Automatically Acquiring Structured Case Representations: The SMART Way

Stella Asiimwe[1], Susan Craw[1], Nirmalie Wiratunga[1], and Bruce Taylor[2]

[1] School of Computing
[2] The Scott Sutherland School
The Robert Gordon University, Aberdeen, Scotland, UK
{sa, smc, nw}@comp.rgu.ac.uk, B.Taylor@rgu.ac.uk

Abstract. Acquiring case representations from textual sources remains an interesting challenge for CBR research. Approaches based on methods in information retrieval require large amounts of data and typically result in knowledge-poor representations. The costs become prohibitive if an expert is engaged to manually craft cases or hand tag documents for learning. Thus there is a need for tools that automatically create knowledge-rich case representations from textual sources without the need to access large volumes of tagged data. Hierarchically structured case representations allow for comparison at different levels of specificity thus resulting in more effective retrieval than can be achieved with a flat structure. In this paper, we present a novel method for automatically creating, hierarchically structured, knowledge-rich cases from textual reports in the SmartHouse domain. Our system, SMART, uses a set of anchors to highlight key phrases in the reports. The key phrases are then used to learn a hierarchically structured case representation onto which reports are mapped to create the corresponding structured cases. SMART does not require large sets of tagged data for learning, and the concepts in the case representation are interpretable, allowing for expert refinement of knowledge.

1 Introduction

Case-based reasoning is an approach to problem-solving that offers a cost-effective solution to the knowledge acquisition bottleneck, since solutions do not have to be designed from scratch in every new problem situation [1]. Textual CBR aims to support problem-solving by making use of knowledge sources that are stored as text. However, the knowledge engineering effort required to extract cases from unstructured or semi-structured textual sources can lessen the advantages gained by developing a CBR system instead of using other problem-solving methodologies like rule-based reasoning.

Techniques in machine learning, natural language processing and information retrieval(IR) have been combined in efforts to identify features for indexing cases. IR-based methods employ shallow statistical inferences that typically result in knowledge-poor representations. This results in poor retrieval effectiveness as the representation determines the cases that will be retrieved.

Current research in Textual CBR aims to create more knowledge-rich case representations to enable effective retrieval and reasoning [13]. Techniques in natural language processing have been explored but their heavy reliance on grammar makes them unattractive in domains where problem-solving experiences were not recorded following strict grammatical structure. Machine learning approaches typically borrow ideas from inductive learning but the reliance on expert-tagged training data can make the cost of developing such systems prohibitive. This has created the need for tools that automatically create knowledge-rich case representations from textual sources without the need to access large volumes of tagged data.

Hierarchically structured case representations allow for case comparison at different levels of specificity, resulting in more effective retrieval than can be achieved with a flat structure. Although there have been efforts to automatically identify and represent cases with knowledge-rich features, these typically lack an underlying structure that links important domain concepts. Thus case decomposition to match problem descriptors at different levels of abstraction is not possible since such cases will have a flat structure.

We present SMART (**S**mart **M**ethod for **A**cquiring **R**epresentation for **T**ext), a tool that automatically creates knowledge-rich hierarchically structured cases from textual reports. SMART identifies domain knowledge inherent in the reports. It then uses the knowledge to learn a hierarchically structured case representation onto which the reports are mapped to obtain similarly structured cases. The rest of the paper is organised as follows. Section 2 gives a description of the data. Section 3 discusses the process of extracting knowledge from the textual reports and using it to create structured cases. The quality of the case content and the effect of structuring the cases are evaluated in Section 4 followed by related work in Section 5, and conclusions in Section 6.

2 SmartHouse Reports

SmartHouse problem-solving experiences are recorded as textual reports. Each report captures the problems/impairments of the person with disabilities and the SmartHouse devices that were installed in their home to assist them in carrying out different tasks. Figure 1 is an excerpt from one report. First, it briefly summarises the person's problems. It may mention a medical condition like *Alzheimer's disease* that results in disabilities, or explicitly state the person's disabilities e.g., *mobility problem*. Disabilities are typically referred to as a type of *problem* e.g., *hearing problem*, a type of *difficulty* e.g., *hearing difficulty*, or a kind of *impairment* e.g., *hearing impairment*. To distinguish disabilities from other terms in the text we refer to them as *disability terms*. We also refer to medical conditions that result in disabilities as *ailment terms*. Typically, the causes of a person's disabilities i.e., disability or ailment terms, are mentioned in the summary and may not be repeated in the problem description text where symptoms and problem-areas are elaborated. Sometimes both the disability term and ailment are mentioned.

