UNIVERSITÄT
DUISBURG
ESSEN

ABTEILUNG FÜR INFORMATIK UND
ANGEWANDTE KOGNITIONSWISSENSCHAFT

FAKULTÄT FÜR INGENIEURWISSENSCHAFTEN

INFORMATIK

Technischer Bericht Nr. 2009-01

GSCL-SYMPOSIUM

SPRACHTECHNOLOGIE UND EHUMANITIES

Wolfgang Hoeppner (Hrsg.)

14.02.2009

ISSN 1863-8554

IMPRESSUM:

Technische Berichte der Abteilung für Informatik und Angewandte Kognitionswissenschaft, Universität Duisburg-Essen

ISSN 1863-8554

Herausgeber:

Abteilung für Informatik und Angewandte Kognitionswissenschaft
Fakultät für Ingenieurwissenschaften
Universität Duisburg-Essen
Campus Duisburg
47048 Duisburg

http://duepublico.uni-duisburg-essen.de/informatik/berichte.xml

Symposium
»Sprachtechnologie und eHumanities«

26.02.2009 - 27.02.2009

UNIVERSITÄT DUISBURG-ESSEN, CAMPUS DUISBURG

GSCL

Gesellschaft für Sprachtechnologie & Computerlinguistik

PROGRAMMKOMITEE UND ORGANISATION

Wolfgang Hoeppner	Universität Duisburg-Essen
Angela Klutsch	Universität Duisburg-Essen
Marc Lechtenfeld	Universität Duisburg-Essen
Nino Simunic	Universität Duisburg-Essen

INHALTSVERZEICHNIS

Vorwort

In den 90er Jahren des letzten Jahrhunderts war ich mehrere Jahre Dekan des Fachbereichs ‚Literatur- und Sprachwissenschaften' an der Gerhard-Mercator Universität Duisburg, die dann 2003 mit der Universität Essen fusioniert wurde und dabei ihren Namenspatron eingebüßt hat.

Damals gab es auf einmal größere Geldtöpfe, die für die informationstechnische Ausstattung der Fachbereiche genutzt werden sollten. In meinem damaligen Fachbereich habe ich natürlich die Professoren angesprochen, ob sie sich nicht für derlei Dinge interessieren könnten. Das Ergebnis war niederschmetternd: die meisten haben gar nicht reagiert, und auf persönliche Befragung traten dann Meinungen zutage, die in Computern ein Teufelswerkzeug sahen, das ihre wissenschaftliche Arbeit verderben würde und Maschinen zu ästhetisch urteilenden Instrumenten missbrauchen würde. Dies war insbesondere in der Literaturwissenschaft so; die Linguisten waren da schon zugänglicher, vermutlich weil sie mit formalen Methoden besser vertraut waren.

Diese Zeiten haben sich geändert. Auch Literaturwissenschaftler haben gelernt, das Internet zu nutzen, und schreiben beispielsweise gerne E-Mails an Fachkollegen. Linguisten nutzen weltweit verfügbare Korpora aus und nehmen diese als Grundlage für ihre Forschungen.

In den letzten Jahren hat die Informationstechnologie die Geisteswissenschaften entdeckt, möglicherweise kann man auch sagen, die Geisteswissenschaften hätten die Informationstechnologie schätzen gelernt. Wie auch immer: es gibt zahlreiche Förderungsinitiativen in diesem Bereich. Und dies hängt auch mit den Digitalisierungsinitiativen der letzten Jahrzehnte zusammen. Je mehr digitalisierte Korpora existieren, desto mehr bisher unbekannte oder ungenutzte Quellen sind ortsunabhängig verfügbar. Die Frage stellt sich jetzt, wie diese Datenmassen einer Wissenschaftsgemeinschaft verfügbar gemacht werden können? Als ein Beispiel soll hier das kürzlich abgeschlossene

BMBF-Projekt Wikinger erwähnt werden[1], an dem die Fraunhofer Gesellschaft (St. Augustin), die Kommission für Zeitgeschichte (Bonn) und die Duisburger Computerlinguistik gemeinsam gearbeitet haben. In diesem Projekt wurde ein System für die semantische Annotation von Texten zum deutschen Katholizismus der letzten 200 Jahre entwickelt.

Das BMBF hat vor mehreren Jahren ein Sonderprogramm unter dem Namen eScience aufgelegt (http://www.bmbf.de/de/298.php), bei dem es um die informationstechnologische Unterstützung zahlreicher Wissenschaften ging; die Geisteswissenschaften waren dort ein Element, wenn auch noch kein sonderlich prominentes, aber das oben erwähnte Projekt Wikinger und auch TextGrid (http://www.textgrid.de/) gehören dazu. Im Rahmen der europäischen Projekte eContent*plus*[2] und Clarin (http://www.clarin.eu/) sind diese Aktivitäten ausgeweitet worden, die deutsche Variante ist die D-SPIN Initiative (http://www.sfs.uni-tuebingen.de/dspin/). Im Jahr 2007 hat das BMBF außerdem ein Programm „Wechselwirkungen zwischen Natur- und Geisteswissenschaften" aufgelegt mit den Pilotanwendungen ‚Archäologie und Altertumswissenschaften' sowie ‚Sprach- und Literaturwissenschaften' (http://www.bmbf.de/foerderungen/7774.php), in dem derzeit 16 Projekte gefördert werden. Die in Deutschland wichtigsten Wissenschaftsorganisationen haben eine Schwerpunktinitiative „Digitale Information"[3] herausgegeben. Zurzeit sind außerdem Aktivitäten bei der DFG für ein Schwerpunktprogramm ‚Digital Humanities' begonnen worden.

Es tut sich also etwas bei der Annäherung zwischen den Geisteswissenschaften und der Informationstechnologie, und das ist für beide Seiten gut. Das Duisburger Symposium ‚Sprachtechnologie und eHumanities' bietet ein breites und interdisziplinäres Forum, auf dem die Verflechtungen zwischen den Geisteswissenschaften und der Informatik diskutiert werden sollen.

1 Lars Bröcker, Stefan Paal, Andreas Burtscheidt, Bernhard Frings, Marc Rössler, Andreas Wagner, Wolfgang Hoeppner. "WIKINGER - Wiki Next Generation Enhanced Repositories". German e-Science Conference 2007. Baden-Baden, Germany, 2007 (http://www.ges2007.de).

2 http://ec.europa.eu/information_society/activities/econtentplus/index_en.htm

3 http://www.dfg.de/aktuelles_presse/das_neueste/ download/pm_allianz_digitale_information_details_080612.pdf

Die meisten Beiträge zu diesem Symposium sind in dem vorliegenden Band enthalten, der auch als Online-Version unter folgender Adresse verfügbar ist:
http://duepublico.uni-duisburg-essen.de/informatik/berichte.xml (ISSN: 1863-8554).

Lassen Sie mich abschließend die Beiträge dieses Bandes kurz charakterisieren. **Baum et al.** beschäftigen sich mit der Analyse gesprochener Korpora aus dem Bereich der Rundfunkdomäne. **Büchel** beschreibt interdisziplinäre XML-Technologien für die automatische Analyse mathematischer Beweise. Um die syntaktische Annotation diachroner Korpora geht es bei **Burghardt/Wolff**. **Fritzinger et al**. untersuchen die Extraktion signifikanter Wortpaare und stellen hierzu Werkzeuge vor. **Ignatova et al.** befassen sich mit der Informationsmotivation von Benutzern in sozialen Frage-Antwort Kontexten. Informationsextraktion im Zusammenhang mit RapidMiner wird von **Jungermann** thematisiert. Eine multilinguale politikwissenschaftliche Medientext-Analyse mit Hilfe von computer- und korpuslinguistischen Verfahren wird in **Kolb et al.** beschrieben. **Mehler et al.** stellen eine Arbeitsumgebung für die geisteswissenschaftlichen Fachwissenschaften dar. Im Kontext des Web 3.0 entwickelt **Scheffel** eine Relationserkennung in Fließtexten. **Toprak et al.** untersuchen Studien zur ‚Sentiment Analysis' in Aufsätzen aus dem Bereich der Bewerbungsanalysen. Aus dem Gebiet der klassischen Philologie untersucht **Vertan** multilinguale Benutzerschnittstellen. **Wagner et. al.** befassen sich mit einem literarischen Gedächtnis System am Beispiel der Kafka-Forschung.

