Towards the Semantification of Technical Documents

Sebastian Furth[1] **and Joachim Baumeister**[1,2]
[1] denkbares GmbH, Friedrich-Bergius-Ring 15, 97076 Würzburg, Germany
[2] University of Würzburg, Institute of Computer Science, Am Hubland, 97076 Würzburg, Germany
{firstname.lastname}@denkbares.com

Abstract

In the domain of engineering large corpora of technical documents are commonly created and used. Applications such as semantic search offer advantages in accessing those documents, but require them to be semantically annotated. Annotating these corpora manually is in most cases not feasible. In recent years a lot of machine learning methods have proved their ability to annotate documents automatically. The downside of these methods is their need for training data. We present a holistic approach for the semantification of technical documents without training data. The approach tackles different challenges such as terminology extraction, semantic annotation, and reviewing. Our approach has been successfully applied to the technical documents corpora of two German machine builders

1 Introduction

Large corpora of technical documents exist in the domain of engineering. In contrast to other corpora they are often multilingual and consist of large, contentually structured and illustrated documents. Examples are operation manuals, installation guides or repair manuals. One of the main characteristics of these documents is the standardized terminology in form of a controlled vocabulary.

Exploiting the information contained in such documents can be useful for a variety of application scenarios. An example of such a scenario is the fast and effective access of information, which can be useful when searching for the repair instructions of a special assembly. *Semantic Search* [Guha *et al.*, 2003] enables such an information access. In contrast to traditional search engines ontologies are used to connect textual content with semantic information which can then be exploited during the retrieval to improve search results.

The connections between text resources and semantic information are created in a process called *Ontology Population* [Buitelaar and Cimiano, 2008], where an ontology structure is filled with instances. These instances describe for example what the main subject (in terms of ontology concepts) of a document is. This is vaguely related to *Subject Indexing* [Hutchins, 1978; Albrechtsen, 1993] which in turn can be considered as part of the more general problem of *Document Classification* [Sebastiani, 2002]. Creating these instances manually requires an in-depth analysis of the underlying documents, which is time-consuming and often cost-intensive.

In the field of *Information Extraction* there exist established methods for the extraction of semantic information from natural language texts. Most of these methods are based on supervised *Machine Learning* approaches, which require a sufficient amount of traning data for good results. In real-world scenarios such training data is often not available and the creation under the cost-benefit ratio not economic. The absence of training data implies in most cases missing test data which leads to a challenge regarding the evaluation, as standard measures like precision, recall and f-measure can not be estimated.

In this paper we present an holistic approach for the automatic semantification of technical documents that does not require training data. We call our approach holistic, as it is an complete process that covers all steps necessary for the semantification of existing technical documents. In our context semantification means the identification and annotation of the main subjects for a given document. The contribution of this paper is a process that relying on well established methods tackles the problem of semantifying technical documents without training data. The remainder of this paper is structured as follows: In Section 2 we give an overview of our approach, Section 3 describes the semantic annotation in detail, Section 4 shows the applicability of our approach in an industrial case study, Section 5 gives an overview of related work while Section 6 shows some future directions regarding our approach before concluding.

2 Process Overview

In this section we give an overview of our approach as depicted in Figure 1. Starting with unstructured technical documents (mainly PDF files) we enrich, segment, and process them in order to reach our goal of semantification. The semantification requires the availability of terminology, which is extracted from various sources. We added an explicit review stage to the process, as the results are in most cases crucial for the performance of target applications and we are not able to evaluate them due to the absence of adequate test data. Reviewed documents are also used as sources for the terminology extraction stage. In a postprocessing stage the data is prepared for target applications.

2.1 Preprocessing

The first stage of our process consists of a series of preprocessing steps. The preprocessing is necessary to prepare the input documents for the semantic annotation. In detail

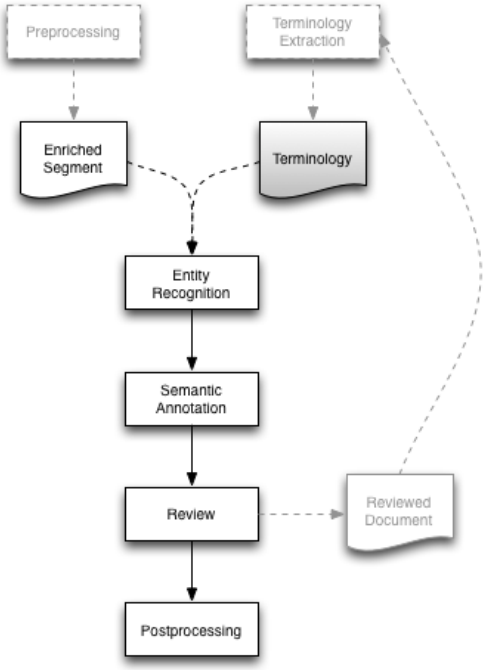

Figure 1: Overview of the semantification process.

the steps of this process stage are (1) the conversion and (2) the segmentation of the input documents as well as (3) the addition of structure to the segments.