The following sections of the SmartHouse reports record the different ways in which the person's disabilities manifest themselves. Each section describes a particular area of difficulty or risk. The excerpt shows a description of the *wandering* problem the person had. Every problem-area is given a summary heading, but they do not always accurately describe the content. *Telephone operation* may be used as a heading of a section describing a person's inability to use their telephone because they had difficulties *hearing* the caller. In another report, the same heading could be used for a description

> Mrs M, Y Street, Edinburgh. Initial Assessment Visit:
>
> *Mr and Mrs M were a couple in their eighties living in Y, and Mrs M had a tendency to wander due to* ***Alzheimer's disease*** *... A number of difficulties were identified:*
>
> ***Wandering:***
>
> *Mrs M had a tendency to leave the house and wander off. She had recently wandered away from the house in the middle of the night. Her husband had found her outside, dressed only in a thin nightgown, trying to open the side door to a neighbor's home... The final choice of equipment consisted of:*
>
> ***Bed occupancy sensor***
>
> *The bedroom light was automatically turned on at 8 pm. When the couple went to bed, the light went off as it sensed their presence in the bed by means of a pressure pad under the mattress...*

Fig. 1. Report Excerpt

of a person's difficulty in using their telephone because their *mobility* problems prevented them from reaching the phone in time to answer the call. The summary and the problem-description sections make up the problem part of a SmartHouse case. We shall refer to each problem-description section as a *sub-problem* since it is only a part of the whole problem description.

Lastly, the report gives a description of the SmartHouse devices installed in the house to help the person cope with their disabilities. In Figure 1, a *Bed occupancy sensor* was installed to help the person with their *wandering*. Each sub-problem can be mapped onto a corresponding list of SmartHouse devices.

SmartHouse device recommendation is based on people's disabilities. Indeed, when an occupational therapist needs to recommend SmartHouse solutions for a new person, she will be more interested in the person's disabilities and areas in which the disabilities manifest themselves, than in the medical condition that caused the disabilities. Therefore, we focus on structuring the problem parts of the reports and aim to base the structure on people's disabilities. Each problem-part is regarded as a document.

3 Creating Structured Cases

The task of creating structured cases from the documents is divided into the following steps:

1. Representing the documents with only those terms that actually describe the problem;
2. Using the representative terms to create a hierarchically structured case representation that reflects important features in the domain; and
3. Mapping the document representations onto concepts in the case representation in order to create structured cases.

It is important that cases capture the knowledge in the original reports if they are to be useful for problem-solving. Hence it is crucial for the case representation to capture important domain concepts and the relationships between them. In the SmartHouse domain, concepts are groupings of descriptors of people's disabilities and their manifestations. Some of this information may be in the summary but the bulk of it is embedded in the sub-problem descriptions of each SmartHouse report. Consequently, we need to extract this information and then use it to represent each difficulty, which is described in a sub-problem, with all the knowledge pertaining to it before exploiting the representations to create a case representation. First, terms that are likely to contain domain knowledge are extracted from each sub-problem. The terms are arranged into useful groupings according to different domain topics, which, with the help of background knowledge, enables us to identify key phrases. The key phrases and any relevant knowledge from other parts of the problem description (e.g., the summary) are all used to represent the sub-problem. However, since we also want the case representation to be based on people's disabilities, for those documents where the disability term is not stated, we enrich the sub-problem's representation by discovering and including, the appropriate disability term. The knowledge in the sub-problem representations is used to learn important domain concepts and the relationships between them which in turn, enables the creation of structured cases.

3.1 Term Extraction

We extract terms in the form of trigrams, bigrams and unigrams since SmartHouse concepts are often characterised by short phrases comprising 2 or 3 words like *hearing impairment* and *unable to communicate*, and only a few single words are highly meaningful. To avoid redundancy, we discard all substrings of terms that appear in the same sentence. All terms that begin and end with a stopword are also discarded. So terms may contain stopwords but will not start or end with one.

Every word appears by itself or as part of a longer phrase. The effect is that each sub-problem is transformed into a set of terms consisting of stemmed phrases. Extracted terms are also meaningful because they have not been distorted by the removal of internal stopwords. The task is described in detail in [2]. Figure 2(a) is a stemmed version of the text describing the *wandering* sub-problem in Figure 1. Figure 2(b) illustrates the extracted terms that are used to represent the sub-problem.