Diese Themenvielfalt zeigt sicher das Potenzial, das informationstechnologische Werkzeuge für die Geisteswissenschaften anbieten können. Und es zeigt wohl auch, dass der Einsatz von Computern keine Existenzbedrohung für die Geisteswissenschaften bedeutet. Das wäre ja auch unangemessen. Eine Maschine ist immer dann brauchbar, wenn es um die Bewältigung großer Datenmengen geht, sie kann aber kaum oder gar nicht über die Qualität dieser Daten für die menschliche Gesellschaft verlässliche Aussagen produzieren. Sie kann Unterstützung bieten, nicht aber qualitative Entscheidungen fällen.

14.2.2009

Wolfgang Hoeppner

DiSCo — A Speaker and Speech Recognition Evaluation Corpus for Challenging Problems in the Broadcast Domain

Doris Baum, Barbara Samlowski, Thomas Winkler, Rolf Bardeli, and Daniel Schneider
Fraunhofer IAIS
Schloss Birlinghoven
Sankt Augustin
Germany

***Abstract*—Systems for speech and speaker recognition already achieve low error rates when applied to high-quality audiovisual broadcast data, such as news shows recorded in a studio environment. Several evaluation corpora exist for this domain in various languages. However, in actual applications for broadcast data analysis, the data requirements are more complex. There are many data types beyond the planned speech of the news anchorperson. For example, interesting live recordings from prominent politicians are often recorded in an environment with challenging acoustic properties. Discussions typically expose highly spontaneous speech, with different speakers talking at the same time. The performance of standard approaches to speech and speaker recognition typically deteriorates under such data characteristics, and dedicated techniques have to be developed to handle these problems. Corresponding evaluation corpora are needed which reflect the challenging conditions of the actual applications.**

Currently, no German evaluation corpus is available which covers the required acoustic conditions and diverse language properties. This contribution describes the design of a new speaker and speech recognition evaluation corpus for the broadcast domain, reflecting the typical problems encountered in actual applications.

I. Introduction

With the computing power, annotated training material, and refined recognition systems available today, speech and speaker recognition produce sufficiently good results for setting up a useful spoken document retrieval system for restricted domains. Current systems for German data such as [1] achieve satisfying error rates for speech recognition and spoken term detection on a test set of broadcast news data recorded in a studio environment. However, the test set used in the evaluation of this system only contains recordings with no background music or noise, no cross-talk and no telephone data. About half of it is planned speech from professional speakers, i.e., anchorpersons reading news. This leaves out a large part of material contained in broadcasts which is of particular interest to audio search engine users in media archives. Examples for such relevant material include:

- Spontaneous speech from emotionally charged situations, often containing hesitations and stammering
- Debates with speakers interrupting each other
- People with foreign or regional accents
- Voice-overs on foreign language interviews
- Live recorded interviews made in noisy environments
- Telephone interviews
- Public speeches, often containing reverberation
- Background music

Performance evaluations including such challenging problems are required to develop and compare new robust algorithms for speech and speaker recognition. Moreover, such evaluations are often asked for by professional users of spoken document retrieval systems, who need these figures in order to assess the business value of a system.

Although evaluation corpora with some of the required characteristics exist for other languages [2], [3], no sufficiently annotated corpus exists for the German language which covers the required range of material. This paper describes efforts to design a new speech and speaker evaluation corpus, DiSCo (Difficult Speech Corpus), with the goal of measuring and improving a system's performance by testing on representative material from the broadcast domain. Section II contains a summary of related work on speech and speaker recognition on broadcast data and in difficult conditions. Section III describes the most important adverse conditions in broadcast data we identified and the problems they pose. Section IV details the considerations and the decisions made during corpus design and the transcription process, and Section V gives results from experiments carried out on a preliminary version of the new corpus.

II. Speech and speaker recognition in broadcast data

Automatic speech recognition has a wide area of potential applications. Accordingly, the number and diversity of difficult environments for speech and speaker recognition is equally high. For example, speech recognition in car [4], [5] or motorcycle environments [6], in meetings [7], or in broadcast data are areas of active and busy research.

In this context, the broadcast domain is especially interesting for two reasons. First, there is ample demand for automatic analysis of speech in broadcast data. Applications

reach from content-based search and browsing in television, movie, and radio archives [1], [8] to content-enrichment tasks like automatic subtitling [9]. Second, although there are a number of challenging problems for speech technology in this domain, there are also large portions of material which are feasible for automatic methods and thus allow realistic applications to be built.

During the 1990s, broadcast news data has been seen as appropriate material for fostering research in speech recognition, see for example the 1996–1999 NIST Broadcast News Recognition Evaluation [10]. Such high quality broadcast data with large amounts of planned speech is no longer considered sufficiently difficult for the evaluation and promotion of speech recognition tasks. It is therefore often enriched by more difficult conversational speech. This can be seen, for example, in the NIST Rich Transcription Evaluation Project [11]. Also, additional languages move into the focus of attention, e.g., Arabic [12] and Chinese [13].

One problem in the broadcast domain that does not stem from a specific acoustic situation is vocabulary size. In addition to the fact that the vocabulary for this domain is usually quite large, it is also subject to perpetual change. No matter how large the dictionary of a speech recognizer is, new words will always move into the focus of interest and often become the most important to be recognized. There are various approaches for coping with such *out of vocabulary* (OOV) words. One very flexible approach, here, is to not only create word transcriptions but also to retain syllable or other subword transcriptions. In this way, retrieval applications can search for out of vocabulary words by using their subword transcriptions [1], [8].

Speaker recognition is a valuable additional tool for the analysis of broadcast data. Often, users are interested in searching for information provided by specific interesting speakers like politicians or celebrities. In addition to this gain in metadata, speaker recognition allows the application of high-performance acoustic models for individual speakers.

The current standard speaker recognition techniques, such as [14], work very well for clean, studio-recorded, wideband speech, even for large sets of speakers [15]. However, the performance declines dramatically for bad recording or transmission channel conditions (e.g., for telephone data [15]) or when there is mismatch between training and test data capturing conditions. This is due to the fact that they use spectral features to capture the shape of a speaker's vocal tract in order to identify him or her, an approach vulnerable to channel variation and spectral noise. In broadcast data, these kinds of problems are often found, thus making reliable speaker recognition a challenging task. To overcome the limitations imposed by spectral features, a number of speaker recognition approaches using high-level features which try to capture the speakers' intonation, pronunciation, and style, have been proposed [16], [17], [18], [19], [20]. High-level features often require more training and test data but are less susceptible to channel variation and varying acoustic conditions. In order to test which of these techniques might be applicable to a system for German broadcast data, development and test data for speaker recognition from the domain is needed.

III. Adverse conditions in broadcast data

For our purposes, broadcast data falls into three categories:

First, data produced in a studio environment with professional equipment and trained speakers, for which the quality of the speech and audio data is rather high. Even in this controlled environment, the speech information can suffer from certain influences, which makes an automatic analysis of speech more difficult.

Second, data from non-studio productions, like live broadcasts from sports events or documentary features, for which environmental conditions can be even more manifold and adverse. As news and documentaries cover real-life situations, practically all environmental noise conditions might also occur in broadcast and have to be taken into account.

Finally, in both situations an overlap of speakers, i.e., either various speakers speaking at the same time or the situation of voice-overs, poses a considerable challenge to existing speech technology.

However, it can be assumed that some conditions are more likely for the broadcast domain than others. For the development of DiSCo the following conditions are considered to be most dominant and representative for broadcast data, and, hence, should be covered by the corpus:

- **Additive Noise.** Additive noise is the main source of degradation for many speech recognition systems and the most manifold as well. Thus, many scientific publications broach the issue of development and evaluation of algorithms for the reduction of additive noise (e.g, [21], [22]). Every sound which is recorded but which is not part of the analyzed speech can be considered as noise as it generally leads to degradation of the speech or speaker recognition performance.
  Typical additive noise in broadcast can be traffic noise, camera clicking, noise from machines, stadium noise during sports events, etc. Music and speech in the background of a speaker are also additive noise in terms of the previous definition. Due to their specific characteristics, both, music and speech, are classified separately in this corpus, as they might introduce additional challenges for speech analysis. Additive noise can be present in every program, but it is more likely to occur in programs like infotainment shows, talk shows, sports event coverage, news event coverage, and light programs.
- **Music in the Background.** Music in the background of a speaker is a common type of additive noise in broadcast programs. But due to its specific harmonic characteristics, the influence of background music on speech analysis is often severe and, therefore, of particular interest in speech recognition for the broadcast domain [23], [24]. Hence, music is classified separately for this corpus. Music is often mixed artificially into the background of a speaker to create a certain atmosphere. But music can also be part of the real acoustic environment of a recording. Music

in the background is used or can be present in several programs, e.g., infotainment shows and documentaries.

