

# Mining Diagnostic Text Reports by Learning to Annotate Knowledge Roles

Eni Mustafaraj, Martin Hoof, and Bernd Freisleben

## 1.1 Introduction

Several tasks approached by using text mining techniques, like text categorization, document clustering, or information retrieval, operate on the document level, making use of the so called bag-of-words model. Other tasks, like document summarization, information extraction, or question answering, have to operate on the sentence level, in order to fulfill their specific requirements. While both groups of text mining tasks are typically affected by the problem of data sparsity, this is more accentuated for the latter group of tasks. Thus, while the tasks of the first group can be tackled by statistical and machine learning methods based on a bag-of-words approach alone, the tasks of the second group need natural language processing (NLP) at the sentence or paragraph level in order to produce more informative features.

Another issue common to all previously mentioned tasks is the availability of labeled data for training. Usually, for documents in real world text mining projects, training data do not exist or are expensive to acquire. In order to still satisfy the text mining goals while making use of a small contingent of labeled data, several approaches in machine learning have been developed and tested: different types of active learning Jones et al. [2003], bootstrapping Ghani and Jones [2002], or a combination of labeled and unlabeled data Blum and Mitchell [1998]. Thus, the issue of the lack of labeled data turns into the issue of selecting an appropriate machine learning approach.

The nature of the text mining task as well as the quantity and quality of available text data are other issues that need to be considered. While some text mining approaches can cope with data noise by leveraging the redundancy and the large quantity of available documents (for example, information retrieval on the Web), for other tasks (typically those restricted within a domain) the collection of documents might not possess such qualities. Therefore, more care is required for preparing such documents for the text mining task.

The previous observations suggest that performing a text mining task on new and unknown data requires handling all of the above mentioned issues, by combining and adopting different research approaches. In this paper, we present an approach to extract knowledge from text documents containing diagnostic problem solving situations in a technical domain (i.e. electrical engineering). In the proposed approach, we have combined techniques from several areas, including NLP, knowledge engineering, and machine learning to implement a learning framework for annotating cases with knowledge roles. The ultimate goal of the approach is to discover interesting problem solving situations (hereafter simply referred to as cases) that can be used by an experience management system to support new engineers during their working activities. However, the performed annotations facilitate the retrieval of cases on demand, allow collecting empirical domain knowledge, and can be formalized with the help of an ontology to also permit reasoning. The experimental results presented in the paper are based on a collection of 500 Microsoft Word documents written in German, amounting to about one million words. Several processing steps were required to achieve the goal of case annotation. In particular, we had to (a) transform the documents in an XML format, (b) extract paragraphs belonging to cases, (c) perform part-of-speech tagging, (d) perform syntactical parsing, (e) transform the results into XML representation for manual annotation, (f) construct features for the learning algorithm, and (g) implement an active learning strategy. Experimental results demonstrate the feasibility of the learning approach and a high quality of the obtained annotations.

The paper is organized as follows. In Section 1.2 we describe our domain of interest, the related collection of documents, and how knowledge roles can be used to annotate text. In Section 1.3 we consider work in natural language processing, especially frame semantics and semantic role labeling, emphasizing parallels to our task and identifying how resources and tools from these domains can be applied to perform annotation. Section 1.4 describes in detail all the preparatory steps for the process of learning to annotate cases. Section 1.5 evaluates the results of learning. Section 1.6 concludes the paper and outlines areas of future work.

## 1.2 Domain Knowledge And Knowledge Roles

### 1.2.1 Domain Knowledge

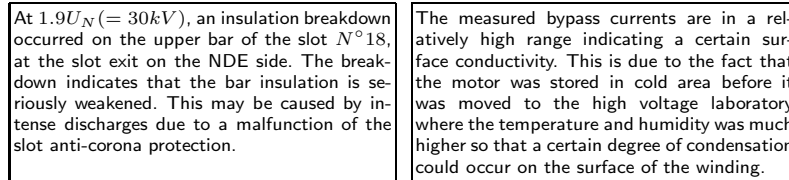
Our domain of interest is predictive maintenance in the field of power engineering, more specifically, the maintenance of insulation systems of high voltage rotating electrical machines. Since in many domains it is prohibitive to allow faults that could result in a breakdown of the system, components 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. Usually, the findings related to the predictive maintenance process are 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 photographs. The focus of the work described here are the textual diagnostic reports.

In the domain 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). As part of their business agreement, the service provider submits to the customer an *official diagnostic report*. Such a report follows a predefined structure template and is written in syntactically correct and parsimonious language. In our case, the language is German.

A report is organized into many sections: summary, reason for 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 photographs of damaged parts.

From a diagnostic point of view, the most important information is found in the evaluations of the measurements and tests performed. As a demonstration, consider the two excerpts in Figure 1.1 (originating from English documents for non-German speaking customers).



**Fig. 1.1.** Excerpts from two evaluations of isolation current measurements.

As it is often the case with diagnosis, while the quantities that are measured or the components that are inspected are the same, the findings depend on a series of contextual factors, and the reasons for these findings could be quite unique (as the examples of Figure 1.1 demonstrate). Usually, human experts need many years of field experience to gain a degree of expertise that allows them to handle any situation. The goal of our project is to mine the text documents for relevant pieces of knowledge acquired during diagnostic problem solving situations.

### 1.2.2 Domain Concepts

In some text mining applications, such as text categorization or information retrieval, the goal is often to discover terms specific to the domain that could

be used as indices for organizing or retrieving information. Indeed, the excerpts of Figure 1.1 contain several of such domain-specific terms: *insulation*, *discharge*, *slot anti-corona protection*, *conductivity*, or *winding*. Still, using these terms as indices or keywords for representing the documents does not contribute to the purpose of our intended application, which is to find knowledge that supports diagnostic problem solving. To exemplify, consider the sentences in Figure 1.2:

- |   |
|---|
| <p>1) The calculated <b>insulating resistance</b> values lay in the safe operating area.<br/> 2) Compared to the last examination, lower values for the <b>insulating resistance</b> were ascertained, due to dirtiness at the surface.</p> |
|---|

**Fig. 1.2.** Two sentences with the same domain concept shown in boldface.

In both sentences, the domain concept *insulating resistance* is found, but from a diagnostic point of view only the second sentence is interesting, because it describes a possible cause for lower values. Thus, more than domain concepts are needed to capture the knowledge expressed in the documents. Our solution to this problem is to label the text with semantic annotations expressed in terms of knowledge roles, which are introduced in the following subsection.

### 1.2.3 Knowledge Roles

Knowledge roles are a concept introduced in CommonKADS Schreiber et al. [2000], a knowledge engineering methodology for implementing knowledge-based systems. More specifically, knowledge roles are abstract names that refer to the role a domain concept plays when reasoning about a knowledge task. Such tasks are, for example, diagnosis, assessment, monitoring, or planning. Although these tasks are found in many domains, their description in CommonKADS is domain-independent. Thus, when describing a diagnosis task, knowledge roles like *finding*, *symptom*, *fault*, *parameter*, or *hypothesis* would be used.