As stated before we are mainly confronted with documents in the PDF format. To simplify the further processing we convert all documents to XML. Therefore we evaluated different PDF conversion tools and chose the Xpdf-based tool "pdf2xml"[1], as the generated XML provides a lot of exploitable information about the document's original structure. In order to achieve our goal of identifying the main subjects for each segment, we first need to split the input documents into segments. Depending on the data quality of the input documents different segmentation methods are used, e. g. structural segmentation based on the PDF outline (provided as PDF bookmarks), formatting or lexical analysis. An example for the latter one is the well established TextTiling [Hearst, 1997] approach. Each segment is enriched with structure using different methods from *Natural Language Processing* like *Tokenization*, *Part-of-Speech Tagging* or *Parsing*.

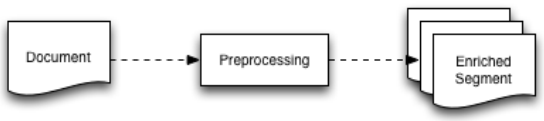

Figure 2: Converting documents to enriched segments.

2.2 Terminology Extraction

A characteristic of technical documents is the usage of a special and relatively fixed and controlled vocabulary. We exploit this characteristic by limiting the set of identifiable subjects to a given set of concepts. Together with related terms they form the *terminology* which is the basis for the semantic annotatation method presented in Section 3. The goal of this processing step is the extraction of the terminology from various sources (see Figure 3).

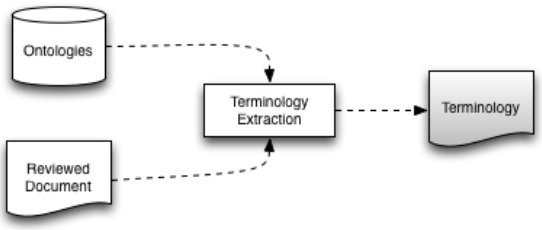

Figure 3: Extracting terminology from different sources.

The set of concepts is derived from the structural description of real world entities like machines. We assume that each concept has a human readable label. These labels are used as the most important element in the set of related terms. This set of terms is complemented by terms derived from concepts that have a relation to our given set of concepts but are not included in the set of identifiable subjects, e. g. assuming that our given set of concepts covers all assemblies of a machine, related concepts could be all parts the assemblies consist of.

A reasonable way to formalize knowledge is the definition of an ontology. There exist a couple of standardized languages for the formalization of ontologies, e. g. RDF(S) [Brickley and Guha, 2004] or OWL [Krötzsch *et al.*, 2012]. Hence it is not surprising that the structural description of real world entities like machines is often provided in the form of an ontology. When confronted with an ontology we use domain-specific SPARQL [Harris and Seaborne, 2012] queries to extract the terminology, i. e. in most cases the labels of concepts. As stated before, our process implies an explicit review step, producing reviewed documents. These documents can also be exploited in terms of terminology extraction.

2.3 Entity Recognition

For each segment we now need to identify occurrences of terminology terms, as our semantic annotation algorithm is based on these terms. So, the extracted terminology is the basis for an entity recognition step. As we are confronted with a controlled vocabulary and thus exactly know what entities (terms) we want to recognize, we use a dictionary-based entity recognition method to identify all occurrences of terminology terms in the segments. At the moment the lookup of terms is based on word stems produced by a standard Porter stemmer [Porter, 1980]. Regarding multi-word terms, we allow order independent matches, i. e. all permutations as well as non-contiguous matches, i. e. ignoring non-matching tokens between tokens belonging to a term.

2.4 Semantic Annotation

After the entity recognition step we are usually confronted with a lot of identified terms, indicating different concepts. For each segment the task is now the inference of the main concepts based on the recognized terms. We use an approach derived from Explicit Semantic Analysis proposed by [Gabrilovich and Markovitch, 2007]. This method will be described in detail in Section 3.