- **Speech in the Background.** Background talk is very critical for speech analysis, as it is rather difficult to separate two or even more speakers [25]. Another speaker in the background – or, even worse: cross-talk situations, i.e., two speakers speaking at the same time with about the same volume – dramatically decreases the performance of speech and speaker recognition systems. Background speech is often present in interviews or in voice-over situations like translations of original speech. Typical programs for background speech, voice-over, and cross-talk are mainly political debates, talk shows, and news.
- **Reverberation.** Reverberation and its effect and compensation in robust speech recognition is a seperate field of research [26]. Reverberation is caused by acoustic characteristics of the room. Studio data is generally low in reverberation, but speech in political debates of the parliament, for example, often suffers from reverberation effects. Similar challenges are echos caused by acoustic feedback. A prominent example is telephone speech in the broadcast environment. Echos mainly occur when the speaker on the telephone uses handsfree devices or listens to the delayed channel of his broadcast device while calling a live show. Parliament debates and call-in shows are qualified for providing data with distortions caused by reverberation and echos.
- **Telephone Speech.** Telephone speech has specific channel characteristics and provides much worse speech quality than high quality studio recordings. Additionally, the channel characteristics also vary for different phone channels (GSM, ISDN, analog connections, etc.). Additive noise and echos can also be present in telephone speech. Thus, telephone speech suffers from many different sources of degradation [27]. In the broadcast domain, telephone speech can mainly be found for telephone interviews and for some live coverages from foreign correspondents (often with additive noise in the background). A sufficient quantity of telephone speech in the broadcast domain is covered by adding a call-in show to the corpus.
- **Speech Diversity.** A more generalized challenge in automatic speech recognition and speech analysis is the diversity of speech. Though most speech in the broadcast domain is quite clear and planned, fast speakers, speakers with different accents and dialects as well as spontaneous speech can also be present in specific programs. All these variations and individual characteristics in speech complicate a reliable automatic speech recognition [28]. A broad selection of different speech and speaker characteristics is achieved by capturing a variety of different programs including news, talk shows, sports shows, etc.

IV. Corpus Design

A. Intended Use and Applications of the Corpus

There are many different types of corpora, each having its own set of demands on the data and the annotations. In Llisterri's guidelines for building spoken corpora [29], two large groups are identified according to their applications and user communities:

The first group consists of corpora developed by the so-called "corpus linguistics community" in order to provide data for linguistic research. Topics of interest include conversation and discourse analysis, children's or child-directed speech, and the development of lexica. Corpora of this type require the data to be as natural as possible. In many cases, spontaneous conversation is preferred. Annotations may include grammatical tagging as well as prosodic information, while exact information about word pronunciation can often be disregarded.

The second group comprises corpora compiled by the "speech community" which focuses on theories of phonetics and phonology as well as on technical and technological applications thereof. Traditionally, corpora developed by this user group are produced in a very controlled environment. Often, prompt sentences are read aloud and recorded under laboratory conditions. The speech community tends to place more emphasis on the pronunciation of words than on prosody or grammatical issues.

As an evaluation corpus for automatic speech and speaker recognition, our database belongs to the second category. However, in order to simulate the real-life situations our recognition system is intended for, we use natural and spontaneous speech gathered from reports and interviews transmitted on television and the internet, rather than controlled recordings of phonetically balanced sentences. Our database is designed to cover a wide range of acoustic situations so as to reflect the many challenges confronting automatic speech and speaker recognition and term detection outlined in the previous sections. It includes speech samples from a number of well-known public figures of interest to train and test speaker recognition in adverse conditions.

B. Types of Data Included

One difficulty in putting together a broadcast corpus suitable for our purposes is the uncertainty in predicting which television programs will contain what type of data. Therefore, a good coverage of programs containing the different adverse situations targeted by the corpus is vital. The following list gives an overview of the recorded material and the special acoustic situations they cover:

- News Broadcasts
  Daily news programs contain different types of data, but the speaking style, in general, is formal and planned. Often, texts are read by professional newscasters. There are longer passages of clean speech, which can be used in comparisons against more complicated data. During reports and commentaries from experts and politicians, however, background noise is often present, and in many cases news summaries are read against a background of music.
- In-depth News Commentaries

Programs of this type provide detailed analysis and discussion of current events. The topics are similar to those dealt with in news broadcasts, but there are also longer interviews with prominent public figures and celebrities. Overlapping tends to occur in discourse between interview partners as well as in passages of foreign speech which are superimposed with simultaneous translations.

- Sports Commentaries
  These shows, which are similar in structure to the programs in the foregoing category, feature news from the world of sports, with German shows often focusing on soccer events. These shows contain informal interviews, on occasion with voice-over translations, as well as a considerable amount of audience and stadium noise.
- Infotainment Shows
  Popular science shows are conducted in a planned but informal speech style. They also contain short passages of spontaneous speech from street interviews. Background music is especially prevalent here, so these recordings serve as test material for dealing with voice over music.
- Political Talk Shows
  In these discussion rounds, politicians, public figures, and other guests debate specific topics. They contain passages of heated argumentation with spontaneous speech and considerable speaker overlap. Moreover, they are a challenging test instance for speaker recognition.
- Parliamentary Debates
  The speeches in these debates are often planned, but the recordings include much background noise from the audience as well as a high level of reverberation. Furthermore, as the speakers are important politicians, the data is a challenging test case for speaker recognition.
- Call-in Shows
  One important application for robust speech recognition is telephone speech. Short telephone interviews can occasionally be found in news broadcasts and commentaries. To increase this type of data in our corpus, we decided to include recordings from a call-in show. The informal style of this type of show increases the spontaneous speech part of the corpus.
- Crime Fiction Series
  As a final especially challenging test case, we included a few installments of crime series. In these programs, several kinds of complex speech material are combined - speaker overlap, excessive background noise, and background music.

C. The Annotation Process

The manual annotation process is designed to be iterative: A preliminary set of annotations is produced and then reviewed by the human annotators in terms of content and formal aspects so that mistakes can be corrected. Where expedient, the annotation guidelines are modified in order to obtain better results in the next cycle.

The recordings are annotated in three phases. In the first phase the data is segmented into utterances and transcribed orthographically. For this process, we use the program Transcriber[1]. During the following two phases, the utterances are classified into groups using an especially developed annotation program, called DIVE. In the second phase, each utterance is labelled according to speaker. The utterances are also classified according to a specific set of features from a given list (see Table I).

TABLE I
ANNOTATION FEATURES AND ATTRIBUTES FOR THE DISCO CORPUS

Feature	Attributes
background noise	yes / no
channel quality	studio / telephone / other
type of speech	spontaneous / planned
speech rate	low / medium (default setting) / high

During the third phase, the data groups are analysed to refine the classes for re-annotation. Different types of background noise are specified.

This procedure is devided into the following steps:

- **Step 1** - Recording the data:
  In the first step, the television programs are recorded by a digital video recorder. The resulting files are saved into separate directories according to program name and into subdirectories indicating the time and date of recording. Each recording comprises three different types of files: an index file, the video files themselves and a text file with additional information such as program subheadings or summaries. At this stage, a first quality check secures that the programs have been recorded properly.
- **Step 2** - Producing the scripts:
  In order to further process the data, several scripts have to be created to convert the files into the required formats. These will be described together with the step in which they are used. Unlike the other tasks described here, this step does not have to be repeated for every recording.
- **Step 3** - Producing the audio files:
  The audio files used for transcribing the recordings have to be extracted from the video file with the help of a script. For automatic speech recognition as well as for human text transcriptions a wave file (16 kHz, 16 bit, stereo) is needed. The following annotation according to classes will be done on the basis of an mp4 audio/video file.
- **Step 4** - Gathering the metadata:
  Another script is necessary to gather information about the recording and the program into an XML-file.
- **Step 5** - Creating the text transcription framework
  Optionally, the transcribers can use an automatically computed transcription and segmentation as a basis for their work.
- **Step 6** - Creating the orthographic transcriptions
  In the first annotation phase, the data are transcribed orthographically. Silence or pure background noise, unintelligible or foreign speech, and speaker overlap are

[1]http://trans.sourceforge.net/

not transcribed, but marked separately. For this step, annotation guidelines detail the transcription conventions for, among other things, numbers, compound words, words with different possible spellings, contractions and hesitations.

