Volume 27 - Number 1 - 2012 - ISSN 2190-6858

JLCL

Journal for Language Technology
and Computational Linguistics

Herausgegeben von /*Edited by*
Lothar Lemnitzer

GSCL Gesellschaft für Sprachtechnologie & Computerlinguistik

Contents

Impressum

Herausgeber	Gesellschaft für Sprachtechnologie und Computerlinguistik (GSCL)
Aktuelle Ausgabe	Band 27 – 2012 – Heft 1
Herausgeber	Lothar Lemnitzer,
Anschrift der Redaktion	Lothar Lemnitzer Berlin-Brandenburgische Akademie der Wissenschaften Jägerstr. 22/23 10117 Berlin lemnitzer@bbaw.de
ISSN	2190-6858
Erscheinungsweise	2 Hefte im Jahr, Publikation nur elektronisch
Online-Präsenz	www.jlcl.org

Editorial

Ich begrüße Sie zur der nächsten thematisch nicht gebundenen Ausgabe des Journals. Es ist das erste Heft des Jahrgangs 27.

Das Heft ist das Ergebnis einer Ausschreibung und der anschließenden Begutachtung der Einreichungen, für die ich mich bei den GutachterInnen bedanken möchte. Nach dem Begutachtungsprozess blieben drei Beiträge übrig, die ich mich nun freue in diesem Heft Ihnen präsentieren zu können.

Henrich, Hinrichs und Suttner präsentieren eine Methode, mit der sie das deutsche Wortnetz GermaNet um Beispielsätze aus der deutschen Wikipedia anreichern und zugleich ein semantisch annotiertes Korpus erstellen konnten.

Haddar, Fehri und Romary präsentieren ein Verfahren, mit dem sie verschiedene lexikalische Ressourcen im HPSG-Stil und für das Arabische zusammenführen. Das Lexical Markup Framework, mit einigen Erweiterungen, dient hier als gemeinsames Zielformat.

Riedl und Biemann zeigen, wie man erfolgreich Texte entlang der durch sie repräsentierten „Topics“ segmentiert. Das Verfahren ist effizienter und zugleich akkurater als der von ihnen dargestellte Stand der Technik.

Zum Schluss noch ein Ausblick auf das zweite Heft dieses Jahrgangs. Es wird voraussichtlich Ende November 2012 erscheinen, hat computerlinguistische Methoden für Texte altüberlieferter Sprachen zum Gegenstand und wird von Herrn Hoenen und Herrn Jügel aus Frankfurt als Gastherausgeber betreut.

Ab dem folgenden Heft wird die Redaktion von mir gemeinsam mit Thierry Declerck geleitet. Die meisten von Ihnen werden Thierry sicher kennen.

Ich wünsche Ihnen eine gute Lektüre

Lothar Lemnitzer

Verena Henrich, Erhard Hinrichs, Klaus Suttner

Automatically Linking GermaNet to Wikipedia for Harvesting Corpus Examples for GermaNet Senses

The comprehension of a word sense is much easier when its usages are illustrated by example sentences in linguistic contexts. Hence, examples are crucially important to better understand the sense of a word in a dictionary. The goal of this research is the semi-automatic enrichment of senses from the German wordnet GermaNet with corpus examples from the online encyclopedia Wikipedia. The paper describes the automatic mapping of GermaNet senses to Wikipedia articles, using proven, state-of-the-art word sense disambiguation methods, in particular different versions of word overlap algorithms and PageRank as well as classifiers that combine these methods. This mapping is optimized for precision and then used to automatically harvest corpus examples from Wikipedia for GermaNet senses. The paper presents details about the optimization of the model for the GermaNet-Wikipedia mapping and concludes with a detailed evaluation of the quantity and quality of the harvested examples. Apart from enriching the GermaNet resource, the harvested corpus examples can also be used to construct a corpus of German nouns that are annotated with GermaNet senses. This sense-annotated corpus can be used for a wide range of NLP applications.

1 Introduction

Different senses of a word are often hard to distinguish – not only for second language learners. This is especially the case when a dictionary makes fine-grained sense distinctions for polysemous words (Palmer et al., 2007). Although the usefulness of meaningful sense descriptions for identifying word senses is self-evident, descriptions alone are often not sufficient to discriminate senses. Kilgarriff et al. (2008) point out that humans grasp the sense of a word in a dictionary much easier when example sentences illustrating the usage of a word in context are available. Consequently, corpus examples are crucially important for comprehensive understanding of senses in dictionaries and other lexical resources such as wordnets.

The purpose of this paper is to describe an automatic method for adding corpus examples to the word senses of a wordnet. While the method described is language-independent, the present paper will focus on the German wordnet GermaNet. Using German as a test case is particularly appropriate since – with the exception of its verb entries – GermaNet's word senses are still lacking illustrative example sentences. This gap in coverage is particularly evident in the case of nouns, which have a total of 77 925 word senses in GermaNet (release 6.0) and which are – with few exceptions

– not accompanied by any example sentences. Due to the large number of missing examples, the task of adding them by purely manual, lexicographic work would be at best an arduous task and require considerable effort and person power. Therefore, the possibility of employing automatic or semi-automatic methods for adding corpus examples would be extremely valuable.

Such an automatic method has to rely on an electronically available resource that should ideally satisfy the following criteria: (i) it should be of sufficient size in order to provide the necessary lexical coverage, (ii) since nouns are the focus of the present paper, the resource should have a comprehensive coverage of nominal word senses and a significant overlap in coverage with GermaNet, and (iii) it should be freely available so that the corpus examples harvested from the resource in question can be freely shared. The requirement that word senses are to be mapped to example sentences by automatic means imposes a further restriction on the type of textual material to be used. Such a mapping needs to perform automatic word sense disambiguation so as to ensure that the candidate word senses from GermaNet are mapped to the appropriate example sentences. The precision of this word-sense-to-example mapping should be extremely high so as to be usable with minimal amount of manual post-correction. Such high precision can be realized only if automatic word sense disambiguation can be performed with high reliability. This is, in turn, the case if the texts from which the examples are harvested exhibit a high degree of thematic coherence so as to provide sufficient cues for contextual disambiguation.

If one takes the requirements just mentioned into account, the web-based encyclopedia Wikipedia[1] becomes a natural choice. Its thematic coverage focuses on articles that typically describe nominal concepts and thus provides the type of lexical coverage needed for the present purpose. It is freely available, of sufficient size, and thematically diverse and comprehensive. Moreover, there is a 76.7% overlap in coverage between Wikipedia and the 4 358 polysemous nouns in GermaNet. In addition, the articles attempt to illustrate a particular target concept and are thus thematically highly coherent. This in turn facilitates automatic word sense disambiguation.

In short, the task at hand consists of an automatic mapping of word senses in GermaNet to articles in Wikipedia and the actual harvesting of corpus examples from the linked Wikipedia articles. The nature of the task of harvesting corpus examples for word senses is closely related to the task of creating a sense-annotated corpus. Both tasks focus on harvesting textual materials whose words will be assigned the corresponding word senses of the sense inventory (i.e., wordnet) in question. Because of this close similarity between the two tasks, it is appropriate to combine all harvested corpus examples into a sense-annotated corpus.

In recent years, the use of Wikipedia has gained considerable popularity in empirically oriented research in theoretical and computational linguistics. The present paper wants to contribute to this growing body of research which thus far has mostly focused on English. To the best of our knowledge the present study is the first of its kind for German

[1] http://www.wikipedia.org/

that links word senses in GermaNet to the corresponding articles in Wikipedia. There has been a considerable body of research for English that investigates the alignment of the Princeton WordNet with Wikipedia (see Section 3). However, we are not aware of any other previous research that tries to align the German Wikipedia to GermaNet.

The semi-automatic enrichtment of GermaNet with examples taken from Wikipedia is valuable not only for users of GermaNet, but also for lexicographers involved in the further construction of GermaNet. The Wikipedia examples offer authentic language materials and thereby free lexicographers from having to construct made-up examples that are not validated by actual language corpora.

The remainder of this paper is structured as follows: After a short description of the resources GermaNet and Wikipedia in Section 2, Section 3 provides an overview of related work. Sections 4 and 5 introduce the mapping of GermaNet to Wikipedia and describe how this mapping can be used to automatically harvest corpus examples for GermaNet senses, respectively. The approach is evaluated in Section 6. Finally, there are concluding remarks and an outlook to future work in Section 7.

2 Resources

2.1 GermaNet

GermaNet (Henrich and Hinrichs, 2010; Kunze and Lemnitzer, 2002) is a lexical semantic network that is modeled after the Princeton WordNet for English (Fellbaum, 1998). It represents word meanings by *lexical units* and groups lexical units that express the same semantic concept into *synsets* (*synonymy sets*). Thus, a synset is a set-representation of the semantic relation of synonymy.

Synsets and lexical units are interlinked by two types of semantic relations: by *conceptual* and by *lexical* relations. Conceptual relations hold between two semantic concepts, i.e., synsets. They include relations such as hypernymy, part-whole relations, entailment, or causation. Lexical relations hold between two individual lexical units. Antonymy, a pair of opposites, is an example of a lexical relation.

GermaNet covers the three word categories of adjectives, nouns, and verbs, each of which is hierarchically structured in terms of the hypernymy relation of synsets. The development of GermaNet started in 1997, and is still in progress. GermaNet's version 6.0 (release of April 2011) contains 93 407 lexical units, which are grouped into 69 594 synsets. At present, GermaNet provides comprehensive example sentences only for its verbs senses.

2.2 Wikipedia

Wikipedia is a web-based encyclopedia that is available for many languages, including German. It is written collaboratively by volunteers and is freely available[2]. The general

[2]Wikipedia is available under the Creative Commons Attribution/Share-Alike license `http://creativecommons.org/licenses/by-sa/3.0/deed.en`

structure of a Wikipedia article starts with a paragraph that briefly defines the presented concept. The rest of the article consists of a detailed description optionally containing references that proof the source of the text, hyperlinks to other Wikipedia articles as well as pictures illustrating the described context. Further, the encyclopedia divides its articles into thematic categories. For those words that have multiple articles, Wikipedia provides disambiguation pages with a short description of each article.

For the present project, a dump of the German Wikipedia as of June 21, 2011 is utilized, consisting of 2.27 mio. pages. The Wikipedia data was extracted by the freely available Java-based library JWPL (Zesch et al., 2008).

3 Related Work

As mentioned in the Introduction, the purpose of this paper is to describe an automatic method for adding corpus examples to the word senses of GermaNet. This task is twofold: (i) it involves the automatic mapping of word senses in GermaNet to articles in Wikipedia and (ii) on the basis of this mapping, it harvests corpus examples for GermaNet's senses. Related work for both these tasks is discussed in the following two subsections.

3.1 Mapping Wikipedia to a Wordnet

Several authors have investigated ways of aligning the Princeton WordNet with the English Wikipedia, with some studies focusing on an alignment of Wikipedia categories to WordNet synsets and others investigating the alignment between Wikipedia articles and WordNet. Toral et al. (2009) utilize several text similarity measures to match Wikipedia categories to WordNet synsets. For the same task, Ponzetto and Navigli (2009) apply a knowledge-rich method which maximizes the structural overlap between the WordNet taxonomy and the category graph extracted from Wikipedia.

Other approaches align articles in Wikipedia – instead of categories – with WordNet synsets. In the study of Wolf and Gurevych (2010), the actual alignment between Wikipedia articles and WordNet synsets has been performed manually on the basis of an automatically extracted set of potential sense alignments. A vector-based similarity measure is applied by Ruiz-Casado et al. (2005) to map articles of the Simple English Wikipedia to their most similar WordNet synset. Suchanek et al. (2007) ignore ambiguity while aligning Wikipedia and WordNet and solve ambiguous mappings manually. Ponzetto and Navigli (2010) calculate conditional probabilities relying on a normalized word overlap measure of the textual sense representation. A threshold-based Personalized PageRank to automatically align articles in Wikipedia with synsets in WordNet is utilized by Niemann and Gurevych (2011). The most recent study we are aware of is the one by Fernando and Stevenson (2012), who first compute similarity between WordNet synsets and Wikipedia articles to perform the alignment and then apply heuristics based on the link structure of Wikipedia to refine their resulting mappings.

All these accounts differ in certain aspects from our approach. Most follow the idea of extending the coverage of an ontology, whereas we focus on the systematic enrichment of an existing resource, i.e., GermaNet, by corpus examples. This is the reason why we perform the mapping on word senses (i.e., lexical units) in GermaNet and not on synsets, as the above-mentioned studies do. Moreover, these studies all focus on English, while our work concerns German. Our approach allows the alignment of multiple Wikipedia articles to a sense in GermaNet, whereas some of the other algorithms assign only the most likely WordNet synset to an article in Wikipedia.

3.2 Harvesting Corpus Examples

The nature of the task of harvesting corpus examples for word senses is closely related to the task of creating a sense-annotated corpus. Both tasks focus on harvesting textual materials whose words will be assigned the corresponding word senses of the wordnet in question. Because of this close similarity between the two tasks, it is appropriate and relevant to review and to characterize the state of the art in creating sense-annotated corpora.

With relatively few exceptions to be discussed shortly, the construction of sense-annotated corpora has focussed on purely manual methods. This is true for SemCor, the WordNet Gloss Corpus, and for the training sets constructed for English as part of the SensEval and SemEval shared task competitions (Agirre et al., 2007; Erk and Strapparava, 2010; Agirre et al., 2004). Purely manual methods were also used for the German sense-annotated corpora constructed by Broscheit et al. (2010) and Raileanu et al. (2002) as well as for other languages including the Bulgarian and the Chinese sense-tagged corpora (Koeva et al., 2006; Wu et al., 2006).

Few previous attempts of (semi-)automatically harvesting corpus data for the purpose of constructing a sense-annotated corpus exist. Yarowsky (1995), for example, developed a semi-supervised method based on a decision-list supervised WSD algorithm that iteratively disambiguates examples starting with a manually created seed set of annotated sentences. The knowledge-based approach of Leacock et al. (1998) – later also used by Agirre and Lacalle (2004) and Mihalcea and Moldovan (1999) – relies on the monosemous relative heuristic for the automatic harvesting of web data for the purposes of creating sense-annotated corpora. By focussing on web-based data, their work resembles the research described in the present paper. However, the underlying harvesting methods differ.

The three studies that are closest in spirit to the approach presented here are those of Santamaría et al. (2003), Henrich et al. (2012), and Henrich et al. (to appear). These studies also rely on automatic mappings between wordnet senses and a second web resource: an automatic association of Web directories (from the Open Directory Project, ODP) to WordNet senses for English (in the case of Santamaría et al. (2003)), a mapping between the German version of the web-based dictionary Wiktionary and GermaNet created by Henrich et al. (2011) (in the case of Henrich et al., 2012), and a mapping between the English Wiktionary and the Princeton WordNet created by Meyer

and Gurevych (2011) (in the case of Henrich et al., to appear). Henrich et al.'s (2012) work has produced the German WebCAGe corpus (short for: *Web-Harvested Corpus Annotated with GermaNet Senses*). WebCAGe has been constructed by harvesting sense-specific example sentences from Wiktionary itself and by harvesting additional textual materials from other web-based textual sources such as Wikipedia, online newspaper materials, and the German Gutenberg text archive[3]. These additional materials were harvested by following the links that accompany example sentences in Wiktionary. The work by Henrich et al. (to appear) applies Henrich et al.'s (2012) approach to English and has led to a sense-annotated corpus for English which they call WebCAP (short for: *Web-Harvested Corpus Annotated with Princeton WordNet Senses*). For both these corpora (Henrich et al., 2012; Henrich et al., to appear) it has to be kept in mind that the example sentences contained in Wiktionary are often artificially constructed by the authors of a Wiktionary entry and are, thus, not authentic materials taken from actual text corpora. Harvesting example sentences from Wikipedia articles – the goal of the present research – results in authentic corpus examples and, thus, provides a significant extension of Henrich et al.'s work.

4 Mapping GermaNet to Wikipedia

As mentioned above, harvesting of corpus examples from Wikipedia presupposes the existence of a mapping from GermaNet to Wikipedia in order to be able to link each target word in question to the appropriate GermaNet sense. Since the words contained in GermaNet and Wikipedia are often ambiguous, this mapping involves lexical disambiguation. The senses of an ambiguous word in GermaNet are each represented by a lexical unit. In Wikipedia, the senses of an ambiguous term are summarized in a 'disambiguation page' that lists all word meanings distinguished in Wikipedia along with short descriptions of each sense. Figure 1 shows a simplified example of such a disambiguation page for the German noun *Brücke*.[4]

The disambiguation page for *Brücke* in Figure 1 lists 9 distinct senses: *Brücke* in the sense of a structure built to span physical obstacles, a sportive excercise, a charge of heraldy, a defensive stance in wrestling, a bridge as a fixed partial denture, a small carpet, an edge in a graph, a structure located on the brain stem (pons), and a bridge of a ship. Each of these senses is summarized by a short description that contains a link to the corresponding Wikipedia article. Additionally, the disambiguation page also lists the use of *Brücke* in named entities such as family names (see the four bullet points in the lower part of Figure 1). Since named entities are not modelled in GermaNet, these additional senses can be ignored and the mapping can be limited to the ordinary senses of the word.

In GermaNet, the word *Brücke* is associated with three distinct lexical units (senses) that are contained in the following synsets:

[3] `http://gutenberg.spiegel.de/`

[4] Note that there are further senses for *Brücke* in Wikipedia that are not shown in the figure for reasons of space.

Figure 1: Disambiguation page for the word *Brücke* in Wikipedia

Sense 1 ('bridge of a ship'): *Kommandobrücke, Brücke* – Schifffahrt; hypernyms: Deck, Schiffsdeck

Sense 2 ('bridge as a structure built to span physical obstacles'): *Brücke* – ein künstlicher Weg zur Überquerung eines Flusses, eines Tales oder Ähnlichem; hypernyms: Übergang, Überweg

Sense 3 ('bridge as a fixed partial denture'): *Brücke* – Zahnmedizin: modellierte Zahnreihe zur Überwindung eines oder mehrerer fehlender Zähne; hypernyms: Zahnersatz

The mapping task between GermaNet and Wikipedia now has to associate the correct GermaNet sense with the corresponding word meaning in Wikipedia. In general, this involves an n:m mapping. In the case that there is no disambiguation page, but the term is contained in Wikipedia, i. e. the term is monosemous, the Wikipedia article itself is used as a candidate for the mapping. Even if each of the resources only lists a single sense, it cannot automatically be assumed that the two entries in question refer to the same sense. Please also note that the titles of the Wikipedia articles are not always identical to the word under consideration. For example, two of the word senses of *Brücke* link to Wikipedia articles with the titles *Pons* 'pons' and *Kommandobrücke* 'bridge of a ship'.

For the mapping between GermaNet and Wikipedia several systems were implemented which basically rely on two different algorithms: Lesk and PageRank.

Lesk: Lesk (1986) introduces a word sense disambiguation algorithm that disambiguates two words by counting the overlaps between their respective sense definitions. Applied to the task at hand, this means that given two bag of words (BOW) for a GermaNet sense s_i and a Wikipedia page p_j, the overlap between these is calculated.

PageRank: PageRank (Brin and Page, 1998) is Google's algorithm for ranking webpages. Given a graph, every node v is initialized with $v = \frac{1}{|nodes|}$. In the following iteration steps every node spreads its mass equally to its neighbour nodes. The process is repeated until the values for each node converge. The resulting PageRank vector $\mathbf{Pr}$ is equivalent to:

$$\mathbf{Pr} = cM\mathbf{Pr} + (1 - c)\mathbf{v}$$

where M is the adjacency matrix for the graph, $\mathbf{v}$ is the vector with the initial values and c is a damping factor, which controls, how much of the initial mass is infused in every iteration step. Since $\sum_i Pr_i = 1$, the resulting value Pr_i may be considered as the probability to end up with node v_i in a random walk over the graph.

Both techniques have in common, that they use bag of words (BOW) for the disambiguation. A bag of words representing a given text is just the set of lemmas occurring in the text, i.e., just the words without syntactic information. Although a BOW is a very basic data structure, it is very common in Information Retrieval to represent whole documents. In the implementation for our algorithms, two kinds of BOWs are used: one representing a Wikipedia page and one representing a sense in GermaNet. In the case of a Wikipedia page, the corresponding article is used for the BOW, in the case of a Germanet sense all synonyms of the given sense and all neighbouring words/synsets up to a certain distance are included in the BOW. There are several parameters, which allow to control which words are actually included in the BOWs (see Section 6.1 for more details about these parameters).

What follows is a detailed description of the different systems we implemented.

1. Lesk: Given two BOWs, one for a given Germanet sense s_i and one for a given Wikipedia page w_j, the overlap between the two is calculated and normalized with respect to the minimum of the two.

2. We have reimplemented the approach by Niemann and Gurevych (2011). Given the two BOWs for GermaNet sense s_i and Wikipedia page w_j, PageRank is run twice on the whole GermaNet graph, initializing only those nodes whose corresponding synsets have at least one lemma in common with both BOWs. To calculate the semantic relatedness between a sense s_i and a Wikipedia page w_j

three similarity measures were applied to the resulting two PageRank vectors $\Pr(s_i)$ and $\Pr(w_j)$: Euklidian distance, cosine, and chi^2:

$$\chi^2(Pr(s_i), Pr(w_j)) = \sum_k \frac{(Pr(s_i)_k - Pr(w_j)_k)^2}{Pr(s_i)_k + Pr(w_j)_k}$$

3. We developed a system called *TextLink*, which is an adaptation of the PageRank algorithm. It uses a special directed bipartite multigraph, which consists on one side of all Wikipedia articles and on the other side of all lemmas which function as a link in Wikipedia – see Figure 2: Wikipedia articles are shown in the upper part of the figure, the lemmas occuring as links in the lower part. For this purpose the whole Wikipedia is scanned for links. Whenever a link is found, the lemma/phrase, which is configured as a link (i.e., the link label), is added as a new node to the graph, if not already existent, connecting it with the two nodes corresponding to the interlinked Wikipedia pages (parallel edges are allowed). Note that the example in Figure 2 is a pretty small excerpt from the whole graph.

