Learning Semantic Annotations for Textual Cases

Eni Mustafaraj¹, Martin Hoof², and Bernd Freisleben¹

¹ Department of Mathematics and Computer Science, University of Marburg
{eni,freisleb}@informatik.uni-marburg.de
² Fachhochschule Kaiserslautern, Kaiserslautern, Germany
m.hoof@et.fh-kl.de

Abstract. In this paper, we propose an approach to attach semantic annotations to textual cases for their representation. To achieve this goal, a framework that combines machine learning algorithms, natural language processing, and domain knowledge to semi-automatically learn semantic annotations for a collection of documents from the field of predictive maintenance is presented.

1 Introduction

Case based reasoning (CBR) is about reusing and adapting the solutions of previous problems for new situations. Textual case based reasoning (TCBR)—the focus of our paper—is concerned with situations where problem solutions are in textual form written by a domain expert, like a physician, an attorney, or a technician. According to Lenz et al. [1], “...the major advantage of TCBR is the ability to make use of domain-specific expertise in order to go beyond pure keyword matching and thus improve the problem solving behavior. However, ..., a well designed knowledge acquisition process is required in order to encode existing knowledge. ...”

Knowledge in a CBR system is usually found in one of the following containers: the vocabulary used; the similarity measure; the case base; and the solution transformation [2]. In TCBR, a document repository could serve as the case base, and usually no adaptation knowledge is needed (although recently, a first approach for textual case adaptation has been published [3]). Therefore, the knowledge modeling process usually deals with the creation of an index vocabulary to retrieve cases and the definition of various similarity measures [1].

In [4], Lamontagne and Lapalme analyze and compare six different TCBR research approaches from several viewpoints. They notice that none of the works proposes a complex process for index selection (this is done either by a domain expert or with methods of the information retrieval field, or with a combination). Furthermore, they could not find clear guidelines on which knowledge engineering phase the efforts should be put: on the conceptual representation of cases, on richer similarity functions, or on a more sophisticated formalized search. They conclude that for the moment the complexity/length of text cases as well as their intended use are the deciding factors in preferring one approach over the other.
In this paper, we demonstrate the importance of the conceptual representation of cases. Our approach for the conceptual representation of cases consists of two phases. In the first phase, we try to semi-automatically learn semantic annotations for textual cases. In the second phase, we use the terms related to semantic annotations as well as domain knowledge to populate a domain ontology that can offer richer similarity measures for case retrieval. We hope that the selected case representation will allow us to use the textual cases not only for TCBR, but also for further tasks like text mining or question answering. In this paper, we concentrate on the first phase, the implementation of a framework that combines machine learning algorithms (using an active learning approach), natural language processing, and domain knowledge to semi-automatically learn semantic annotations for textual cases.

The paper is organized as follows. In Section 2, an application in the predictive maintenance domain and its related document collection of textual cases is described. Our framework for semi-automatically assigning semantic annotations to textual cases is discussed in Section 3. In Section 4, we compare our approach to other proposals in the TCBR literature and identify differences and similarities. Section 5 concludes the paper and outlines areas of future research.

2 Textual Cases in Predictive Maintenance

2.1 Domain Description

Our domain of interest is predictive maintenance in the field of power engineering, more specifically, the maintenance of insulation systems of high voltage rotating electrical machines. A detailed account of the domain specific problems is presented in [5]. Typically, predictive maintenance is the consequence of a monitoring, or a monitoring and diagnosis process. Since in many domains it is prohibitive to allow faults that could result in a breakdown of the system, components of the system are periodically or continuously monitored to look for changes in the expected behavior, in order to undertake predictive maintenance actions when necessary.

Monitoring and diagnosis for predictive maintenance is somehow different from the better known problem of diagnosis for troubleshooting. During troubleshooting, it is important to find the source of a problem that has already happened, while in predictive maintenance, diagnostic tests are used to recognize possible symptoms that could lead to problems. Afterwards, the monitoring and diagnostic findings can be documented in several forms: the measured values in a relational database; the evaluations of measurements/tests in diagnostic reports written in natural language; or the recognized symptoms in photos.

In the past, we have built an experience management system relying on CBR methodology [6] that supports interactive diagnostic analysis. The CBR contribution consists of using the problem description structure imposed by the existing database to find similar measurements or machines, in order to retrieve the documents where the problem solutions (diagnostic evaluations) are described.
The user will then decide whether the retrieved documents are relevant for the current problem solving. This is certainly a progress compared to the previous situation (where no such system existed and users could only rely on their experience), but we would like to find more sophisticated approaches to make the reuse of expertise stored in the text documents more easily accessible to the users, especially to the newcomers in the domain. Before presenting the intended goals for the reuse of cases at the end of subsection 2.3, we first describe the available collection of documents.