3.2 Topic Identification

SmartHouse reports can be expressed in terms of *topics* which are essentially, people's *disabilities*. Terms in the report collection can also be regarded as belonging to certain topics. Terms that are important to a given topic will have strong associations with each other. For example, in a topic regarding *mobility problems*, terms like *wheelchair*, *crutches*, and *mobility* will be highly related. Thus finding these topics is vital to identifying terms that actually contain useful knowledge. Latent Semantic Indexing (LSI) finds common dimensions in which both the documents and terms can be expressed in the same space i.e., the topic space [5]. We use the term *topics* to refer to *concepts* created by LSI so that they are not confused with other concepts in later parts of the paper. LSI employs the Singular Value Decomposition of a term × document ($m \times n$) matrix A:

$$A_{(m\times n)} = U_{O_{(m\times m)}} \times S_{O_{(m\times n)}} \times V^T_{O_{(n\times n)}}$$

(a) Stemmed Text

wander mrs m had a tendency to leave the house and wander off. she had recently wander away from the house in the middle of the night. her husband had found her outside, dress only in a thin nightgown, try to open the side door to a neighbor home...

(b) Extracted terms

wander, had a tendency, tendency to leave, leave the house, house and wander, wander away, house, night, thin nightgown, dress, neighbor home, open, door

(c) Key Phrases

wander, had a tendency, leave the house, night, dress, open, door

(d) Sub-problem Representation

alzheimers disease, wander, had a tendency, leave the house, night, dress, open, door

(e) Enriched Sub-problem Representation

dementia, alzheimers disease, wander, had a tendency, leave the house, night, dress, open, door

Fig. 2. The Different Stages of Document Processing

where, U_O represents the term matrix, V_O^T is the document matrix, and S_O is a diagonal matrix containing singular values arranged in descending order. Each column in U_O represents a *topic* and it captures the importance of every term to that topic. The r highest singular values identify the r most important topics in U_O. Thus the most important topics are represented by a $U_{(m \times r)}$ matrix shown shaded in Figure 3.

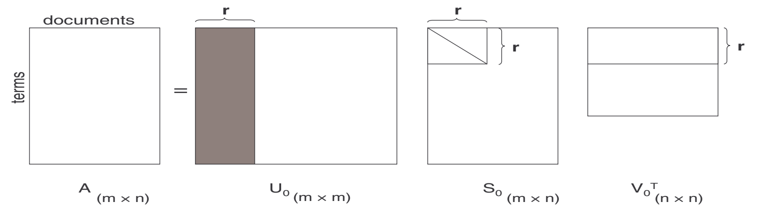

Fig. 3. Singular Value Decomposition and Latent Semantic Indexing

In our work, we obtain the matrix A by representing the documents as an incidence term × document matrix. Entry a_{ij} is the product of a local log frequency weighting and a global log entropy weighting of a term *i* in document *j*. Details can be found in [2].

We are only interested in term-topic associations as expressed in the $U_{(m \times r)}$ matrix. The weights in $S_{(r \times r)}$ reflect importances of the corresponding topics in $U_{(m \times r)}$. Multiplying $U_{(m \times r)}$ by $S_{(r \times r)}$ leads to the accentuation of the entries in $U_{(m \times r)}$. So the weights of terms in the accentuated $U_{(m \times r)}$ matrix become a measure of the *importance* of the individual terms to the key topics in the document collection. We use the top ($r = 9$) singular values to obtain the nine most important topics. It is these groupings of terms as topics that, with the help of background knowledge, we exploit in order to identify key phrases.

3.3 Key Phrase Identification

The text describing people's problems must in some way be related to the disability or the ailment causing the difficulties. Knowing this enables us to target our search for key phrases to terms that have a strong association with the disability terms and ailment terms. People's disabilities are referred to as types of *difficulty*, *problem* or *impairment* in the SmartHouse reports. Pattern-matching with these words enables the identification of disability terms explicitly stated in the reports. Thus, we are able to extract disability terms like *hearing impairment* and *mobility problem*. A list of known ailment terms like *alzheimers disease* and *multiple sclerosis* that result in disabilities is compiled using brochures from the website of Tunstall[1] and used to identify ailment terms in the text.

Disability terms and ailment terms act as *anchors* with which key phrases are identified. A term's *anchor* is a disability term or ailment term with which it occurs in the *same* document. The importance of a term in a given topic is reflected by its corresponding entry in the accentuated $U_{(m \times r)}$ matrix. Therefore, terms whose importance scores are higher than some threshold value, can be deemed to be key. So choosing an appropriate threshold is essential to identifying key phrases. We make use of background knowledge in the form of anchors, to inform our choice of this threshold.