Indeed, if we consider again the sentences in Figure 1.2, it is reasonable to represent the second sentence with knowledge roles as shown in Figure 1.3:

Knowledge Role	Text Phrase
<i>Observed Object:</i>	insulating resistance
<i>Symptom:</i>	lower values
<i>Cause:</i>	dirtiness at the surface

**Fig. 1.3.** Knowledge roles for sentence 2 of Figure 1.2.

Such a representation can have several advantages. Given a certain value of an *Observed Object*, a list of *Symptoms* that should be checked during the diagnosis could be retrieved. Or, given a certain *Symptom*, possible *Causes* for it could be listed, and so forth.

Understandably, we are interested in performing the text annotation with knowledge roles automatically. To achieve this goal, we draw on research in natural language understanding as described in Section 1.3.

It might be argued that one could simply use a combination of keywords to retrieve the information. For example, for sentences like that in Figure 1.2, one might write a query as below:

[low | small | high | large] && [value] && [insulating resistance]

for retrieving symptoms. Or one can search for:

[due to] | [caused by] | [as a result of] ...

to retrieve sentences containing causes. While this approach may be appealing and in some occasions even successful, there are several reasons why it could not be applied in our application:

- A large number of words (adjectives, nouns, adverbs, or verbs) can be used to describe changes (considered as symptoms in our domain), and no one can know beforehand which of them is used in the text.
- While verbs are very important for capturing the meaning of a sentence, they also abound in numbers. For example, to express an observation, any of the following verbs can be used: *observe*, *detect*, *show*, *exhibit*, *recognize*, *determine*, *result in*, *indicate*, etc. Furthermore, adverbs and negations can change their meaning and therefore need to be considered. Thus, instead of using verbs as keywords, we use them to bootstrap the annotating process, and incorporate them within semantic frames, like the frame *Observation* for the group above.
- Often, meaning emerges from the relation between different words, instead of the words separately, and this is exactly what we encountered in the diagnostic cases.

The knowledge roles used for annotating cases are abstract constructs in knowledge engineering, defined independently of any natural language constructs. Thus, a contribution of this work lies in trying to bridge the gap between knowledge roles and the natural language constructs whose meaning they capture. For this purpose, frame semantics, as described in the next section, is an ideal place to start.

## 1.3 Frame Semantics and Semantic Role Labeling

### 1.3.1 Frame Semantics

In frame semantics theory Fillmore [1976], a frame is a “script-like conceptual structure that describes a particular type of situation, object, or event and the participants involved in it” Ruppenhofer et al. [2005]. Based on this theory, the Berkeley FrameNet Project<sup>1</sup> is creating an online lexical resource for the English language by annotating text from the 100 million words British National Corpus.

The structure of a frame contains lexical units (pairs of a word with its meaning), frame elements (semantic roles played by different syntactic dependents), as well as annotated sentences for all lexical units that evoke the frame. An example of a frame with its related components is shown in Figure 1.4.

Annotation of text with frames and roles in FrameNet has been performed manually by trained linguists. An effort to handle this task automatically is being carried out by research in semantic role labeling, as described in the next subsection.

### 1.3.2 Semantic Role Labeling

Automatic labeling of semantic roles was introduced in Gildea and Jurafsky [2002]. In this work, after acknowledging the success of information extraction systems that try to fill in domain-specific frame-and-slot templates (see Section 1.3.4), the need for semantic frames that can capture the meaning of text independently of the domain was expressed. The authors envision that the semantic interpretation of text in terms of frames and roles would contribute to many applications, like question answering, information extraction, semantic dialogue systems, as well as statistical machine translation or automatic text summarization, and finally also to text mining.

After this initial work, research on semantic role labeling (SRL) has grown steadily, and in the years 2004 and 2005 Carreras and Màrquez [2004, 2005] a shared task at the CoNLL<sup>2</sup> was defined, in which several research institutions compared their systems. In the meantime, besides FrameNet, another corpus with manually annotated semantic roles has been prepared, PropNet Palmer and Gildea [2005], which differs from FrameNet in the fact that it has general semantic roles not related to semantic frames. PropNet is also the corpus used for training and evaluation of research systems on the SRL shared task. A similar corpus to FrameNet for the German language is been created by the Salsa project Erk et al. [2003a], and a discussion on the differences and similarities among these three projects is found in Ellsworth et al. [2004].

SRL is approached as a learning task. For a given target verb in a sentence, the syntactic constituents expressing semantic roles associated to this

<sup>1</sup> <http://framenet.icsi.berkeley.edu/>

<sup>2</sup> Conference of Natural Language Learning

Frame <i>Evidence</i>	
<u>Definition:</u> The <b>Support</b> , a phenomenon or fact, lends support to a claim or proposed course of action, the <b>Proposition</b> , where the <b>Domain_of_Relevance</b> may also be expressed.	
<u>Lexical units:</u> <i>argue.v, argument.n, attest.v, confirm.v, contradict.v, corroborate.v, demonstrate.v, disprove.v, evidence.n, evidence.v, evince.v, from.prep, imply.v, indicate.v, mean.v, prove.v, reveal.v, show.v, substantiate.v, suggest.v, testify.v, verify.v</i>	
<u>Frame Elements:</u>	
<b>Proposition</b> [PRP]	This is a belief, claim, or proposed course of action to which the Support lends validity.
<b>Support</b> [SUP]	Support is a fact that lends epistemic support to a claim, or that provides a reason for a course of action.
...	
<u>Examples:</u>	
And a [SUP sample tested] <b>REVEALED</b> [PRP some inflammation].	
It says that [SUP rotation of partners] does not <b>DEMONSTRATE</b> [PRP independence].	

**Fig. 1.4.** Information on the frame *Evidence* from FrameNet.

verb need to be identified and labeled with the right roles. SRL systems usually divide sentences word-by-word or phrase-by-phrase and for each of these instances calculate many features creating a feature vector. The feature vectors are then fed to supervised classifiers, such as support vector machines, maximum entropy, or memory-based learners. While adapting such classifiers to perform better on this task could bring some improvement, better results can be achieved by constructing informative features for learning. A thorough discussion of different features used for SRL can be found in Gildea and Jurafsky [2002], Pradhan et al. [2005].

### 1.3.3 Frames and Roles for Annotating Cases

On the one hand, in knowledge engineering there are knowledge tasks and knowledge roles to represent knowledge; on the other hand, in natural language understanding there are semantic frames and semantic roles to represent meaning. When knowledge related to a knowledge task (like diagnosis) is represented by natural language, it is reasonable to expect that some knowledge roles will map to some semantic roles. The question is how to find these mappings, and more importantly, how to label text with these roles?