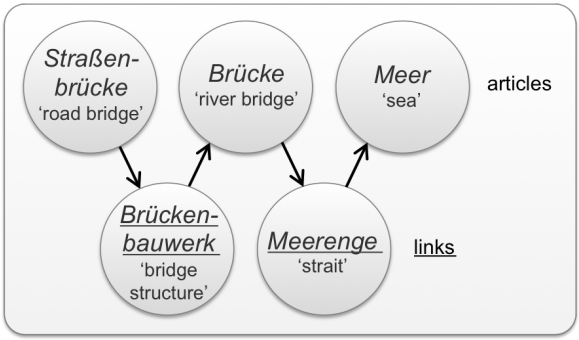


Figure 2: Bipartite graph illustration for *Brücke* ('bridge' architecture)

More formally, the definition of the graph is $G(V, A)$ with vertices $V = W + L$, $W \cap L = \emptyset$, where W is the set of all Wikipedia pages, $w_i \in W$ refers to a specific Wikipedia page w_i, L describes the set of all hyperlinks, and $l_k \in L$ is a hyperlink h with anchor text (label) l_k. Given two Wikipedia pages w_i and w_j and a hyperlink $h(l_k, w_i, w_j)$, directing from Wikipedia page w_i to page w_j, whose anchor text (label) is l_k: For every such hyperlink h we create two arcs $a_s, a_t \in A$ with $a_s = w_i l_k$ and $a_t = l_k w_j$ (parallel arcs allowed).

To better understand the construction of the graph, see the second Wikipedia article in Figure 3 (which will be described in Section 5) entitled with *Brücke* 'bridge': the mouseover symbol on the left side illustrates that the link labelled

with *Meerenge* 'strait' connects the two Wikipedia articles *Brücke* and *Meer* 'sea' with each other.

In order to calculate a mapping between the GermaNet senses s of a given lemma and the corresponding set of Wikipedia pages w, for each sense a BOW is created. Given a BOW for sense s_i all link nodes whose labels are contained as a lemma in the BOW are initialized and PageRank is run with just one iteration and a damping factor $c = 1$. Sense s_i is then mapped to the Wikipedia page w_j which maximizes the resulting value and which is above a certain threshold.

Alternatively we applied three iterations, slightly modifying the original PageRank algorithm in that we added up the values in each iteration step, so that the value for vertex $v_i = \sum_{k=1}^{3} v_i^{(k)}$. Note that this is a slight alteration of the original PageRank algorithm because we iterate exactly three times and not until the node values remain constant as it is the idea in the original PageRank algorithm. Experiments showed better results with this procedure, which can be regarded as a weighted breadth-first-search of distance three with the exception that nodes can be visited more than once.

4. Combination of two different systems: we tested, if any combination of two systems (out of the three systems described in 1., 2., and 3. above) might give better results, thus showing that the power of Lesk and PageRank lie in different fields and act to some degree in a complementary way.

For all of the algorithms just described, we use thresholding for the mapping between GermaNet senses and Wikipedia articles: a mapping is established only if the numeric value computed for a putative mapping by the WSD algorithm is above a certain threshold. This threshold has been computed by a series of test runs on the training corpus (described in Section 6.1).

5 Harvesting Corpus Examples

Once the GermaNet word senses have been mapped to Wikipedia articles, these articles need to be mined for relevant corpus examples that include the target word in question. Notice that the target word often occurs more than once in a given text. In keeping with the widely used heuristic of "one sense per discourse" (Gale et al., 1992), multiple occurrences of a target word in a given text are all automatically assigned to the same GermaNet sense.

In a morphologically rich language like German, the automatic harvesting of example sentences requires some lexical preprocessing of the Wikipedia articles in order to be able to robustly identify the occurrences of the target word under consideration. Automatic detection of target words is performed by the software tool used by Henrich et al. (2012) for the construction of WebCAGe. This tool splits the text up into individual sentences, performs tokenization, lemmatization, and compound splitting. Apart from lemmatization, compound splitting is also necessary because the target word can be part

of a compound. Since the constituent parts of a compound are not usually separated by blank spaces or hyphens, German compounding poses a particular challenge for target word identification.

Figure 3 shows the combined result of the GermaNet to Wikipedia mapping and the harvesting of example sentences for each of the Wikipedia articles associated with the GermaNet senses of the German noun *Brücke*. The occurrences of the target words are highlighted in the running text by surrounding boxes. Because of the sense mapping between GermaNet and Wikipedia, each target word occurrence is automatically associated with a corresponding GermaNet sense.

The primary use of the harvested examples in the present study is to enrich the GermaNet lexical units by corpus examples from Wikipedia. However, an interesting and highly useful by-product of this work is the construction of a large sense-annotated corpus of Wikipedia data for German, which will be referred to as WikiCAGe (short for: *Wikipedia-Harvested Corpus Annotated with GermaNet Senses*). This by-product is particularly valuable because sense-annotated corpora for German are in short supply.

6 Evaluation

The two tasks to be solved in this research (the mapping and the harvesting) require separate evaluations. This section presents both evaluation steps: Section 6.1 evaluates the automatic mapping of word senses in GermaNet to articles in Wikipedia. The harvesting of the corpus examples, which relies on this mapping, is analysed in Section 6.2.

6.1 Evaluation of the Automatic Mapping

In order to be able to evaluate the automatic alignment of lexical units (senses) in GermaNet to articles in Wikipedia, three experienced lexicographers created two manually annotated gold standards:

1. The gold standard that was used for training, i.e., to identify the best performing systems and to fine-tune the most reliable parameter settings, consists of 30 polysemous nouns. These 30 nouns comprise a total of 862 potential sense mappings between GermaNet senses and Wikipedia articles of which 82 were manually classified as correct. The nouns were manually chosen with the goal of including examples with different numbers of senses, ranging from 2 to 6 distinct senses. On average, the 30 nouns exhibit 3.7 senses in GermaNet. This degree of polysemy is considerably higher compared to the average number of 2.3 word senses of polysemous nouns in GermaNet. The reason for choosing a set of nouns with a higher than average degree of polysemy for training was deliberate so as to provide ample data for a fine-grained adjustment of the parameter and threshold settings with respect to all classifiers used for the GermaNet-Wikipedia mapping.

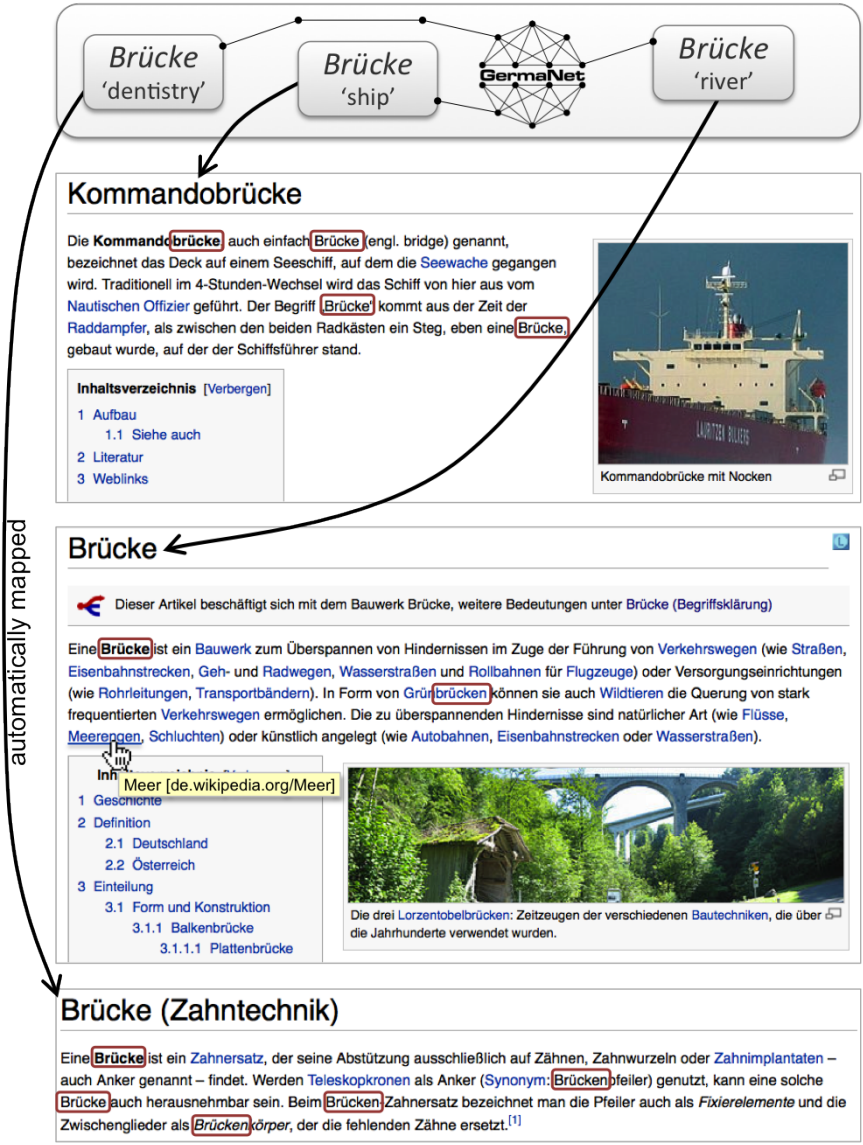


Figure 3: Mapping example for the word *Brücke* with corpus examples

2. For testing the best algorithm setup, another gold standard of 270 randomly chosen polysemous nouns with an average of 2.4 senses was created.[5] These 270 nouns comprise a total of 4308 sense mappings of which 446 were classified as correct.

The first gold standard has been used to identify the best performing systems and to fine-tune the most reliable parameter settings. All systems that are evaluated use one or two bag of words for the disambiguation. Which words are actually included in the BOW is a matter of parameter setting. In the case of Wikipedia, the choices are the following: (i) whether the BOW consists of the title and the first paragraph of an article or of the entire page, (ii) whether to include in the BOW the Wikipedia categories linked to the article or not, and (iii) whether the anchor words of ingoing resp. outgoing links should be included in the BOW or not.

The experiments with the training corpus show constantly better results when the BOW representing a Wikipedia page consists of the title and the first paragraph instead of the entire page. This is not surprising since the first paragraph of a Wikipedia article usually serves as a short definition of the presented concept. Further, the results are much better when the anchor words of the links are included in the BOW of a Wikipedia page. This can be explained by the fact that a term, which is configured as a link directing to that page, is usually semantically closely related to the term described on the page.

In the case of GermaNet, the BOW includes all synonyms from the target word synset and can be expanded to include synsets that are linked to the target word by conceptual or lexical relations. This expansion is again a matter of parameter setting and includes the following choices: (i) the graph distance between the target word and the candidate synset, (ii) a weighting parameter that is proportional to the graph distance, and (iii) whether to include or exclude the hyponymy relation among the conceptual relations used for expansion.

The parameter settings just described determine the strength of association between a GermaNet sense and a Wikipedia article. This numerical score can then be used for thresholding. That is, the association strength is considered a match only if the score is above a given threshold.

The mapping algorithm follows a maximal matching strategy of the GermaNet-Wikipedia bipartite graph. Another choice point concerns the interaction of thresholding and maximal match calculation. Thresholding can either be incorporated into the maximal match calculation in the sense that candidate matches below a given threshold are discarded when the overall optimal mapping is calculated or thresholding can be applied after maximal match calculation. In the latter scenario, which empirically turned out to be superior, thresholding is in effect used to prune individual sense mappings from the maximal match result.

[5] By choosing the set of 270 polysemous nouns at random, we ensure that the degree of ambiguity closely matches the average number of 2.3 word senses per polysemous noun in GermaNet.

Since our primary goal is to extract example sentences in an automated way, the priority is the optimization of precision, neglecting recall. Therefore we focused on configurations which resulted in a precision of 0.85 or better.

Table 1 gives an overview of the results for the three individual mapping algorithms introduced in Section 4 (shown in rows 1 to 3) as well as for all pairwise combinations of the three individual algorithms (shown in rows 4 to 6). Precision is determined as the ratio between correctly identified mappings (i.e., true positives) and the overall number of automatically proposed mappings (i.e., true positives plus false positives). Recall is the ratio of true positives compared to the overall number of correct mappings in the gold standard (i.e., true positives and false negatives). F-score represents the harmonic mean between recall and precision. Among the individual algorithms, *Niemann/Gurevych* yields the best precision (0.85) for the test corpus and performs best in terms of F-score for both the training and the test corpora.[6]

Table 1: Evaluation results

System	Training			Testing		
	Prec.	**Rec.**	**F**	**Prec.**	**Rec.**	**F**
Lesk	**0.95**	0.25	0.40	0.81	0.27	0.41
Niemann/Gurevych	0.91	**0.29**	**0.44**	**0.85**	**0.30**	**0.44**
Textlink	0.90	0.22	0.35	0.79	0.23	0.36
Lesk + Niemann/Gur.	0.96	**0.32**	**0.48**	0.88	**0.35**	**0.50**
Lesk + Textlink	**1.00**	0.25	0.40	**0.90**	0.21	0.34
Niemann + TextLink	0.94	0.23	0.37	0.87	0.22	0.35

In order to test whether the three individual algorithms may yield better results when they are combined with one another, all pairwise combinations were evaluated as well. Here, the combination of the *Lesk* and the *Niemann/Gurevych* algorithms achieved the best F-score for both training and test corpora. It is therefore this combined algorithms that was used as the basis for the automatic harvesting of corpus examples.

6.2 Evaluation of the Automatic Harvesting of Corpus Examples

The algorithm for harvesting corpus examples is evaluated in terms of precision- and recall and an error analysis is provided. We also assess the effectiveness of our harvesting approach by comparing the overall size of WikiCAGe to existing sense-annotated corpora for German.

[6] Note that we have also conducted experiments with PageRank itself as in the approach by Agirre and Soroa (2009), but as these experiments – surprisingly – perform worse than the Lesk algorithm, we have not included the results in the table. For the task at hand, the results for PageRank are in an acceptable range only in combination with error measures well-known in the area of Information Retrieval as in the account of *Niemann/Gurevych*.

In order to inspect the quality of the harvested corpus examples, 261 automatically annotated Wikipedia articles were manually verified and, where required, post-corrected. We will make this manually verified excerpt of WikiCAGe freely available on the web. A precision of 0.89 with a recall of 0.91 prove the viability of the proposed method for automatic harvesting of sense-annotated data. In practise, this means that human post-correction is needed on average only for one out of ten harvested corpus examples in order to eliminate the remaining noise in the annotated data.

An analysis of those harvested corpus examples that are tagged with a wrong GermaNet word sense shows three predominant error types: (i) errors that are caused by an erroneous mapping between GermaNet and Wikipedia, (ii) errors that clash with the heuristic "one sense per discourse", and (iii) errors that are due to the software tool used for the detection of the target words. Erroneous mappings between word senses in GermaNet and articles in Wikipedia make up 6.0% of the total errors. An inspection of the "one sense per discourse" heuristic shows that this heuristic is violated by 3.3% of all marked target word occurrences. The last identified error type, i.e., errors that are due to the identification of the target word in the text, make up 3.0%.

Altogether, the presented approach has mapped 1 030 polysemous nouns from GermaNet to Wikipedia. Since GermaNet contains a total of 4 358 polysemous nouns, this amounts to a coverage of 23.6% for all such nouns and of 30.8% for all polysemous nouns that occur both in GermaNet and Wikipedia.

The successful mappings yield a total of 24 344 tagged word tokens occurring in 18 868 example sentences. This means that for each of the 1 030 nouns approximately 18 examples sentences are harvested on average. The large number of 18 868 harvested example sentences also leads to a sizable corpus of sense-annotated data. Table 2 shows a comparison of WikiCAGe to other existing sense-annotated corpora for German, i.e., to the manually constructed resources by Broscheit et al. (2010) and Raileanu et al. (2002) and the automatically created resource WebCAGe by Henrich et al. (2012). The number of sense tagged words that are listed separately per word class show that WebCAGe and the corpus by Broscheit et al. contain occurrences for words of all the three word classes of adjectives, nouns, and verbs, whereas WikiCAGe and the corpus by Raileanu et al. are limited to nouns only. By comparison, the overall number of sense-tagged words in WikiCAGe (24 344) is considerably larger than in all the other corpora.

7 Conclusion and Future Work

In this paper, we have presented an automatic method for enriching GermaNet senses with example sentences from Wikipedia. This method has the desirable side-effect of yielding a sense-annotated corpus for German, which we refer to by the name WikiCAGe, at the same time. We plan to make the excerpt of WikiCAGe, that was already

Table 2: Comparing WikiCAGe to other sense-tagged corpora of German.

		WikiCAGe	**WebCAGe**	**Broscheit et al., 2010**	**Raileanu et al., 2002**
Sense tagged words	adj. (a)	0	211	6	0
	nouns (n)	1 030	1 499	18	25
	verbs (v)	0	897	16	0
	a/n/v	1 030	2 607	40	25
Number of tagged word tokens		24 344	10 750	approx. 800	2 421
Domain independent		yes	yes	yes	medical domain

manually post-corrected for the evaluation of the presented algorithm, available to the larger research community.[7]

The algorithms used for the GermaNet to Wikipedia mapping and for the automatic harvesting of corpus examples were optimized for precision, resulting in an enrichment of 23.6% of all polysemous nouns in GermaNet. The motivation for optimizing on precision is to minimize the noise in the harvested data. The precision of 89% achieved for the automatic automatic harvesting of Wikipedia examples is sufficient to use the WikiCAGe corpus as is for NLP applications such as word sense disambiguation and statistical machine translation, whose statistical models are robust enough to cope with noisy training data. In future work, we plan to explore the precision vs. recall trade-off in order to increase the coverage of the methods described in this paper. This will increase the need for manual post-inspection of the harvested examples. However, since this post-inspection will not require any editing but just discarding of examples that do not match the candidate word sense, the amount of noise in the data does not have to be as tightly controlled. This in turn means that there is a priori no tight restriction on boosting recall and thus coverage.

Another direction for future work concerns the selection of those examples that best illustrate the use of a particular GermaNet word sense. As noted in Section 6.2, an average of 18 examples is harvested for each polysemous noun in GermaNet. In order to be able to select the most appropriate example(s) one needs to formulate clear criteria for what counts as a good example. Here we intend to build on the work of Kilgarriff et al. (2008). They specify the following properties of a good example: (i) it should represent a typical, exhibiting frequent and well-dispersed pattern of usage, (ii) it should be informative, helping to elucidate the definition, and (iii) it should be intelligible to learners, avoiding gratuitously difficult lexis and structures, puzzling or distracting

[7] `http://www.sfs.uni-tuebingen.de/en/wikicage.shtml`

names, anaphoric references or other deictics which cannot be understood without access to the wider context. Kilgarriff et al. further describe how these properties can be applied in practise to given example sentences, e.g., by using features such as the length of a sentence or the frequencies of words in a sentence.

Acknowledgments

The research reported in this paper was jointly funded by the SFB 833 grant of the DFG and by the CLARIN-D grant of the BMBF. We would like to thank Marie Hinrichs as well as the anonymous JLCL reviewers for their helpful comments on earlier versions of this paper. We are very grateful to Reinhild Barkey, Valentin Deyringer, Sarah Schulz, and Johannes Wahle for their help with the evaluation reported in Section 6.

References

Agirre, E., Aldabe, I., Lersundi, M., Martínez, D., Pociello, E., and Uria, L. (2004). The basque lexical-sample task. In Mihalcea, R. and Edmonds, P., editors, *Senseval-3: 3rd International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 1–4, Barcelona, Spain. Association for Computational Linguistics.

Agirre, E. and Lacalle, O. L. D. (2004). Publicly available topic signatures for all WordNet nominal senses. In *Proceedings of the 4th International Conference on Languages Resources and Evaluations*, LREC '04, pages pp. 1123–1126.

Agirre, E., Màrquez, L., and Wicentowski, R., editors (2007). *SemEval '07: Proceedings of the 4th International Workshop on Semantic Evaluation*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Agirre, E. and Soroa, A. (2009). Personalizing pagerank for word sense disambiguation. *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, (April):33–41.

Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. In *7th International World-Wide Web Conference*, WWW '98.

Broscheit, S., Frank, A., Jehle, D., Ponzetto, S. P., Rehl, D., Summa, A., Suttner, K., and Vola, S. (2010). Rapid bootstrapping of word sense disambiguation resources for German. In *Proceedings of the 10. Konferenz zur Verarbeitung Natürlicher Sprache*.

Erk, K. and Strapparava, C., editors (2010). *SemEval '10: Proceedings of the 5th International Workshop on Semantic Evaluation*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Fellbaum, C., editor (1998). *WordNet: an Electronic Lexical Database*. MIT Press.