[1] https://sourceforge.net/projects/pdf2xml/

2.5 Review

Depending on the requirements regarding the data quality, we propose a manual review of the results of the semantic annotation by domain experts. As the availability of domain experts is a crucial element in this step, we propose the usage of an appropriate iteractive review tool (see Figure 4 for an example) that helps to decrease the review time.

For our task such a review tool needs to fulfill at least the following requirements: (1) Display the hierarchical segmentation of a specific document, (2) display the main subjects for each segment, (3) allow the addition and deletion of subjects. In order to minimize the review time for each document we additionally propose the usage of a visual component and the highlighting of critical annotations. The visual component should be able to display the semantic similarity of identified subjects, as in technical documents the subject in a sequence of segments often stays constant or at least semantically similar. An example for this claim is a technical document that covers the mounting and unmounting of assemblies. In such a document the probability is high that the corresponding segments of a specific assembly are in a sequence. In the visual component we then expect characteristic patterns like the steps displayed in Figure 4. Additionally, we propose that segments without any annotations or with a lot of semantically unrelated annotations should be automatically detected and highlighted.

There exist various metrics for the computation of semantic similarities. Examples for approaches based on WordNet [Fellbaum, 1998] were proposed among others by Jiang et al. [Jiang and Conrath, 1997] or Lin [Lin, 1998]. These metrics might be adapted due to the specifity of the used terminology.

Figure 4 shows a sample review tool. In the left the title of the current document is displayed and a status for the document (new, in progress, reviewed) can be specified by the reviewer. Below, the hierarchical segmentation of the document is displayed in a tree view element. The tree view can be used for checking and navigating through the segmentation. Clicking on an element in the tree view loads the information regarding the semantic annotations for the selected segment. The loaded information is displayed in the right part of the application. In the upper part a visual component (Visual Report) displays the results based on semantic similarity[2]. Missing annotations are indicated using a red placeholder. At the bottom of the right part detailed information (Details) about the semantic annotations are available. They can be accessed by scrolling the view or by clicking on a data point in the visual component. For a thorough review it may be necessary to look up the text of a segment, thus we provide direct access to the text in the original document. The detail view also provides possibilities for the addition and removal of concepts.

2.6 Postprocessing

The final step in the proposed process is concerned with postprocessing tasks. Such tasks typically handle the resource preparation for the target applications, evaluate the results or apply measurements to the extracted data.

[2]In the example we use taxonomic information for the computation of semantic similarity.

3 Semantic Annotation of Technical Documents

For the identification of the main subjects of a segment we use an approach derived from Explicit Semantic Analysis [Gabrilovich and Markovitch, 2007]. It was originally developed for the determination of semantic relatedness of texts and is based on a semantic interpreter which copes with a fixed set of concepts, representing each of them as an attribute vector of words. The concepts correspond to Wikipedia articles. The words are extracted from the article text and assigned weights using the TFIDF scheme [Salton and Buckley, 1988]. The semantic interpreter is realized as an inverted index that maps each word into a list of concepts in which it appears. When confronted with an input document, the relevance of the concepts contained in the index can be computed by using the semantic interpreter. For each word in the input document the inverted index is asked for the corresponding concepts and their TFIDF weights. The relevance of the concepts is computed by summing up the weights. The result is a weighted vector of concepts, where the top-ranked concept is the most relevant for the underlying document. The semantic relatedness of texts can then be determined by comparing the computed weighted concept vectors.

3.1 Building the Semantic Interpreter

In the presented approach we also use a semantic interpreter. However its purpose is not the determination of semantic relatedness of texts but the identification of the main subjects of a segment. Therefore terms and concepts are extracted from the terminology. Instead of TFIDF weights we use acquired domain knowledge to manually specify the weights, e. g. assuming we have a hierarchy of assemblies, then labels of the direct predecessors and successors of an assembly are weighted higher than the transitive ones. Another example are parts lists where we determine the weight of the parts' labels as a function of the components they are used in, i. e. parts that are used in only one component get the highest weight. In the following let $C = \{c_j\}$ be the set of concepts, $T = \{t_i\}$ be the set of terms, $\langle k_j \rangle$ be an inverted index entry for term t_i, where the weight k_j represents the strength of the association between term t_i and concept c_j.