A knowledge task like diagnosis or monitoring is not equivalent to a semantic frame. The former are more complex and abstract, and can usually be divided into several components, which in turn can be regarded equivalent to semantic frames. By analyzing the textual episodes of diagnostic evaluations, we noticed that they typically contain a list of observations, explanations based on evidence, and suggestions to perform some activities. Thus, we consulted FrameNet for frames like Observation, Change, Evidence, or Activity. Indeed, these frames are all present in FrameNet, for example, Activity is

present in 10 subframes, or different meanings of Change are captured in 21 frames. The frame Evidence was shown in Figure 1.4, and besides the two roles of Proposition and Support, it has also roles for Degree, Depictive, Domain\_of\_Relevance, Manner, Means, and Result. When one carefully reads the definition of the roles Proposition and Support and looks at the examples (Figure 1.4), one can conclude that Proposition is similar to Cause and Support to Symptom in a diagnosis task.

The problem is to determine which frames to look for, given that there are currently more than six hundred frames in FrameNet. The key are the lexical units related to each frame, usually verbs. Starting with the verbs, one gets to the frames and then to the associated roles. This is also the approach we follow. We initially look for the most frequent verbs in our corpus, and by consulting several sources (since the verbs are in German), such as im Walde [2003], VerbNet<sup>3</sup>, and FrameNet, we connect every verb with a frame, and try to map between semantic roles in a frame and knowledge roles we are interested in. One could also use the roles of FrameNet, but they are linguistically biased, and as such are not understandable by domain users that will annotate training instances for learning (a domain user would directly know to annotate *Cause*, but finds *Proposition* somehow confusing.)

In this work, FrameNet was only used as a lexical resource for consultation, that is, to find out which frames are evoked by certain lexical units, and what the related semantic roles are. Since the language of our corpus is German, we cannot make any statements about how useful the FrameNet frames could be to a learning system based on English annotated data corresponding to the defined frames.

Finally, it should be discussed why such an approach to annotating text cases with frames and roles could be beneficial to text mining. For the purpose of this discussion, consider some facts from the introduced domain corpus. During the evaluation of the learning approach, we manually annotated a subcorpus of unique sentences describing one specific measurement (high-voltage isolation current). In the 585 annotated sentences, the frame Evidence was found 152 times, 84 times evoked by the verb *zurückführen* (trace back to), 40 times by the verb *hindeuten* (point to), and 28 times by 9 other verbs. Analyzing the text annotated with the role *Cause* in the sentences with *zurückführen*, 27 different phrases expressing causes of anomalies pointed to by the symptoms were found. A few of these expressions appeared frequently, some of them occasionally, some others rarely. In Table 1.1, some of these expressions are shown.

If for every sentence with the frame *Evidence* the text annotated with *Symptom* and *Cause* is extracted, this text can then be processed further with other text mining techniques for deriving domain knowledge, which is not available in any of the analyzed texts alone. For example, one could get answers to questions like: which are the most frequent symptoms and what

---

<sup>3</sup> <http://www.cis.upenn.edu/~bsnyder3/cgi-bin/search.cgi>

**Table 1.1.** Some phrases annotated with the role *Cause*.

German Phrase	English Translation	Frequency
Verschmutzungseinflüsse	influences of pollution	10
leitende Verschmutzungen	conducting pollutions	8
Ionisation in Klemmenbereich	ionization in the terminal area	3
äussere Entladungen	external discharges	1

causes can explain them; what problems (i.e. causes) do appear frequently in a specific type of machine, etc. Thus, such an annotation with frames and roles preprocesses text by generating very informative data for text mining, and it can also be used in the original form for information retrieval. Still, such an approach makes sense in those cases when text contains descriptions of repetitive tasks, which are then expressed by a small number of underlying semantic frames. Since data and text mining try to extract knowledge from data of the same nature in the same domain, we find that annotation of text with knowledge roles could be a valuable approach.

Before explaining in detail the process of learning to automatically annotate text with knowledge roles (based on the SRL task) in Section 4, we briefly discuss the related field of information extraction.

### 1.3.4 Information Extraction

Information extraction (IE), often regarded as a restricted form of natural language understanding, predates research in text mining, although today, IE is seen as one of the techniques contributing to text mining Weiss et al. [2004]. Actually, the purpose of IE is very similar to what we are trying to achieve with role annotation. In IE it is usually known in advance what information is needed, and part of text is extracted to fill in slots of a predefined template. An example, found in Mooney and Bunescu [2005], is the job posting template, where, from job posting announcements in Usenet, text to fill slots like: title, state, city, language, platform, etc. is extracted and stored in a database for simpler querying and retrieval.

Usually, methods used by IE have been based on shallow NLP techniques, trying to extract from a corpus different types of syntactic rules that match syntactic roles to semantic categories, as for example in Riloff and Schelzenbach [1998].

With the advances in NLP and machine learning research, IE methods have also become more sophisticated. Actually, SRL can also be seen as a technology for performing information extraction, in those cases when text is syntactically and semantically more demanding and expressive. All these technologies are intended to be used for extracting knowledge from text, despite their differences in implementation or scope.

## 1.4 Learning to Annotate Cases with Knowledge Roles

To perform the task of learning to annotate cases with knowledge roles, we implemented a software framework, as shown in Figure 1.5. Only the preparation of documents (described in Section 1.4.1) is performed outside of this framework. In the remainder of the section, every component of the framework is presented in detail.

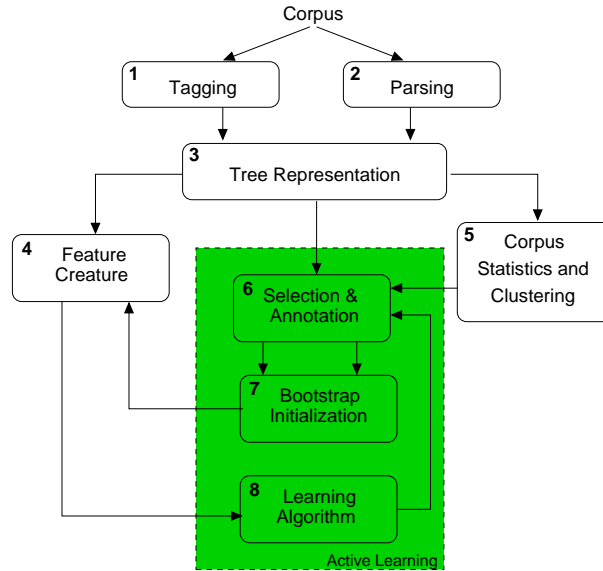


Fig. 1.5. The Learning Framework Architecture.