Fernando, S. and Stevenson, M. (2012). Mapping WordNet synsets to Wikipedia articles. In Chair), N. C. C., Choukri, K., Declerck, T., Doğan, M. U., Maegaard, B., Mariani, J., Odijk, J., and Piperidis, S., editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation*, LREC '12, pages 590–596. European Language Resources Association (ELRA).

Gale, W. A., Church, K. W., and Yarowsky, D. (1992). One sense per discourse. In *DARPA Speech and Natural Language Workshop*.

Henrich, V. and Hinrichs, E. (2010). GernEdiT – The GermaNet Editing Tool. In Calzolari, N., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., and Tapias, D., editors, *Proceedings of the 7th International Conference on Language Resources and Evaluation*, LREC '10, pages 2228–2235. European Language Resources Association (ELRA).

Henrich, V., Hinrichs, E., and Vodolazova, T. (2011). Semi-Automatic Extension of GermaNet with Sense Definitions from Wiktionary. In *Proceedings of 5th Language & Technology Conference*, LTC '11, pages 126–130.

Henrich, V., Hinrichs, E., and Vodolazova, T. (2012). WebCAGe — A Web-Harvested Corpus Annotated with GermaNet Senses. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 387–396.

Henrich, V., Hinrichs, E., and Vodolazova, T. (to appear). An automatic method for creating a sense-annotated corpus harvested from the web. In *International Journal of Computational Linguistics and Applications*, volume 3.2.

Kilgarriff, A., Husák, M., McAdam, K., Rundell, M., and Rychly, P. (2008). *GDEX: Automatically finding good dictionary examples in a corpus*, pages 425–432. Institut Universitari de Linguistica Aplicada, Universitat Pompeu Fabra.

Koeva, S., Lesseva, S., and Todorova, M. (2006). Bulgarian sense tagged corpus. In *Proceedings of the 5th SALTMIL Workshop on Minority Languages: Strategies for Developing Machine Translation for Minority Languages, Genoa, Italy*, pages 79–86.

Kunze, C. and Lemnitzer, L. (2002). GermaNet – representation, visualization, application. In *Proceedings of the 3rd International Language Resources and Evaluation*, LREC '02, pages 1485–1491.

Leacock, C., Miller, G. A., and Chodorow, M. (1998). Using corpus statistics and WordNet relations for sense identification. *Comput. Linguist.*, 24(1):147–165.

Lesk, M. (1986). Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th annual international conference on Systems documentation*, SIGDOC '86, pages 24–26, New York, NY, USA. ACM.

Meyer, C. M. and Gurevych, I. (2011). What psycholinguists know about chemistry: Aligning wiktionary and wordnet for increased domain coverage. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, IJCNLP '11, pages 883–892.

Mihalcea, R. and Moldovan, D. I. (1999). An automatic method for generating sense tagged corpora. In *Proceedings of the American Association for Artificial Intelligence*, AAAI '99, pages 461–466, Menlo Park, CA, USA. American Association for Artificial Intelligence.

Niemann, E. and Gurevych, I. (2011). The people's web meets linguistic knowledge: automatic sense alignment of Wikipedia and WordNet. In *Proceedings of the 9th International Conference on Computational Semantics*, IWCS '11, pages 205–214, Stroudsburg, PA, USA. Association for Computational Linguistics.

Palmer, M., Dang, H. T., and Fellbaum, C. (2007). Making fine-grained and coarse-grained sense distinctions, both manually and automatically. *Natural Language Engineering*, 13(2):137–163.

Ponzetto, S. P. and Navigli, R. (2009). Large-scale taxonomy mapping for restructuring and integrating Wikipedia. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, IJCAI '09, pages 2083–2088, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Ponzetto, S. P. and Navigli, R. (2010). Knowledge-rich word sense disambiguation rivaling supervised systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1522–1531, Stroudsburg, PA, USA. Association for Computational Linguistics.

Raileanu, D., Buitelaar, P., Vintar, S., and Bay, J. (2002). Evaluation corpora for sense disambiguation in the medical domain. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, LREC '02.

Ruiz-Casado, M., Alfonseca, E., and Castells, P. (2005). Automatic assignment of Wikipedia encyclopedic entries to WordNet synsets. In *In: Proceedings of the Atlantic Web Intelligence Conference, AWIC-2005. Volume 3528 of Lecture Notes in Computer Science*, pages 380–386. Springer Verlag.

Santamaría, C., Gonzalo, J., and Verdejo, F. (2003). Automatic association of web directories with word senses. *Comput. Linguist.*, 29(3):485–502.

Suchanek, F. M., Kasneci, G., and Weikum, G. (2007). Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA. ACM.

Toral, A., Ferrández, O., Agirre, E., and Muñoz, R. (2009). A study on linking Wikipedia categories to WordNet synsets using text similarity. In *Proceedings of the International Conference*, RANLP '09, pages 449–454, Borovets, Bulgaria. Association for Computational Linguistics.

Wolf, E. and Gurevych, I. (2010). Aligning sense inventories in wikipedia and wordnet. In *Proceedings of the First Workshop on Automated Knowledge Base Construction*, pages 24–28.

Wu, Y., Jin, P., Zhang, Y., and Yu, S. (2006). A chinese corpus with word sense annotation. In *Proceedings of the 21st international conference on Computer Processing of Oriental Languages: beyond the orient: the research challenges ahead*, ICCPOL'06, pages 414–421, Berlin, Heidelberg. Springer-Verlag.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, ACL '95, pages 189–196, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zesch, T., Müller, C., and Gurevych, I. (2008). Extracting lexical semantic knowledge from Wikipedia and Wiktionary. In *Proceedings of the Conference on Language Resources and Evaluation*, LREC '08.

Kais HADDAR, Hela FEHRI, Laurent ROMARY

A prototype for projecting HPSG syntactic lexica towards LMF

Abstract

The comparative evaluation of Arabic HPSG grammar lexica requires a deep study of their linguistic coverage. The complexity of this task results mainly from the heterogeneity of the descriptive components within those lexica (underlying linguistic resources and different data categories, for example). It is therefore essential to define more homogeneous representations, which in turn will enable us to compare them and eventually merge them.

In this context, we present a method for comparing HPSG lexica based on a rule system. This method is implemented within a prototype for the projection from Arabic HPSG to a normalised pivot language compliant with LMF (ISO 24613 - Lexical Markup Framework) and serialised using a TEI (Text Encoding Initiative) based representation. The design of this system is based on an initial study of the HPSG formalism looking at its adequacy for the representation of Arabic, and from this, we identify the appropriate feature structures corresponding to each Arabic lexical category and their possible LMF counterparts.

1 INTRODUCTION

HPSG (Head-driven Phrase Structure Grammar) syntactic lexica have been developed as part of various applications such as parsing of natural language and construction of electronic dictionaries (Blache, 1995; Levine and Meurers, 2006; Pollard and Sag, 1994). The evaluation, reclaim and exploitation of the results provided by these applications are often seen as complex tasks because they are generally not based on normalised lexical resources. Additionally, the corresponding lexical resources are not described on the basis of the same underlying descriptors (or "data categories", to use the terminology of ISO 12620:2009 - see Ide and Romary, 2004). It is therefore important to define a conceptual framework that allows the definition of a pivot language between such resources in order to construct normalised representations from existing ones using merging and interoperability mechanisms. In line with the principles articulated in (Romary and Ide, 2004), the pivot language should be based on a standardised abstract meta-model combined with data categories. This in turn makes it possible to implement the pivot language using any kind of concrete syntax, i.e. an XML vocabulary, that maps onto the abstract model in an isomorphic way.

This paper follows these modelling principles with the main objective of proposing a method for the transformation of HPSG grammar lexica into a normalised pivot language that conforms to the principles of the LMF standard (ISO 24613), a framework that has been designed by the ISO committee TC 37/SC 4. More specifically, the pivot language will be used to estimate the real coverage of existing HPSG syntactic lexica and to merge them into integrated resources. It is worth noting that the same process can also be applied to lexica defined under other unification formalisms.

The proposed method takes into account both the specificities of the HPSG formalism as adapted to the Arabic language and the possibility of applying LMF to this formalism. This paper accordingly provides a precise overview of both formalisms with a view to identifying the adaptations that can be brought to HPSG and the data categories that must be added to LMF in accordance with ISO standard 12620. This study will enable us to elaborate a rule-based system for projecting HPSG syntactic lexicons towards LMF in a systematic way.

Section 2 of this paper presents the main reference works that have either covered standardisation attempts in the language resource domain or actual methods for projecting information across formalisms. We then briefly present in section 3 the HPSG formalism and the linguistic phenomena that may be covered by this formalism. We subsequently introduce in section 4 the LMF platform and its main principles. Section 5 focuses on our method of projecting an HPSG grammar lexicon for the Arabic language towards LMF as well as the experimentation of this method. We conclude on possible further ways this work could be extended to other types of lexical resources.

2 NORMALISATION AND PROJECTION ACTIVITIES

There have been several works dealing with the use of HPSG lexica for the processing of the Arabic language, including (Abdelkader, 2006), (Chabchoub, 2005), and (Elleuch, 2004). Still, the corresponding lexical resources are small and each of them concentrates on a particular task or syntactic phenomenon. Despite this, when considered together, they are highly complementary and merging them could definitely lead to a much richer lexicon containing a wide variety of lexical categories for the Arabic language. The fusion operation is, however, quite complex. The HPSG formalism can be implemented in different ways depending on the underlying theoretical assumptions as well as the actual language being dealt with. For instance, some features can be found in one lexicon but not in another one depending on the underlying linguistic viewpoint. For example, a feature like /slash/ will only appear in the context of elliptical or relative constructs. Additionally, the actual technical implementation when computerised may come in various organisations (e.g. feature granularity) and formats (e.g. XML, binary). This makes the recovery of the corresponding lexical content from one application to another extremely complex.

In order to achieve the reuse of such resources as well as their fusion, on the basis of standardised data categories, it is necessary to adopt a comprehensive normalisation strategy. To this end, research has been continuously carried out in recent years (see, for example, Ide and Véronis, 1995; Monachini and Calzolari, 1999; Atkins *et al.*, 2002) so that a community of researchers using a given formalism can benefit from the results, lexicons and resources developed by other communities using various formalisms. These endeavours have taken a range of dictionary models as a basis and suggested lexical abstraction adapted to automatic language processing, and at the same time has sought to retain the best compromise between simplicity and wide coverage. Specific attempts (see Eagles, 1996, for an example of a cross-formalism survey) have been made to standardise under-categorisation processes. They rely on a comparison between linguistic formalisms and NLP lexicons so that it is possible to carry out transformations from one formalism into another.

The continuous stream of projects and activities such as *GENELEX* (Genelex, 1994), *EAGLES*, *ISLE*, *MULTEXT*, *TEI* (Lemnitzer *et al.*, to appear), together with the mass of

expertise that these have contributed to, has led to the finalisation of an ISO standard on the representation of computerised lexical structures, namely ISO 24613 LMF within ISO committee TC 37/SC 4[1]. Published in 2008, this initiative has already been followed up by several attempts to provide reference implementations compliant with the future standard. In the domain of morphological lexica, for instance, the *Morphalou* project (Romary *et al.*, 2004) provides a full-form lexicon for French comprising 540,000 inflected forms. Nguyen and colleagues (Nguyen *et al.*, 2006) also describe the implementation of LMF for a full-featured lexicon for NLP purposes. A morphological lexicon, *ArabicLDB* (Khemakhem 2006), has been proposed for Arabic: it exemplifies in particular how to implement roots and vocalic patterns for Arabic morphology.

Similarly, several tools have already been proposed to help construct lexical databases in conformity with LMF together with standardised data categories. In particular, Lexus (Ringersma & Kemps-Snijders, 2007) is an online environment allowing one to both model a lexical structure compliant with the LMF meta-model and import lexical content accordingly. An endeavour to develop an editor with a constraint checker for the Arabic language has recently been proposed (Hasni *et al.*, 2006).

From the point of view of cross-formalism mapping, we can identify two main trends. The first one corresponds to the simplified use of specialised concept lexica. It presents the risk of getting ill-formed structures during analysis because it does not take into account specificities of each formalism. The second approach uses a rule-based system that takes the role of a parser. This approach is more efficient than the first as it always yields well-formed representations. For example, the method presented in (Kasper *et al.*, 1995) translates an HPSG grammar into a TAG (Tree Adjoining Grammar) representation. The underlying translator implements an algorithm that fulfils TAG specific constraints. These constraints define the mapping between the concepts used in the two formalisms. Conversely, in (Yoshinaga *et al.*, 2002), the authors propose an algorithm for converting LTAG into HPSG. While these formalisms treat the same set of constraints, the algorithm consists of mapping the constraints of LTAG one by one into HPSG equivalent ones.

The problem of evaluating and comparing grammars is treated by (Fehri *et al.*, 2006) and (Loukil, 2006). The proposed solution uses LMF as a pivot language and translates the input lexica into an LMF compliant structure.

The knowledge database DIINAR.1[2] encompasses 19,457 verbs, 70,702 deverbal entries, verbal nouns, active and passive participles, 'analogous' adjectives, nouns 'of time and operating place', 39,099 nominal stems, 445 tool-words and a prototype of 1,384 proper names. From this database, a large lexicon can be generated. As an application this database is associated with a morphological analyser called AraParse. This analyser uses a large stem-based lexicon generated form DIINAR.1.

Every entry is associated with morpho-syntactic specifiers at word-level and ensuring grammar-lexis relations between the lexical basis of a given word-form and other word-

[1] http://www.iso.org/iso/home/store/catalogue_tc/catalogue_tc_browse.htm?commid=297592

[2] DIctionnaire INformatisé de l'ARabe version 1 (see http://silat.univ-lyon2.fr/Presentation%20DIINAR.html)

formatives. The total amount of minimal words (e.g. of lemmas with their prefixes and suffixes) generated from the database is 7,774,938.

The AraParse lexicon contains:

− all the 121,522 unvocalised stem-entries of the DIINAR.1 database,

− all the vocalic schemes of each stem,

− all possible combinations of (prefixes, suffixes) for each couple of stem and vocalic schemes, and a set of specifiers (Genelex, 1994) containing morpho-syntactic information,

− a specifier of compatibility with possible clitics for each triple of stem, vocalic scheme, prefixes/suffixes combination.

The lexicon is organised in a letter tree structure. The principal advantage of the tree structure is that it greatly facilitates access while at the same time considerably reducing the size of the lexicon.

3 HPSG ARABIC LEXICON

HPSG has been proposed since the beginning of the eighties by Pollard and Sag (Pollard and Sag, 1994). It belongs to a family of formalisms based on constraints and descends from other previous unification formalisms such as GPSG (Generalised Phrase Structure Grammar), CG (Categorical Grammar) and LFG (Lexical Functional Grammar). It was initially designed to represent Romance or Germanic languages but, in the last decade, has been extensively applied to a wide variety of other language families.

In view of the specificities of the Arabic language (e.g. hierarchy types and agreement), its representation in HPSG requires the modification of some of HPSG's feature values and the addition of some Arabic specific features. To illustrate these specificities, we provide some examples of core features that are routinely needed in the context of the linguistic description of Arabic:

- **CFORM:** this feature is used for the description of the consonantal pattern of verbs and can take one of the values triliteral *thulāthī,* three consonant root or quadriliteral *rubā'ī,* four consonant root. This feature is necessary to identify different schemas that are useful for referencing the canonical or derivative form of the concerned lexical entry (e.g. *ktb* is *thulāthī* and *zqzq* is *rubā'ī*).
- **DENUDE:** is used for triliteral verbs and can take one of the values denuded *mujarrid*, when no extra letters are combined to the root, or increased *mazīd* when the root is combined with extra letters (e.g. *kataba* [wrote] is *mujarrid* and *'inkataba* [was written] is *mazīd*). It is useful in the same contexts as CFORM.
- **DIMINUTIVE:** can take one of the values non diminutive *ghair muṣaghar* or diminutive *ṣīghit al-ttaṣghīr*. With this feature we can distinguish canonical forms from inflected ones (e.g. *kalb* [dog], *kulayeb* [small dog]).
- **RELATIVE:** this feature has the same role as DIMINUTIVE. It can take one of the values relative *manṣūb* or non relative *ghair manṣūb* (e.g. *tūnisī* [Tunisian] is *manṣūb*).
- **NATURE:** is used to give the semantic role of a noun (e.g. 'gift' vs. 'giver'). Among the values that can be taken by this feature, we have: agent noun, *ism fā'il* (e.g. kātib [writer]), patient noun, *ism maf'ul* (e.g. *maktūb* [written]), verbal adjective, *ṣifa muchabbaha* (e.g. *shujā'* [courageous]). Every value taken by this feature represents a lexical entry and a derived or inflected form.

- **RADICAL:** This feature gives the root of a verb (e.g. the root of *kataba* is *ktb*).

We have also modified the definition of the feature NFORM. NFORM gives the different forms that an Arabic noun can have. The values of this feature are: *mutaṣṣarf muchtak* (inflectional derivative), *mutaṣṣarf jāmed* (inflectional inert) and *ghair mutaṣṣarf* (non inflectional). With this feature we can know if a canonical form can have inflected (or even derived) forms or not.

As in (Dahdah, 1992), we consider that an Arabic word can be a noun *ism*, a particle *harf* or a verb *fi'l*. We can mention here that other categorisations have been used, which usually add the adjective as a fourth category. In what follows we are going to give a preview on the considered word categories.

3.1 Nouns

What distinguishes the Arabic language from other languages is the fact that the lexical category for a noun can be broken down into several subcategories to distinguish between frozen (e.g. proper nouns *asmā' al-'alam*, place nouns *asmā' al-makān*), non-frozen (e.g. adjectives *al-ṣṣifa al-muchabbaha*, noun-agent *ism elfā'il*) and inflected nouns (e.g. demonstrative pronouns *asmā' al-ichāra*, pronouns *al-ḍamā'ir*).

Note that all nouns share the same AVM (Attribute Value Matrix) model, represented in Figure 1, where they only differ from one another depending on associated feature values. In the case of adjectives, we add to this skeleton the feature MOD in feature HEAD. The feature MAJ is used to introduce the lexical category of a word (e.g. verb, noun and preposition). The feature DEFN gives the noun the property of definiteness.

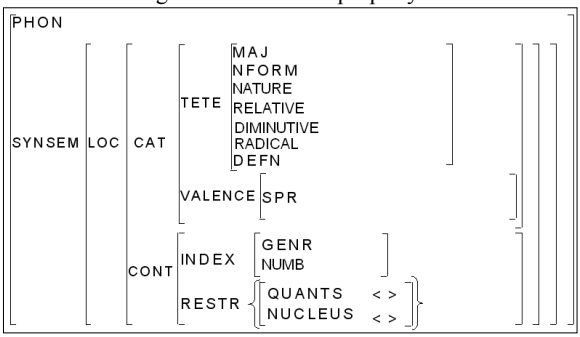


Figure 1: A noun AVM model

In a noun AVM, all morphological features are regrouped in the HEAD feature. The only syntactic feature is SPR. This feature introduces the element that precedes a word (e.g. a demonstrative pronoun is a potential value of a noun SPR). Although agreement features are considered as semantic features in HPSG, they are founded in the morphological part in LMF. For each subcategory of the category noun, we must specify an adequate AVM, for example:

Inert variable noun (*elism elmutassaref eljamed)*: concrete noun' *ism al-thāt*, or abstract noun' *ism al-ma'na*. Figure 2 and Figure 3 are two examples of a concrete noun and an abstract noun.

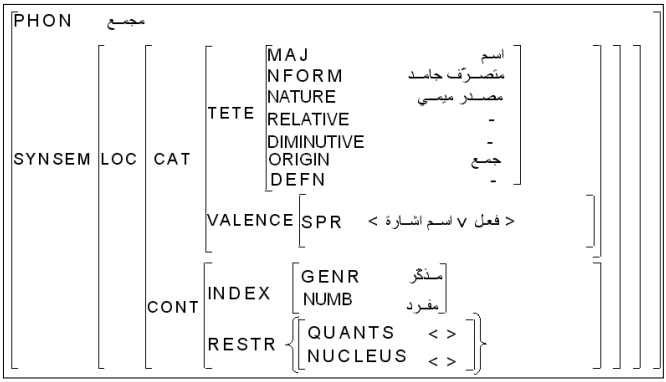


Figure 2: AVM of the noun *majmaʿ* (collector**)**

Figure 2 shows that the noun *majma'* (collector) is an inert indefinite noun, non diminutive and non relative. This information is given respectively by the features NFORM, DEFN, DIMINUTIVE and RELATIVE. *Almajma'* is the definite form of *majma'*. The features NATURE, ROOT and SPR show that this noun represents an m-initial infinitive *maṣdar mīmī* having like root *jm'* and that it can be preceded by a verb or a demonstrative pronoun. Note that the noun *majmaʿ* (collector) is a masculine noun and singular.

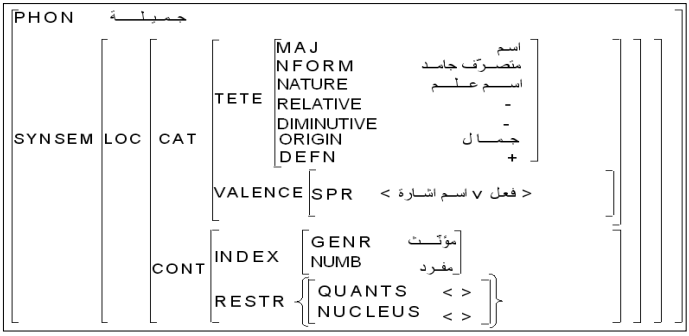


Figure 3: AVM of the proper noun *jamīla*

In Figure 3, we notice that the features that changed value are the features NATURE, DEFN ORIGIN, and GENR since *jamīla* is a proper noun, definite and feminine. The *masdar* (ORIGIN) value of the proper noun *jamīla* is *jamāl*. This noun can be preceded by a verb or by a demonstrative pronoun (the SPR value). In Arabic, a proper noun can be used as an adjective and in this case it is necessary to apply some modifications to the appropriate AVM.