- **Step 7** - Combining transcription and metadata
  The orthographic transcription and the collected information about the respective recording are combined into a single XML file.
- **Step 8** - Creating the classification framework
  As in step 5, a skeleton classification file is automatically created to aid the annotators in their task of dividing the data according to speaker and the selected features.
- **Step 9** - Classifying the data
  In the second annotation phase, the utterances are tagged according to speaker as well as to a set of predetermined features.
- **Step 10** - Analysing the annotated data
  The resulting groups of data are analysed and a new set of classes are determined according to which the utterances are to be annotated a second time.
- **Step 11** - Reiteration of classification
  During the third annotation phase, the refined class features are used to re-annotate the utterances.
  Following these steps, a well annoted corpus with rich information for the evaluation and development of speech technology is derived.

V. Experimental results

A. Linguistic analysis

Preliminary linguistic analysis has been performed for a subset of the recorded programs in order to gain insight into the distribution of important parameters for speech recognition. This preliminary corpus contains approximately four hours of speech from five different German television programs covering a political discussion show, a foreign affairs report, an interview show, a regional infotainment show, and a sports show. Figure 1 shows the fraction of the corpus covered by each of these programs.

Transcribable speech accounts for 77.6 percent of the total time, i.e., three hours and ten minutes. The remaining part comprises 16.5 percent of silence or pure background noise, 4.7 percent of unintelligible speech, and 1.8 percent of speaker overlap, with two or more people speaking at the same time. The amount of time taken up by periods of silence, unintelligible speech, and speaker overlap vary from program to program. As can be seen in Table II, discussion shows contain more speaker overlap than news commentaries and a program featuring international news includes larger amounts of foreign speech, which has been tagged as unintelligible.

One important aspect of linguistic corpus analysis is the assessment of word type distributions. Frequency lists can be produced which record the different word forms or types that the corpus consists of together with the number of tokens belonging to each of these word types, i.e., the number of times that one particular word form appears in the corpus. Depending on the aim of the analysis, the definition for distinguishing word types can vary. For some studies, e.g., determining the vocabulary of a language, it may be advisable to count different grammatical forms of a word as one word type or to differentiate between words that are spelled in the same way but have different meanings [30]. Generally the word types of a corpus are not distributed equally. On the contrary, studies often show that while a few word types appear very often, a large number occur seldom or only once, i.e., they follow a Zipfian distribution [31].

The fact that corpora regularly contain a few strongly represented words and a large percentage of ”hapax legomena”, i.e., word types which appear only once, poses a challenge for corpus-based research and applications of speech technology, where representative data are required [32]. This is also true for evaluation corpora such as the DiSCo database. Besides word transcripts, some speech recognizers can also produce transcripts on the subword level, allowing for vocabulary independent speech search. As our speech recognition system produces both word and syllable transcripts, the type-token relations for this corpus will be analyzed on the level of both, syllables as well as words.

The database collected so far contains 34,387 word tokens that can be divided into 6,305 orthographic or 6,067 phonological word types. The latter are distinguished by their standard pronunciation. Accordingly, homophones are counted as one type and stammered words are not treated as separate types.

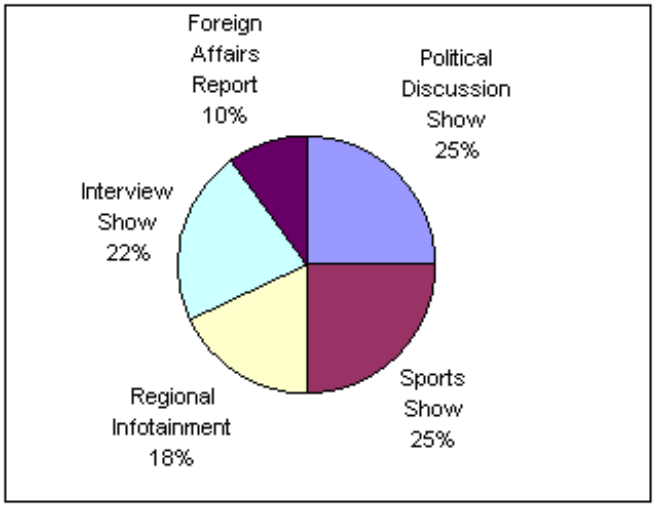


Fig. 1. Contribution of different broadcast formats to the total length of the preliminary corpus.

TABLE II
Distribution of silence, unintelligible speech and overlapping speeech according to program type

Program	Silence	Unintelligible	Overlap
Interview show	14.54%	3.21%	0.88%
Political discussion show	11.83%	1.14%	4.89%
Foreign affairs report	17.89%	15.19%	0.34%
Regional infotainment with dialect	18.49%	2.60%	0.93%
Sports show	20.88%	6.40%	0.82%

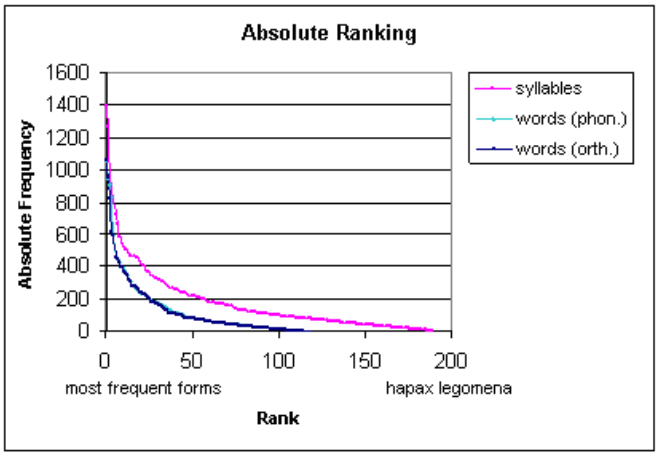


Fig. 2. Absolute word and syllable frequencies according to their frequency rank.

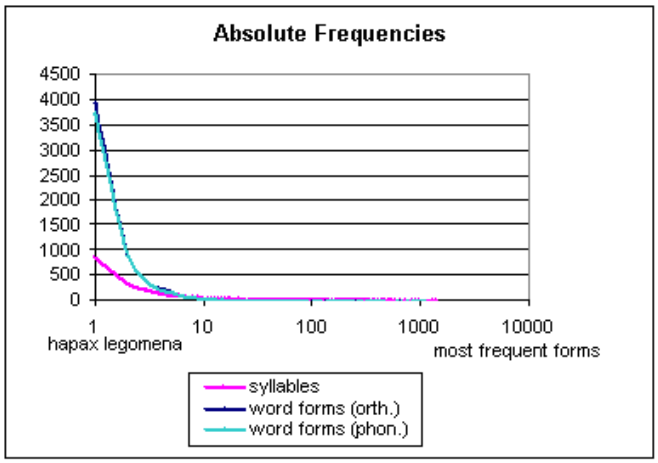


Fig. 3. Word type frequency vs. occurrence per frequency.

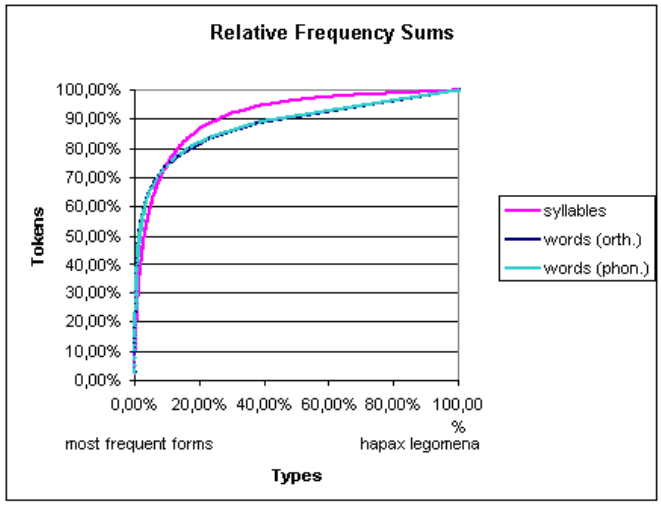


Fig. 4. Running total of relative type and token frequencies.