3.2 Using Document Characteristics for Term Weighting

To determine the main subject of a segment, we first represent a segment as a list of terms. The terms correspond to annotations made by the dictionary-based entity recognition method used in a preceding process stage. In contrast to [Gabrilovich and Markovitch, 2007] we also take document characteristics into account by weighting the terms. We consider several document specific information like relevance in the document (segment frequency)[3], formatting (bold, italics, underscoring) or the position in the segment (headline). In the following let $S = \{t_i\}$ be the segment, and let $\langle v_i \rangle$ be its weight vector, where v_i is the weight of term t_i.

3.3 Ranking Concepts

For each segment we then use the semantic interpreter to get a ranked list of concepts. The ranking is done using the algorithm given as pseudo code in listing 1.

[3]As we split the document in segments, the segment frequency corresponds to the document frequency in other corpora.

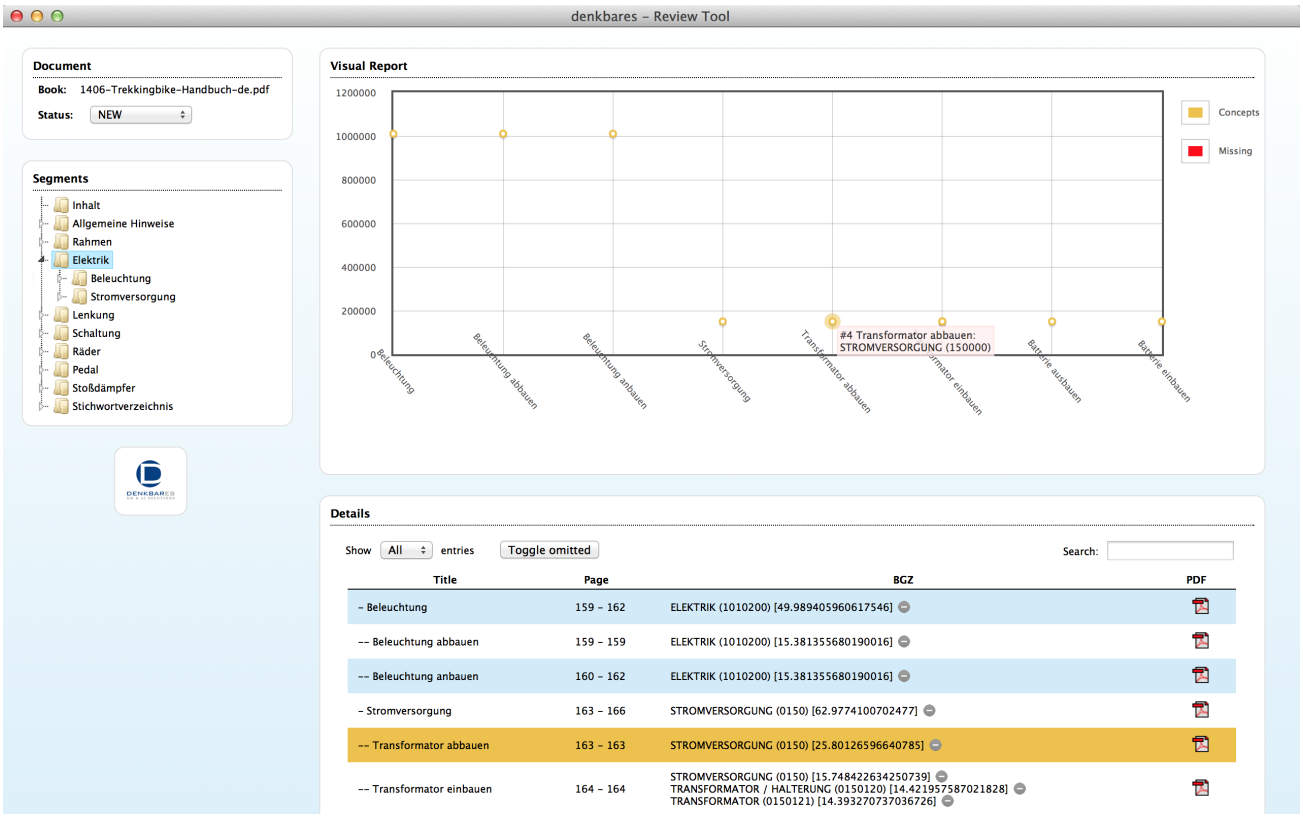

Figure 4: A tool for the manual review of semantic annotations, containing the hierarchical segmentation (left), a visual report (top right) and a detail view (bottom right).

```
getRankedConcecpts(S, ⟨v_i⟩)
 Map<Concept, Double> ranking
 for each t_i in S
  ⟨k_j⟩ = SemanticInterpreter.get(t_i)
  for each k_j
   wtdrelatedness = k_j * v_i
   ranking.update(c_j, wtdrelatedness)
 ranking.sort(WeightedRelatedness, DESC)
 return ranking
```

Listing 1: An algorithm for the term-based ranking of concepts.