3.2 Particles

Particles are words that serve to situate events and objects in relation to time and to space. They give a text a coherent sequence. Particles represent another category for an Arabic

word and can be construction letters *hurūf mabān* or significance letters *hurūf mabān*. Significance letters are divided into two subcategories: the first regroups particles that have no effect (e.g. morphological, grammatical) on the word whereas the second includes particles that have some declination effects on the noun (e.g. prepositions, particles of the vocative) or on the verb (e.g. elision particles, subjunctive particles) or on both (e.g. conjunctions).

For particles, we proposed two different AVM models; one is used for prepositions and the other for particles. Both AVM models are illustrated respectively in Figure 4 and Figure 5.

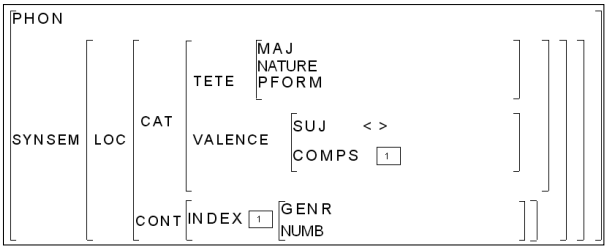


Figure 4: AVM model of a preposition

Preposition morphological features are regrouped in the HEAD feature. Note that agreement features are relative to the object introduced by this preposition.

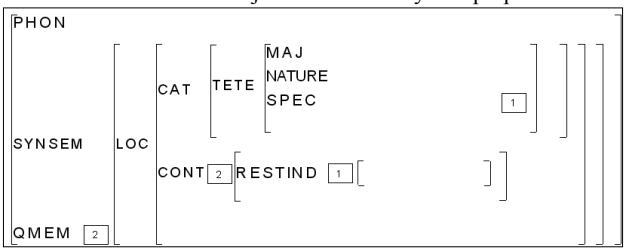


Figure 5: AVM model of an elision or subjunctive particle

Note that a preposition AVM is different from that of an elision or subjunctive particle. Elision and subjunctive particles are words that can precede verbs. This difference resides in the features HEAD, VALENCE and CONT. In the preposition AVM we remark the existence of the feature PFORM as a morphological feature. In the tool AVM, the feature SPEC replaces it. Additionally, the features VALENCE and INDEX and agreement features exist for a preposition but not for a tool. In the following figures we are going to give some examples of AVMs that correspond to different categories of particles.

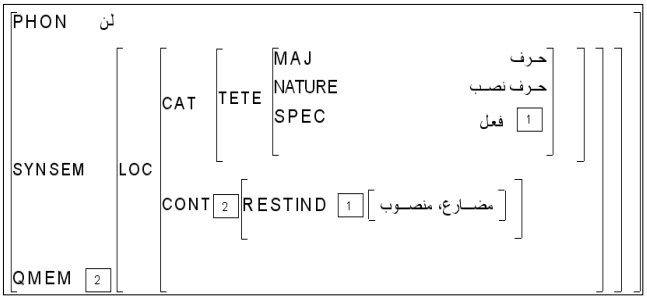


Figure 6: AVM of the particle *lan*

Figure 6 represents the tool AVM of the Arabic word *lan*. This word is a particle (value of MAJ) that belongs to significance letters (value of NATURE) and that precedes a verb (SPEC) in the subjunctive mood *manṣub*. The verb must be conjugated in imperfect tense *mudhāraʿ* (value of RESTIND).

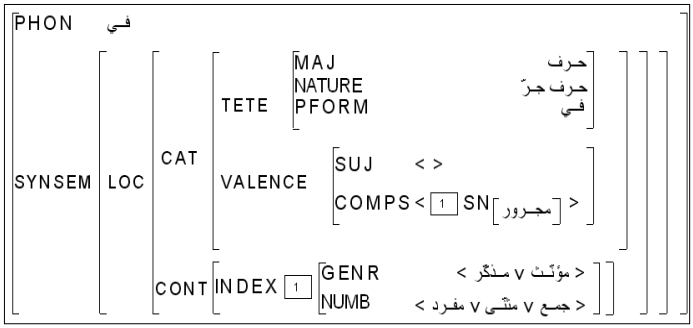


Figure 7: AVM of the particle *fī*

Figure 7 represents an example of a preposition AVM. In this AVM we note the existence of the feature VALENCE. The particle *fī* admits an object described by the feature COMPS. This object must be a genitive nominal phrase *majrūr*. The agreements of this object are expressed in the feature INDEX.

3.3 Verbs

A verb usually indicates a real action on the part of the subject that occurs over a period of time (e.g. *kataba* [wrote] and *qara'* [read]). It is a fundamental element to which the sentence constituents are connected directly or indirectly. In Arabic, the basic source of all the forms of a verb is called the root of the verb. The root is not a real word; rather it is a sequence of three consonants that can be found in all the words that are related to it. Most roots are composed of three letters, a very few are composed of four or five letters. The verb is therefore the stem of a word family (Ammar and Dichy, 1999).

Schemes are applicable to roots and these applications produce a new verb. For example, from the root *kharaja*, meaning "to go out", we obtain the verb "to make go out" by doubling the central consonant to make *kharraja*. The scheme can be considered as a formal representation established by three or four consonants *f`l* that are totally vocalised, or as a

mould containing the root. Altogether there are 19 verbal schemes that can be either nude[3] or increased by taking three consonants from the root and modifying the vowels, redoubling the second letter of the root, or inserting affixes (prefix, infix and suffix). The longer verbs conjugate with the same prefixes and suffixes as the original verb. Therefore, a root can generate most of the 19 verbs and the corresponding schemes can give 22 different conjugation patterns. In fact, there is a scheme *fa'ala* that can have three variations different from conjugation according to the nature of the vowel used in the second consonant of the root: *yaf' ulu*, *yaf' ilu*, and *yaf' alu*. Also, the scheme *fa 'ila* can give two variations different from conjugation for the same reason (Dahdah, 1992).

The AVM model for verbs is illustrated in Figure 8.

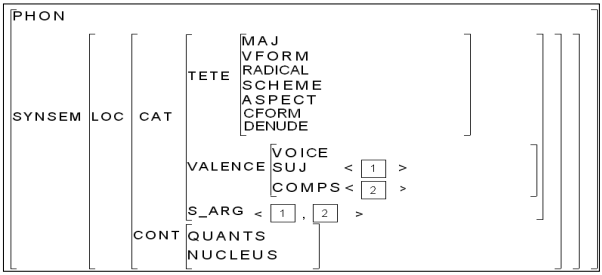


Figure 8: Model of an AVM of a verb

The morphological verb features are always given in the feature HEAD, the syntactical ones in the feature VALENCE and the semantic ones in the feature CONT. In the following figure we give an example of a verb AVM using *kataba*.

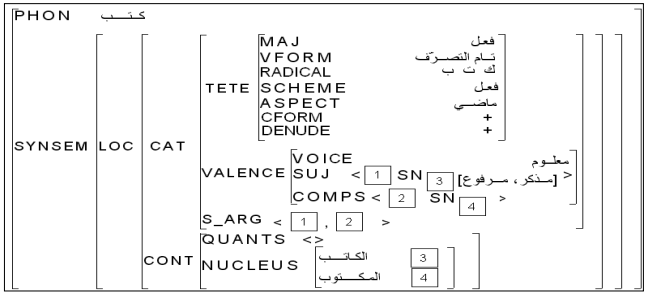


Figure 9: AVM of the verb *kataba*

The example in Figure 9 shows that the verb *kataba* is conjugated in the perfect tense, in active voice and has as a root *ktb*. This verb can subcategorise a subject and an object. These values are contained in the feature S-ARG describing a structure list. This feature is considered as a valence feature concatenation. In addition, we remark that a subject carries a specification on its index: the nominal category should be masculine and nominative.

[3] the verb appears in its canonical representation (as opposed to "augmented")

4 LMF MODEL

After presenting the HPSG formalism adapted for the Arabic language and defining the appropriate AVM for every lexical category (noun, particle and verb), we now describe the ISO LMF specification platform under the specific perspective of the projection of lexical structures. Through this study we can understand LMF specificities and subsequently identify the common points that are processed by the two abstract models (HPSG and LMF). As a result we can extrapolate a method allowing the projection from HPSG lexicons into LMF.

The objective of LMF is to propose a modular data model that is independent from any particular lexicographic theory and allows abstraction from concrete representation (e.g. proprietary syntax, XML structure based on the TEI guidelines, database model, etc.). The modelling framework, initially experimented with in the terminological domain (Romary, 2001), operates at the conceptual level: it aims to identify the essential components of a generic lexicographic model, to describe the constraints governing their arrangement, and to identify the descriptors (data categories) that are associated with them. The LMF standard is based on a core model together with a set of five extensions, as explained in the following subsections.

4.1 Core part

The core model of LMF specifies the concepts of lexicon, word, form and sense in keeping with a semasiological view of lexical structures[4]. It describes information concerning a lexicon and the basic hierarchy of the information that can be included in a lexical entry. The core model is illustrated in Figure 10.

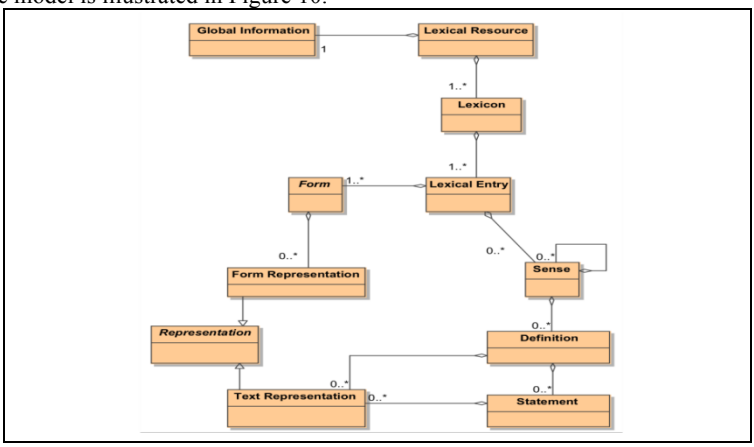


Figure 10: LMF core model

[4] Similarly, the TMF standard (ISO 16642) is dedicated to onomasiological structures as encountered in conceptual systems and terminologies.

In Figure 10, the Lexical Resource component is a singleton that represents the entire resource, seen as a container for one or more lexicons. The Lexicon component is the informational locus for all lexical entries of a source language within the database. A lexicon must contain at least one lexical entry and must not allow certain subclasses. The Global Information component contains the administrative information and other general attributes of a lexicon (e.g. the metadata associated to a lexical resource). The Lexical Entry component may represent a word, a composed expression, or an affix in a given language.

With the semasiological perspective in mind, the Lexical Entry component instantiates the link between the Form and Sense components. A lexical entry may have one or several different forms and may have none or several different meanings. The Entry Relation component allows one to represent cross-references between two or more lexical entries within or across lexicons. It can contain attributes that describe the type of relationship.

The LMF core model can be extended to satisfy further requirements bound to the treatment of specific lexicographic aspects. Several possible extensions are described in the LMF standards, among which we may mention the morphological extension, the syntactic extension, the semantic extension, the inflectional paradigm extension and the multilingual annotations extension. These extensions must be selected according to the needs of the designer of a specific lexical model. In our case we will put a specific emphasis on the morphological, syntactic and semantic extensions, as presented in the following sections.

4.2 Morphological extension

The goal of this extension is to provide mechanisms that support the development of the NLP lexicons describing the morphology of the lexical entries.

Example 4.1 (The Arabic word *'ayn* [eye]):
The object diagrams of Figure 11 and Figure 12 represent two different ways to describe the inflectional part of the Arabic word '*ayn* (eye).

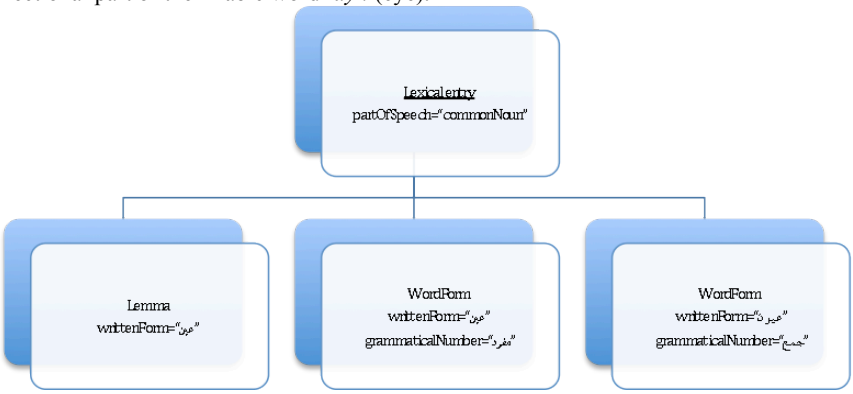


Figure 11: Objects diagram representing the inflectional part of *'ayn* "ع-ي-ن" without inflectional paradigm

As mentioned earlier, the LMF structure depicted in Figure 11 can be implemented in any specific format and in particular may be serialised according to any kind of XML representation as long as it is isomorphic to the underlying LMF model. In the rest of the

paper, we will more specifically apply our examples using the Text Encoding Initiative (TEI) framework, benefiting from a widely accepted background for our concrete representation, and also making full use of the customisation facilities offered by the TEI infrastructure. The elementary lexical structure presented in Figure 11 can easily be serialised in TEI, as follows[5]:

```
<entry>
    <gramGrp>
      <pos>commonNoun</pos>
    </gramGrp>
    <form type="lemma">
      <orth>عين</orth>
    </form>
    <form type="inflected">
      <orth>عين</orth>
      <gramGrp>
        <number>مفرد</number>
      </gramGrp>
    </form>
    <form type="inflected">
      <orth>عيون</orth>
      <gramGrp>
        <number>جمع</number>
      </gramGrp>
    </form>
  </entry>
```

Note that in Figure 11 above, two inflected forms of the singular word *'ayn* (eye) and of the plural word '*ayūn* are represented without passing through an inflectional paradigm. In this case, every inflected form must be described in an object of the class InflectedForm.

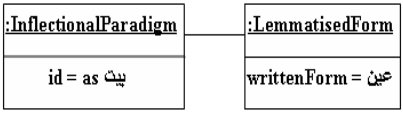


Figure 12: Diagram of objects representing the inflectional part of *'ayn* with inflectional paradigm

In Figure 12, the two inflected forms of *'ayn* must be generated automatically using the inflectional paradigm. The paradigm used called "*as bayt*" (house) consists of inserting the letter *ū* in the fourth position of the word *'ayn*. It can be shared with other lexical entries where the inflectional part is like *'ayn* (e.g. *bayt* and *bayūt*).

4.3 Syntactic extension

The syntactic extension of the LMF standard aims at providing ways to describe the word properties when combined with other words and phrases in a sentence.

Example 4.2 (the Arabic word *kataba*):

[5] TEI elements belong to the namespace http://www.tei-c.org/ns/1.0

The verb *kataba* (wrote) subcategorises a subject that must be a nominal phrase (NP) and an object that must also be a nominal phrase. This syntactic behaviour is described in Figure 13, taking into account that a verb can admit more than one syntactic behaviour.

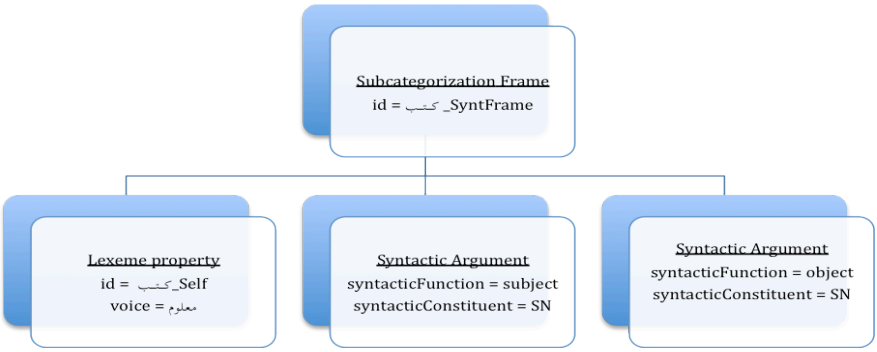


Figure 13: Diagram of objects representing the syntactic behaviour of the verb *kataba* "كتب".

Figure 13 shows how the syntactic behaviour is represented in an object of the class Sub-categorisation Frame. When a verb has more than one frame, each version of this verb is considered as a new entry and will be projected in LMF differently (*kataba alwaladu* and *kataba alwaladu risālata*). This object is combined with as many objects of the class Syntactic Argument as the number of constituents of the verb *kataba* requires.

4.4 Semantic extension

With the semantic extension of LMF, it is possible to describe a semantic profile together with the relations with other meaning within the lexical database. The extension also provides the means of linking syntactic and semantic description, typically at the argument level.

Example 4.3 (the Arabic word *kataba*):
In the example described below, we present an object diagram illustrating the relationship between the syntactic and the semantic part of the verb *kataba* (wrote).

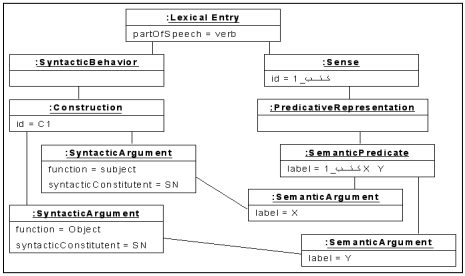


Figure 14: Object diagram representing the relationship between the syntax and the semantic of the verb *kataba*

In Figure 14, the subject is labelled as X and the object as Y. If we suppose that X represents *al-walad* (the boy) and Y *al-dars* (the lesson), then the association of these constituents with the verb *kataba* gives the significance of *the boy wrote the lesson* expressed in the object SemanticPredicate.

From the overview of the basic LMF mechanisms presented above we can see how sub-categorisation phenomena, which are essential in the HPSG formalism, can be taken into account in the LMF standard. The main difference between the two representation models essentially resides in the manner in which the lexical entries are actually organised. A canonical (or derived) form with all its inflected forms constitutes one single lexical entry in LMF. In HPSG, however, each form, whether it is derived, canonical or inflected, constitutes a unique lexical entry. We can also identify features in HPSG that are specific to the Arabic language and have no equivalent in LMF as it stands as a published standard. For these, we will have to provide specific extensions by describing new data categories, which will then be submitted to the Data Category Registry (ISOCat.org). For instance, most data categories presented in section 3 for the morphological description of the Arabic language have at present no equivalent in ISOCat.

5 PROJECTING HPSG LEXICAL STRUCTURES IN LMF

In this section, we present the proposed method for the projection of a syntactic HPSG lexicon into an LMF compliant representation. This method is designed on the basis of the LMF meta-model and on the above-mentioned extensions applied to this model, incorporating the characteristics of the HPSG theoretical framework. The method that we propose is articulated around two essential steps, namely the identification of a projection rule system and the projection process itself.

5.1 Identification of projection rule system

The first phase consists of studying the various lexical categories represented in HPSG in order to identify the nature and the information associated to each feature of an AVM adapted to the Arabic language. During projection, each such feature will be transformed into an LMF class attribute. The intrinsic nature of a feature — whether morphological, syntactic or semantic — helps us to know to which LMF component the feature is going to be projected. We can then limit the number of the classes that will be affected by the projection accordingly.

The feature type (e.g. morphological, syntactical) helps us to identify in which class the projection is going to be made. If we take the case of the feature RADICAL, the feature keeps the same value for the canonical (or derived) form and its inflected forms. We can say, therefore, that it is a feature that relates to the class LexicalEntry. However, if we take the case of the feature SCHEME, we note that this feature changes from an inflected form to another and in this case it relates to the class InflectedForm. In the next paragraph, we present the specific rules that we have identified for the morphological features.

5.1.1 Projection rules for morphological features

The rules corresponding to morphological features are divided into two types: those that can be applied to all lexical categories (noun and verb, particle, non-inflected noun and non-inflected verb) and those that may only be applied to specific categories.

Example of a rule applicable to verbs only:

R_{1m} : ($Feature_{HPSG}$=PHON) ∧ (Value(SCHEME)∈FDC)→*in LemmatisedForm: att=lemma* ∧ *val=Valeur(PHON)* ∧ *in InflectedForm: att=orthography* ∧ *val=Value(PHON)*

In the rule R_{1m}, FDC designates the set of all models representing the canonical and derived forms relative to a verb (FDC ={CaCaCa, CaCCaCa, CaCaCaCa,CaaCaCa,CaCaCaCaCa, CaCaaCaCa,CiCCaCaCa, CiCCaCaCa, CaCCaCaCa, CiCCaCCaCa, CiCCaCCaCa, CiCCaaCaCa, CaCaCCaCa, CiCCaCaCa, CiCCaCaCaCa} { فعل، فعلل، فعّل، فاعل، أفعل، تفعّل، تفاعل،انفعل، افتعل، افعلّ، استفعل، افعوعل، افعولّ، افعالّ، تفعلل، افعللّ، افعنلل}). Note that the function Value allows returning an HPSG feature value. We can take the case of the verb *'akhraja* أخــرج (to extract). The model for this verb is *'af'ala* أفــعــل and belongs to FDC. Therefore, after having applied the rule R_{1m}, a new attribute is added in the class LemmatisedForm and named lemma and the value is equal to *'akhraja* أخــرج, and another is added in the class InflectedForm with the name orthography, and its value is equal to *'akhraja* أخــرج.