The corpus can be divided into 59,788 syllable tokens, which belong to 2,653 types. As for the phonological word forms, syllable types are defined by pronunciation.

A chart representing the absolute word and syllable frequencies according to their rank shows that although there are more different syllables than words, the syllable frequencies decline less rapidly than word frequencies and require more ranking steps to reach the lowest syllable frequency (Figure 2). One reason for this is that they start out on a higher level, as the most frequent word form, the German article *die*, appearing 1059 times, is subsumed by the corresponding syllable, which occurs 1396 times. It becomes apparent here as well as in the following charts that the curves for orthographic and phonological words follow very similar paths.

Another way of visualizing the results is by plotting the frequency of the word types against the number of occurrences for this frequency, e.g., how many word types appear only once in the corpus (Figure 3). This puts more emphasis on uncommon word types than the frequency ranking approach [31]. Here, it becomes obvious that the number of syllables that occur only once is significantly lower than the number of singly occurring word forms.

The last fact is also confirmed by an analysis of relative frequencies. While about 60 percent of the corpus's word types occur only once, comprising 10 percent of the total corpus, the percentage of hapax legomena syllables is slightly above 30 percent. Only 1.4 percent of the corpus is made up of unique syllables. Figure 4, which represents the running total of relative type and token frequencies, starting with the most frequent types, shows that both on word and on syllable level, 75% of the corpus can be represented by the top ten percent of word or syllable types. Furthermore it can be seen that uncommon syllable types make up less of the corpus in comparison to rare word types.

On the whole, both the phonological and the orthographic word forms of our corpus follow the expected Zipfian distribution. In absolute numbers, there are less different syllable types that have to be recognized, while a relative analysis shows that frequent syllables comprise more and infrequent syllables less of the corpus than the corresponding word types.

B. Automatic Speech Recognition

Preliminary experiments using automatic speech recognition (ASR) have been carried out on a subset of the recorded programs. The goal of this first pre-evaluation was to gain insight into the challenges of the individual recording types. We used an ASR setup based on the configuration described in [1] with an increased recognition vocabulary of 200,000 words and a trigram language model.

In order to eliminate the effect of automatic segmentation errors on the speech recognition result, a manual segmentation into speech segments was carried out before the actual transcription. Table III shows the word error rate on the manually segmented speech portion of the selected evaluation files.

The resulting error rates reflect the varying level of acoustic and linguistic complexity of the recordings. The lowest overall

TABLE III
OVERALL WORD ERROR RATE (WER) ON SELECTED PROGRAMS

Program	WER
News show, only planned speech	16.1 %
Interview show	29.4%
Political discussion show	39.9%
Foreign affairs report	41.8%
Regional infotainment with dialect	52.0%
Sports show	64.5%

word error rate can be observed on the planned speech portion of a broadcast news show, read by a professional speaker in a silent studio environment. The results on the interview program indicate that spontaneous speech presents an additional challenge to speech recognition, even if the interviewed people in the analyzed show are *media professionals* such as politicians. The recognition rate degrades further if the prevalent speech type changes from interview to discussion. Here, the speech of the participants is not only highly spontaneous, it can also be emotional, and vary greatly in speed. Speakers interrupt each other frequently, and this cross-talk makes speech recognition even harder.

As stated above, an additional challenge for speech recognition algorithms is background noise. A large part of the evaluated foreign affairs report contains voice-overs with the translation of a non-German recording, with two active voices confusing the speech recognizer. Moreover, the number of OOV words in the report is higher than the average OOV rate observed in the data annotated so far.

The performance of an ASR system drops if the mismatch between training and evaluation data increases. This is particularly the case when dialectal speech is to be recognized and the dialect was not present in the training set. Without any additional acoustic adaptation [33] the word error rate increases significantly.

The sports show has some challenging acoustic and linguistic properties, posing additional problems for the speech recognition system. There is a large portion of highly spontaneous speech in interviews, as well as a high number of OOVs due to the frequent occurrence of proper names of athletes or sports clubs. Moreover, recordings from sport events usually take place in a rather loud acoustic environment, including intense crowd noise or noise from the sport itself (such as motor car noise).

Although the observed word error rates are rather high, it is still possible to use the resulting transcripts for spoken document retrieval [34]. Corresponding information retrieval experiments with Spoken Term Detection comparable to [1] will be carried out in future evaluations.

To gain further insight into the challenges of the various programs, they were manually labeled according to the type of speech used. One of the labels *spontaneous*, *planned*, and *unsure* was assigned to each segment of speech manually. Only those segments labeled as *spontaneous* or as *planned* were used for the further analysis, to investigate the difficulty they pose for automatic speech recognition. Table IV shows the amount of planned and spontaneous speech in each of the programs. It becomes apparent that the political discussion show contains mostly spontaneous speech and that the foreign affairs report has mostly planned speech – from reporters and interpreters. For the rest of the programs, there is a rather balanced proportion of both speech types.

TABLE IV
TIME OF SPEECH TYPES IN THE SELECTED PROGRAMS

Program	Minutes planned	Minutes spontaneous
Interview show	8:06	8:17
Political discussion show	3:39	31:32
Foreign affairs report	17:05	0:49
Regional infotainment with dialect	20:50	15:19
Sports show	19:40	27:31

TABLE V
WORD ERROR RATES (WER) ON PLANNED AND SPONTANEOUS SPEECH

Program	WER planned	WER spontaneous
Interview show	18.30%	39.00%
Political discussion show	27.90%	40.50%
Foreign affairs report	39.70%	76.10%
Regional infotainment with dialect	48.50%	54.40%
Sports show	58.70%	68.30%
Weighted sum	44.60%	52.00%

The word error rates were recalculated for both classes, the results are shown in Table V. Speech type alone can not explain the differences in error rates between the programs – other factors have to be taken into account. While for the interview show the word error rate of planned speech is almost as low as for the news broadcast in Table III (18.3% vs. 16.1%), the other programs have a much higher word error rate on their planned speech part. We suspect that the reasons for this are background music, noise, and overdubbing of translations in the case of the foreign affairs report, dialect and talking speed in the case of the regional infotainment show, and stadium and audience noise in the case of the sports show. For spontaneous speech, the word error rate is about 40% in clean acoustic environments with practised speakers talking standard German, such as in the interview and political discussion shows. This rate deteriorates further if dialect is used or in noisy environments.

Altogether, spontaneous speech poses severe problems for automatic speech recognition, increasing the word error rate, often by at least 20% – but it is, of course, not the only challenge to be tackled. So a corpus for evaluation of difficult speech must facilitate more annotation categories, like dialect, talking speed, and background noise.

VI. CONCLUSION

Taking speech and speaker recognition in real world scenarios to the next level is only possible with a corpus documenting exactly those challenges which are just out of reach of the current state of the art. For the German language, no such corpus has been available. Our experiments show that the types

of broadcast material selected for our corpus covers very well the kind of material that is difficult to handle by state-of-the-art algorithms. Once completed, the DiSCo corpus will serve as a solid foundation for the evaluation of progress in the domains it covers and will thus help developing more robust speech and speaker recognition algorithms.

VII. Acknowledgments

The work for this contribution was supported by the projects CONTENTUS[2], MoveOn[3] and VITALAS[4].

References

[1] D. Schneider, J. Schon, and S. Eickeler, "Towards large scale vocabulary independent spoken term detection: Advances in the Fraunhofer IAIS Audiomining System," in *Proceedings of the ACM SIGIR Workshop "Searching Spontaneous Conversational Speech" held at SIGIR '08*, J. Köhler, M. Larson, F. Jong de, W. Kraaij, and R. Ordelman, Eds., Singapore, 24 July 2008 2008. [Online]. Available: http://ilps.science.uva.nl/SSCS2008/Proceedings/sscs08_proceedings.pdf

[2] J. Garofolo, E. Voorhees, C. Auzanne, and B. Stanford, V.and Lund, "Design and preparation of the 1996 hub-4 broadcast news benchmark test corpora," in *Proceedings of the DARPA Speech Recognition Workshop.*, 1997, pp. 15–21.