Key phrases are identified for each sub-problem in turn. The task is to determine whether an extracted term is key or not. Consider the term *wander* which was extracted from the *wandering* sub-problem in which the anchor is *alzheimers disease*(Figure 1). In order to determine if *wander* is key, we find a topic in the accentuated $U_{(m \times r)}$ matrix, in which *wander*'s anchor i.e., *alzheimers disease*, has the highest importance score. The portion of the accentuated $U_{(m \times r)}$ matrix in Figure 4 shows *alzheimers disease* as having its highest importance score(7.29) in topic 3. Thus *7.29* is used to set the threshold for determining if *wander* is a key phrase.

We set the lower limit to 30% of the anchor's importance score, which is a good compromise since terms that are unimportant will typically have negative scores. Hence the threshold will be *2.19*. So terms whose importance scores are equal or above this threshold will be regarded as key. *wander*'s importance score of 7.48 is above this threshold and consequently, it will be identified as key. Other terms like *dementia* (whose score is 8.02) will also be selected as key. The effect is that terms that are nearly as important as their anchor(s), are identified as key. Figure 2(c) shows highlighted terms for the *wandering* sub-problem of Figure 1. Identified key phrases compare very well with those that an expert would deem to be key [2].

Each sub-problem is made independent of the other parts of the same document by representing it with the key phrases for that sub-problem, plus knowledge in other

[1] http://www.tunstall.co.uk

$$\begin{array}{lcccccc}
 & & & \downarrow & & & \\
 & \text{Topic 1} & \text{Topic 2} & \text{Topic 3} & \text{Topic 4} & \dots & \text{Topic n} \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\text{lock operation} & 1.79 & 5.87 & -0.85 & -1.93 & 2.06 & 2.76 \\
\longrightarrow \textbf{alzheimers disease} & \mathbf{0.08} & \mathbf{0.55} & \boxed{\mathbf{7.29}} & \mathbf{-1.20} & \mathbf{-0.47} & \mathbf{0.62} \\
\text{unable to hear} & 6.11 & -0.71 & -0.03 & -1.47 & 0.92 & -1.98 \\
\text{dementia} & 1.69 & -4.96 & 8.02 & 0.86 & 0.33 & 0.69 \\
\text{use a wheelchair} & 0.36 & 2.56 & -0.07 & 1.08 & -0.47 & 2.46 \\
\text{wander} & -1.74 & -1.12 & 7.48 & 0.92 & 0.30 & 0.51 \\
\text{abnormal gait} & 1.30 & 2.84 & 0.88 & 0.87 & 0.42 & 2.30 \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots
\end{array}$$

Fig. 4. Accentuated $U_{(m \times r)}$ Matrix Showing Term Importance for Key Topics

parts of that document, that pertains to that sub-problem. To achieve this, each sub-problem is represented by its key phrases and the anchors for that document. This is illustrated in Figure 2(d) where the anchor *alzheimers disease* has been added. The case representation is to be based on people's disabilities and so we need to ensure that each sub-problem's representation also reflects the person's disability. Thus for those documents where the ailment is mentioned instead of the disability, the appropriate disability term is identified and used to enrich the sub-problem's representation.

3.4 Enriching Sub-problem Representations

Representation enrichment is carried out for those sub-problems where a disability term is not mentioned. Each sub-problem is represented by all the knowledge in the document that pertains to it i.e., its key phrases and knowledge from anchors. This enables us to find interactions between terms in one sub-problem and those in another sub-problem of a different document. These interactions are used to generate association rules. What we need are rules whose conclusions are disability terms. That way, sub-problems in which the body of the rule appears can be enriched with the disability term. We argue that since representative terms are typically 2 or 3 word phrases, co-occurences of different terms is significant. So we make use of co-occurences of terms in the different sub-problems to mine association rules relating them.

The higher the number of co-occurences between any two terms, the more the evidence that they are related. Thus the number of co-occurences determines the support for the consequent rule. We do not expect to have a high number of co-occurences since the terms do not comprise only single words. So we use a low support threshold of 3. Only those rules whose conclusions are disability terms are taken into account and of these, we select only those whose conclusions apply for every term in the body i.e., rules with a confidence score of 100%. Consequently, we are able to generated rules like *alzheimers disease*→*dementia* $\{4\}$, where *4* is the support for the rule. The rules allow us to associate an ailment like *alzheimers disease* with the disability *dementia* or a term like *wander* with *dementia*, hence discovering disability terms for documents that did not have any.

Discovered disability terms are added to the text representing each sub-problem. Figure 2(e) illustrates the resulting representation for the *wandering* sub-problem of Figure 1; *dementia* is the discovered disability term. All knowledge pertaining to each sub-problem, including that in the summary, is now included in the sub-problem representation. Hence we represent the whole problem part of the original document with its sub-problems. The sub-problem representations now contain domain knowledge that can be used to create a case representation. Formal Concept Analysis [9] is used to generate a hierarchy of concepts from the sub-problem representations. The hierarchy is what is transformed into a representation for our textual cases.