### 1.4.1 Document Preparation

In Section 1.2.1 it was mentioned that our documents are official diagnostic reports hierarchically structured in several sections and subsections, written by using MS<sup>®</sup> Word. Actually, extracting text from such documents, while preserving the content structure, is a difficult task. In completing it we were fortunate twice. First, with MS<sup>®</sup> Office 2003 the XML based format WordML was introduced that permits storing MS<sup>®</sup> Word documents directly in XML. Second, the documents were originally created using a MS<sup>®</sup> Word document template, so that the majority of them had the same structure. Still, many problems needed to be handled. MS<sup>®</sup> Word mixes formatting instructions with content very heavily and this is reflected also in its XML format. In addition, information about spelling, versioning, hidden template elements, and so on are also stored. Thus, one needs to explore the XML output of

the documents to find out how to distinguish text and content structure from unimportant information. Such a process will always be a heuristic one, depending on the nature of the documents. We wrote a program that reads the XML document tree, and for each section with a specified label (from the document template) it extracts the pure text and stores it in a new XML document, as the excerpt in Figure 1.6 shows.

```

<section title="Measurements">
  <subsection title="Stator_Winding">
    <measurement title="Visual_Control">
      <submeasurement title="Overhang_Support">
        <evaluation>
          Die Wickelkopfabsttzung AS und NS befand sich in einem ...
        </evaluation>
        <action>Keine</action>
      </submeasurement>
    ...
  
```

**Fig. 1.6.** Excerpt of the XML representation of the documents.

Based on such an XML representation, we create subcorpora of text containing measurement evaluations of the same type, stored as paragraphs of one to many sentences.

#### 1.4.2 Tagging

The part-of-speech (POS) tagger (TreeTagger<sup>4</sup>) that we used Schmid [1995] is a probabilistic tagger with parameter files for tagging several languages: German, English, French, or Italian. For some small problems we encountered, the author of the tool was very cooperative in providing fixes. Nevertheless, our primary interest in using the tagger was not the POS tagging itself (the parser, as is it shown in Section 1.4.3 performs tagging and parsing), but getting stem information (since the German language has a very rich morphology) and dividing the paragraphs in sentences (since the sentence is the unit of operation for the next processing steps).

The tag set used for tagging German is slightly different from that of English<sup>5</sup>. Figure 1.7 shows the output of the tagger for a short sentence<sup>6</sup>.

As indicated in Figure 1.7, to create sentences it suffices to find the lines containing: ". \\$. ." (one sentence contains all the words between two such lines). In general, this is a very good heuristic, but its accuracy depends on the nature of the text. For example, while the tagger correctly

<sup>4</sup> <http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger>

<sup>5</sup> <http://www.ims.uni-stuttgart.de/projekte/corplex/TagSets/stts-table.html>

<sup>6</sup> Translation: A generally good external winding condition is present.

Es	PPER	es
liegt	VVFIN	liegen
insgesamt	ADV	insgesamt
ein	ART	ein
guter	ADJA	gut
äusserer	ADJA	äußer
Wicklungszustand	NN	<unknown>
vor	PTKVZ	vor
.	\$.	.

**Fig. 1.7.** A German sentence tagged with POS-tags by TreeTagger.

tagged abbreviations found in its list of abbreviations (and the list of abbreviations can be customized by adding abbreviations common to the domain of the text), it got confused when the same abbreviations were found inside parentheses, as the examples in Figure 1.8 for the word ‘ca.’ (circa) show.

If such phenomena occur often, they become a problem for the further correct processing of sentences, although one becomes aware of such problems only in the course of the work. A possible solution in such cases is to use heuristics to replace erroneous tags with correct ones for the types of identified errors.

an	APR	an	(	\$(	(
ca.	ADV	ca.	ca	NE	<unknown>
50	CARD	50	.	\$.	.
%	NN	%	20	CARD	20

**Fig. 1.8.** Correct and erroneous tagging for the word ‘ca.’

The more problematic issue is that of words marked with the stem <unknown>. Actually, their POS is usually correctly induced, but we are specifically interested in the stem information. The two reasons for an <unknown> label are a) the word has been misspelled and b) the word is domain specific, and as such not seen during the training of the tagger. On the positive side, selecting the words with the <unknown> label directly creates the list of domain specific words, useful in creating a domain lexicon.

A handy solution for correcting spelling errors is to use a string similarity function, available in many programming language libraries. For example, the Python language has the function “get\_close\_matches” in its “difflib” library. An advantage of such a function is having as a parameter the degree of similarity between strings. By setting this value very high (between 0 and 1) one is sure to get really similar matches if any at all.

Before trying to solve the problem of providing stems for words with the <unknown> label, one should determine whether the stemming information substantially contributes to the further processing of text. Since we could not know that in advance, we manually provided stems for all words labeled as

<unknown>. Then, during the learning process we performed a set of experiments, where: a) no stem information at all was used and b) all words had stem information (tagger + manually created list of stems). Table 1.2 summarizes the recall and precision of the learning task in each experiment.

**Table 1.2.** Results of experiments for the contribution of stem information on learning.

Experiment	Recall	Precision
a) no stems (only words)	90.38	92.32
b) only stems	91.29	93.40

These results show approximately 1% improvement in recall and precision when stems instead of original words are used. We can say that at least for the learning task of annotating text with knowledge roles stem information is not necessarily important, but this could also be due to the fact that a large number of other features (see Section 1.4.5) besides words are used for learning.

Still, the reason for having a list of stems was not in avoiding more data due to word inflections, but in capturing the word composition, a phenomenon typical for the German language. For example, all the words in the first row of Table 1.3 are compound words that belong to the same semantic category identified by their last word ‘wert’ (value), i.e. they all denote values of different measured quantities, and as such have a similar meaning. This similarity cannot be induced if one compares the words in the original form, something possible by comparing the word representations of the second row.

**Table 1.3.** Original words (first row), words composed of stems (second row).

Ableitstromwerte, Gesamtstromwerte, Isolationswiderstandswerte, Isolationstromwerte, Kapazitätswerte, Ladestromwerte, Stromwerten, Verlustfaktor-anfangswert, etc.		
Ableit-Strom-Wert,	Gesamt-Strom-Wert,	Isolation-Widerstand-Wert,
Isolation-Strom-Wert,	Kapazität-Wert,	Lade-Strom-Wert,
Verlustfaktor-Anfang-Wert,	etc.	

Unfortunately, there are only a few tools available for morphological analysis of German words. We tried Morphy Lezius [2000], which is publicly available, but it was not able to analyze any of our domain-specific words. Therefore, we had to perform this task by hand.

### 1.4.3 Parsing

Syntactical parsing is one of the most important steps in the learning framework, since the produced parse trees serve as input for the creation of features used for learning. Since we are interested in getting qualitative parsing results, we experimented with three different parsers: the Stanford parser (Klein 2005), the BitPar parser Schmid [2004], Schiehlen [2004], and the Sleepy parser Dubey [2003]. What these parsers have in common is that they all are based on unlexicalized probabilistic context free grammars (PCFG) Manning and Schütze [1999], trained on the same corpus of German, Negra<sup>7</sup> (or its superset Tiger<sup>8</sup>), and their source code is publicly available. Still, they do differ in the degree they model some structural aspects of the German language, their annotation schemas, and the information included in the output. Figure 1.9 shows the output of the same sentence<sup>9</sup> parsed by each parser, and in the following, we discuss each of them.