[3] S. Galliano, E. Geoffrois, G. Gravier, J.-F. Bonastre, D. Mostefa, and K. Choukria, "Corpus description of the ester evaluation campaign for the rich transcription of french broadcast news," in *Proceedings of the 5th international Conference on Language Resources and Evaluation (LREC 2006).*, 2006, pp. 315–320.

[4] A. Moreno, B. Lindberg, C. Draxler, G. Richard, K. Choukri, S. Euler, and J. Allen, "SPEECHDAT-CAR. a large speech database for automotive environments," in *Proceedings of the 2nd International Conference on Language Resources and Evaluation, (LREC 2000)*, Athens, Greece, 2000.

[5] Y. Gong, "Speech recognition in noisy environments: a survey," *Speech Communication*, vol. 16, no. 3, pp. 261–291, 1995.

[6] T. Winkler, T. Kostoulas, R. Adderley, C. Bonkowski, T. Ganchev, J. Köhler, and N. Fakotakis, "The moveon motorcycle speech corpus," in *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, ELRA, Ed., Marrakech, Morocco, May 2008.

[7] E. Shriberg, R. Dhillon, S. Bhagat, J. Ang, and H. Carvey, "The ICSI Meeting Recorder Dialog Act (MRDA) Corpus," in *Proceedings of 5th SIGdial Workshop on Discourse and Dialogue*, M. Strube and C. Sidner, Eds., 2004, pp. 97–100.

[8] M. Akbacak, D. Vergyri, and A. Stolcke, "Open-vocabulary spoken term detection using graphone-based hybrid recognition systems," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2008).*, 2008, pp. 5240–5243.

[9] P. Wambacq, P. Vanroose, X. Yang, J. Duchateau, and D. H. V. Uytsel, "Speech recognition for subtitling purposes," in *Proceedings 5th International Conference Languages & The Media*, November 2004, p. 46.

[10] "The 1999 NIST Evaluation Plan for Recognition of Broadcast News, in English," 1999. [Online]. Available: http://www.nist.gov/speech/tests/bnr/1999/bnews_99_spec.html

[11] "Rich transcription evaluation project." [Online]. Available: http://www.nist.gov/speech/tests/rt/

[12] D. Vergyri, A. Mandal, W. Wang, A. Stolcke, J. Zheng, M. Graciarena, D. Rybach, C. Gollan, R. Schlater, K. Kirchoff, A. Faria, and N. Morgan, "Development of the sri/nightingale arabic asr system," *to appear in Proceedings of Interspeech 2008, Brisbane, Australia*, 2008.

[13] S. Chu, H. kwang Kuo, Y. Y. Liu, Y. Qin, Q. Shi, and G. Zweig, "The IBM Mandarin Broadcast Speech Transcription System," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2007)*, vol. 2, April 2007, pp. II–345 – II–348.

[2]http://theseus-programm.de/scenarios/de/contentus.html

[3]http://www.m0ve0n.net/

[4]http://vitalas.ercim.org/

[14] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," in *Digital Signal Processing*, vol. 10, 2000, pp. 19–41.

[15] D. A. Reynolds, "Large population speaker identification using clean and telephone speech," *IEEE Signal Processing Letters*, vol. 2, no. 3, pp. 46–48, March 1995.

[16] D. Reynolds, W. Andrews, J. Campbell, J. Navratil, B. Peskin, A. Adami, Q. Jin, D. Klusacek, J. Abramson, R. Mihaescu, J. Godfrey, D. Jones, and B. Xiang, "The SuperSID project: Exploiting high-level information for high-accuracy speaker recognition," in *Proceedings of the International Conference on Audio, Speech, and Signal Processing*, vol. 4, Hong Kong, 2003, pp. 784–787. [Online]. Available: http://www.clsp.jhu.edu/ws2002/groups/supersid/icassp03_overview.pdf

[17] D. A. Reynolds, J. P. Campbell, W. M. Campbell, R. B. Dunn, T. P. Gleason, D. A. Jones, T. F. Quatieri, C. B. Quillen, D. E. Sturim, and P. A. Torres-Carrasquillo, "Beyond cepstra: Exploiting high-level information in speaker recognition," in *Workshop on Multimodal User Authentication*, Santa Barbara, California, December 2003, pp. 223–229.

[18] M. K. Sönmez, E. Shriberg, L. Heck, and M. Weintraub, "Modeling dynamic prosodic variation for speaker verification," in *International Conference on Spoken Language Processing (ICSLP98)*, vol. 7, Sydney, Australia, 1998, pp. 3189–3192.

[19] F. Weber, L. Manganaro, B. Peskin, and E. Shriberg, "Using prosodic and lexical information for speaker identification," in *IEEE International Conference on Acoustics, Speech, and Signal Processing 2002 (ICASSP '02)*, vol. 1, 2002, pp. 141–144. [Online]. Available: http://www.icsi.berkeley.edu/ftp/pub/speech/papers/icassp02-spid.pdf

[20] B. Peskin, J. Navratil, J. Abramson, D. Jones, D. Klusacek, D. A. Reynolds, and B. Xiang, "Using prosodic and conversational features for high-performance speaker recognition: report from JHU WS'02," in *IEEE International Conference on Acoustics, Speech, and Signal Processing 2003 (ICASSP '03)*, vol. 4, April 2003, pp. 792–795. [Online]. Available: http://www.icsi.berkeley.edu/ftp/global/pub/speech/papers/icassp03-peskin2.pdf

[21] H. Hirsch and D. Pearce, "The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions," in *6th International Conference on Spoken Language Processing (ICSLP'00)*, Bejing, China, October 2000.

[22] J. Ming, "Noise compensation for speech recognition with arbitrary additive noise," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 14, no. 3, pp. 833–844, May 2006.

[23] B. Raj, V. Parikh, and R. Stern, "The effects of background music on speech recognition accuracy," in *ICASSP '97: Proceedings of the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '97)-Volume 2*. Washington, DC, USA: IEEE Computer Society, 1997, p. 851.

[24] P. Vanroose, "Blind source separation of speech and background music for improved speech recognition," in *Proceedings of the 24th Symposium on Information Theory in the Benelux*, 2003, pp. 103–108.

[25] J. R. Hershey, S. J. Rennie, P. A. Olsen, and T. T. Kristjansson, "Super-Human Multi-Talker speech recognition: A graphical modeling approach," *Computer Speech & Language*, vol. In Press, Accepted Manuscript, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/B6WCW-4V8VS6S-1/2/e7f3b944847579 52bb02a525b3b44772

[26] S. Thomas, S. Ganapathy, and H. Hermansky, "Recognition of reverberant speech using frequency domain linear prediction," *Signal Processing Letters, IEEE*, vol. 15, pp. 681–684, 2008. [Online]. Available: http://dx.doi.org/10.1109/LSP.2008.2002708

[27] P. Moreno and R. Stern, "Sources of degradation of speech recognition in the telephone network," *Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on*, vol. i, pp. I/109–I/112 vol.1, Apr 1994.

[28] M. Nakamura, K. Iwano, and S. Furui, "Differences between acoustic characteristics of spontaneous and read speech and their effects on speech recognition performance," *Computer Speech & Language*, vol. 22, no. 2, pp. 171–184, Apr. 2008. [Online]. Available: http://www.sciencedirect.com/science/article/B6WCW-4PCPFHV-1/2/17f5b4d652317795d7e57a65744c1c97

[29] J. Llisterri, "Preliminary recommendations on spoken texts," Expert Advisory Group on Language Engineering Standards, Tech. Rep., 1996. [Online]. Available: http://www.ilc.cnr.it/EAGLES96/spokentx/spokentx.html

[30] G. Kennedy, *An introduction to corpus linguistics*, ser. Studies in language and linguistics. London: Longman, 2003.
[31] E. Leopold, "Das Zipfsche Gesetz," *Künstliche Intelligenz*, no. 2/02, p. 34, 2002.
[32] J. Sinclair, "Corpus and text: Basic principles," in *Corpus and Text: Basic Principles*. Oxford: Oxbow Books, 2005, pp. 1–16.
[33] P. C. Woodland, "Speaker adaptation for continuous density HMMs: a review," in *ITRW on Adaptation Methods for Speech Recognition*, 2001, pp. 11–19. [Online]. Available: http://publications.eng.cam.ac.uk/2000/
[34] J. Garofolo, G. Auzanne, and E. Voorhees, "The TREC spoken document retrieval track: A success story," in *Proceedings of the Content Based Multimedia Information Access Conference*, 2000. [Online]. Available: http://citeseer.ist.psu.edu/garofolo00trec.html

Datenbank- und XML-Technologien im Projekt NAPROCHE

Gregor Büchel

1. Einleitung

Das Projekt NAPROCHE (=: Natural Language Proof Checking), das an der Universität Bonn gegründet wurde[1], untersucht Konzepte und Methoden der maschinellen Prüfung von mathematischen Beweisen, die in einer natürlichen Sprache (z.B. Deutsch oder Englisch) abgefasst sind und maschinenlesbar vorliegen. Das NAPROCHE-Projekt ist ein interdisziplinäres Vorhaben von Mathematikern, Linguisten und Informatikern.