The algorithm basically iterates through all terms t_i in a segment S, asks for the inverted index entry $\langle k_j \rangle$ of all concepts c_j related to term t_i and sums up the product of term weight v_i and relation strength k_j, we call it weighted relatedness. The temporary results are saved in a map which gets sorted for the final result in descending order on the weighted relatedness score. This score expresses the relevance of the concepts for the segment, i. e. a higher score means higher relevance.

3.4 Determining the Sprint Group

The algorithm described in the last section produces a ranking of relevant concepts. We now need to identify the most relevant concepts — we call it the sprint group[4].

For the determination of the sprint group we propose two different strategies. The first one simply uses a threshold, the second one is based on statistical outlier tests. So the basic approach for determining the sprint group is taking the score of the most relevant concept. Based on this score we add all concepts to the sprintgroup that are within a specified threshold, e. g. 90% of the highest score. Basically this yields good results, but there are scenarios where it does not fit. An example for such a scenario is when all concepts have low scores, i. e. no concept is really relevant for the segment. Using the basic approach the majority of the concepts would enter the sprint group. To tackle this issue we propose the usage of statistical outlier tests. Using such tests we can determine whether scores exist that offset from the rest. A simple test is for example to compute the interquartile range ($IQR = Q_{75} - Q_{25}$) and then to treat all scores that are higher than $Q_{75} + \alpha * IQR$ as outliers. There are more sophisticated outlier tests like Grubbs' test for outliers [Grubbs, 1969].

[4]Corresponding to the sprint group in cycling races, that offsets against the peloton.

4 Case Study

We have already applied our approach to corpora of two German mechanical engineering companies. In the following we describe the procedure for an engineering company for harvesting technology.

4.1 The data set

The corpus contains about 9000 technical PDF documents, covering different machines. Each document has up to 2000 pages and is of a certain type, e. g. repair manual, operation manual, circuit diagram or installation guide. The documents address different target groups ranging from maintenance staff to end users what influences the structure and the level of detail.

The terminology was mainly extracted from two ontologies. The first ontology describes relations of assemblies,

products, and machines, e. g. that the cylinder block assembly is part of the engine assembly, which itself is a part of a certain product or machine — in the following we will refer to this ontology as *core ontology*. The second ontology describes in detail which parts are build in a special assembly, e. g. that a certain valve is part of the cylinder head — we called this ontology the *parts ontology*. Assemblies and parts have had labels attached as literals using the RDFS property `rdfs:label` and language attributes. We used SPARQL to extract concepts (assemblies) and terms. Concepts were represented using their URI while the labels discribed above were used as terms.

4.2 Processing the corpus

The corpus of technical documents ran through the complete process as described in Section 2. The documents were provided in the PDF format and got converted to XML. Then a segmentation algorithm used the included PDF bookmarks to segment the documents. Structure was added to the produced segments, using a standard whitespace tokenizer and a maximum-entropy part-of-speech tagger. Then a dictionary-based entity recognition algorithm annotated all occurrences of terms extracted from the core and parts ontologies. A semantic intepreter with domain-specific weights (see next section) identified the main subjects of each segment. The results were reviewed using the review tool depicted in Figure 4. The reviewed results were finally converted into an XML format compatible with the target application.

4.3 Weighting term-concept relations

In the following we describe the weighting of the term-concept relations in detail. The $\langle k_j \rangle$ values indicating the strength of the association between term t_i and concept c_j were computed differently for assembly and part terms. For terms extracted from the core ontology we definded the weight as $k_j = \frac{1}{\#edges\ between\ concepts}$, i. e. the label of the concept in focus will get the maximum weight of 1, which means that this label indicates the concept best. Predecessors and successors in the assembly hierarchy got lower weights, e. g. the parents and children got the weight 0.5, grandparents and grandchildren the weight 0.33.