Example of a rule applied only to nouns:

R_{3m}: ($Feature_{HPSG}$=PHON) ∧ (Value(NOMB)=SINGULAR) ∧ (Value(GENR) = MASCULIN) ∧ (Value(DIMINUTIVE)=non diminutive) → *in LemmatisedForm: att=lemma* ∧ *val=Value(PHON)* ∧ *in InflectedForm: att=orthography* ∧ *val=Value(PHON)*

The rule R_{3m} is applied to the canonical or derived forms of a noun. The application of this rule results in the addition of two new attributes: the first one is added to the class LemmatisedForm and named lemma and has as its value the HPSG feature PHON, and the second is added to the class InflectedForm. The second attribute is named orthography and has as its value the HPSG feature value.

Example of a rule applicable to verbs and nouns only:

R_{5m}: ∀ $Feature_{HPSG}$ ∃ $attribute_{LMF}$: $attribute_{LMF}$ ≡ $Feature_{HPSG}$ ∧ ¬ Variable(Value ($Feature_{HPSG}$)) → *in LexicalEntry: att = attribute*$_{LMF}$ ∧ *val = Value*($Feature_{HPSG}$)

The rule R_{5m} is applied to the features that always take the same values for the canonical form (or derived) and its inflected forms. Let us note here that the function Variable is a function that returns true if a feature keeps the same value for the canonical form (or derived) and all its inflected forms. Figure 15 represents an example of the application of the rule R_{5m}.

Canonical form	Value(MAJ)
ذهب	verb

Inflected form	Value(MAJ)
ذهب	verb
ذهبت	verb
ذهبا	verb
ذهبنا	verb
ذهبوا	verb
...	verb

⟹ Variable (Value (MAJ))=false ⟹ R_{5m}

Figure 15: Example of the application of the rule R_{5m}

We can observe that in Figure 15, the value of the feature MAJ remains unchanged for the verb *dhahaba* ذهب (to go) and for all its inflected forms. In this case, we must apply the rule R_{5m} since the feature MAJ has its equivalent in LMF that is equal to the attribute PartOfSpeech.

Example of a rule applicable to particles, non-inflected nouns and non-inflected verbs:

R_{9m}: $Feature_{HPSG} = MAJ \rightarrow$ *in LexicalEntry: att = GrammaticalCategory* $\wedge$ *val = Value(MAJ)*

The rule R_{9m} is applied only to the features MAJ and PHON given that these features exist in any type of particles.

5.1.2 Identified rules for syntactic features

The identified rules for syntactic features are considered to be paradigms. Several lexical entries can have the same syntactic behaviour and in this case they share the same projection rule through their identifier. Rule R_{1syn} is an example of this in a case where the value of the HPSG feature can have more than one value at a time (complex). This rule is defined formally as follows:

R_{1syn}: *Complex* $Value(Feature_{HPSG}) \rightarrow$ *in SyntacticArgument: att = function* $\wedge$ *val = function*$(Feature_{HPSG})$ $\wedge$ *att = SyntacticConstituent* $\wedge$ *val =* $Value(Feature_{HPSG})$

Among the features to which we apply the rule R_{1syn} are SPR, TOPIC, ATTRIBUT and COMPS. Note that rule R_{1syn} must be applied as many times as there are values for the feature in question.

The features SUJ and COMPS will be projected by using rule R_{1syn} because their values are composed. On the other hand, VOICE will be projected by using rule R_{2syn} as this feature admits its equivalent in LMF and its value is simple:

R2syn : atomic (value (attributeHPSG)) $\wedge \exists$ *attributeLMF : attributeLMF*$\equiv$ *attributeHPSG* ® *in : Self att = name (attributeLMF)* $\wedge$ *val = value (AttributeHPSG)*

5.1.3 Projection rules for semantic features

Semantic features are represented in the feature CONT, which contains a list of quantifiers. The semantic part, which we consider here, is represented by the feature NUCLEUS whose value is generally an AVM composed of the features agent-noun and patient-noun if it is about a verb, but is empty otherwise. So far we have identified only one projection rule applicable to the semantic features illustrated by R_{1sem}.

R_{1sem}: *if Nucleus* $\neq$ <> $\rightarrow$ *in SemanticArgument: att=agent-noun* $\wedge$ *val=value(agent-noun)*

The same rule R_{1sem} can be applied to the feature patient-noun. The attribute will be projected to the class SemanticArgument. Note that a lexical entry projection must be made in the appropriate locus (i.e. component) of the LMF model. The lexicon under work already contains other lexical entries that have been projected. If we take the case of the verb *dhahaba* (he goes) and the inflected form *dhahabnā* (we go), we observe that in HPSG these two lexical entries have two independent AVMs. Whereas, at the time of the projection, the two entries only represent one lexical entry of which *dhahaba* (he goes) is a canonical form and *dhahabnā* (we go) its inflected form.

5.2 Projection process

The projection phase, the goal of which is to apply the corresponding projection rules to all features characterising a lexical entry, is based upon three essential stages. These stages are applied iteratively on all lexical entries included in the lexicon. These stages are illustrated in Figure 16.

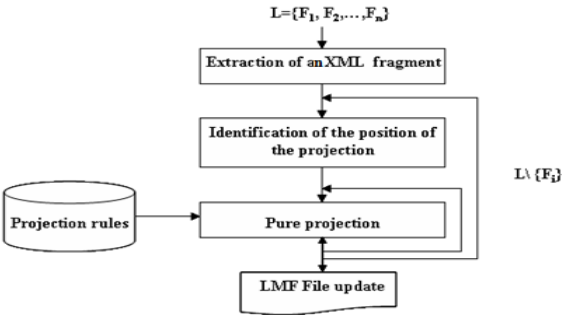
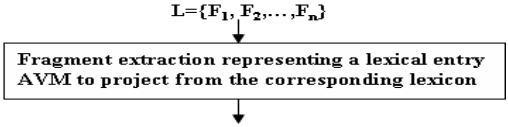


Figure 16: Stages of proposed method

The input for our process is a set of lexical entries that can represent verbs, nouns, particles or a combination of these categories. A projection starts with the first open lexicon. In the following paragraphs we are going to give an idea of the method stages required for the extraction of every lexical entry, its XML fragment, the identification of its projection position and the projection using the adequate rules.

5.2.1 Extraction of XML fragments for AVMs associated with lexical entries

The first phase consists in extracting the XML file fragment, which represents the lexical entry AVM to be projected. A fragment extraction phase is essential because the projection is made on a word by word basis. Figure 17 illustrates this stage.



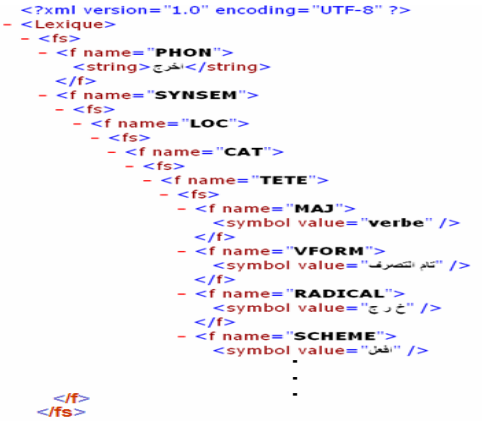


Figure 17: XML fragment extraction of a lexical entry in conformity with ISO 24610-1

The example in Figure 17 concerns the verb *akhraja* أخـرج (to take out). At this stage the description of the various features characterising this verb is encoded according to the ISO-TEI standard for feature-structures (ISO 24610-1).

5.2.2 Identification of projection position

The projection basic algorithm uses some tests that concern the verification of the lexical entry form to be projected and the position of the projection. Figure 18 illustrates the position of these tests at the time of the projection process.

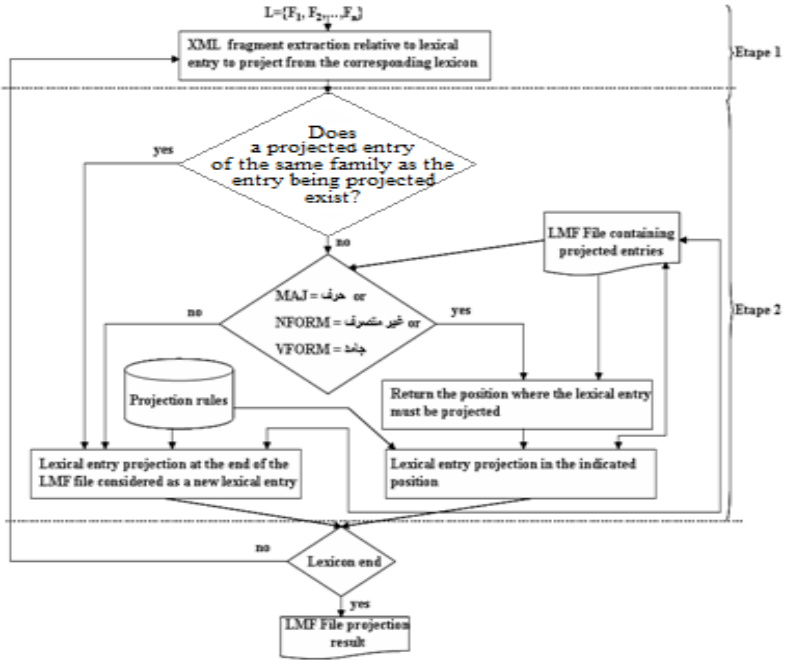


Figure 18: Overview flow chart

As Figure 18 indicates, the process first consists of verifying if the lexical entry under projection can admit inflected forms or not. This verification is essential as in the case where inflected forms exist it is necessary to know in which position the projection must be made. We need to remember that in LMF a lexical entry is composed of the canonical or derived form and all its inflected forms whereas our starting point is a lexicon containing different lexical entries that can be canonical, derived or inflected forms represented according to the HPSG formalism. These entries are organised according to the choices made by the lexicon's designer. Therefore, the LMF compliant output contains a number of lexical entries that must be lower or equal to the number of existing lexical entries in the HPSG lexicon. Let us note that features that allow us to know if a lexical entry admits the inflected forms or not are NFORM and VFORM for nouns and verbs respectively. For particles we have no inflected forms.

The process then moves to projection position verification. If the lexical entry to be projected can admit inflected forms, it is necessary to browse the LMF file containing the projected lexical entries to know if a lexical entry of the same class has been projected. This research is based on:

- the values of the features RADICAL and DENUDE in the case of a verb representing a canonical form or an inflected form of a canonical form,
- the values of the features RADICAL and SCHEME for the rest of the verbs,
- the values of the features NATURE and RADICAL for the nouns.

5.2.3 Pure projection

The phase "pure projection" consists of browsing the XML file already extracted in the previous phase in order to extract features representing the lexical entry to project. On the basis of the survey of the various AVMs already done in the first stage, we now know the different features forming every lexical category and can apply the corresponding projection rule. Figure 19 illustrates this stage.

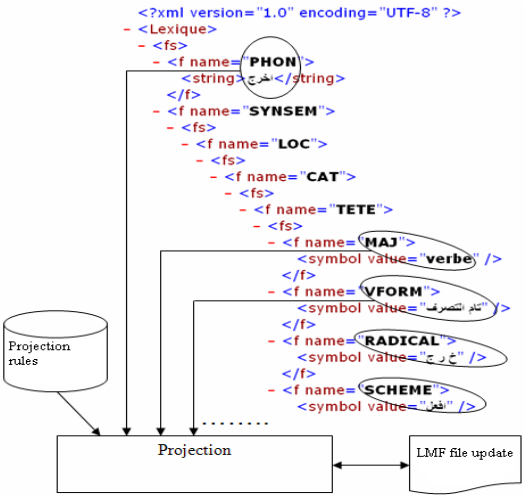


Figure 19: Projection of a word's features AVM

For every lexicon entry, we extract a feature together with its value and project them using the adequate projection rule until arriving at the end. The example in Figure 19 relates to the verb *akhraja* "أخـــرج" (to take out).

The proposed method is independent from the HPSG lexicon organisation. The order of the lexical entries in the lexicon does not have any impact on the projection. We can find, for example, in the HPSG lexicon a canonical form before its inflected forms or the opposite. Also, the addition or the adoption of an HPSG feature does not influence the result obtained from the projection. Our established projection rule system processes all possible cases that can arise in the Arabic language. The projection of another HPSG lexicon using another language than Arabic is possible. It is sufficient to modify some projection rules in accordance with the particularities of the new language.

This method helps us to then process both the conception and the implementation phases, which is the subject of the following section.

6 THE IMPLEMENTATION PHASE

Having presented the proposed method for the projection of HPSG into LMF in the previous section, in this section we are going to describe the achieved prototype in order to validate this method.

6.1 General architecture of the achieved prototype

The prototype allows projection of one or several existing HPSG syntactic lexicons into LMF. The projection will give us a normalised representation of these lexica and therefore encourages their merging. Our prototype is composed of two modules. The first concerns the projection phase and is applied after having chosen and opened one or several HPSG lexicons. The second concerns the generation of the LMF file resulting from the projection. Figure 20 depicts these different modules.

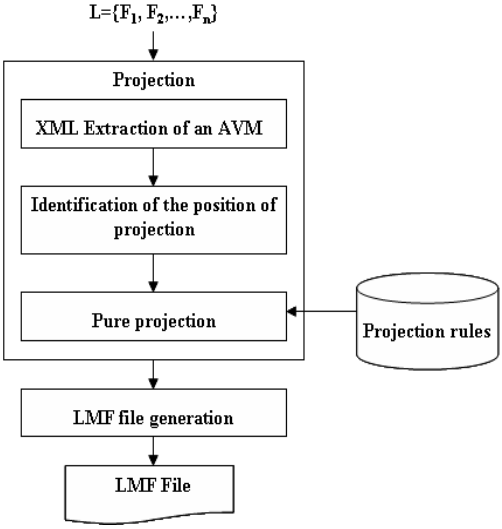


Figure 20: Prototype's architecture

In order to execute the projection, the user must open at least one lexicon that is represented in HPSG. The system will then browse every open lexicon entry by entry to extract the XML fragment relative to the corresponding entry. For every extracted XML fragment the system also extracts every attribute and its value and projects them using the base of the suitable projection rules.

The HPSG lexicon to be projected is in turn composed of one or several AVMs. An AVM is itself composed of features and values. A feature value can be a simple value or a composed value (list or AVM). As for the LMF lexicon, the result of the projection is composed of a set of elements. Every element can be composed of other elements and/or the data categories (DC). Every data category constitutes of an attribute having a value.

6.2 Some prototype functionalities

The implemented prototype allows the projection of a lexicon represented in XML and respects the standard format of representation of feature structures introduced (Hasni *et al.*, 2006). Figure 21 illustrates the projection process.

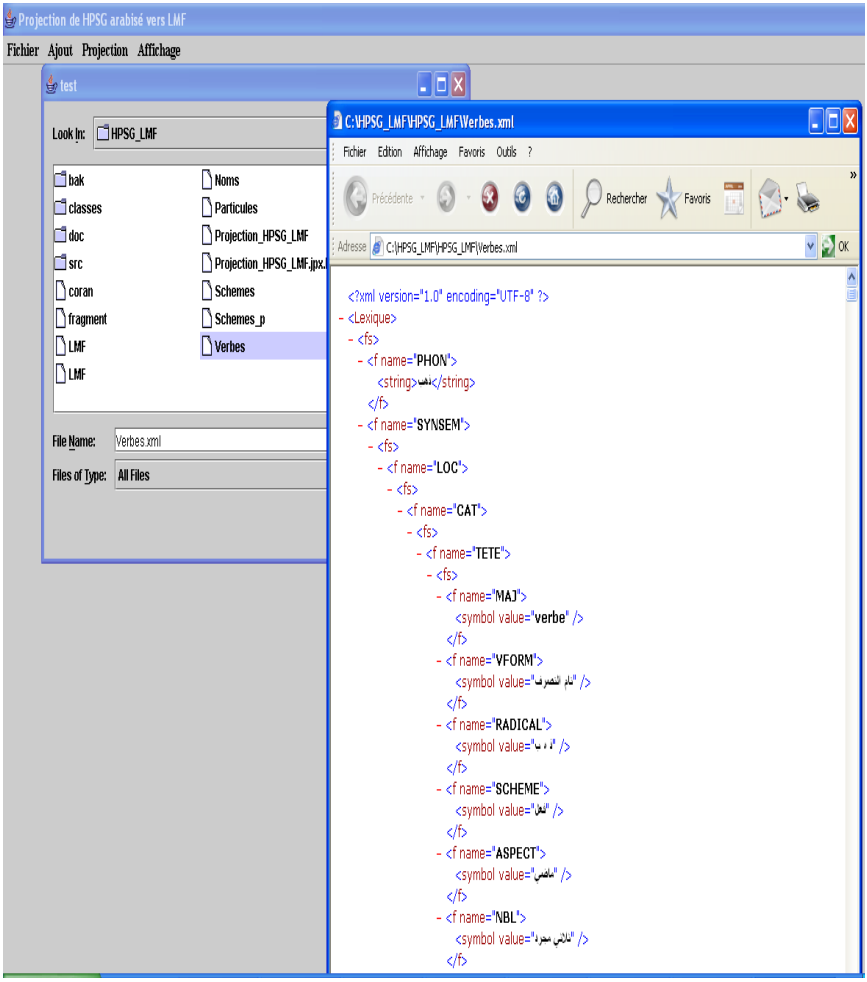


Figure 21: Opening of the file "Verbes.xml"

The displayed file in Figure 21 corresponds to the projection result file. It reflects the application of the rules relative to a verb that we have detailed previously. This file takes into account the DTD that is represented in ISO/TC37/SC4 N130 rev.9 2006.

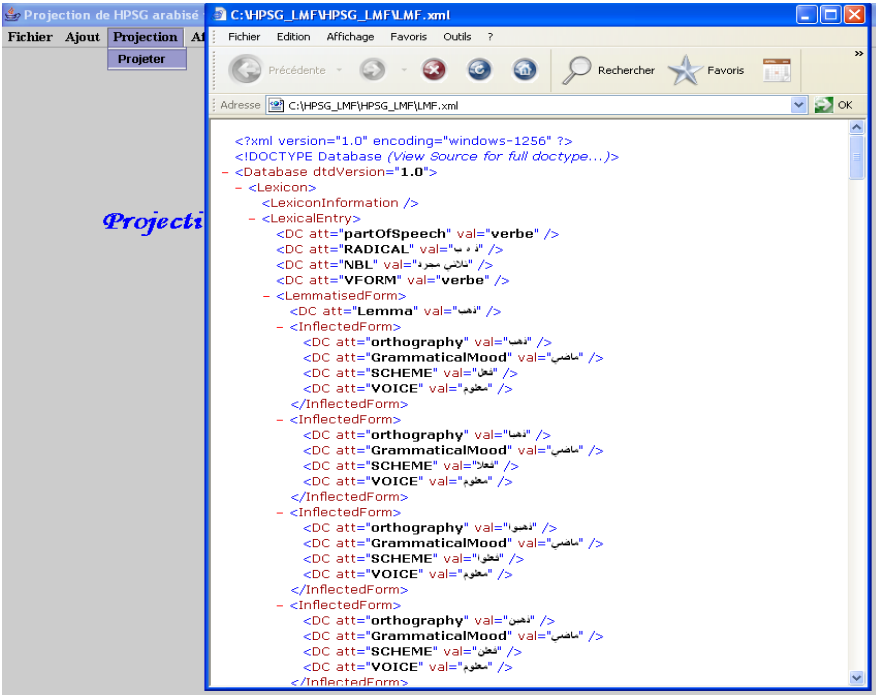


Figure 22: Projection of the file "Verbes.xml"

The menu in Figure 22 displays the lexical entries of the constructed files. Figure 23 is an example of the verbs that are found in the file "Verbes.xml".

Figure 23: Display of the entries contained in the file "Verbes.xml"

The graphical interface of Figure 23 shows the feature values of a verb from the lexicon in question. This interface also provides navigation facilities in the lexicon by showing the characteristics of the verbs that precede and follow the verb shown.

6.3 Evaluation

In order to evaluate the constructed system we projected 10 Arabic HPSG lexicons into LMF language. Projected lexicons have varied structures and contents and allow the obtaining of a normalised lexicon in conformity with LMF and without any loss of information. These lexicons contain different features that we have added to Arabic adapted HPSG in order to bind every canonical or derived form to its inflected forms. These lexicons can also contain canonical and/or derived forms without inflected forms. The obtained LMF lexicon contains 3,000 verbs, 450 nouns and 50 particles.

A Lexicon projection result in conformity with LMF can result in loss of information because projected lexicons possess features that do not exist in the base of chosen features. Data categories in HPSG are not standardised and every user can define his proper data in order to achieve his goal. Therefore, the same feature can exist in several HPSG lexicons under different writing formats. HPSG lexicons that generate some lexicons that do not conform to LMF and result in loss of information are those that contain inflected forms and do not use the features that we have already added.