3.5 Formal Concept Analysis

In Formal Concept Analysis (FCA), a formal context is a triple (O, A, I) where O is a set of objects, A a set of attributes and $I \subseteq O \times A$ is a binary relation that indicates which objects have which attributes. Figure 5 shows a context of sub-problem objects and their possible features which form the set of attributes. The crosses indicate attributes of each object. FCA makes use of a formal context to produce a hierarchy of concepts. A formal concept is a pair $(o \subseteq O, a \subseteq A)$ such that every object in o is described by every attribute in a and conversely, every attribute in a covers every object in o. The objects associated with a concept are called its extent, and the attributes describing the concept are called its intent. In Figure 5, the set of objects $\{$*sub-problem 1*, *sub-problem 4*$\}$ have the set of attributes $\{$*wander*$\}$ in common. Conversely, the set of attributes $\{$*wander*$\}$ shares a common set $\{$*sub-problem 1*, *sub-problem 4*$\}$ of objects to which they belong. No other object has this set of attributes. This results in a concept whose intent is $\{$*wander*$\}$ and extent $\{$*sub-problem 1*, *sub-problem 4*$\}$.

	ATTRIBUTES				
OBJECTS	dementia	hearing impairment	wander	alzheimers disease	mobility problem
sub-problem 1	X		X		
sub-problem 2					X
sub-problem 3		X			
sub-problem 4			X	X	

Fig. 5. Context for some SmartHouse Sub-problems

3.6 Constructing the Case Representation

FCA can generate a hierarchy of concepts from a context of SmartHouse sub-problems and their representative terms. However, some of the terms do not discriminate between disabilities. A phrase like *intercom operation* can be used in cases where the person had a *hearing impairment* (and could therefore not hear the buzzer), or where the person had a *mobility problem* and had difficulty reaching the intercom because of its positioning.

In order to disambiguate terms with respect to people's disabilities, each is tagged with the corresponding disability term. Thus *intercom operation.mobility problem* will be different from *intercom operation.hearing problem* where the term after the period is the disability term.

We apply FCA to a context of sub-problems and their now *tagged* representative terms in order to obtain formal concepts for the case representation. Figure 6 illustrates a portion of the resulting concept hierarchy. Every node represents a concept and the nodes are ordered by a concept-subconcept relationship. The highest node represents the most general concept while the lowest one represents the most specific concept. To prevent cluttering, an attribute is attached to the top-most concept that has the attribute in its intent. The attribute occurs in all intents of concepts that are reachable by descending the subtree from which it is attached. For example, node 5 represents a concept whose intent is $\{$*dementia*, *flood.dementia*$\}$. Using tagged attributes ensures that there are clear demarcations between the different disabilities. Nodes 1, 2, 3, and 4 are concepts representing some of the most common disabilities. There is also a clear distinction between concepts that may be shared between disabilities as these may warrant different sets of SmartHouse solutions. Nodes 5 and 7 are *flooding* problems due to the disabilities *dementia* and *learning problems* respectively. The case representation is obtained by removing the tags from attributes in each intent. Each level 1 sub-tree of the case representation represents a disability and the shared concepts provide the different levels of abstraction in that sub-tree. Figure 7 is a portion of the case representation illustrating the *dementia* sub-tree. Nodes 1 and 2 in Figure 7 correspond to nodes 6 and 5 of Figure 6 respectively.

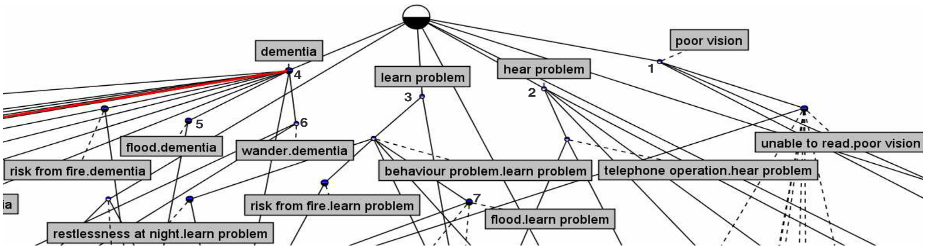
dementia
4
learn problem
3
hear problem
2
poor vision
1
5
6
flood.dementia
wander.dementia
unable to read.poor vision
risk from fire.dementia
behaviour problem.learn problem
telephone operation.hear problem
a
risk from fire.learn problem
7
flood.learn problem
restlessness at night.learn problem