This approach was not feasible for terms from the parts ontology, because there are parts that are semantically different but have the same label (e. g. "valve" or "screw"). The more parts have the same label, the less suitable are they for the inference of a particular concept, i. e. their weight should be adapted accordingly. We decided to define the weight for terms from the parts ontolgy as $k_j = \frac{1}{concept\ frequency}$ where $concept\ frequency$ is the number of concepts that have a part represented by a particular label. This procedure shifts the focus from concepts to labels for terms from the parts ontology. The maximum weight of 1 is assigned to parts that have a unique label and are built in only one assembly. Parts with common labels that are used in a variety of assemblies get lower weights, e. g. parts with the label "screw" are built in more than 500 assemblies, so the weight is as low as 0.002.

4.4 Evaluation

The evaluation of our approach covers the performance regarding the semantic annotation of the segments, i. e. the identification of the main subject. As described above no training or test data were supplied, so we used documents that were reviewed by domain experts using the proposed review tool.

This allowed us to measure different key performance indicators, ranging from precision, recall, and f-measure to the number of corrections that needed to be made by the domain expert. We additionally measured the time needed for the correction of the automatically generated results for a couple of chapters using the proposed review tool.

For the evaluation we selected five documents from the corpus. These documents covered different machines, document types and languages. Table 1 shows the results, where the first three columns correspond to precision, recall, and f-measure and the forth and fifth column show the number of corrections made by a domain expert — the minus (-) indicates the removal of an assigned concept and the plus (+) the addition of a missing concept.

Document	P	R	F	-	+
d1-SYS-de	0,67	1,00	0,80	5	0
d2-RHB-de	0,85	0,87	0,86	16	13
d3-RHB-fr	0,81	0,74	0,77	4	6
d4-RHB-de	1,00	0,92	0,96	0	3
d5-RHB-de	0,77	0,77	0,77	31	30
Overall	0,82	0,83	0,82	56	52

Table 1: Precision, Recall, F-Measure and Number of Corrections.

The results show the overall applicability of our approach. Averaged over the five documents we yield a f-measure of 82%. In these documents 108 corrections needed to be done by the domain expert. As the availability of a domain expert is critical, we also estimated the time needed for a correction. The correction time was measured for randomly selected chapters from the documents above[5]. For each of the selected chapters we measured the number of corrections as well as the total time needed for applying them (see Table 2) — we measured an average correction time of 18 seconds per correction.

Document	# Corrections	∅ Time/Correction
d1-SYS-de (3)	2	22 s
d1-SYS-de (6)	1	20 s
d2-RHB-de (4)	5	8 s
d2-RHB-de (7)	10	16 s
d4-RHB-de (8)	1	28 s
d4-RHB-de (10)	1	16 s
d5-RHB-de (3)	10	20 s
d5-RHB-de (7)	12	14 s
Overall	42	18 s

Table 2: Measuring the correction effort: number of corrections and average time.

5 Related Work

To the best of our knowledge we are not aware of another holistic approach for the problem of the semantification of technical documents, although there exist alternative approaches for single steps of our approach. We use standard methods for the preprocessing, structural enrichment

[5]We left out the French document due to the absence of a French domain expert.

and entity recognition, so we will not consider them in this section, but focus on the terminology extraction, semantic annotation, and the review tool.

Regarding the term extraction, alternative approaches use combinations of statistic, linguistic, contextual or semantic information for the identification and selection of relevant terms, e. g. the C-Value/NC-Value approach [Frantzi *et al.*, 2000] or the TRUCKS-System [Maynard *et al.*, 2008]. As we are confronted with a limited and controlled vocabulary, our approach of extracting terms from ontologies is superior, because we have complete control over the results.

Regarding the semantic annotation, which in our case is the identification of the main subject for a segment or document, latent approaches exist, e. g. Latent Dirichlet Allocation (LDA) [Blei *et al.*, 2003] or Latent Semantic Analysis (LSA) [Deerwester *et al.*, 1990]. We want to identify a concrete (or explicit) concept, so the latent approaches do not fit for our problem.

Regarding the review of semantic annotations we do not know of another tool for the review of the main subject of a segment or document. Ontosophie [Celjuska and Vargas-Vera, 2004] is a system for the population of an event ontology and uses supervised machine learning for learning extraction rules. These rules also compute a confidence value which is used to determine whether a human reviewer needs to accept an extracted information. The idea of our review tool is to guide a human reviewer through an entire book and highlight critical annotations for rapid correction.