We can deduce that to get a lexicon in conformity with LMF without any loss of information necessitates three conditions in the HPSG lexicon source of projection: the first of these is to add the features that bind canonical or derived forms to their inflected forms in the case where the HPSG lexicon to be projected contains inflected forms; the second is to add all HPSG features in the feature basis; the third is to reject all schemes of triliteral verbs in order to avoid conflict between two verbs that may be written in the same way *kharaja* (خَـرَجَ) and *kharija* (خَـرِجَ).

Our system may also be considered to be extensible. We opted for a simple design assuring module autonomy and we have implemented a projection prototype of the HPSG into LMF in an object oriented language encouraging the use of expandable software. We can thus extend our work by the addition of projection rules that allow the use of lexicons belonging to other formalisms. However, prototype portability is not assured as it is designed to manipulate the Arabic language and to use only Windows operating systems that support this language. Our achieved prototype permits not only the recuperation and the fusion of HPSG lexicons without data redundancy but also allows processing of several variations such as the orthographic variation. At the lemma level, for two etymologically bound forms having identical pronunciation and belonging to different inflectional paradigms, our system contains two distinct and separate lexical entries. Furthermore, the prototype allows projection of lexical entries that are categorised as grammatical words. We find in the noun category pronouns (e.g. personal, demonstrative), proper nouns, abstract nouns, etc. in the particle category, we find the significance letters (e.g. conjunctions) and the construction letters.

7 CONCLUSION AND PERSPECTIVES

In this article we have developed a system allowing the projection of an HPSG syntactic lexicon into an LMF compliant lexical model. This system allows us to project lexical

entries of different lexical categories from any HPSG lexicon. This projection will help us to either recover some existing HPSG lexicons, or to merge them and/or to integrate them with other lexicons in order to create richer and larger resources.

HPSG and LMF norm studies carried out so far suggest a method composed of two stages. The proposed method uses a projection rule system able to cover the different features that characterise lexical entries relative to the Arabic language.

The proposed method experimentation is intended to test the feasibility of the achieved system and to discern method limits. Evaluation of the prototype has been based on projection of Arabic HPSG lexicons that are constructed within the framework of several research works. These lexicons have varied structure and content that helped us to identify necessary conditions for the success of projection into LMF.

As for perspectives, we want to define the criteria allowing the formal verification of the projection success. Additionally, we want to try to supply applications conceived in unification grammars from lexical databases in conformity with LMF. This will hopefully encourage the reuse and the enrichment of existing lexicons. Finally, we can exploit projection of the LTAG grammars into HPSG while taking advantage of the phase that allows the conversion of canonical elementary trees into lexical entries that are specified in HPSG. This phase can be considered as an intermediate phase for the passage into LMF: from LTAG into HPSG and from HPSG into LMF.

REFERENCES

Abdelkader, A. (2006). Etude et analyse de la phrase nominale arabe en HPSG. Mémoire de mastère, SINT, FSEGS, Sfax.

Ammar, S., Dichy, J. (1999). Les verbes arabes. Collection Bescherelle, Hatier, France.

Atkins S. *et al.* (2002) From Resources to Applications. Designing The Multilingual ISLE Lexical Entry. In Proceedings of LREC 2002, Las Palmas, Canary Islands, Spain.

Blache, P. (1995). Une introduction à HPSG. 2LC-CNRS – http://www.lpl.univ-aix.fr/~blache/papers/Intro-hpsg.ps.gz

Chabchoub, S. (2005). Etude et représentation des verbes arabes en HPSG en vue de construire un conjugueur automatique. Mémoire de mastère, SINT, FSEGS, Sfax.

Dahdah, A. (1992). Dictionnaire des règles de la langue arabe dans des tableaux. Librairie de Nachirun Lebanon, 5ème édition, Beyrouth.

Eagles (1996). EAGLES Recommendations on subcategorisation. Eagles document EAG-CLWG-SYNLEX.

Elleuch, S. (2004). Analyse syntaxique de la langue arabe sur le formalisme d'unification HPSG. Mémoire de mastère, SINT, FSEGS, Sfax.

Fehri, H. *et al.* (2006). Un système de projection du HPSG arabisé vers la plate-forme LMF. Dans les actes du colloque JETALA, Institut d'Etudes et de Recherches pour l'arabisation (IERA), Rabat du 05-07 juin.

Genelex (1994). Projet Eureka GENELEX, Rapport sur LE MULTILINGUISME. Version 2.0, France.

Hasni, E.*et al.* (2006). Un éditeur et vérificateur des contraintes lexicales pour la langue arabe. Dans les actes du colloque international sur l'informatique et ses applications, IA 2006, Oujda, 31 octobre et 1-2 novembre.

Ide N., Romary L. (2004). A Registry of Standard Data Categories for Linguistic Annotation, In Proceedings of the Fourth Language Resources and Evaluation Conference (LREC). Lisbon – http://hal.inria.fr/inria-00099858

Ide N. , Véronis J., (1995). Encoding Dictionaries, in Ide N. and Véronis J. (Eds.), The Text Encoding Initiative : Background and Context, Kluwer Academic Publishers, Dordrecht, pp 167–179.

ISO 12620:2009 Terminology and other language and content resources -- Specification of data categories and management of a Data Category Registry for language resources.

ISO 24610-1:2006 Language resource management -- Feature structures -- Part 1: Feature structure representation.

ISO 24613:2008 Language resource management - Lexical markup framework (LMF).

Kasper, R. *et al.* (1995). Compilation of HPSG to TAG. In proceeding of the 33rd ACL, pp 92–99.

Khemakem, A. (2006). ArabicLDB : une base lexicale normalisée pour la langue arabe. Mémoire de mastère, SINT, FSEGS, Sfax.

Lemnitzer, L., Romary, L., Witt, A, "Representing human and machine dictionaries in Markup languages", Dictionaries. An International Encyclopedia of Lexicography. Supplementary volume: Recent developments with special focus on computational lexicography Mouton de Gruyter (Ed.) (to appear) – http://hal.inria.fr/inria-00441215

Levine R. D. , Meurers D. (2006). "Head-Driven Phrase Structure Grammar: Linguistic Approach, Formal Foundations, and Computational Realization". Keith Brown (Ed.): Encyclopedia of Language and Linguistics, Second Edition. Oxford: Elsevier, pp 690-704.

Loukil, N. (2006). Une proposition de représentation normalisée des lexiques des grammaires d'unification. In Actes de TALN, Presse Universitaire de Louvain, 10 – 13 Avril 2006, Belgique.

Monachini M., Calzolari N. (1999). Standardization in the Lexicon. In: H. van Halteren (Ed.): Syntactic Wordclass Tagging. Kluwer, Dordrecht, pp 149–173.

Nguyen T. M. H. *et al.* (2006). A lexicon for Vietnamese language processing, Language Resources and Evaluation 40, 3-4 2006, pp 291-309 – http://hal.inria.fr/inria-00201451

Pollard, C., Sag., I.A. (1994). Head-Driven Phrase Structure Grammar. Publié par la presse dans l'université de Chicago, Edition Golgoldmittu, Chicago, LSLI.

Ringersma, J., & Kemps-Snijders, M. (2007). Creating multimedia dictionaries of endangered languages using LEXUS. In H. van Hamme, & R. van Son (Eds.), Proceedings of Interspeech 2007 (pp. 65-68). Baixas, France: ISCA-Int.Speech Communication Assoc.

Romary, L., Ide, N. (2004). International Standard for a Linguistic Annotation Framework, Natural Language Engineering 10, 3-4, pp 211-225 – http://hal.inria.fr/hal-00164624

Romary, L. *et al.* (2004). Standards going concrete: from LMF to Morphalou. In Coling 2004, August 29, Geneva, Switzerland, pp 22-28 – http://hal.inria.fr/inria-00121489

Romary, L. (2001). An abstract model for the representation of multilingual terminological data: TMF - Terminological Markup Framework. In Proc. Tama 2001, Antwerpen, Belgium – http://hal.inria.fr/inria-00100405

Yoshinaga, N., Miyao, Y. (2002). Grammar conversion from LTAG to HPSG. In Proc.of the sixth ESSLLI Student Session, University of Tokyo, pp 309-324.

Martin Riedl, Chris Biemann

Text Segmentation with Topic Models

This article presents a general method to use information retrieved from the Latent Dirichlet Allocation (LDA) topic model for Text Segmentation: Using topic assignments instead of words in two well-known Text Segmentation algorithms, namely TextTiling and C99, leads to significant improvements. Further, we introduce our own algorithm called TopicTiling, which is a simplified version of TextTiling (Hearst, 1997). In our study, we evaluate and optimize parameters of LDA and TopicTiling. A further contribution to improve the segmentation accuracy is obtained through stabilizing topic assignments by using information from all LDA inference iterations. Finally, we show that TopicTiling outperforms previous Text Segmentation algorithms on two widely used datasets, while being computationally less expensive than other algorithms.

1 Introduction

Text Segmentation (TS) is concerned with "automatically break[ing] down documents into smaller semantically coherent chunks" (Jurafsky and Martin, 2009). We assume that semantically coherent chunks are also similar in a topical sense. Thus, we view a document as a sequence of topics. This semantic information can be modeled using Topic Models (TMs). TS is realized by an algorithm that identifies topical changes in the sequence of topics.

TS is an important task, needed in Natural Language Processing (NLP) tasks, e.g. information retrieval and text summarization. In information retrieval tasks, TS can be used to extract segments of the document that are topically interesting. In text summarization, segmentation results are important to ensure that the summarization covers all themes a document contains. Another application could be a writing aid to assist authors with possible positions for subsections.

In this article, we use the Latent Dirichlet Allocation (LDA) topic model (Blei et al., 2003). We show that topic IDs, assigned to each word in the last iteration of the Bayesian inference method of LDA, can be used to improve TS significantly in comparison to methods using word-based features. This is demonstrated on three algorithms: TextTiling (Hearst, 1997), C99 (Choi, 2000) and a newly introduced algorithm called TopicTiling. TopicTiling resembles TextTiling, but is conceptually simpler since it does not have to account for the sparsity of word-based features.

In a sweep over parameters of LDA and TopicTiling, we find that using topic IDs of a single last inference iteration leads to enormous instabilities with respect to TS error rates. These instabilities can be alleviated by two modifications: (i) repeating the

inference iterations several times and selecting the most frequently assigned topic ID for each word across several inference runs, (ii) storing the topic IDs assigned to each word for each iteration during the Bayesian inference and selecting most frequently assigned topic ID (the mode) per word. Both modifications lead to similar stabilization, however (ii) needs less computational resources. Furthermore, we can also show that the standard parameters recommended by Griffiths and Steyvers (2004) do not always lead to optimal results.

Using what we have learned in these series of experiments, we evaluate the performance of an optimized version of TopicTiling on two datasets: The Choi dataset (Choi, 2000) and a more challenging Wall Street Journal (WSJ) corpus provided by Galley et al. (2003). Not only does TopicTiling deliver state-of-the-art segmentation results, it also performs the segmentation in linear time, as opposed to most other recent TS algorithms.

The paper is organized as follows: The next section gives an overview of TS algorithms. Then we introduce the method of replacing words by topic IDs, lay out three algorithms using these topic IDs in detail, and show improvements for the topic-based variants. Section 5 evaluates parameters of LDA in combination with parameters of our TopicTiling algorithm. In Section 6, we apply the method to various datasets and end with a conclusion and a discussion.

2 Related Work

Topic segmentation can be divided into two sub-fields: (i) linear topic segmentation and (ii) hierarchical topic segmentation. Whereas linear topic segmentation deals with the sequential analysis of topical changes, hierarchical segmentation concerns with finding more fine grained subtopic structures in texts.

One of the first unsupervised linear topic segmentation algorithms was introduced by Hearst (1997): TextTiling segments texts in linear time by calculating the similarity between two blocks of words based on the cosine similarity. The calculation is accomplished by two vectors containing the number of occurring terms of each block. LcSeg, a TextTiling-based algorithm, was published by Galley et al. (2003). In comparison to TextTiling, it uses *tf-idf* term weights, which improves TS results. Choi (2000) introduced an algorithm called C99, that uses a matrix-based ranking and a clustering approach in order to relate the most similar textual units. Similar to the previous introduced algorithms, C99 uses words. Utiyama and Isahara (2001) introduced one of the first probabilistic approaches using Dynamic Programming (DP) called U00. DP is a paradigm that can be used to efficiently find paths of minimum cost in a graph. Text Segmentation algorithms using DP, represent each possible segment (e.g. every sentence boundary) as an edge. Providing a cost function that penalizes common vocabulary across segment boundaries, DP can be applied to find the segments with minimal cost.

Related to our work are a modified C99 algorithm, introduced by Choi et al. (2001) that uses the term-representation matrix in latent space of LSA in combination with a term frequency matrix to calculate the similarity between sentences and two DP

approaches described in Misra et al. (2009) and Sun et al. (2008): here, topic modeling is used to alleviate the sparsity of word vectors. The algorithm of Sun et al. (2008) follows the approach described in Fragkou et al. (2004), but uses a combination of topic distributions and term frequencies. A Fisher kernel is used to measure the similarity between two blocks, where each block is represented as a sequence of sentences. The kernel uses a measure that indicates how much topics two blocks share, combined with the term frequency, which is weighted by a factor that indicates how likely the terms belong to the same topic. They use entire documents as blocks and generate the topic model using the test data. This method is evaluated using an artificially garbled Chinese corpus. In a similar fashion, Misra et al. (2009) extended the DP algorithm U00 from Utiyama and Isahara (2001) using topic models. Instead of using the probability of word co-occurrences, they use the probability of co-occurring topics. Segments with many different topics have a low topic-probability, which is used as a cost function in their DP approach. (Misra et al., 2009) trained the topic model on a collection of the Reuters corpus and a subset of the Choi dataset, and tested on the remaining Choi dataset. The topics for this test set have to be generated for each possible segment using Bayesian inference methods, resulting in high computational cost. In contrast to these previous DP approaches, we present a computationally more efficient solution. Another approach would be to use an extended topic model that also considers segments within documents, as proposed by Du et al. (2010). A further approach for text segmentation is the usage of Hidden Markov Model (HMM), first introduced by Mulbregt et al. (1998). Blei and Moreno (2001) introduced an Aspect Hidden Markov Model (AHMM) which combines an aspect model (Hofmann, 1999) with a HMM. The limiting factor of both approaches is that a segment is assumed to have only one topic. This problem has been solved by Gruber et al. (2007) who extends LDA to consider the word and topic ordering using a Markov Chain. In contrast to LDA, not a word is assigned to a topic, but a sentence, so the segmentation can be performed sentence-wise.

In early TS evaluations, Hearst (1994) measured the fitting of the estimated segments using precision and recall. But these measures are considered inappropriate for the task, since the distance of a falsely estimated boundary to the correct one is not considered at all. With P_k (Beeferman et al., 1999), a measure was introduced that regulates this problem. But there are issues concerning the P_k measure, as it uses an unbalanced penalizing between false negatives and false positives. WindowDiff (WD) (Pevzner and Hearst, 2002) solves this problem, but most published algorithms still use the P_k measure. In practice, both measures are highly correlated. While there are newer published metrics (see Georgescul et al. (2006), Lamprier et al. (2007) and Scaiano and Inkpen (2012)), in practice still the two metrics P_k and WD are commonly used.

To handle near misses, P_k uses a sliding window with a length of k tokens, which is moved over the text to calculate the segmentation penalties. This leads to following pairs: $(1, k), (2, k + 1), ..., (n - k, n)$, with n denoting the length of the document. For each pair (i, j) it is checked whether positions i and j belong to the same segment or to different segments. This is done separately for the gold standard boundaries and the

estimated segment boundaries. If the gold standard and the estimated segments do not match, a penalty of 1 is added. Finally, the error rate is computed by normalizing the penalty by the number of pairs $(n - k)$, leading to a value between 0 and 1. A value of 0 denotes a perfect match between the gold standard and the estimated segments. The value of parameter k is assigned to half of the number of tokens in the document divided by the number of segments, given by the gold standard.

According to Pevzner and Hearst (2002), a drawback of the P_k measure is its unawareness of the number of segments between the pair (i, j). WD is an enhancement of P_k: the number of segments between position i and j are counted, where again the distance between the positions is parameterized by k. Then the number of segments is compared between the gold standard and the estimated segments. If the number of segments are not equal, 1 is added to the penalty, which is again normalized by the number of pairs to get an error rate between 0 and 1.

The first hierarchical algorithm was proposed by Yaari (1997), using the cosine similarity and agglomerative clustering approaches. A hierarchical Bayesian algorithm based on LDA is introduced by Eisenstein (2009). In our work, however, we focus on linear topic segmentation.

LDA was introduced by Blei et al. (2003) and is a generative model that discovers topics based on a training corpus. Model training estimates two distributions: A topic-word distribution and a topic-document distribution. As LDA is a generative probabilistic model, the creation process follows a generative story: First, for each document a topic distribution is sampled. Then, for each document, words are randomly chosen, following the previously sampled topic distribution. Using the Gibbs inference method, LDA is used to apply a trained model for unseen documents. Here, words are annotated by topic IDs by assigning the most probable topic ID on the basis of the two distributions. Note that the inference procedure, in particular, marks the difference between LDA and earlier dimensionality reduction techniques such as Latent Semantic Analysis.

3 Text Segmentation Datasets

In this paper we use two datasets: A document collection generated based on the Brown corpus and a more challenging corpus generated using WSJ documents.

3.1 Choi Dataset

The Choi dataset (Choi, 2000) is commonly used in the field of TS (see e.g. Misra et al. (2009); Sun et al. (2008); Galley et al. (2003)). It is an artificially generated corpus generated from the Brown corpus and consists of 700 documents. Each document consists of ten segments. The document generation was performed extracting consecutive snippets of 3-11 sentences from different documents from the Brown corpus. 400 documents consist of segments with a sentence length of 3-11 sentences and there are 100 documents each with sentence counts of 3-5, 6-8 and 9-11.

3.2 Galley Dataset

Galley et al. (2003) present two corpora for written language, each having 500 documents, which are also generated artificially. In comparison to Choi's dataset, the segments in its 'documents' vary from 4 to 22 segments, and are composed by concatenating full source documents. Use of full documents make this corpus a more realistic one in comparison to the one provided by Choi. One dataset is generated based on WSJ documents of the Penn Treebank (PTB) project (Marcus et al., 1994) and the other is based on Topic Detection Track (TDT) documents (Wayne, 1998). As the WSJ dataset seems to be harder (consistently higher error rates across several works), we use this dataset for experimentation.

4 From Words to Topics

4.1 Method to Represent Words with Topic IDs

The method (see also Misra et al., 2009; Sun et al., 2008) for using information gained by topic models is conceptually simple: Instead of using words directly as features to characterize textual units, we use their topic IDs as assigned by Bayesian inference. LDA inference assigns a topic ID to each word in the test document in each inference iteration step, based on a TM trained on a training corpus. The first series of experiments use the topic IDs assigned to each word in the last inference iteration. Figure 1 depicts the general setup.

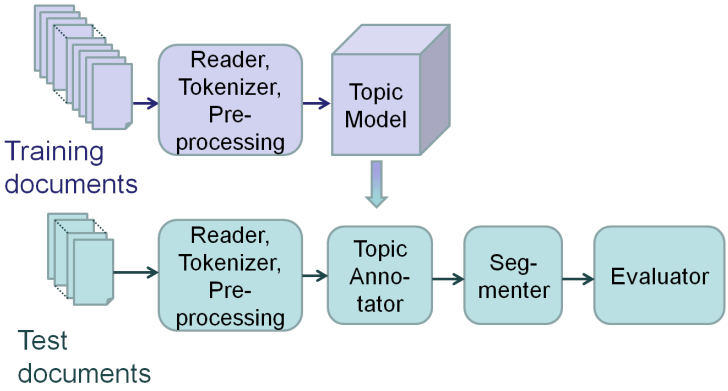


Figure 1: Basic concept of text segmentation using Topic Models

First, preprocessing steps[1] like tokenizing, sentence segmentation, part-of-speech tagging or filtering are applied to the training and test documents.

[1] we use the DKPro framework, `http://code.google.com/p/dkpro-core-asl/`

The training data used to estimate the topic models should ideally be from the same domain as the test documents. Since no information about the test data should inform the training, no test documents should be used for the topic model estimation, even though topic models belong to the unsupervised learning paradigm. The topic model is estimated once in advance and can then be used for inference on the test documents: LDA inference assigns a topic ID to each word in the test document and generates a document topic distribution.

An example of a text annotated with topic IDs, taken from the WSJ test data, is presented in Figure 2. One can clearly see the boundary by looking at the most probable topic IDs. The first text is about a telecommunication company, having mostly topic ID 2 assigned to words. The second segment is about an anti-government rally in South Africa. Most words of this segment are annotated with topic ID 37. The topic IDs are not assigned statically per word, but converge from Gibbs Sampling inference, which iterates over the words and re-samples topic IDs according to the per-document topic distribution and the per-topic word probabilities from the previous inference step. For example, the word *people* (marked bold in Figure 2) is marked with topic 37 since this topic is highly probable in the document. Using this word in a different context would most likely lead to a different topic ID.