**Stanford Parser** - The Stanford parser is an ambitious project that tackles the task of generating parse trees from unlabeled data independently of the language. For the moment, the parser is distributed with parameter files for parsing English, German, and Chinese. We tested the parser on our data and noticed that the POS tags were often erroneously induced (in the sentence with only 8 words of Figure 1.9 there are 3 such errors—CARD tags for 2 nouns and 1 adjective), which then resulted in erroneous parse trees. But, in those cases when the tagging was performed correctly, the parse trees were also correct. Still, the parser could not parse long sentences, perhaps due to the fact that it was trained in the part of the Negra corpus with sentences having up to 10 words. Trying the parser with long English sentences instead, produced excellent results. We concluded that at this phase of implementation, the Stanford parser could not be used with our corpus of German sentences that contain an average of up to 18 words per sentence.

**BitPar Parser** - This parser is composed of two parts, the parser itself Schmid [2004] and the parameter files (chart rules, lexicon, etc.) from Schiehlen [2004]. Published experimental results claim robust performance, due to the use of sophisticated annotation and transformation schemata for modeling grammars. Another advantage of the parser is that its lexicon can be extended very easily with triples of domain-dependent words, their tags, their frequency counts in corpus, thus avoiding the tagging errors typical for unlexicalised parsers. These tagging errors damage the parse results, as can be seen from the results of the Stanford parser. Our critique for the described BitPar is that it usually produces trees with more nodes than

---

<sup>7</sup> <http://www.coli.uni-saarland.de/projects/sfb378/negra-corpus/>

<sup>8</sup> <http://www.ims.uni-stuttgart.de/projekte/TIGER/TIGERCorpus/>

<sup>9</sup> English translation: “On NS were detected circa 5 torn wedge’s safety bands.”

<i>Stanford Parser</i>	<pre>(ROOT (NUR (S   (PP (APPR Auf) (CARD NS))   (VAFIN wurden)   (VP     (AP (ADV ca.)       (NM (CARD 5)         (CARD gerissene)         (CARD Keilsicherungsbandagen)))     (VVPP festgestellt))) (\$..))</pre>
<i>BitPar Parser</i>	<pre>(utt: (S.fin:   (PP: (APPR: Auf)     (NN: NS))   (VWFN: wurden)   (AP: (AVP-MAD: (ADV-MAD: ca.))     (CARD: 5))   (NP.nom: (AP: (ADJA%: gerissene))     (NN.nom: Keilsicherungsbandagen))   (VVPP%: festgestellt)) (\\$.: .))</pre>
<i>Sleepy Parser</i>	<pre>(TOP (S   (PP-MO (APPR-AD Auf)     (NE-NK NS) )   (VAFIN-HD wurden)   (NP-SB     (ADV-MO ca.) (CARD-NK 5)     (ADJA-NK gerissene)     (NN-NK Keilsicherungsbandagen))   (VP-OC (VVPP-HD festgestellt))) (\$. .))</pre>
<p>Auf NS wurden ca. 5 gerissene Keilsicherungsbandagen festgestellt.  On NS were ca. 5 torn wedge's safety bands detected.</p>	

**Fig. 1.9.** Parsing output of the same sentence from the three parsers

the other parsers and the annotation of nodes contains specialized linguistic information, not very appropriate for creating features for learning.

**Sleepy Parser** - This parser has been specifically tuned for the German language, and while it is a statistical parser like the others, it uses different annotation schemas and incorporates grammatical functions (SB–subject, OC–clausal object, MO–modifier, HD–head, etc.) or long-distance dependencies between terms. In contrast to the two other parsers, it also has a highly tuned suffix analyzer for guessing POS tags Dubey [2005], which contributes to more accurate tagging results than the other parsers, al-

though some domain-dependent words are not always correctly tagged. Erroneous parsing is also encountered for very long sentences.

### *Choosing a Parser*

All the tested parsers make errors during parsing. In the end, the criteria upon which we based our choice of the parser were: speed and output information. Sleepy was the fastest and had the most informative output (it prints the log value expressing the likelihood of parsing, and it labels the majority of nodes with their grammatical function). Actually, choosing a parser upon these criteria instead of the accuracy of parsing could be regarded as inappropriate. Our justification is that a metric to measure the accuracy of parsing on new data does not exist. These parsers have all been trained on the same corpus, and at least the two German parsers tuned up to the point where their results are almost the same. Thus, *a priori* their expected accuracy in a new corpus should be equal, and accuracy is not a criterion for choosing one over the other. Given the difficulty of evaluating the accuracy of the parse trees and their presumed similarity, we based the choice of parser on the qualities that contributed most to our task, namely speed and informative output.

#### 1.4.4 Tree Representation

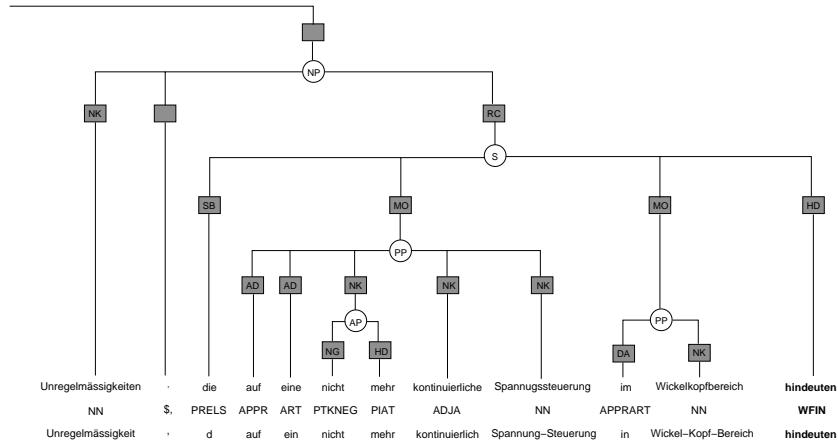
The bracketed parse tree and the stem information of tagging serve as input for the step of creating a tree data structure. The tree is composed of terminals (leaf nodes) and non-terminals (internal nodes), all of them known as constituents of the tree. For export purposes as well as for performing exploration or annotation of the corpus, the tree data structures are stored in XML format, according to a schema defined in the TigerSearch<sup>10</sup> tool. The created tree, when visualized in TigerSearch, looks like the one shown in Figure 1.10<sup>11</sup>. The terminals are labeled with their POS tags and also contain the corresponding words and stems; the inside nodes are labeled with their phrase types (NP, PP, etc.); and the branches have labels, too, corresponding to the grammatical functions of the nodes. The XML representation of a portion of the tree is shown in Figure 1.11.