Mathematische Beweise bedürfen der Schriftform, damit die Behauptungen neuer Sätze korrekt aus Axiomen oder aus bereits bewiesenen Sätzen deduziert werden können. Diese Deduktion ist eine Kette logischer Schlüsse, die die Evidenz der vorgelegten Behauptung herstellt. Wird der Text eines mathematischen Beweises durch einen Rechner erfasst, können zunächst die Vorteile genutzt werden, die für beliebige Texte, die mit Textverarbeitungsprogrammen erfasst werden, gelten (Reproduzierbarkeit, Erweiterbarkeit, umfangreiche Alphabete von Sonderzeichen usw.). Für Mathematiker stellt sich darüber hinausgehend die Frage nach einem Mehrwert: Unter welchen Bedingungen kann der Rechner als Beweisprüfer eingesetzt werden?

2. Der Rechner als Beweisprüfer: MIZAR

Dass der Rechner als Beweisprüfer eingesetzt werden kann, belegen Proof Checking Programme, wie z.B. MIZAR[2]. Bei der Nutzung solcher Programme werden jedoch direkt zwei Probleme sichtbar, die den mit der obigen Frage vermuteten Mehrwert schmälern:
(1) MIZAR, wie andere Proof Checker auch, verlangt, dass der Beweis in einer eigenen formalen Sprache, deren Grammatik bei MIZAR durch Produktionsregeln in Backus-Naur-Form (BNF) spezifiziert ist[3], erfasst wird. Faktisch bedeutet das, dass ein mathematischer Beweis, der mit einem beliebigen Texteditor E erfasst worden ist, unter MIZAR ein zweites Mal editiert werden muss. Dieses soll an einem Beispiel illustriert werden: Es soll folgende elementare Eigenschaft der Anordnungsrelation „<“ („kleiner als“) im Körper der reellen Zahlen bewiesen werden:

Lemma: „Gilt $x < y$ und ist $a < 0$, so folgt $ax > ay$.“
„***Beweis.*** Nach Voraussetzung ist $y-x > 0$ und $-a > 0$. Nach Axiom (A.3)[4] folgt daraus $(-a)(y-x) > 0$, d.h. $ax – ay > 0$, also $ax > ay$.“ [FORST, S.15]

Der Beweis dieses Lemmas kann in MIZAR als Text einer formalen Sprache, der MIZAR Beweissprache, die einer Programmiersprache ähnlich ist, editiert werden:

```
environ
vocabularies ARYTM, ARYTM_1;
notations REAL_1, XXREAL_0, NUMBERS;
constructors REAL_1, XXREAL_0;
registrations XREAL_1, XREAL_0,REAL_1, ORDINAL1;
requirements SUBSET, BOOLE, NUMERALS, REAL, ARITHM;
theorems XREAL_1;
begin
reserve x, y, a for Real;
 a < 0 & x < y implies a * x > a * y
proof
```

[1] Das Projekt NAPROCHE wurde gegründet von Prof. Dr. Peter Koepke (Lehrstuhl für Mathematische Logik, Universität Bonn) und Prof. Dr. Bernhard Schröder (Lehrstuhl für Germanistik/Linguistik, Universität Duisburg-Essen). Der Verfasser des Beitrags arbeitet seit SS2007 im Projekt NAPROCHE mit. Die Homepage des Projekts hat die URL: http://www.math.uni-bonn.de/people/naproche/
[2] http://www.mizar.org/
[3] http://www.mizar.org/language/syntax.xml
[4] Axiom (A.3) ist folgendes Anordnungsaxiom: „Sind $x > 0$ und $y > 0$, so folgt $xy > 0$.“ [FORST, S.14].

```
assume
A1:  a<0;
assume
A2:  x<y;
set l=-a;
set m=l*x;
set n=l*y;
    l>0 by A1, XREAL_1:60; then
    l*x<l*y by A2,XREAL_1:70; then
    -m>-n by XREAL_1:26;
hence a*x>a*y;
end;
```

Die MIZAR Schlüsselwörter entsprechen natürlich sprachlichen Wortformen, die typischerweise in einfach verfassten mathematischen Beweisen verwendet werden: assume / angenommen; implies / (so) folgt; set / setze; then / dann; usw. Der Beweis verlangt die Anwendung bereits bewiesener Sätze oder Axiome. In MIZAR geschieht dies durch Referenz auf Namen von Sätzen, die in MIZAR Artikeldateien bereits bewiesen sind (z.B. Satz XREAL_1:60 im obigen MIZAR Beweis).

(2) Damit die ATP-Komponente[5] von MIZAR insbesondere Satzreferenzen korrekt auflösen kann, müssen eine Reihe von Umgebungsinformationen (Environment) gesetzt sein, die auf Bibliotheken bereits in MIZAR definierter Begriffe, Operatoren und bewiesener Sätze referenzieren. D.h. ein menschlicher Erfasser muss umfassend die korrekten Environment Schlüsselwörter kennen, die zu seinem Beweis nötig sind. Dieses ist eine Schwierigkeit beim Verfassen von MIZAR Beweisen[6].

NAPROCHE versucht das Problem der Mehrfacherfassung aufzuheben, indem der Vorgang der maschinellen Beweisprüfung direkt mit dem Editierwerkzeug für mathematische Beweise verbunden wird. Bei Mathematikern ist das Verfassen von mathematischen Texten in einer TeX-Sprache (TeX, LaTeX, TeXmacs, …) verbreitet. TeX-Sprachen verwalten mathematische Ausdrücke in Form von Klammerausdrücken.

3. Arbeitsgebiete des Projekts NAPROCHE

Im Projekt NAPROCHE können grob drei Arbeitsgebiete skizziert werden: (1) Texttechnologie: Im Zentrum steht die maschinelle Transformation von mathematischen Beweisen, die als Texte natürlicher Sprachen maschinenlesbar in Form von Klammerungssprachen (z. B. TeX-Sprachen) vorliegen, in XML Texte, deren Grammatik möglichst geeignet mathematische Beweisstrukturen abbildet. (2) Maschinelle Semantik[7]: Hierbei wird der in (1) gewonnene Text in eine Proof Representation Structure (PRS) überführt, die auf die Diskursrepräsenta-

[5] Automated theorem proving.

[6] Als Werkzeug hierfür wurde im Rahmen einer von mir betreuten Bachelor Arbeit ein Knowledge Management System für MIZAR (KMS-MIZAR) entwickelt [CHOU]. Das KMS-MIZAR besteht aus einem Datenbanksystem (DBS) zur Verwaltung von MIZAR Definitionen, Theoremen, Notationen und Registrierungen. Diese Entitäten wurden mit einer Akquisitionskomponente aus den MIZAR Abstractfiles extrahiert und in das DBS, das mit einem Oracle™ DBMS und Java/JDBC implementiert ist, eingefügt. Die Extraktion operiert mit vollständiger Erkennungsrate, da sie auf Grundlage der MIZAR BNF-Grammatik entwickelt wurde. Das DBS umfasst im Moment folgendes Mengengerüst:

Entitäten	**Anzahl**
Absfiles	1019
Definitions	11175
Theorems	47064
Notations	545
Registrations	7882

[7] Arbeitsgruppe von Prof. Dr. Schröder.

tionstheorie aufbaut ([KAMP], [SiKl], [FISS], [KOLE]). (3) Mathematische Logik[8]: Der im PRS Format vorliegende Beweis wird in ein FOL Format (First Order Logic) transformiert (z.B. TPTP[9]) [KÜHL]. Mit einem FOL Beweiser erhält der Anwender eine Rückmeldung, ob der Beweis korrekt ist oder in welchem Beweisschritt ein Fehler auftritt. Im folgenden wird auf Arbeiten im Arbeitsgebiet (1) eingegangen.