Mr:62 .:97 Pohs:2 ,:2 previously:4 executive:2 vice:2 president:2 and:17 chief:2 operating:2 officer:2 ,:72 was:2 named:2 interim:2 president:2 and:73 chief:2 executive:2 officer:2 after:17 David:2 M:27 .:36 Harrold:65 ,:2 a:84 company:2 founder:2 ,:26 resigned:2 from:91 the:34 posts:2 for:62 personal:61 reasons:2 in:84 August:2 .:58 Cellular:70 said:54 Robert:2 J:61 .:42 Lunday:2 Jr:18 .:31 ,:44 its:57 chairman:2 and:73 another:25 founder:2 ,:31 resigned:2 from:91 the:57 company:2 's:24 board:2 to:10 pursue:2 the:10 sale:55 of:67 his:28 telephone:31 company:42 ,:74 Big:10 Sandy:50 Telecommunications:31 Inc:2 .:74
APARTHEID:37 FOES:37 STAGED:41 a:37 massive:37 anti-government:37 rally:37 in:40 South:37 Africa:37 .:19 More:29 than:34 70:45 ,:26 000 **people:37** *filled:17 a:22 soccer:37 stadium:88 on:46 the:34 outskirts:37 of:93 the:24 black:37 township:37 of:45 Soweto:37 and:37 welcomed:11 freed:37 leaders:37 of:98 the:57 outlawed:37 African:37 National:45 Congress:87 .:72 It:79 was:55 considered:37 South:37 Africa:37 's:33 largest:90 opposition:67 rally:37 .:37*

Figure 2: Excerpt from a test document, taken from Galley's WSJ corpus. Each word is followed by a colon and a number, which represents the topic ID.

In the example, all tokens are used for topic model estimation — it is also possible to filter tokens by parts-of-speech or very short sentences for the purpose of model estimation and inference. This is expected to lead to even sparser topic distributions.

Once the topic IDs are assigned, most previous segmentation algorithm can be applied, using the topic ID of each word instead of the word itself.

In this work, we implement topic-based versions of C99 (Choi, 2000), TextTiling (Hearst, 1994) and develop a new TextTiling-based method called TopicTiling. Our aim is to find a simplified algorithm that could solve the segmentation problem using topic IDs.

4.2 Text Segmentation Algorithms using Topic Models

4.2.1 C99 using Topic Models

The topic-based version of the C99 algorithm (Choi, 2000), called C99LDA, divides the input text into minimal units on sentence boundaries. A similarity matrix $S_{m \times m}$ is computed, where m denotes the number of units (sentences). Every element s_{ij} is calculated using the cosine similarity (e.g. Manning and Schütze, 1999) between unit i and j. For these calculations, each unit i is represented as a T-dimensional vector, where T denotes the number of topics selected for the topic model. Each element t_k of this vector contains the number of times topic ID k occurs in unit i. Next, a rank matrix R is computed to improve the contrast of S: Each element r_{ij} contains the number of neighbors of s_{ij} that have lower similarity scores then s_{ij} itself. This step increases the contrast between regions in comparison to matrix S. In a final step, a top-down hierarchical clustering algorithm is performed to split the document into m segments. This algorithm starts with the whole document considered as one segment and splits off segments until the stop criteria are met, e.g. the number of segments or a similarity threshold. At this, the ranking matrix is split at indices i, j that maximize the inside density function D.

$$D = \sum_{k=1}^{m} \frac{\text{sum of ranks within segment } k}{\text{area within segment } k} \tag{1}$$

As a threshold-based criterion, the gradient δD is introduced as $\delta D^{(n)} = D^{(n)} - D^{(n-1)}$. The threshold can then be calculated by $\mu + c \times \sigma$, where mean μ and the standard deviation σ are calculated from the gradients[2].

4.2.2 TextTiling using Topic Models

In TTLDA, the topic-based version of TextTiling (TT) (Hearst, 1994), documents are represented as a sequence of n topic IDs instead of words. TTLDA splits the document into *topic-sequences*, instead of sentences, where each sequence consists of w topic IDs. To calculate the similarity between two topic-sequences, called *sequence-gap*, TTLDA uses k topic-sequences, named *block*, to the left and to the right of the sequence gap. This parameter k defines the so-called *blocksize*. The cosine similarity is applied to compute a similarity score based on the topic frequency vectors of the adjacent blocks at each sequence-gap. A value close to 1 indicates a high similarity among two blocks, a value close to zero denotes a low similarity. Then for each sequence-gap a *depth score* d_i is calculated for describing the sharpness of a gap, given by

$$d_i = 1/2(hl(i) - s_i + hr(i) - s_i).$$

[2] $c = 1.2$ as in Choi (2000).

The function $hl(i)$ returns the highest similarity score on the left side of the sequence-gap index i that does not increase and $hr(i)$ returns the highest score on the right side. Then all local maxima positions are searched based on the depth scores.

In the next step, these obtained maxima scores are sorted. If the number of segments n is given as input parameter, the n highest depth scores are used, otherwise a cut-off function is used that applies a segment only if the depth score is larger than $\mu - \sigma/2$, where mean μ and the standard deviation σ are calculated based on the entirety of depth scores. As TTLDA calculates the depth on every topic-sequence using the highest gap, this could lead to a segmentation in the middle of a sentence. To avoid this, a final step ensures that the segmentation is positioned at the nearest sentence boundary.

4.2.3 TopicTiling

This section introduces our own Text Segmentation algorithm called TopicTiling which is based on TextTiling, but conceptually simpler. TopicTiling assumes a sentence s_i as the smallest basic unit. Between each position p between two adjacent sentences, a *coherence score* c_p is calculated. To calculate the coherence score, we exclusively use the topic IDs assigned to the words by inference: Assuming an LDA model with T topics, each block is represented as a T-dimensional vector. The t-th element of each vector contains the frequency of the topic ID t obtained from the respective block. The coherence score is calculated by cosine similarity for each adjacent "topic vector". Values close to zero indicate marginal relatedness between two adjacent blocks,

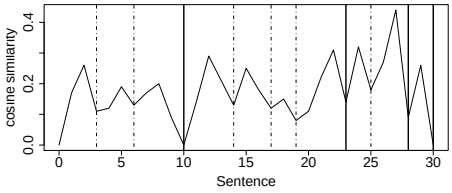


Figure 3: Similarity scores plotted for a document. The vertical lines indicate all possible segment boundaries. The solid lines indicate segments chosen by the threshold criterion, when the number of segments is not given in advance.

whereas values close to one denote a substantial connectivity. Next, the coherence scores are plotted to trace the local minima (see Figure 3). These minima are utilized as possible segmentation boundaries. But rather using the c_p values itself, a *depth score* d_p is calculated for each minimum (cf. TextTiling, Hearst (1997)). In comparison to TopicTiling, TextTiling calculates the depth score for each position and than searches for maxima. The depth score measures the deepness of a minimum by looking at the highest coherence scores on the left and on the right and is calculated using this formula

(cf. depth score formula in the previous section):

$$d_p = 1/2 * (hl(p) - c_p + hr(p) - c_p)$$

The functionality of the function hl (highest peak on the left side) and hr (highest peak on the right side) is illustrated in Figure 4. The function $hl(p)$ iterates to the

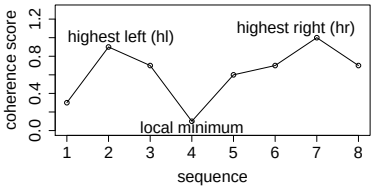


Figure 4: Illustration of the highest left and the highest right peak according to a local minimum.

left as long as the score increases and returns the highest coherence score value. The same is done, iterating in the other direction with the $hr(p)$ function. According to the illustration, $hl(4) = 0.93$, the score value at position 2, and $hr(4) = 0.99$ from the value at position 7.

If the number of segments n is given as input, the n highest depth scores are used as segment boundaries. Otherwise, a threshold is applied (cf. TextTiling). This threshold predicts a segmentation if the depth score is larger than $\mu - \sigma/2$, with μ being the mean and σ being the standard variation calculated on the depth scores.

The algorithm runtime is linear in the number of possible segmentation points, i.e. the number of sentences: for each segmentation point, the two adjacent blocks are sampled separately and combined into the coherence score. This is the main differences to the dynamic programming approaches for TS described in (Utiyama and Isahara, 2001; Misra et al., 2009).

4.3 Experiment: Word-based vs. Topic-based Methods

To show the impact of the topic-based representation introduced in Section 4.1, we show results for TT and C99 using words and topic IDs, and for TopicTiling.

4.3.1 Experimental Setup

As laid out in Section 4.1, an LDA Model is estimated on a training dataset and used for inference on the test set. To ensure that we do not use information from the test set, we perform a 10-fold Cross Validation (CV) for all reported results. To reduce the variance stemming from the random nature of sampling and inference, the results for each fold are calculated 30 times using different LDA models.

While we aim at not using the same *documents* for training and testing by using a folded CV scheme, it is not guaranteed that all testing data is unseen, since the same

source *sentences* can find their way in several artificially crafted *documents*. We could detect that all *sentences* from the training subset also occur in the test subset, but not in the same combinations. This makes the Choi data set artificially easy for supervised approaches. This problem, however, affects all evaluations on this dataset that use any kind of training, be it LDA models in Misra et al. (2009) or *tf-idf* values in Fragkou et al. (2004) and Galley et al. (2003).

The LDA model is trained with $T = 100$ topics, 500 sampling iterations and symmetric hyperparameters as recommended by Griffiths and Steyvers (2004)($\alpha = 50/N$ and $\beta = 0.01$), using the JGibbsLda implementation of Phan and Nguyen (2007). Unseen data is annotated with topic information, using LDA inference, sampling $i = 100$ iterations. Inference is executed sentence-wise, since sentences form the minimal unit of our segmentation algorithms and we cannot use document information in the test setting. The performance of the algorithms is measured using P_k and WindowDiff (WD) metrics, cf. Section 2. The C99 algorithm is initialized with a 11×11 ranking mask, as recommended in Choi (2000). TT is configured according to Choi (2000) with sequence length $w = 20$ and block size $k = 6$.

4.3.2 Results

The experiments are executed in two settings using the C99 and TT implementations[3]: using words (C99, TT) and using topics (C99LDA, TTLDA). TT and C99 use stemmed words and filter out words using a stopword list. C99 additional removes words using predefined regular expressions. In the case of topic-based variants, no stopword filtering or stemming was deemed necessary. Table 1 shows the result of the different algorithms with segments provided and unprovided.

Method	**Segments provided**		**Segments unprovided**	
	P_k	WD	P_k	WD
C99	11.20	12.07	12.73	14.57
C99LDA	4.16	4.89	8.69	10.52
TT	44.48	47.11	49.51	66.16
TTLDA	1.85	2.10	16.41	21.40
TopicTiling	2.65	3.02	4.12	5.75
TopicTiling (filtered)	**1.50**	**1.72**	**3.24**	**4.58**

Table 1: Results by segment length for TT with words and topics (TTLDA), C99 with words and topics (C99LDA) and TopicTiling using all sentences and using only sentences with more than 5 word tokens (filtered).

We note that WD values are always higher than the appropriate P_k values. But we also observe that these measures are highly correlated. First we discuss results for the setting with number of segments provided (see column 2-3 of Table 1). A significant improvement for C99 and TT can be achieved when using topic IDs. In case

[3]We use the implementations by Choi available at `http://code.google.com/p/uima-text-segmenter/`.

of C99LDA, the error rate is at least halved and for TTLDA the error rate is reduced by a factor of 20. The newly introduced algorithm TopicTiling as described above does not improve over TTLDA. Analysis revealed that the Choi corpus includes also captions and other "non-sentences" that are marked as sentences, which causes TopicTiling to introduce false positive segments since the topic vectors are too sparse for these short "non-sentences". We therefore filter out "sentences" with less than 5 words (see bottom line in Table 1). This leads to smaller errors values in comparison to the results achieved with TTLDA. Without the number of segments given in advance (see columns 3-4 in Table 1), we again observe significantly better results, comparing topic-based methods to word-based methods. But the error rates of TTLDA are unexpectedly high. We discovered in data analysis that TTLDA estimates too many segments, as the topic ID distributions between adjacent sentences within a segment are often too diverse, especially in face of random fluctuations from the topic assignments. Estimating the number of segments is better achieved using TopicTiling instead of TTLDA even without any additional sentence filtering. As we aimed to find a simple algorithm that can cope with the topic-based approach, we will use TopicTiling for the next series of experiments.

5 Sweeping the Parameter Space of LDA

Aside from the main parameter, the number of topics or dimensions T, surprisingly little attention has been spent to understand the interactions of hyperparameters, the number of sampling iterations in model estimation and interference, and the stability of topic assignments across runs using different random seeds in the LDA topic model. While progress in the field of topic modeling is mainly made by adjusting prior distributions (e.g. Sato and Nakagawa, 2010; Wallach et al., 2009), or defining more complex mixture models (Heinrich, 2011), it seems unclear whether improvements, reached on intrinsic measures like perplexity or on application-based evaluations, are due to an improved model structure or could originate from sub-optimal parameter settings or due to the randomized nature of the sampling process.

These subsections address these issues by systematically sweeping the parameter space and evaluating LDA parameters with respect to text segmentation results achieved by TopicTiling.

5.1 Experimental Setup

Again, the Choi dataset (see Section 3.1) is used, applying a 10-fold CV as described in Section 4.3.1. To assess the robustness of the TM, we sweep over varying configurations of the LDA model, and plot the results using Box-and-Whiskers plots: the box indicates the quartiles and the whiskers are maximally 1.5 times Interquartile Range (IQR) or equal to the data point that has not a distance larger than 1.5 times IQR. The following parameters are subject to our exploration:

- T: Number of topics used in the LDA model. Common values vary between 50 and 500.
- α : Hyperparameter that regulates the sparseness topic-per-document distribution. Lower values result in documents being represented by fewer topics (Heinrich, 2004). Recommended: $\alpha = 50/T$ (Griffiths and Steyvers, 2004)
- β : Reducing β increases the sparsity of topics, by assigning fewer terms to each topic, which is correlated to how related words need to be, to be assigned to a topic (Heinrich, 2004). Recommended: $\beta = \{0.1, 0.01\}$ (Griffiths and Steyvers, 2004; Misra et al., 2009)
- m Model estimation iterations. Recommended / common settings: $m = 500-5000$ (Griffiths and Steyvers, 2004; Wallach et al., 2009; Phan and Nguyen, 2007)
- i Inference iterations. Recommended / common settings: 100 (Phan and Nguyen, 2007)
- d Mode of topic assignments. At each inference iteration step, a topic ID is assigned to each word within a document (represented as a sentence in our application). With this option ($d = true$), we count these topic assignments for each single word in each iteration. After all i inference iterations, the most frequent topic ID is chosen for each word in a document.
- r Number of inference runs: We repeat the inference r times and assign the most frequently assigned topic per word at the final inference iteration for the segmentation algorithm. High r values might reduce fluctuations due to the randomized process and lead to a more stable word-to-topic assignment.
- w Window: We introduce a so-called *window parameter* that specifies the number of sentences to the left and to the right of position p that define two *blocks*: $s_{p-w}, s_{p-w+1}, \ldots, s_p$ and $s_{p+1}, \ldots, s_{p+w}, s_{p+w+1}$.

All introduced parameters parameterize the TM. Other works stabilize topic assignments by averaging assignments probed from every 50-100th iteration. Examining this effect more closely, we look at the mechanisms of using several inference runs r to find the correct segments and the mode of topic assignments d. Further, we did not find previous work that systematically varies TM parameters in combination with measures other than perplexity.

5.2 Parameter Sweeping Evaluation

5.2.1 Number of Topics T

To provide a first impression of the data, a 10-fold CV is calculated and the segmentation results are visualized in Figure 5. Each box plot is generated from the P_k values of 700

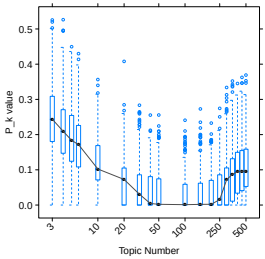


Figure 5: Box plots for different number of topics T. Each box plot is generated from the average P_k value of 700 documents, $\alpha = 50/T$, $\beta = 0.1$, $m = 1000$, $i = 100, r = 1$.

documents. As expected, there is a continuous range of topic numbers, namely between 50 and 150 topics, where we observe the lowest P_k values. Using too many topics leads to overfitting of the data and too few topics result in too general distinctions to grasp text segment information. This general picture is in line with other studies that determine an optimum for T, (cf. Griffiths and Steyvers, 2004), which is specific to the application and the data set.

5.2.2 Estimation and Inference iterations

The next step examines the robustness of the model estimation iterations m needed to achieve stable results. 600 documents are used for training an LDA model and the remaining 100 documents are segmented using this model. This evaluation is performed using 100 topics (as this number leads to stable results according to Figure 5) and performed using 20 and 250 topics. To assess stability across different model estimation runs, we trained 30 LDA models using different random seeds. Each box plot in Figures 6 is generated from 30 mean values, calculated from the P_k values of the 100 documents. The variation indicates the score variance for the 30 different models.

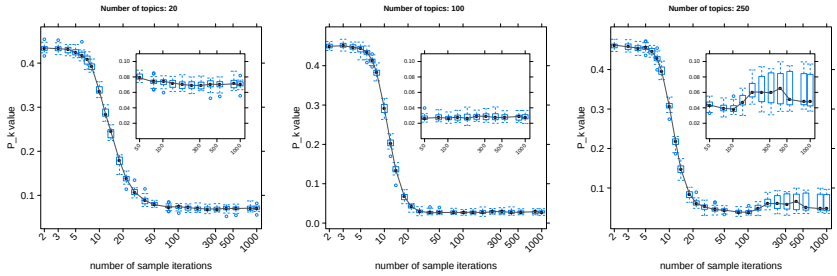
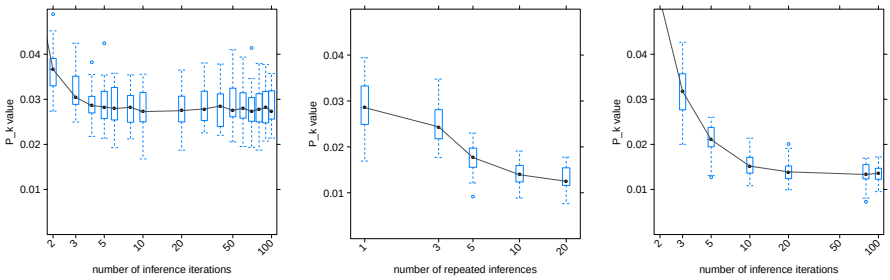


Figure 6: Box plots with different model estimation iterations m, with T=20,100,250 (from left to right), $\alpha = 50/T$, $\beta = 0.1$, $i = 100$, $r = 1$. Each box plot is generated from 30 mean values calculated from 100 documents.

Using 100 topics (see Figure 6), the burn-in phase starts with 8–10 iterations and the mean P_k values stabilize after 40 iterations. But looking at the inset for large m values, significant variations between the different models can be observed: note that the P_k error rates are between 0.021 - 0.037. As expected using 20 and 250 topics leads to worse results as with 100 topics. Looking at the plot with 250 topics, a robust range for the error rates can be found between 20 and 100 sample iterations. With more iterations m, the results get both worse and unstable: as the 'natural' topics of the collection have to be split in too many topics in the model, perplexity reduction that drives the estimation process leads to random fluctuations, which the TopicTiling algorithm is sensitive to. Manual inspection of models for $T = 250$ revealed that in fact many topics do not stay stable across estimation iterations. In the next step we sweep over several inference iterations i using 100 topics. Starting from 5 iterations, error rates do not change much, see Figure 7a. But there is still substantial variance, between about 0.019 - 0.038 for inference on sentence units.



(a) number of inferences i **(b)** repeated inference runs r **(c)** mode method d = true

Figure 7: Figure a) shows the box plots for different inference iterations i, Figure b) shows the box plots for several inference runs r and Figure c) presents the usage of the mode method $d = true$. All remaining parameters are set to the default values.

5.2.3 Repeat the inference r times

To decrease this variance, we assign the topic not only from a singe inference run, but repeat the inference calculations several times, denoted by the parameter r. Then the frequency of assigned topic IDs per token is counted across the r single runs, and we assign the most frequent topic ID (frequency ties are broken randomly). The box plot for several evaluated values of r is shown in Figure 7b. This log-scaled plot shows that both variance and P_k error rate can be substantially decreased. Already for $r = 3$, we observe a significant improvement in comparison to the default setting of $r = 1$ and with increasing r values, the error rates are reduced even more: for $r = 20$, variance and error rates are cut in less than half of their original values using this simple operation.

5.2.4 Mode of topic assignment d

In the previous experiment, we use the topic IDs that have been assigned most frequently at the last inference iteration step. Now, we examine something similar, but for all i inference steps of a single inference run: we select the mode of topic ID assignments for each word across all inference steps. The impact of this method on error and variance is illustrated in Figure 7c. Using a single inference iteration, the topic IDs are almost assigned randomly. After 20 inference iterations P_k values below 0.02 are achieved. Using further iterations, the decrease of the error rate is only marginal. In comparison to the repeated inference method, the additional computational costs of this method are much lower as the inference iterations have to be carried out anyway in the default application setting. Note that this is different from using the overall topic distribution as determined by the inference step, since this winner-takes-it-all approach reduces noise from random fluctuations. As this parameter stabilizes the topic IDs at low computational costs, we recommend using this option in all setups where subsequent steps rely on single topic assignments.