6 Conclusion and Future Work

We proposed a holistic approach for the semantification of technical documents without training data. We defined a process for tackling a couple of challenges, such as terminology extraction, semantic annotation and reviewing. We use standard techniques for the preprocessing and the structural enrichment of the documents. The core of our approach is the semantic annotation which is based on Explicit Semantic Analysis and domain ontologies. This allows for the easy adaptation to new corpora.

We already applied our approach to the techical documents corpora of two mechanical engineering companies. We were able to achieve promising results on these corpora (average f-measure of 82%). We also developed a tool for the manual review and correction of semantic annotations. Experiments with domain experts showed that the average correction time is 18 seconds — which allows for the complete review of a large technical document in a couple of minutes.

For the future we plan to improve the weighting of the term-concept relations. We will investigate different directions: (1) a general applicable weighting scheme, (2) more sophisticated domain specific weighting schemes and (3) the adaption of the weights using the manually reviewed documents. Regarding the review tool we will test other visualization techniques in order to improve the review time and results. We also plan to improve the evaluation of our approach by (1) building or using a public available test corpus and (2) comparing our results to established supervised Machine Learning approaches, using manually reviewed documents as training data.

References

[Albrechtsen, 1993] Hanne Albrechtsen. Subject analysis and indexing: from automated indexing to domain analysis. *Indexer*, 18:219–219, 1993.

[Blei *et al.*, 2003] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[Brickley and Guha, 2004] Dan Brickley and Ramanathan V. Guha. RDF Vocabulary Description Language 1.0: RDF Schema. *W3C Recommendation*, 10, 2004.

[Buitelaar and Cimiano, 2008] Paul Buitelaar and Philipp Cimiano. *Ontology learning and population: bridging the gap between text and knowledge*, volume 167. Ios Press, 2008.

[Celjuska and Vargas-Vera, 2004] David Celjuska and Maria Vargas-Vera. Ontosophie: A semi-automatic system for ontology population from text. In *Proceedings of the 3rd International Conference on Natural Language Processing (ICON)*, 2004.

[Deerwester *et al.*, 1990] Scott Deerwester, Susan T. Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

[Fellbaum, 1998] Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA, 1998.

[Frantzi *et al.*, 2000] Katerina Frantzi, Sophia Ananiadou, and Hideki Mima. Automatic recognition of multi-word terms:. the c-value/nc-value method. *International Journal on Digital Libraries*, 3(2):115–130, 2000.

[Gabrilovich and Markovitch, 2007] Evgeniy Gabrilovich and Shaul Markovitch. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proceedings of the 20th international joint conference on artificial intelligence*, volume 6, page 12, 2007.

[Grubbs, 1969] Frank E. Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11(1):1–21, February 1969.

[Guha *et al.*, 2003] Ramanathan Guha, Rob McCool, and Eric Miller. Semantic search. In *Proceedings of the 12th international conference on World Wide Web*, pages 700–709. ACM, 2003.

[Harris and Seaborne, 2012] Steve Harris and Andy Seaborne. SPARQL 1.1 query language. Proposed recommendation, W3C, 2012.

[Hearst, 1997] Marti A. Hearst. TextTiling: segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64, 1997.

[Hutchins, 1978] William J. Hutchins. The concept of 'aboutness' in subject indexing. In *Aslib Proceedings*, volume 30, pages 172–181. MCB UP Ltd, 1978.

[Jiang and Conrath, 1997] Jay J Jiang and David W Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. *arXiv preprint cmp-lg/9709008*, 1997.

[Krötzsch *et al.*, 2012] Markus Krötzsch, Peter F. Patel-Schneider, Sebastian Rudolph, Pascal Hitzler, and Bijan Parsia. OWL 2 Web Ontology Language Primer (Second Edition). Technical report, W3C, October 2012.

[Lin, 1998] Dekang Lin. An information-theoretic definition of similarity. In *ICML*, volume 98, pages 296–304, 1998.

[Maynard *et al.*, 2008] Diana Maynard, Yaoyong Li, and Wim Peters. NLP techniques for term extraction and ontology population. In *Proceeding of the 2008 conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge*, pages 107–127, 2008.

[Porter, 1980] M. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[Salton and Buckley, 1988] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5):513–523, 1988.

[Sebastiani, 2002] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47, 2002.