2.2 Document Collection Description

In the field of predictive maintenance, two parties are involved: the service provider (the company that has the know-how to perform diagnostic procedures and recommend predictive maintenance actions) and the customer (the operator of the machine). The customer is interested in receiving a high quality service that will prevent unexpected interruption of power generation as the result of machine failures. Therefore, the customer agrees to pay for the diagnostic evaluation services to get appropriate guarantees from the service provider, which are given in the form of an official diagnostic report. The report follows a predefined structure template and is written in syntactically correct and parsimonious language. In our case, the language is German.

A document is organized into many sections: summary, reason of the inspection, data of the inspected machine, list of performed tests and measurements, evaluations of measurement and test results, overall assessment and recommendations, as well as several attachments with graphical plots of numerical measurements or photos of damaged parts. The fact that the documents are written with Microsoft® Office Word, which uses a proprietary format to store the text, has been a problem for the automatic processing of the reports in a first phase. With the introduction of MS Office 2003, Microsoft® presented its new format, WordML, an XML-conform format that permits the processing of MS Word documents also outside the MS Office platform. By using XML DOM tools, we could automatically transform MS Word documents into XML documents containing the text we were interested in, while preserving the original structure (division in sections, subsections, paragraphs). The preservation of document structure is very important, because every diagnostic statement is relevant within a defined context. Currently, we have a repository with diagnostic reports of 500 machines, with a total of more than one million words.

2.3 Understanding and Reusing Textual Cases

A diagnostic document contains information about all tests and measurements performed on the insulation system of the machine. In the following discussion, every description of a performed test/measurement is considered a separate textual case. A document may contain many of such cases (depending on how many tests/measurements were performed), which nevertheless, can be easily accessed using the XML document representation. For the collection of 500 diagnostic
documents, since every document has an average of ten different tests, the size of our collection amounts to approximately 5000 textual cases, consisting of one to many sentences.

A textual case is usually divided in two parts: <evaluation> and <action>. However, this division should not be seen as a division in (problem description, problem solution) as we usually expect in CBR. Most of the times, in the <evaluation> part, both the depiction of symptoms (problem description) and the reveal of possible causes (problem solution) appear; whereas in the <action> part, only different kinds of recommendations are given, which cannot be always seen as a problem solution (although the presence of text other than “none” in the <action> part is a good heuristic that the <evaluation> part contains interesting diagnostic information). The annotated example in Figure 1 describes such a situation; the original German text was translated to English to illustrate the annotations for non-German speaking readers.

<measurement title = "Loss_Factor">
  <evaluation>[ measured_quantity The loss factor and capacitance curves] of [ machine_component all three phases] follow [ symptom a similar course], which indicates [ cause an uniform aging process] for [ machine_component the stator winding insulation]. [ measured_quantity The loss factor curves] show in the first measurement step [ symptom a negative course], which indicates [ cause detaching between the insulation and the copper]. This will lead during operation to [ symptom corona discharges] in [ machine_component the stator terminal], which, nevertheless, are of secondary importance, due to [ cause the relatively low nominal voltage].
  </evaluation>
  <action> Ongoing observation of [ measured_quantity the loss factor curves] in following inspections.
  </action>
</measurement>

Fig. 1. An annotated textual case of a diagnostic measurement.

in dem Gebiet

The reason for annotating the text in Figure 1 is not only to show the presence of (problem description, problem solution) in the <evaluation> part of a textual case, but also to highlight the kind of language used to express domain meaning. Since we are considering text related to diagnostic evaluations, language expressions that can be annotated as: measured_quantity (diagnosis vehicle), machine_component (diagnosis object), symptom, or cause, will appear almost in every sentence. We refer to such annotations as knowledge roles, a term borrowed from the CommonKADS methodology [7]. It is important to mention that by attaching a layer of annotated knowledge roles to textual cases, the problem of retrieval redundancy that appears when querying only by domain concepts (used as keywords) could be avoided. For example, the domain concept insulation resistance appears in both of the following sentences’ excerpts: “...insulation resistance values are in the expected range ...” and
“...[symptom, lower insulation resistance values] were ascertained due to ...”, but from the diagnostic viewpoint we are interested in those cases where this term is found inside a symptom expression.