5.2.5 Hyperparameters α and β

In many previous works, hyperparameter settings $\alpha = 50/T$ and $\beta = \{0.1, 0.01\}$ are commonly used. In the next series of experiments we investigate how different parameters of these both parameters can change the TS task. Analyzing the α values, shown in Figure 8, we can see that the recommended values for $T = 100$, $\alpha = 0.5$ lead to sub-optimal settings, and an error rate reduction of about 40% can be achieved by setting $\alpha = 0.1$. Regarding values of β, we find that P_k rates and their variance are

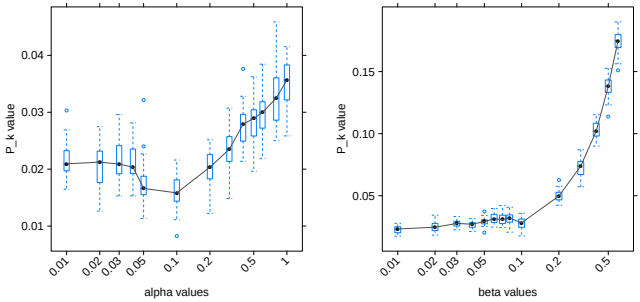
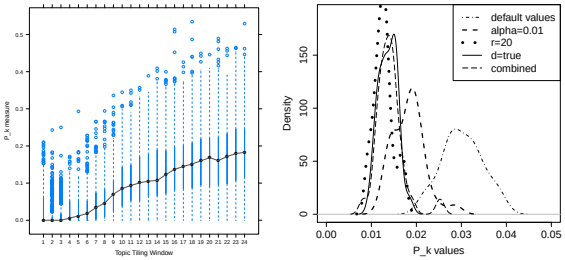


Figure 8: Box plot for several alpha (left) and beta (right) values with $m = 500$, $i = 100$, $T = 100$, $r = 1$ and $\beta = 0.1$ (left image) and $\alpha = 0.5$ (right image).

relatively stable between the recommended settings of 0.1 and 0.01. Values larger than 0.1 lead to much worse performance. Regarding variance, no patterns within the stable range emerge, see Figure 8.

5.2.6 Window Parameter w

The optimal window parameter has to be specified according to the documents that are segmented.



(a) window parameter **(b)** Error distributions

Figure 9: Figure a) represents the box plots for varying window parameter w with $m = 500$, $i = 100$, $T = 100$, $\alpha = 50/T$, $\beta = 0.1$, $r = 1$. The Density of the error distribution for the system according to Table 2 is shown in Figure b).

Using the Choi corpus we observe that the window parameter could be increased to a size of 3 before the error rate increases. Since the segment sizes vary from 3-11 sentences we expect a decline for $w > 3$, which is confirmed by the results shown in Figure 9a.

5.3 Putting it all together

Until this point, we have examined different parameters with respect to stability and error rates one at the time. Now, we combine what we have learned from this and strive at optimal system performance. Table 2 shows P_k error rates for the different systems. At this, we fixed the following parameters: $T = 100$, $m = 500$, $i = 100$, $\beta = 0.1$. For the computations we use 600 documents for the LDA model estimation, apply TopicTiling to the 100 remaining documents and repeat this 30 times with different random seeds.

System	P_k	error red.	σ^2	var. red.
default	0.0302	0.00%	2.02e-5	0.00%
$\alpha = 0.1$	0.0183	39.53%	1.22e-5	39.77%
$r = 20$	0.0127	57.86%	4.65e-6	76.97%
$d = true$	0.0137	54.62%	3.99e-6	80.21%
combined	0.0141	53.45%	9.17e-6	54.55%

Table 2: Comparison of single parameter optimizations, and combined system. P_k averages and variance are computed over 30 runs, together with reductions relative to the default setting. Default: $\alpha = 0.5$, $r = 1$, $d = false$. combined: $\alpha = 0.1$, $r = 20$, $d = true$

We observe massive improvements for optimized single parameters. The α-tuning results in an error rate reduction of 39.77% in comparison to the default configurations. Using $r = 20$, the error rate is cut in less than half its original value. Also for the mode mechanism ($d = true$) the error rate is halved but slightly worse than when using the repeated inference. Regarding the practice to assign the most frequent topic ID selected from every 50-100th iteration, we conclude that – at least in our application – a much smaller number of iterations suffices when taking assignments from all iterations. Here, allowing long inference periods to account for possible topic drifts seems not required. Using combined optimized parameters does not result to additional error decreases. We attribute the slight decline of the combined method in both the error rate P_k and the variance to complex parameter interactions that shall be examined in further work. In Figure 9b, we visualize these results in a density plot. It becomes clear that repeated inference leads to slightly better and more robust performance (higher peak) than the mode method. We attribute the difference to situations, where there are several highly probable topics in our sampling units, and by chance the same one is picked for adjacent sentences that belong to different segments, resulting in failure to recognize the segmentation point. However, since the differences are miniscule, only using the mode method might be more suitable for practical purposes since its computational cost is lower.

6 Comparison to other Algorithms

In a last series of experiments, we compare the performance of TopicTiling to other TS algorithms on several datasets. All LDA models for these series were created using $T = 100$, $\alpha = 50/T$, $\beta = 0.01$, $m = 500$, $i = 100$.

6.1 Evaluation on the Choi Dataset

The evaluation uses the 10-fold CV setting as described in Section 4.3.1. For this dataset, no word filtering based on parts of speech was deemed necessary. The results for different parameter settings are listed in Table 3. Using only the window parameter

seg. size	**3-5**		**6-8**		**9-11**		**3-11**	
	P_k	WD	P_k	WD	P_k	WD	P_k	WD
d=false,w=1	2.71	3.00	3.64	4.14	5.90	7.05	3.81	4.32
d=true,w=1	3.71	4.16	1.97	2.23	2.42	2.92	2.00	2.30
d=false,w=2	1.46	1.51	1.05	1.20	1.13	1.31	1.00	1.15
d=true,w=2	**1.24**	**1.27**	**0.76**	**0.85**	**0.56**	**0.71**	**0.95**	**1.08**
d=false,w=5	2.78	3.04	1.71	2.11	4.47	4.76	3.80	4.46
d=true,w=5	2.34	2.65	1.17	1.35	4.39	4.56	3.20	3.54

Table 3: Results based on the Choi dataset with varying parameters.

without the mode ($d = false$), the results demonstrate a significant error reduction with a window of 2 sentences. An impairment is observed when using a too large

window (w=5) (cmp. Section 5.2.6). We can also see that the mode method improves the results when using a window of 1, except for the documents having small segments ranging from 3-5 sentences. The lowest error rates are obtained with the mode method and a window size of 2. As described in Section 4.2.3, the algorithm is also able to automatically estimate the number of segments using a threshold value (see Table 4).

	3-5		**6-8**		**9-11**		**3-11**	
	P_k	WD	P_k	WD	P_k	WD	P_k	WD
d=false,w=1	**2.39**	**2.45**	4.09	5.85	9.20	15.44	4.87	6.74
d=true,w=1	3.54	3.59	1.98	2.57	3.01	5.15	2.04	2.62
d=false,w=2	15.53	15.55	0.79	0.88	1.98	3.23	1.03	1.36
d=true,w=2	14.65	14.69	**0.62**	**0.62**	**0.67**	**0.88**	**0.66**	**0.78**
d=false,w=5	21.47	21.62	16.30	16.30	6.01	6.14	14.31	14.65
d=true,w=5	21.57	21.67	17.24	17.24	6.44	6.44	15.51	15.74

Table 4: Results on the Choi dataset without providing the number of segments

As can be seen the optimized parameters leads to worse results for segments of length 3-5. This is caused by the smoothing effect of the window parameter which leads to less detected boundaries. But the results of the other documents are comparable to the ones shown in Table 3. Some results (see segment length 6-8 and 3-11 with parameter d=true and w=2) are even better than the results with segments provided which is attributed to the remaining variance in the probabilistic inference computations. The threshold method can outperform the setup with a given number of segments, since not recognizing a segment produces less error in the measures than predicting a wrong segment. Table 5 presents a comparison of the performance of TopicTiling compared to different algorithms in the literature.

Method	**3-5**	**6-8**	**9-11**	**3-11**
TT (Choi, 2000)	44	43	48	46
C99 (Choi, 2000)	12	9	9	12
U00 (Utiyama and Isahara, 2001)	9	7	5	10
LCseg (Galley et al., 2003)	8.69			
F04 (Fragkou et al., 2004)	5.5	3.0	1.3	7.0
M09 (Misra et al., 2009)	2.2	2.3	4.1	2.3
TopicTiling (d=true, w=2)	**1.24**	**0.76**	**0.56**	**0.95**

Table 5: Lowest P_k values for the Choi data set for various algorithms in the literature with provided segment number.

It is obvious that the results are far better than current state-of-the-art results. Using a one-sample t-test with $\alpha = 0.05$ we can state significant improvements in comparison to all other algorithms. With error rates below the 1% range, TS on the Choi dataset can be considered as solved. However, since the dataset is comparatively easy, and test data has probably been seen during model training (cf. Section 4.3), we assess the performance of our algorithm on a second dataset.

6.2 Evaluation on Galley's WSJ Dataset

The evaluation on Galley's WSJ dataset is performed, using a topic model created from the WSJ collection of the PTB. The dataset for model estimation consists of 2499 WSJ articles, and is the same dataset Galley used as a source corpus. The evaluation generally leads to higher error rates than in the evaluation for the Choi dataset, as shown in Table 6.

Parameters	**All words**		**Filtered**	
	P_k	WD	P_k	WD
d=false,w=1	37.31	43.20	37.01	43.26
d=true,w=1	35.31	41.27	33.52	39.86
d=false,w=2	22.76	28.69	21.35	27.28
d=true,w=2	21.79	27.35	19.75	25.42
d=false,w=5	14.29	19.89	12.90	18.87
d=true,w=5	**13.59**	**19.61**	**11.89**	**17.41**
d=false,w=10	14.08	22.60	14.09	22.22
d=true,w=10	13.61	21.00	13.48	20.59

Table 6: Results for Galley's WSJ dataset using different parameters with using unfiltered documents (column 2-3) and with filtered documents using only verbs, nouns (proper and common) and adjectives (column 3-4).

This table shows results of the WSJ data when using all words of the documents for training a topic model and assigning topic IDs to new documents. It also shows results using only nouns (proper and common), verbs and adjectives[4]. Considering the unfiltered results, we observe that performance benefits from using the mode assigned topic ID and a window larger than one. In case of the WSJ dataset, we find the optimal setting for the window parameter to be 5. As the test documents contain whole articles, which consist of at least 4 sentences, a larger window is advantageous here, yet a value of 10 is too large. Filtering the documents for parts of speech leads to $\sim 1\%$ absolute error rate reduction, as can be seen in the last two columns of Table 6. Again, we observe that the mode assignment always leads to better results, gaining at least 0.6%. Especially the window size of 5 helps TopicTiling to decrease the error rate to a third of the value observed with d=false and w=1. Table 7 shows the results we achieve with the threshold-based estimation of segment boundaries for the unfiltered and filtered data.

In contrast to the results obtained with the Choi dataset (see Table 4) no decline occurs, when using the threshold approach in combination with the window method. We attribute this due to the small segments and documents in the Choi dataset. Part-of-speech-based filtering is always advantageous over using all words here. Also a decrease of both error rates, P_k and WD, is detected when using the mode and using a larger window size. An improvement is even gained for a window of size 10. This can be attributed to the fact that using small window sizes, too many boundaries are detected.

[4] as identified by the Treetagger `http://code.google.com/p/tt4j/`

Parameters	All words		Filtered	
	P_k	WD	P_k	WD
d=false,w=1	53.07	72.78	52.63	72.66
d=true,w=1	53.42	74.12	51.84	72.57
d=false,w=2	46.68	65.01	44.81	63.09
d=true,w=2	46.08	64.41	43.54	61.18
d=false,w=5	30.68	43.73	28.31	40.36
d=true,w=5	28.29	38.90	26.96	36.98
d=false,w=10	19.93	32.98	18.29	29.29
d=true,w=10	**17.50**	**26.36**	**16.32**	**24.75**

Table 7: Table with results the WSJ dataset without providing the number of segments. Columns 2 and 3 show the results when using all words of the documents. Columns 4 and 5 show the results with part-of-speech-based filtering.

As the window approach smooths the similarity scores, this leads to less segmentation boundaries and improved results.

Table 8 presents the results of other algorithms, as published in Galley et al. (2003), in comparison to TopicTiling. Again, TopicTiling improves over the state of the art.

Method	P_k	WD
C99 Choi (2000)	19.61	26.42
U00 Utiyama and Isahara (2001)	15.18	21.54
LCseg Galley et al. (2003)	12.21	18.25
TopicTiling (d=true,w=5)	**11.89**	**17.41**

Table 8: List of results based on the WSJ dataset. Values for C99, U00 and LCseg as stated in Galley et al. (2003).

The improvements with respect to LCseg are significant using a one-sample t-test with $\alpha = 0.05$.

7 Conclusion

In this article we showed that replacing words in documents by topic IDs, as assigned by the Bayesian inference method of LDA, leads to better results in the Text Segmentation task. This technique is applied in the TT and C99 algorithms. Additionally, we introduced a simplified algorithm based on TT called TopicTiling that outperforms the topic-based versions of TT and C99. In contrast to other TS algorithms using topic models (Misra et al. (2009); Sun et al. (2008)), the runtime of TopicTiling is linear in the number of sentences. This makes TopicTiling a fast algorithm with complexity of $O(n)$ (n denoting the number of sentences) as opposed to $O(n^2)$ of the dynamic programming approach as discussed in Fragkou et al. (2004).

During sweeping the parameter space of LDA and TopicTiling (see Section 5) we show that repeating the Bayesian inference several times and using the most frequently assigned topic IDs in the last iteration not only reduces the variance, but also improves

overall results. We obtain almost equal performance, when selecting the most frequent topic ID (mode) assigned per word across each inference step. Although the error rates are slightly higher in our experiments, this method is preferred, as the computational cost is much lower than repeating the inference step several times. This method is not only applicable to Text Segmentation, but in all applications where performance crucially depends on stable topic ID assignments per token. Using the Choi dataset and the Galleys WSJ dataset we can show significant improved results in comparison to actual state-of-the-art algorithms.

For further work, we would like to devise a method to detect the optimal setting for the window parameter w automatically, especially in a setting where the number of target segments is not known in advance. This is an issue that is shared with the original TextTiling algorithm. Moreover, we will extend the usage of our algorithm to more realistic corpora.

More interesting is the perspective on possible applications. Equipped with a highly reliable segmentation mechanism, we would like to apply text segmentation as a writing aid to assist authors with feasible segmentation boundaries. This could be applied in an interactive manner by giving feedback about the coherence during the writing process. As the author is responsible for accepting such segmentation, the need for automatically determining the number of segments would be dispensable, and subject to tuning to the author's preferences.

Another direction of research that is more generic for approaches based on topic models is the question of how to automatically select appropriate data for topic model estimation, given only a small target collection. Since topic model estimation is computationally expensive, and topic models for generic collections (think Wikipedia) might not suit the needs of a specialized domain (such as with the WSJ data), it is a promising direction to look at target-domain-driven automatic corpus synthesis.

8 Acknowledgments

This work has been supported by the Hessian research excellence program "Landes-Offensive zur Entwicklung Wissenschaftlich-ökonomischer Exzellenz" (LOEWE) as part of the research center "Digital Humanities".

References

Beeferman, D., Berger, A., and Lafferty, J. (1999). Statistical models for text segmentation. *Machine learning*, 34(1):177–210.

Blei, D. M. and Moreno, P. J. (2001). Topic segmentation with an aspect hidden markov model. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 343–348, New Orleans, Louisiana, USA.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

Choi, F. Y. Y. (2000). Advances in domain independent linear text segmentation. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 26–33, Seattle, WA, USA.

Choi, F. Y. Y., Wiemer-Hastings, P., and Moore, J. (2001). Latent semantic analysis for text segmentation. In *Proceedings of EMNLP*, pages 109–117, Pittsburgh, PA, USA.

Du, L., Buntine, W., and Jin, H. (2010). A segmented topic model based on the two-parameter poisson-dirichlet process. *Machine Learning*, 81(1):5–19.

Eisenstein, J. (2009). Hierarchical text segmentation from multi-scale lexical cohesion. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 353–361, Boulder, CO, USA.

Fragkou, P., Petridis, V., and Kehagias, A. (2004). A Dynamic Programming Algorithm for Linear Text Segmentation. *Journal of Intelligent Information Systems*, 23(2):179–197.

Galley, M., McKeown, K., Fosler-Lussier, E., and Jing, H. (2003). Discourse segmentation of multi-party conversation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, volume 1, pages 562–569, Sapporo, Japan.

Georgescul, M., Clark, A., and Armstrong, S. (2006). An analysis of quantitative aspects in the evaluation of thematic segmentation algorithms. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, pages 144–151, Sydney, Australia.

Griffiths, T. L. and Steyvers, M. (2004). Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101:5228–5235.

Gruber, A., Rosen-Zvi, M., and Weiss, Y. (2007). Hidden topic markov models. In *In Proceedings of Artificial Intelligence and Statistics*, San Juan, Puerto Rico.

Hearst, M. A. (1994). Multi-paragraph segmentation of expository text. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 9–16, Las Cruces, NM, USA.

Hearst, M. A. (1997). TextTiling : Segmenting Text into Multi-paragraph Subtopic Passages. *Computational Linguistics*, 23(1):33–64.

Heinrich, G. (2004). Parameter estimation for text analysis. Technical report, University of Leipzig, http://www.arbylon.net/publications/text-est.pdf.

Heinrich, G. (2011). Typology of mixed-membership models: Towards a design method. In *Machine Learning and Knowledge Discovery in Databases*, volume 6912 of *Lecture Notes in Computer Science*, pages 32–47. Springer Berlin / Heidelberg. 10.1007/978-3-642-23783-6 3.

Hofmann, T. (1999). Probabilistic Latent Semantic Analysis. In *Proceedings of Uncertainty in Artificial Intelligence, UAI'99*, pages 289–296, Stockholm, Sweden.

Jurafsky, D. and Martin, J. H. (2009). *Speech and Language Processing: An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics*. Pearson International Edition.

Lamprier, S., Amghar, T., Levrat, B., and Saubion, F. (2007). ClassStruggle. In *Proceedings of the 2007 ACM symposium on Applied computing - SAC '07*, page 600, New York, New York, USA. ACM Press.

Manning, C. and Schütze, H. (1999). *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA.

Marcus, M., Kim, G., Marcinkiewicz, M. A., Macintyre, R., Bies, A., Ferguson, M., Katz, K., and Schasberger, B. (1994). The Penn Treebank: Annotating predicate argument structure. In *Proceedings of the workshop on Human Language Technology*, pages 114–119, Plainsboro, NJ, USA.

Misra, H., Yvon, F., Jose, J. M., and Cappe, O. (2009). Text Segmentation via Topic Modeling: An Analytical Study. In *Proceeding of the 18th ACM Conference on Information and Knowledge Management*, pages 1553–1556, Hong Kong.

Mulbregt, P. v., Carp, I., Gillick, L., Lowe, S., and Yamron, J. (1998). Text segmentation and topic tracking on broadcast news via a hidden markov model approach. In *Proceedings of 5th International Conference on Spoken Language Processing*, Sydney, Australia.

Pevzner, L. and Hearst, M. A. (2002). A Critique and Improvement of an Evaluation Metric for Text Segmentation. *Computational Linguistics*, 28(1):19–36.

Phan, X.-H. and Nguyen, C.-T. (2007). GibbsLDA++: A C/C++ implementation of latent Dirichlet allocation (LDA). http://jgibblda.sourceforge.net/.

Sato, I. and Nakagawa, H. (2010). Topic Models with Power-Law Using Pitman-Yor Process Categories and Subject Descriptors. *Science And Technology*, (1):673–681.

Scaiano, M. and Inkpen, D. (2012). Getting more from segmentation evaluation. In *Proceedings Conference of North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 362–366, Montreal, Canada.

Sun, Q., Li, R., Luo, D., and Wu, X. (2008). Text segmentation with LDA-based Fisher kernel. *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies*, pages 269–272.

Utiyama, M. and Isahara, H. (2001). A statistical model for domain-independent text segmentation. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 499–506, Toulouse, France.

Wallach, H., Mimno, D., and McCallum, A. (2009). Rethinking LDA: Why priors matter. In *NIPS*, Vancouver, B.C., Canada.

Wayne, C. (1998). Topic detection and tracking (TDT): Overview & perspective. In *Proceedings of the Broadcast News Transcription and Understanding Workshop*, Lansdowne, Virginia.

Yaari, Y. (1997). Segmentation of expository texts by hierarchical agglomerative clustering. In *Proceedings of the Conference on Recent Advances in Natural Language Processing*, Tzigov Chark, Bulgaria.

Autorenindex

Author Index

Chris Biemann
TU Darmstadt - FB 20, Informatik
biem@cs.tu-darmstadt.de

Hela Fehri
Mir@cl Laboratory, Faculty of Sciences economics and management of Sfax, Sfax University
hela.fehri@yahoo.fr

Kais Haddar
Mir@cl Laboratory, Faculty of Sciences of Sfax, Sfax University
kais.haddar@yahoo.fr

Verena Henrich
Seminar für Sprachwissenschaft, Universität Tübingen
verena.henrich@uni-tuebingen.de

Erhard Hinrichs
Seminar für Sprachwissenschaft, Universität Tübingen
eh@sfs.uni-tuebingen.de

Martin Riedl
TU Darmstadt - FB 20, Informatik

riedl@ukp.informatik.tu-darmstadt.de

Laurent Romary
Institut fr deutsche Sprache und Linguistik, Humboldt-Universit¨t Berlin
laurent.romary@inria.fr

Klaus Suttner
Seminar für Sprachwissenschaft, Universität Tübingen
suttner@sfs.uni-tuebingen.de