Fig. 6. Concept Lattice with Tagged Intents

3.7 Case Creation

Normally, an occupational therapist would record a person's disabilities, problem areas and symptoms, under pre-defined groupings: *wheelchair* would be recorded under *mobility problem*; *unable to hear buzzer* under *hearing problem*. Similarly, the task of creating structured cases from the document representations involves mapping the representations onto concepts in the case representation. For each problem representation, concepts in which *all* elements of the intent are contained in the problem's representative terms, are instantiated as present. All remaining concepts in the case representation

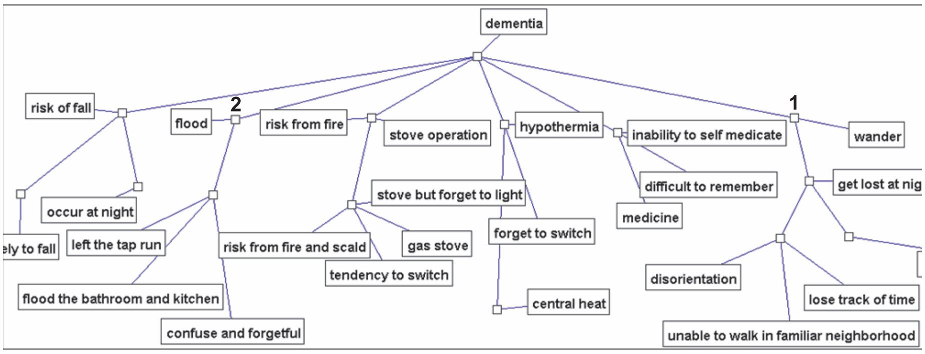

Fig. 7. Portion of Case Representation

are instantiated as absent. The result is a hierarchically structured case similar to the representation shown in Figure 7. Every sub-problem has a matching solution package that solves that sub-problem. So we attach solution packages to leaf nodes of the sub-problems since it has proved to result in good retrieval performance [2].

4 Evaluation

We evaluate SMART by testing the different representations for retrieval tasks. Apache Lucene[2], was used to perform retrieval on the original reports and the case representations. SMART performed retrieval on the structured cases. Lucene ranks documents according to the cosine-distances between the documents and query vectors in IR's Vector Space Model [11]. The model treats each document as a bag-of-words and relies on frequency information of query terms in the document and in the whole collection. SMART structures the query before making pair-wise comparison between nodes in the query structure and those in the structure of each activated case. Similarity *sim*, between any two nodes is given by:

$$sim = 1 - \frac{\sum \text{distance to nearest common parent node}}{\sum \text{distance to the root node}}$$

This captures some interesting intuitions of our notion of similarity. For instance, if two nodes are completely dissimilar, then their nearest common parent node will be the root node of the tree and their similarity score will thus be 0. On the other hand, if two nodes are siblings, then the distance to the nearest common parent node will be 1 for each of them. The similarity score will depend on their depth in the structure; the lower they are, the higher the similarity. Three retrieval methods were carried out:

1. Lucene on original documents(referred to as *Lucene*);
2. Lucene on case representations(*Lucene-plus*); and
3. SMART's retrieval module on the structured cases(SMART).

[2] http://lucene.apache.org

Ten queries were used, each handcrafted by the expert who also provided their solutions. The queries were written in the same manner as the problem descriptions but it was ensured that no case had all the query words. Queries were typical problem-descriptors e.g "*forgetful, unable to self-medicate*". Lucene treats each query as a bag-of-words so in order to make a fair comparison between the three methods, the queries were treated as a bag-of-words for the SMART method as well. Consequently, cases whose concepts contain at least one query word were activated for comparison to the query structure.

Experiments were run for 10 different queries. Precision was measured as the proportion of relevant cases in the top 3 retrieved cases. Recall was the ratio of relevant cases in the top 5 retrieved cases to the total number of cases that have the solution for the query. Relevance of a retrieved case was ascertained by determining its similarity to the expert solution.

4.1 Discussion of Results

Results are presented in Figure 8. We observe good precision for Lucene-plus over Lucene in 7 of the 10 queries. This shows that in some of the cases, the content is more focused to problem-solving, resulting in more effective retrieval than the original documents. Precision values for queries 2 and 6 show the case content to be as adequate for problem-solving, as the original reports.

