legal case-based adaptation with expert examples. In: Case-Based Reasoning Research and Development, pp. 45–59. Springer (2009)
6. Bridge, D., Gomes, P., Seco, N.: Analysing air incident reports: workshop challenge. In: Proc. of the 4th Workshop on Textual Case-Based Reasoning (2007)

7. Bridge, D., Healy, P.: Ghostwriter-2.0: Product reviews with case-based support. In: Research and Development in Intelligent Systems XXVII, pp. 467–480. Springer (2011)
8. Brüninghaus, S., Ashley, K.D.: Progress in textual case-based reasoning: predicting the outcome of legal cases from text. In: Proceedings of the National Conference on Artificial Intelligence. vol. 21, p. 1577. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999 (2006)
9. Carthy, J., Wilson, D.C., Wang, R., Dunnion, J., Drummond, A.: Using t-ret system to improve incident report retrieval. In: Computational Linguistics and Intelligent Text Processing, pp. 468–471. Springer (2004)
10. Cassidy, D., Carthy, J., Drummond, A., Dunnion, J., Sheppard, J.: The use of data mining in the design and implementation of an incident report retrieval system. In: Systems and Information Engineering Design Symposium, 2003 IEEE. pp. 13–18. IEEE (2003)
11. Johnson, C.: Using case-based reasoning to support the indexing and retrieval of incident reports. In: Proceeding of European Safety and Reliability Conference (ESREL 2000): Foresight and Precaution, Balkema, Rotterdam, the Netherlands. pp. 1387–1394. Citeseer (2000)
12. Khoo, C.S.G.: Automatic identification of causal relations in text and their use for improving precision in information retrieval. Ph.D. thesis, The University of Arizona (1995)
13. Lamontagne, L., Lapalme, G.: Textual reuse for email response. In: Advances in Case-Based Reasoning, pp. 242–256. Springer (2004)
14. Leacock, C., Miller, G.A., Chodorow, M.: Using corpus statistics and wordnet relations for sense identification. Computational Linguistics 24(1), 147–165 (1998)
15. Lenz, M., Burkhard, H.D.: Case retrieval nets: Basic ideas and extensions. In: KI-96: Advances in Artificial Intelligence, pp. 227–239. Springer (1996)
16. Miller, G.A.: Wordnet: a lexical database for english. Communications of the ACM 38(11), 39–41 (1995)
17. Orecchioni, A., Wiratunga, N., Massie, S., Chakraborti, S., Mukras, R.: Learning incident causes. In: Proc. of the 4th Workshop on Textual Case-Based Reasoning (2007)
18. Pearl, J.: Causality: models, reasoning and inference, vol. 29. Cambridge Univ Press (2000)
19. Pechsiri, C., Piriyakul, R.: Explanation knowledge graph construction through causality extraction from texts. Journal of Computer Science and Technology 25(5), 1055–1070 (2010)
20. Recio-Garcıa, J.A., Dıaz-Agudo, B., González-Calero, P.A.: Textual cbr in jcolibri: From retrieval to reuse. In: Proceedings of the ICCBR 2007 Workshop on Textual Case-Based Reasoning: Beyond Retrieval. pp. 217–226. Citeseer (2007)
21. Swanson, R., Gordon, A.S.: Say anything: Using textual case-based reasoning to enable open-domain interactive storytelling. ACM Transactions on Interactive Intelligent Systems (TiiS) 2(3), 16 (2012)
22. Tsatsoulis, C., Amthauer, H.A.: Finding clusters of similar events within clinical incident reports: a novel methodology combining case based reasoning and information retrieval. Quality and Safety in Health Care 12(suppl 2), ii24–ii32 (2003)
23. Wilke, W., Bergmann, R.: Techniques and knowledge used for adaptation during case-based problem solving. In: Tasks and Methods in Applied Artificial Intelligence, pp. 497–506. Springer (1998)
24. Wilson, D.C., Bradshaw, S.: Cbr textuality. In: Proceedings of the Fourth UK Case-Based Reasoning Workshop. pp. 67–80. Citeseer (1999)