#### 1.4.5 Feature Creation

Features are created from the parse tree of a sentence. A feature vector is created for every constituent of the tree, containing some features unique to the constituent, some features common to all constituents of the sentence, and some others calculated with respect to the target constituent (the predicate verb).

<sup>10</sup> <http://www.ims.uni-stuttgart.de/projekte/TIGER/TIGERSearch/>

<sup>11</sup> English translation: "...irregularities, which point to a not anymore continuous steering of voltage in the area of the winding head."



**Fig. 1.10.** Representation of a parsed tree in the TigerSearch tool. Due to space reasons, only a branch of the tree is shown.

```

...
<t lemma="Spannung-Steuerung" word="Spannungssteuerung" pos="NN"
  id="sentences._108_28" />
<t lemma="in" word="im" pos="APPRART"
  id="sentences._108_29" />
<t lemma="Wickel-Kopf-Bereich" word="Wickelkopfbereich" pos="NN"
  id="sentences._108_30" />
<t lemma="hindeuten" word="hindeuten" pos="VVFIN" id="sentences._108_31" />
</terminals>
<nonterminals>
<nt id="sentences._108_500" cat="PP">
  <edge idref="sentences._108_3" label="NK" />
  <edge idref="sentences._108_2" label="DA" />
  <edge idref="sentences._108_1" label="DA" />
</nt>
...

```

**Fig. 1.11.** XML representation of a portion of the parse tree from Figure 1.10.

A detailed linguistic description of possible features used by different research systems for the SRL task is found in Pradhan et al. [2005]. In this subsection, we only list the features used in our system and give example values for the leaf node **Spannungssteuerung** of the parse tree in Figure 1.10.

*Phrase type* **NN**  
*Grammatical function* **NK**  
*Terminal* (is the constituent a terminal or non-terminal node?) **1**  
*Path* (path from the target verb to the constituent, denoting u(up) and d(down) for the direction) **uSdPPd**  
*Grammatical path* (like Path, but instead of node labels, branches labels are considered) **uHDdMOdNK**  
*Path length* (number of branches from target to constituent) **3**  
*Partial path* (path to the lowest common ancestor between target and constituent) **uPPuS**  
*Relative Position* (position of the constituent relative to the target) **left**  
*Parent phrase type* (phrase type of the parent node of the constituent) **PP**  
*Target* (lemma of the target word) **hindeuten**  
*Target POS* (part-of-speech of the target) **VVFIN**  
*Passive* (is the target verb passive or active?) **0**  
*Preposition* (the preposition if the constituent is a PP) **none**  
*Head Word* (for rules on head words refer to Collins [1999]) **Spannung-Steuerung**  
*Left sibling phrase type* **ADJA**  
*Left sibling lemma* **kontinuierlich**  
*Right sibling phrase type* **none**  
*Right sibling lemma* **none**  
*Firstword, Firstword POS, Lastword, Lastword POS* (in this case, the constituent has only one word, thus, these features get the same values: Spannung-Steuerung and NN. For non-terminal constituents like PP or NP, first word and last word will be different.)  
*Frame* (the frame evoked by the target verb) **Evidence**  
*Role* (this is the class label that the classifier will learn to predict. It will be one of the roles related to the frame or none, for an example refer to Figure 1.12.) **none**

If a sentence has several clauses where each verb evokes a frame, the feature vectors are calculated for each evoked frame separately and all the vectors participate in the learning.

#### 1.4.6 Annotation

To perform the manual annotation, we used the Salsa annotation tool (publicly available) Erk et al. [2003b]. The Salsa annotation tool reads the XML representation of a parse tree and displays it as shown in Figure 1.12. The user has the opportunity to add frames and roles as well as to attach them to a desired target verb. In the example of Figure 1.12 (the same sentence of Figure 1.10), the target verb *hindeuten* (point to) evokes the frame Evidence, and three of its roles have been assigned to constituents of the tree. Such an assignment can be easily performed per point-and-click. After this process, an element <frames> is added to the XML representation of the sentence, containing information about the frame. Excerpts of the XML code are shown in Figure 1.13.

#### 1.4.7 Active Learning

Research in IE has indicated that using an active learning approach for acquiring labels from a human annotator has advantages over other approaches of selecting instances for labeling Jones et al. [2003]. In our learning framework, we have also implemented an active learning approach. The possibilities for designing an active learning strategy are manifold; the one we have implemented uses a committee-based classification scheme that is steered by corpus statistics. The strategy consists of the following steps:

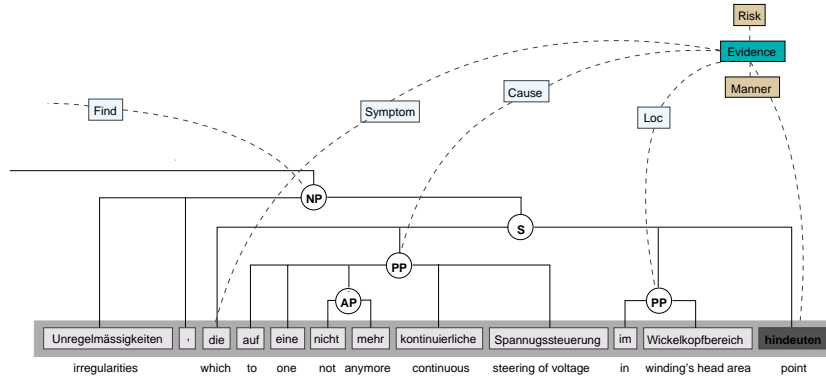


Fig. 1.12. Annotation with roles with the Salsa tool.

```

<frames>
  <frame name="Evidence" id="sentences._108__f1">
    <target><fenode idref="sentences._108_31"/></target>
    <fe name="Symptom" id="sentences._108_f1_e1">
      <fenode idref="sentences._108_22"/>
    </fe>
    <fe name="Cause" id="sentences._108_f1_e2">
      <fenode idref="sentences._108_509"/>
    </fe>
    <fe name="Loc" id="sentences._108_f1_e5">
      <fenode idref="sentences._108_510"/>
    </fe>
    ...
  </frame>
</frames>

```

Fig. 1.13. XML Representation of an annotated frame.

- a) Divide the corpus in clusters of sentences with the same target verb. If a cluster has fewer sentences than a given threshold, group sentences with verbs evoking the same frame into the same cluster.
- b) Within each cluster, group the sentences (or clauses) with the same parse sub-tree together.
- c) Select sentences from the largest groups of the largest clusters and present them to the user for annotation.
- d) Bootstrap initialization: apply the labels assigned by the user to groups of sentences with the same parse sub-tree.
- e) Train all the classifiers of the committee on the labeled instances; apply each trained classifier to the unlabeled sentences.
- f) Get a pool of instances where the classifiers of the committee disagree and present to the user the instances belonging to sentences from the next largest clusters not yet manually labeled.
- g) Repeat steps d)–f) a few times until a desired accuracy of classification is achieved.