Query	Relevant Cases	Precision			Recall		
		Lucene	Lucene-plus	SMART	Lucene	Lucene-plus	SMART
1	3	0.67	1.00	1.00	0.67	1.00	1.00
2	2	0.67	0.67	0.67	1.00	1.00	1.00
3	8	1.00	0.67	1.00	0.50	0.38	0.50
4	6	0.67	1.00	1.00	0.33	0.50	0.67
5	7	0.67	1.00	1.00	0.43	0.71	0.57
6	6	1.00	1.00	1.00	1.00	1.00	1.00
7	5	0.33	0.67	1.00	0.20	0.80	0.80
8	3	0.67	1.00	0.67	1.00	1.00	1.00
9	3	0.33	0.67	0.67	1.00	1.00	1.00
10	7	1.00	0.33	1.00	0.43	0.43	0.57

Fig. 8. Test Results for Precision and Recall

In Query 10, Lucene obtains better precision than Lucene-plus. This is not surprising because query 10 was the most complex of all, comprising words from *Alzheimer's*-related cases, those regarding *memory* problems, and those with *wandering* problems. When the words are fewer (Lucene-plus), IR-based ranking of the relevant documents goes down for this complex query as reflected by Lucene and Lucene-plus' (equal) values of recall. However, SMART has good performance for this query. This further illustrates the superiority of similarity measures that take into account relationships between concepts, over bag-of-words approaches.

Analysis of overall performance indicates that SMART performed best with an average precision of 0.90 compared to Lucene's 0.70 and Lucene-plus' 0.80. SMART's average recall is 0.81, Lucene's is 0.66 and Lucene-plus has 0.78. This shows that hierarchically structured case representations allow for effective similarity matching even when cases are activated in a naïve way such as the bag-of-words we used. Comparing structures the way we do ensures that structures that are different are interpreted as such; a *mobility-related* case will get a similarity score of 0 when compared to a case regarding *hearing* problems since nodes in the two cases will have the root node as their nearest common parent.

4.2 Comparison with a Manually Created Case Representation

The case representation was also compared to one that was manually created in an earlier SmartHouse CBR project [14]. For the manual exercise, ten general types of problem were identified and then the more specific problems described in reports were manually extracted and clustered under the ten general classes. Figure 9 shows some of the ten general concepts and subconcepts for the *ChallengingBehaviour* category.

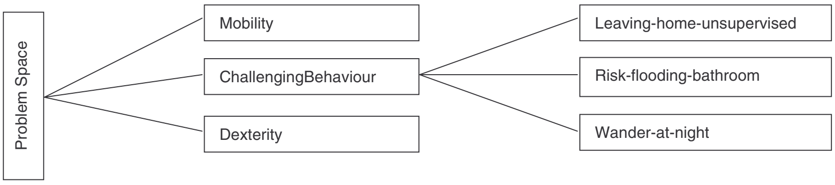

Fig. 9. Manually Crafted Hierarchy

Consider the concept *challenging behaviour* under which the subconcepts *leaving-home-unsupervised*, *risk-flooding-bathroom* and *wander-at-night* are clustered. In *dementia* patients, *flooding* and *wandering* are due to *memory* problems while in people with *learning problems*, the acts are *intentional*. Different solutions will be recommended depending on the underlying disability. Thus cases with the *challenging behaviour* concept of Figure 9 will require the user to have knowledge of the previous people's disabilities before they can be re-used unlike those created using SMART's case representation. This goes to show that case representations that are crafted by humans are also prone to error. While the automatically created case representation will not be perfect, it is much easier for an expert to amend an imperfect representation than to create one from scratch.

5 Related Work

Current research which focuses on automatically or semi-automatically creating knowledge-rich case representations can be classified under 2 broad categories; The first category extract predictive features for representing the textual cases. SOPHIA [7] employs distributional clustering [8] to create word-groups that co-occur in similar documents; the word clusters can represent the textual documents. Wiratunga et. al [15] exploits keyword co-occurence patterns to generate association rules that aid extraction of features to represent the textual cases. Thompson [12] applies Ripper [4] a rule induction system, to obtain features for text categorization tasks in the law domain. Ripper features can be a set of word stems contained in the documents. These approaches result in better retrieval effectiveness than IR-based ones but the representative features still lack an underlying structure relating the different concepts. Like SOPHIA and Wiratunga et. al [15], SMART exploits co-occurence patterns of terms in the different sub-problems to learn association rules in order to enrich the case representation. However, SMART goes further to identify domain concepts in the enriched representations and to learn their relationships in order to obtain a conceptual structure. Techniques like Latent Semantic Indexing produce features with an underlying structure; the features are linear combinations of terms in the document collection. However, the interpretability of the representations and underlying concepts remains a gray area. Thus expert initiated refinement of knowledge is difficult for these features. SMART exploits the linear combinations of terms provided by LSI (in the form of topics) to identify important features that are used to create an interpretable structure.