Let us suppose that the whole collection of documents is tagged with such semantic annotations. Combined with the fact that sentences appear in defined contexts (the XML markups separating different kind of measurements), such annotations could be beneficial in several scenarios:

– Establish statistical domain knowledge based on evidence in cases. Example: “In X % of cases, symptom Y appears in machine component Z”. This can be achieved by using association rules for text mining.
– Generate hypothesis sets for differential diagnosis. Example: “When encountering symptom S look for causes C1, C2, ..., Cn.”
– Question answering. Example: “What can influence surface conductivity?”

Although these scenarios cross the usual borders of case based reasoning, they can be made possible only as the result of the existence of textual cases and their representation through semantic annotations combined with similarity measures based on domain ontologies. Since such a case representation is missing, we face the problem of finding ways to automatically annotate the textual cases. A solution to this problem is presented in the next section.

3 A Framework for Learning Semantic Annotations

3.1 Semantic Roles in Natural Language Learning

Our approach for learning to annotate knowledge roles is inspired by approaches for semantic role labeling in the natural language learning domain. In 2004 and 2005, semantic role labeling has been the shared task in the CoNLL conference with the aim “to come up with machine learning strategies which address the semantic role labeling problem on the basis of only partial syntactic information, avoiding the use of full parsers and external lexico-semantic knowledge bases” [8].

In this context, a semantic role is the relationship that a syntactic constituent has with a predicate. Two annotated corpora exist to support research in semantic role labeling. The one used in the CoNLL task is the PropBank corpus\(^3\), an enrichment of the Penn Treebank with predicate-argument structures. The other one is the FrameNet corpus\(^4\) that is based on frame semantics. PropBank uses predicate independent labels (A0-A5, A6), where for example, A0 stands for the agent, A1 for the patient or theme, etc. Instead, FrameNet uses human-friendly, predicate dependent labels, for example, in the frame Reading\(^5\) the labels are reader, text, means, place, etc.

\(^3\) http://www.cis.upenn.edu/ace/  
\(^4\) http://framenet.icsi.berkeley.edu/  
\(^5\) This frame can be found at http://framenet.icsi.berkeley.edu/index.php?option=com_wrapper&Itemid=118&frame=Reading
The task of learning semantic roles for naturally occurring, domain independent text—as the text in the CoNLL challenge—is very difficult. We believe that trying the same task in a specific domain reduces its complexity (far fewer predicates and roles, less ambiguity, etc.) and increases the chances for better results. Nevertheless, we still have to face the difficulty of lacking an annotated corpus for supervised learning. For this purpose, we will use an active learning approach, where the user is asked to annotate the examples with the highest anticipated value.

3.2 Verbs and Semantic Verb Classes

For the semantic role labeling task, the predicate (the verb) of the sentence is given as an input, because the central interest is in labeling its arguments. Indeed, verbs determine the number and kinds of arguments that may appear in a sentence, besides influencing the meaning and the structure of the sentence. There has been much research in computational linguistic for establishing semantic verb classes, where verbs grouped in a class behave similarly (they may take arguments with the same meaning), refer to [9] for English verbs and [10] for German verbs. For example, the class Observation in the German language groups the verbs: bemerken, erkennen, erfahren, feststellen, realisieren, registrieren, and ermitteln.

Verbs play an important role in our annotation framework. First, the types of knowledge roles that can be learned were defined based on the most frequent verbs appearing in our collection of textual cases. Furthermore, we used the semantic verb classes for including synonymous verbs with lower frequency. In Table 1, relations between syntactic arguments (subject, direct object, and complements) and some domain-specific knowledge roles (Finding, Location, Measurement, etc.) for our textual cases are given for several of the most frequent German verbs in our collection.

<table>
<thead>
<tr>
<th>Verb</th>
<th>Subject</th>
<th>Direct Object</th>
<th>Complements</th>
</tr>
</thead>
<tbody>
<tr>
<td>feststellen</td>
<td>(passive voice) Finding</td>
<td>Location</td>
<td>(e.g., symptom, cause, condition)</td>
</tr>
<tr>
<td>(to detect)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ergeben</td>
<td>Measurement,</td>
<td>Finding</td>
<td>–</td>
</tr>
<tr>
<td>(to result)</td>
<td>Measured_quantity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>hindeuten</td>
<td>Finding</td>
<td>–</td>
<td>Finding</td>
</tr>
<tr>
<td>(to point to)</td>
<td>(e.g., condition,</td>
<td></td>
<td>(e.g., symptom, cause)</td>
</tr>
<tr>
<td></td>
<td>symptom)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following annotated German sentences exemplify the relation between the verbs in Table 1 and the attached knowledge roles.
[Location Im Nutaustrittbereich] wurden [Finding stärkere Glimmentladungsspuren] festgestellt. (Translation: In the slot exit area stronger signs of corona discharge were detected.)