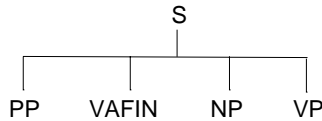
In the following, the rationale behind choosing these steps is explained.

*Steps a), b), c):* In these steps, statistics about the syntactical structure of the corpus are created, with the intention of capturing its underlying distribution, so that representative instances for labeling can be selected.

*Step d):* This step has been regarded as applicable to our corpus, due to the nature of the text. Our corpus contains repetitive descriptions of the same diagnostic measurements on electrical machines, and often, even the used language has a repetitive nature. Actually, this does not mean that the same words are repeated (although often standard formulations are used, especially in those cases when nothing of value was observed). Rather, the kind of sentences used to describe the task has the same syntactical structure. As an example, consider the sentences shown in Figure 1.14.

[ <sub>PP</sub> Im Nutaustrittsbereich] wurden [ <sub>NP</sub> stärkere Glimmentladungsspuren] festgestellt. <i>In the area of slot exit stronger signs of corona discharges were detected.</i>
[ <sub>PP</sub> Bei den Endkeilen] wurde [ <sub>NP</sub> ein ausreichender Verkeildruck] festgestellt. <i>At the terminals' end a sufficient wedging pressure was detected.</i>
[ <sub>PP</sub> An der Schleifringbolzenisolation] wurden [ <sub>NP</sub> mechanische Beschädigungen] festgestellt. <i>On the insulation of slip rings mechanical damages were detected.</i>
[ <sub>PP</sub> Im Wickelkopfbereich] wurden [ <sub>NP</sub> grossflächige Decklackablätterungen] festgestellt. <i>In the winding head area extensive chippings of the top coating were detected.</i>

**Fig. 1.14.** Examples of sentences with the same structure.



**Fig. 1.15.** Parse tree of the sentences in Figure 1.14.

What all these sentences have in common is the passive form of the verb *feststellen* (wurden festgestellt), and due to the subcategorization of this verb, the parse tree on the level of phrases is identical for all sentences, as indicated by 1.15. Furthermore, for the frame Observation evoked by the verb, the assigned roles are in all cases: NP—Finding, PP—Observed\_Object. Thus, to bootstrap initialization, we assign the same roles to sentences with the same sub-tree as the manually labeled sentences.

*Step e):* The committee of classifiers consists of a maximum entropy (Max-Ent) classifier from Mallet McCallum [2002], a Winnow classifier from SNoW Carlson et al. [2004], and a memory-based learner (MBL) from TiMBL Daelemans et al. [2004]. For the MBL, we selected  $k=5$  as the number of the nearest neighbours. The classification is performed as follows: if at least two classifiers

agree on a label, the label is accepted. If there is disagreement, the cluster of labels from the five nearest neighbours is examined. If the cluster is not homogenous (i.e. it contains different labels), the instance is included in the set of instances to be presented to the user for manual labeling.

*Step f)*: If one selects new sentences for manual annotation only based on the output of the committee-based classifier, the risk of selecting outlier sentences is high Tang et al. [2002]. Thus, from the instances' set created by the classifier, we select those belonging to large clusters not manually labeled yet.

## 1.5 Evaluations

To evaluate this active learning approach on the task of annotating text with knowledge roles, we performed a series of experiments that are described in the following. It was explained in Section 1.4.1 that based on the XML structure of the documents we created subcorpora with text belonging to different types of diagnostic tests. After such subcorpora have been processed to create sentences, only unique sentences are retained for further processing (repetitive, standard sentences do not bring any new information, they only disturb the learning and therefore are discarded). Then, lists of verbs were created, and by consulting the sources mentioned in Section 1.3.3, verbs were grouped with one of the frames: Observation, Evidence, Activity, and Change. Other verbs that did not belong to any of these frames were not considered for role labeling.

### 1.5.1 Learning Performance on the Benchmark Datasets

With the aim of exploring the corpus to identify roles for the frames and by using our learning framework, we annotated two different subcorpora and then manually controlled them, to create benchmark datasets for evaluation. Some statistics for the manually annotated subcorpora are summarized in Table 1.4. Then, to evaluate the efficiency of the classification, we performed 10-fold cross-validations on each set, obtaining the results shown in Table 1.5, where recall, precision, and the  $F_{\beta=1}$  measure are the standard metrics of information retrieval.

**Table 1.4.** Statistics for the benchmark datasets.

Subcorpus	Cases No.	Sentences No.	Unique Sentences No.	Annotated Roles No.
Isolation Current	491	1134	585	1862
Wedging System	453	775	602	1751

**Table 1.5.** Learning results for the benchmark datasets.

Subcorpus	Recall	Precision	$F_{\beta=1}$ measure
Isolation Current	0.913	0.934	0.92
Wedging System	0.838	0.882	0.86

We analyzed some of the classification errors and found that they were due to parsing anomalies, which had forced us in several occasions to split a role among several constituents.

### 1.5.2 Active Learning versus Uniform Random Selection

In order to evaluate the advantages of active learning, we compared it to the uniform random selection of sentences for manual annotations. Some results for both approaches are summarized in Table 1.6 and Table 1.7. Recall, precision, and  $F_{\beta=1}$  measure were calculated after each iteration, in which 10 new sentences manually labeled were added to the training set. The results of active learning ( $F_{\beta=1}$  measure) are 5–10 points better than those of random learning. For this experiment, the step d) of the active learning strategy was not applied, since it is very specific to our corpus.

**Table 1.6.** Random Learning Results.

Sentences No.	Recall	Precision	$F_{\beta=1}$ measure
10	0.508	0.678	0.581
20	0.601	0.801	0.687
30	0.708	0.832	0.765
40	0.749	0.832	0.788

**Table 1.7.** Active Learning Results.

Sentences No.	Recall	Precision	$F_{\beta=1}$ measure
10	0.616	0.802	0.697
20	0.717	0.896	0.797
30	0.743	0.907	0.817
40	0.803	0.906	0.851

### 1.5.3 Bootstrapping Based on Other Sets

During the annotation of the two benchmark datasets, we noticed that the two subcorpora, although different in nature (set\_1: *Isolation Current* contains evaluations of numerical measurements performed on the three phases of the machine, set\_2: *Wedging System* describes visual inspections on the wedging components of the machine) had very often the frame Observation or Change in common, while the frame Evidence appeared almost only in the first set, and the frame Activity almost always in the second. Thus, we tested whether text annotated with the same roles in one set could bootstrap the learning in the second, and the results are summarized in Table 1.8.

**Table 1.8.** Results for bootstrapping based on other labeled sets..

Training File	Testing File	Recall	Precision
Isolation Current	Wedging System	0.765	0.859
Wedging System	Isolation Current	0.642	0.737

We consider these results as very promising, since they give a hint at the possibility of using previously annotated text from other subcorpora to bootstrap the learning process, something that would alleviate the process of acquiring manual annotations for new text.