The second category typically employs information extraction systems to obtain structured representations of the textual sources. The DiscoTEX framework [6] constructs a structured template from text, with pre-defined slots. The slots are obtained by first tuning an information extraction system using a corpus of documents annotated with the filled templates. SMILE [3] makes use of an information extraction system Autoslog [10] to extract relevant information in order to learn indexing concepts for textual cases. Common among these systems is the significant amount of manual intervention required for tuning the information extractors. SMART does not employ an information extraction system but, makes use of information extraction techniques and background knowledge to extract key phrases which are used to learn a conceptual structure in an unsupervised process.

SMART overcomes the short-comings in the systems mentioned above by combining their complimentary strengths. The result is an automatically created knowledge-rich, hierarchically structured, case representation. The case representation and its concepts are interpretable, allowing for expert refinement of knowledge.

6 Conclusions

This paper presents SMART, a novel approach to automatically obtaining a structured case representation from semi-structured textual sources. SMART makes use of background knowledge to determine a set of anchors that help to highlight key phrases in the text. The key phrases are then used to learn a hierarchically structured case representation onto which the textual reports are mapped before they are used in reasoning. The novelty is in SMART's ability to learn important domain concepts that are interpretable, the relationships between them, and to use the concepts to create hierarchically structured cases, without the need for tagged data.

We have evaluated the quality of the case content against original documents and found the case knowledge to be as adequate and sometimes better focused for problem-solving. We have also obtained better retrieval effectiveness using the case structure than we did with a flat structure using a high performance IR tool. This is a useful feature for domains where the ability to match problem descriptors at various levels of abstraction is crucial but more importantly, where case and adaptation knowledge are scarce.

The approaches presented are generally applicable for knowledge modelling in semi-structured textual sources where domain knowledge is embedded in the free-form text of the section content. In domains like the medical and SmartHouse where concepts are shared between different entities e.g., symptoms among different diseases, there is a requirement to have a conceptual structure that is interpretable in addition to a knowledge-rich case representation, while ensuring that the system is a cost-effective solution to problem-solving. SMART fulfills these requirements by harnessing a number of techniques.

References

1. A. Aamodt and E. Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 1994.
2. S. Asiimwe, S. Craw, B. Taylor, and N. Wiratunga. Case authoring: from textual reports to knowledge-rich cases. In *Proc of the 7th Int. Conf. on Case-Based Reasoning*, pages 179–193. Springer, 2007.
3. S. Brüninghaus and K. D. Ashley. The role of information extraction for textual CBR. In *Proc of the 4th Int. Conf. on Case-Based Reasoning*, pages 74–89. Springer, 2001.
4. W. W. Cohen and Y. Singer. Context-sensitive learning methods for text categorization. In *SIGIR*, pages 307–315, 1996.
5. S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391-407, 1990.
6. U. Nahm and R. Mooney. Text mining with information extraction, 2002.
7. D. Patterson, N. Rooney, V. Dobrynin, and M. Galushka. Sophia: A novel approach for textual case-based reasoning. In *Proc of the 19th Int. Joint Conf. on Artificial Intelligence*, pages 15–20, 2005. Professional Book Center.
8. F. C. N. Pereira, N. Tishby, and L. Lee. Distributional clustering of english words. In *Meeting of the Association for Computational Linguistics*, pages 183–190, 1993.
9. U. Priss. Formal concept analysis in information science. *Annual Review of Information Science and Technology*, 40, 2006.
10. E. Riloff. Automatically generating extraction patterns from untagged text. In *Proc of the Thirteenth National Conf. on Artificial Intelligence*, pages 1044–1049, 1996.
11. G. Salton and M. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, New York, 1983.
12. P. Thompson. Automatic categorization of case law. In *Proc of the 8th Int. Conf. on Artificial intelligence and law*, pages 70–77, 2001. ACM Press.
13. R. O. Weber, K. D. Ashley, and S. Brüninghaus. Textual Case-Based Reasoning. *Knowledge Engineering Review*, 20(3):255–260, 2005.
14. N. Wiratunga, S. Craw, B. Taylor, and G. Davis. Case-based reasoning for matching SmartHouse technology to people's needs. *Knowledge Based Systems*, 17(2-4):139–146, 2004.
15. N. Wiratunga, R. Lothian, and S. Massie. Unsupervised feature selection for text data. In *Proc of the 8th European Conf. on Case-based Reasoning*, pages 340–354, 2006.