[MeasuredQuantity Die gemessenen Ladestromwerte] ergaben [Finding einen horizontalen Kurvenverlauf], der [Finding auf leitende Verschmutzungen der Hauptisolierung] hindeutet. (Translation: The measured charging current values resulted in a horizontal curve, which points to conducting dirtiness in the main insulation.)

In order to keep the learning process simple (lack of annotated training instances), knowledge roles like symptom, cause, or condition are all annotated with the upper-concept Finding, with the specialization taking place in a second phase. The next subsection will explain how the domain knowledge (expressed in the dependency of knowledge role on the verbs), natural language processing, and machine learning will be combined in a learning framework to annotate the textual cases.

3.3 Framework Components

The learning framework has the architecture shown in in Figure 2 and in the following we describe the functionality of each component.

PoS Tagging. TreeTagger\(^6\) was used for part-of-speech (PoS) tagging of German text. During the PoS tagging process, stems are also attached to every word. Since we deal with text in a restricted, technical domain, a large number of words were marked as \(<\text{unknown}>\). From these domain specific words, we created a lexical resource to provide the correct stems and to be used later for constructing a domain ontology.

Chunking & Parsing. A German parser tool could not be included in the framework (the parser could not produce results for every sentence); therefore, we used only chunking performed again by the TreeTagger. The tool can tag

\(^6\) http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/
only 3 types of chunks: noun phrases (NP), verb phrases (VP), and prepositional phrases (PP).

**Feature Creation.** The learning algorithm needs features that will be created from the words in the sentence, their PoS and chunk tags, as well as the structure of the sentence. A large number of features for the task of semantic role labeling has been presented in [11] and [12]. The features we used in our learning framework are: stems, PoS tags, phrase type, chunk sequence, target verb, relative position to the target verb, and local context.

**Active Learning Strategy.** The active learning strategy needs user interaction. The aim is to annotate training instances for the learning algorithm keeping the burden on the user low. The strategy is implemented in two phases: a) active initialization and b) active learning. In the active initialization phase, the pool of sentences containing a desired target verb (or a target semantic verb class) is divided into clusters that have an identical or similar value for the chunk sequence feature (a chunk sequence represents a sentence with the labels of its chunk components, e.g., 'PP NP PP NP VP NP VP'). A few sentences from the largest clusters are selected and presented to the user for labeling. Then the mappings “class label”-“chunk sequence components” are spread throughout the sentences of the selected clusters, creating in this way a large group of annotated sentences. In the next phase of active learning, the previously annotated sentences are passed to the learning algorithm to train the classifier. After the trained classifier performs the labeling, some of the sentences where the class label probabilities are splitted among several classes are presented to the user for relabeling, while the sentences where all labels have probability 1.0 are added to the pool of training instances. Phase b) is then repeated.

**Learning Algorithm.** We selected maximum entropy as a learning approach [13]. Besides being a successful learning approach widely used in natural language processing, a maximum entropy classifier provides probabilities for its confidence in each class label, and we use this information for the active learning strategy.

### 3.4 Results and Discussion

We present here only learning results for the knowledge roles **Location** and **Finding** of the most frequent verb feststellen (to detect) (refer to Table 1). We extracted 940 distinct sentences containing this verb and selected 2 sentences (based on the most frequent prepositions with different meanings) from each of the 5 largest clusters created during active initialization. These 10 sentences were annotated by the user, and during the spreading process a total of 172 sentences were then automatically annotated. For testing we left aside 10 of these sentences and additionally annotated manually 40 other sentences selected from the remaining clusters. It should be noted that some errors were introduced in the label-spreading phase, which for the moment—until we improve the active initialization strategy—were corrected manually. The accuracy of the classifier is calculated as the proportion of the correctly labeled chunk phrases to the total number of chunk phrases in the testing set. In the first iteration the accuracy amounted to 61%, and improved to 73% in the 3 further iterations, while the
user was asked to relabel only 10 new sentences per iteration. We consider an accuracy of 73% with a user contribution of only 40 annotated instances as a promising result. But there are several ways for improvement, besides further iterations/relabelling. One of these ways is, for example, a pre-learning phase for the recognition of named entities (in the described domain, named entities are, for example, the Measured Quantity, Machine Component, etc.) that will be used as an additional feature for learning. Finally, it should be mentioned that German language tools are not yet as advanced as English language tools and introduce a larger number of errors that decrease the accuracy of the learning process.