## 1.6 Conclusions

In this paper, we have presented an approach for extracting knowledge from text documents containing descriptions of knowledge tasks in a technical domain. Knowledge extraction in our approach is based on the annotation of text with knowledge roles (a concept originating in knowledge engineering), which we map to semantic roles found in frame semantics. The framework implemented for this purpose is based on deep NLP and active learning. Experiments have demonstrated a robust learning performance, and the obtained annotations were of high quality. Since our framework is inspired by and founded upon research in semantic role labeling (SRL), the results indicate that SRL could become a highly valuable processing step for text mining tasks.

In future work, we will consider the advantages of representing annotated text by means of knowledge roles and the related frames. Besides the previously explained uses for semantic retrieval of cases and the extraction of empirical domain knowledge facts, such a representation could also permit looking for potentially interesting relations in text and can be exploited to populate application and domain ontologies with lexical items.

## 1.7 Acknowledgements

The Insulation Competence Center of ALSTOM Ltd. Switzerland kindly permitted the use of the text documents for research purposes. Katrin Erk, Sebastian Pado, Amit Dubey, Sabine Schulte im Walde, Michael Schiehlen, and Helmut Schmid provided their linguistic tools and were an invaluable source of information and support. We are grateful to all of them.

## References

- A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proc. of the Workshop on Computational Learning Theory, COLT '98, Madison, WI*, pages 92–100, 1998.
- A. J. Carlson, C. M. Cumby, N. D. Rizzolo, J. L. Rosen, and D. Roth. SNoW: Sparse Network of Winnow. 2004. URL <http://l2r.cs.uiuc.edu/~cogcomp/software.php>.
- X. Carreras and L. Màrquez. Introduction to the coNLL shared task: Semantic role labeling. In *Proc. of 8th Conference of Natural Language Learning*, pages 89–97, Boston, MA, 2004.
- X. Carreras and L. Màrquez. Introduction to the coNLL-2005 shared task: Semantic role labeling. In *Proc. of 9th Conference of Natural Language Learning*, pages 152–165, Ann Arbor, MI, June 2005.
- M. Collins. *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania, 1999.
- W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch. TiMBL: Tilburg Memory Based Learner. 2004. URL <http://ilk.uvt.nl/downloads/pub/papers/ilk0402.pdf>.
- A. Dubey. *Statistical Parsing for German*. PhD thesis, University of Saarland, Germany, 2003.
- A. Dubey. What to do when lexicalization fails: Parsing German with suffix analysis and smoothing. In *Proc. of 43rd Annual Meeting of ACL, Ann Arbor, MI*, pages 314–321, 2005.
- M. Ellsworth, K. Erk, P. Kingsbury, and S. Padó. PropBank, SALSA, and FrameNet: How design determines product. In *Proc. of the LREC 2004 Workshop on Building Lexical Resources from Semantically Annotated Corpora, Lisbon, Portugal*, 2004.
- K. Erk, A. Kowalski, and S. Padó. The Salsa annotation tool-demo description. In *Proc. of the 6th Lorraine-Saarland Workshop, Nancy, France*, pages 111–113, 2003a.
- K. Erk, A. Kowalski, S. Padó, and M. Pinkal. Towards a resource for lexical semantics: A large German corpus with extensive semantic annotation. In *Proc. of 41st Annual Meeting of ACL, Sapporo, Japan*, pages 537–544, 2003b.

- C. J. Fillmore. Frame semantics and the nature of language. In *Annals of the New York Academy of Sciences: Conf. on the Origin and Development of Language and Speech*, volume 280, pages 20–32, 1976.
- R. Ghani and R. Jones. A comparison of efficacy of bootstrapping of algorithms for information extraction. In *Proc. of LREC 2002 Workshop on Linguistic Knowledge Acquisition, Las Palmas, Spain*, 2002.
- D. Gildea and D. Jurafsky. Automatic labeling of semantic roles. In *Computational Linguistics*, volume 23, pages 245–288, 2002.
- S. Schulte im Walde. *Experiments on the Automatic Induction of German Semantic Verb Classes*. PhD thesis, Universität Stuttgart, Germany, 2003.
- R. Jones, R. Ghani, T. Mitchell, and E. Riloff. Active learning for information extraction with multiple view features sets. In *Proc. of Adaptive Text Extraction and Mining, EMCL/PKDD-03, Cavtat-Dubrovnik, Croatia*, pages 26–34, 2003.
- W. Lezius. Morphy - German morphology, part-of-speech tagging and applications. In *Proc. of 9th Euralex International Congress, Stuttgart, Germany*, pages 619–623, 2000.
- C. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, MA, 1999.
- A. K. McCallum. MALLETT: A machine learning for language toolkit, 2002. URL <http://mallet.cs.umass.edu>.
- R. J. Mooney and R. Bunescu. Mining knowledge from text using information extraction. *SIGKDD Explor. Newsl.*, 7(1):3–10, 2005.
- M. Palmer and D. Gildea. The proposition bank: An annotated corpus of semantic roles. In *Computational Linguistics*, volume 31, pages 71–106, 2005.
- S. Pradhan, K. Hacioglu, V. Kruglery, W. Ward, J. H. Martin, and D. Jurafsky. Support vector learning for semantic argument classification. *Machine Learning Journal, Kluwer Academic Publishers*, 59:1–29, 2005.
- E. Riloff and M. Schelzenbach. An empirical approach to conceptual frame acquisition. In *Proc. of 6th Workshop on Very Large Corpora, Montreal, Canada*, pages 49–56, 1998.
- J. Ruppenhofer, M. Ellsworth, M. R. L. Petruck, and C. R. Johnson. *FrameNet: Theory and Practice*. 2005. URL <http://framenet.icsi.berkeley.edu/book/book.html>.
- M. Schiehlen. Annotation strategies for probabilistic parsing in German. In *Proc. of CoLing'04, Geneva, Switzerland*, 2004.
- H. Schmid. Efficient parsing of highly ambiguous context-free grammars with bit vectors. In *Proc. of CoLing'04, Geneva, Switzerland*, 2004.
- H. Schmid. Improvement in part-of-speech tagging with an application to German. In *Proc. of the ACL SIGDAT-Workshop, Dublin, Ireland*, pages 47–50, 1995.
- G. Schreiber, H. Akkermans, A. Anjewierden, R. deHoog, N. Shadbolt, W. VandeVelde, and B. Wielinga. *Knowledge Engineering and Manage-*

- ment: The CommonKADS Methodology*. The MIT Press, Cambridge, MA, 2000.
- M. Tang, X. Luo, and S. Roukos. Active learning for statistical natural language parsing. In *Proc. of the ACL 40th Anniversary Meeting, Philadelphia, PA*, pages 120–127, 2002.
- S. Weiss, N. Indurkha, T. Zhang, and F. Damerou. *Text Mining: Predictive Methods for Analyzing Unstructured Information*. Springer, New York, NY, 2004.