4 Related Work

As previously mentioned, in [4] six different systems related to TCBR were analyzed. Each of the described systems has unique and interesting characteristics. For example, FAQFinder uses WordNet for calculating semantic similarities, DRAMA uses noun phrases to build vectors for indexing, or PRUDENCIA uses the structure of the documents (division in subsections) to transform them to structured cases annotated with some predefined metadata. We examine in more detail SMILE and CBRANSWERS, since they are more relevant to our work.

SMILE [14] stands out from other approaches for its extensive use of machine learning techniques to many aspects of TCBR. In SMILE, some domain relevant, abstract concepts called factors are used to annotate whole sentences. In our view, these factors display a higher abstraction level compared to our semantic annotation, a quality that makes them more difficult to learn. In further work, the SMILE authors advocated the necessity of using state-of-the-art NLP/IE methods for processing textual cases [15]. They introduced a new feature, Propositional Patterns (ProPs), which could be useful for preserving meaning and generalization. Nevertheless, an approach how to derive ProPs automatically from text was not presented.

CBRANSWERS [16] was used to build several real-word applications of TCBR, like SIMATIC or FALLQ. CBRANSWERS introduced several novelties to CBR: information completion, information entities, and case-retrieval nets, which could be used for both structured and textual cases. An important contribution of this work is the emphasis on knowledge engineering for case representation, which consists of several knowledge layers. However, at the time of publication of this work, the author viewed advanced NLP techniques behind Part-of-Speech as not powerful enough (lack of robustness, lack of handling a large corpus, difficulties of integration in CBR systems, etc.) to be used extensively for case acquisition. We think that in the meantime many of the reasons for such skepticism have become outdated.

CBR is used very frequently for diagnosis tasks, although TCBR approaches in this domain are in minority. In [5], we have already identified the differences of

---

7 Natural Language Processing (NLP), Information Extraction (IE)
our domain problem to that of the Cassiopee system, which helps troubleshooting the CFM 56-3 engines of the Boeing 737. Varma describes in [17] a TCBR project in which hundred megabytes of archived text notes generated by field engineers in the domain of servicing medical imaging machines were “mined” with the purpose of automatically distinguishing good text notes from bad ones in a diagnostic sense. Some differences to our problem domain are: the conciseness of the used language, documents are informal notes without structure, and relevant keywords instead of higher concepts are used as case features for indexing. Nevertheless, the use of a supervised learning algorithm based on n-grams with a minimal user contribution for annotating the cases as useful or not is a valuable approach.

Recently, a new approach for handling textual cases was presented in the field of Conversational CBR. Gupta and Aha tackle the problem of index extraction methods for domain independent text documents and propose FACIT (a framework for Feature Acquisition and Case Indexing from Text)[18], which is based on generative semantic lexicons (also called sublanguage or generative ontologies) and represents the extracted features in a logical form that allows for a subsequent automatic induction of subsumption taxonomies that will serve to index the cases [19]. We share several ideas with this work: the use of deep NLP techniques (like full parsing) or domain ontologies; the similarity between their features and relations to our semantic annotations; and also perhaps the generative approach (although it is seen in literature [20] as not as tractable as the discriminative approach). Our work differs from that of Gupta et al. in its scope and the followed procedure, since we do not have domain lexical resources and will create a domain ontology after having semantically annotated the textual cases.

5 Conclusions

In this paper, we have presented a machine learning approach, combined with domain knowledge, natural language processing, and user participation for annotating textual cases. Annotation of textual cases with domain relevant concepts (also called knowledge roles) is a form for conceptual case representation, which allows the reuse of case knowledge (besides TCBR) for several other knowledge tasks, like question answering and text mining. The presented learning framework, inspired by semantic role labeling in natural language learning, showed promising results that give us confidence in the validity of the approach. The addition of new features, like named entities of the domain, could further improve the results.

Nevertheless, further work is needed to integrate the proposed case representation in the cycle of case-based reasoning. While such representation can be already used for unrestrained retrieval of cases, similarity measures—based, for example, on a domain and application ontology—that will restrain retrieval, need to be implemented. In a last step, the textual cases will have to be integrated into the existing CBR system of structured cases.
References