4. Grundlegende Formate, verwandte Arbeiten

MathML [MaML] wird als plattformunabhängiges Austauschformat für mathematische Texte eingesetzt. MathML beruht auf XML und stellt eine formale Semantik für mathematische Symbole, Formeln und Ausdrücke bereit. Zur Semantik von Beweisen wurde von [SrKo] eine auf XML beruhende Annotationssprache entwickelt (ProofML). ProofML wurde unter Berücksichtigung von OMDoc Konzepten [OMDo] weiterentwickelt [FISS]. Wichtige Ergebnisse zur Fachsprachenforschung auf dem Gebiet der Mathematik sind in [BECK] und [EISE] zusammengestellt.

5. Natürliche Sprache eines mathematischen Beweises als Transformationsgegenstand

Natürliche Sprache kommt in einem mathematischen Beweis vor und hat einige zentrale Aufgaben, wie den Fluss der mathematischen Argumente für potentielle Leser zu strukturieren, logische Figuren hervorzuheben und bestimmte Formelanteile zu erläutern. Das nachfolgende Beispiel ist der Beweis eines Satzes[10] aus der elementaren Differential- und Integralrechnung, dass sich zwei Stammfunktionen[11] einer gegebenen reellen Funktion nur um eine Konstante c unterscheiden:

Satz 2:
Voraussetzung: $F : I \to \Re$ ist eine Stammfunktion von $f : I \to \Re$.
Behauptung: $G: I \to \Re$ ist genau dann eine Stammfunktion von $f : I \to \Re$, wenn F – G eine Konstante ist.
Beweis[12]**: „**a) [$\Leftarrow$] Sei $F - G = c$ mit einer Konstanten $c \in \Re$. Dann ist $G' = (F - c)' = F' = f$. [D.h. G ist damit Stammfunktion von f.]
b) [$\Rightarrow$] Sei G Stammfunktion von f, also $G' = f = F'$. Dann gilt $(F - G)' = 0$, daher ist F – G konstant. (§16, Corollar3 [zum Satz von Rolle]).“

Betrachtet man die sprachlichen Elemente dieses Satzes und seines Beweises, dann lassen sich diese in drei Bereiche einteilen: **A) Allgemeinsprachliche Wörter** der deutschen oder einer anderen natürlichen Sprache. Z.B.: also, dann, daher, genau, gilt, ist, sei, wenn. **B) Mathematische Begriffe**, die als fachsprachlich bestimmte Wörter einer natürlichen Sprache (z.B. der deutschen Sprache) vorliegen und eindeutig in einen entsprechenden Terminus einer anderen natürlichen Sprache (z.B. Englisch) übersetzt werden können. Z.B.: „Menge“ / „set“ , „Stammfunktion“ / „antiderivative“. **C) Mathematische Symbole**, **Ausdrücke** und **Formeln**. Z. B.: G‘ als Symbol der 1. Ableitungsfunktion der Funktion G. Die Bereiche A) und B) repräsentieren zusammengenommen den natürlich sprachlichen Anteil des Beweises. In dem obigen Beispiel können folgende Leistungsmerkmale der natürlichen Sprache in der Formulierung des Satzes und seines Beweises beobachtet werden: (1) **Erläuterung von Formeldaten**: Mit dem Wort „sei“ wird typischerweise die Setzung eines Symbols

[8] Arbeitsgruppe von Prof. Dr. Koepke.
[9] http://www.tptp.org/
[10] Dieser Satz wird in diesem Beitrag aus Abkürzungsgründen durchgängig als „Satz 2“ zitiert.
[11] Mit I wird ein Intervall in der Menge der reellen Zahlen $\Re$ bezeichnet. Es gilt $I \subseteq \Re$. Eine diffenenzierbare Funktion $F : I \to \Re$ heisst **Stammfunktion** einer Funktion $f : I \to \Re$, wenn für alle $x \in I$ gilt: $F'(x) = f(x)$ (hierbei ist $F'(x)$ die erste Ableitung von $F(x)$ an der Stelle x). Beispiel: Die Funktion $F(x) = x^3$ ist Stammfunktion der Funktion $f(x) = 3*x^2$.
[12] Der Beweis ist aus [FORST], S.140. Textergänzungen im Beweis sind in [...] gesetzt.

eingeleitet. Mit der Setzung „Sei F – G = c“ wird eine reelle Konstante c als Differenz von F und G deklariert. Die Rechtfertigung für diese Setzung folgt aus der Beweisrichtungsvoraussetzung, die besagt, das F – G konstant ist. (2) **Strukturierung der Argumentationsfolge**: Mit Formulierungen, wie „dann gilt“, „dann ist“ oder „man sieht“ können mehrere Einzelargumente pauschal zusammengefasst werden und als ein Schluss dargestellt werden. (3) **Hervorhebung logischer Figuren**: Die Verknüpfung „genau dann A, wenn B“ beschreibt eine Äquivalenz der Aussagen A und B, die auch durch die Formel A $\Leftrightarrow$ B ausgedrückt werden kann.

Eine Aufgabe des NAPROCHE-Projekts besteht in der computerlinguistischen Untersuchung der natürlich sprachlichen Bestandteile mathematischer Beweise mit dem Ziel der maschinellen Klassifikation der erläuternden, der strukturierenden oder der logischen Funktion kurzer Wortfolgen in mathematischen Beweisen. Dieses Ziel ist Bestimmungsaspekt für die Gestaltung der Grammatik (DTD) des XML Zielformats der Transformation von TeX-Sprachen.

Der Umfang des allgemein sprachlichen Wortschatzes in mathematischen Lehrbüchern ist in der Regel klein. Das Lehrbuch „Grundlagen der Analysis“[13] von Edmund Landau [LAND] benötigt zur Formulierung von 301 mathematischen Sätzen mit zugehörigen Beweisen ein deutschsprachiges Vokabular mit ca. 580 Einträgen. Die Mehrzahl der Schlagwörter sind Wortformen der deutschen Allgemeinsprache. Ein weiteres Ziel der Texttechnologie im Rahmen des NAPROCHE Projekts ist die Entwicklung von Hilfsmitteln zur Trennung des allgemeinsprachlichen Anteils vom fachsprachlichen Anteil in mathematischen Beweisen. Zu diesem Zweck wird ein Datenbanksystem (DBS), das auf Landaus Deutsch-Englischem Glossar aufbaut, entwickelt.

6. Transformation eines maschinenlesbaren mathematischen Beweises in ein XML Format für natürlich sprachliche Beweise

In NAPROCHE wurde ein XML-Format definiert, das folgende Bedingungen erfüllen soll: (a) Es soll an die Gliederung von Standardtypen mathematischer Beweise (z.B. direkter Beweis, Widerspruchsbeweis, vollständige Induktion) angepasst sein. Es soll auf weitere Beweistypen erweiterbar sein (z.B. transfinite Induktion). (b) Es soll bezüglich mathematischer Formelanteile mit MathML übereinstimmen. (c) Es soll hinsichtlich PRS weiterverarbeitbar sein. Die Grammatik dieses XML-Formats kann mittels einer Dokumenttyp Definition (DTD) beschrieben werden (Satz.DTD)[14]. Die Hauptelemente dieser DTD werden nachfolgend erläutert:

(1) Das XML Wurzelelement ist <Satz>. Ein <Satz> kann mehrere <Voraussetzungen> beinhalten. Er hat eine Behauptung (<Beh>) und einen Beweis (<Bew>). Haupttestgegenstand der Entwicklung des Konverters waren Beweise, die innerhalb der Beweisschritte auf externe Hilfssätze bzw. Lemmata verwiesen. Für den Fall, dass der Beweis den Beweis integrierter Lemmata enthält, wurde für bestimmte Anwendungsfälle eine gesonderte DTD definiert (Bonn.DTD).

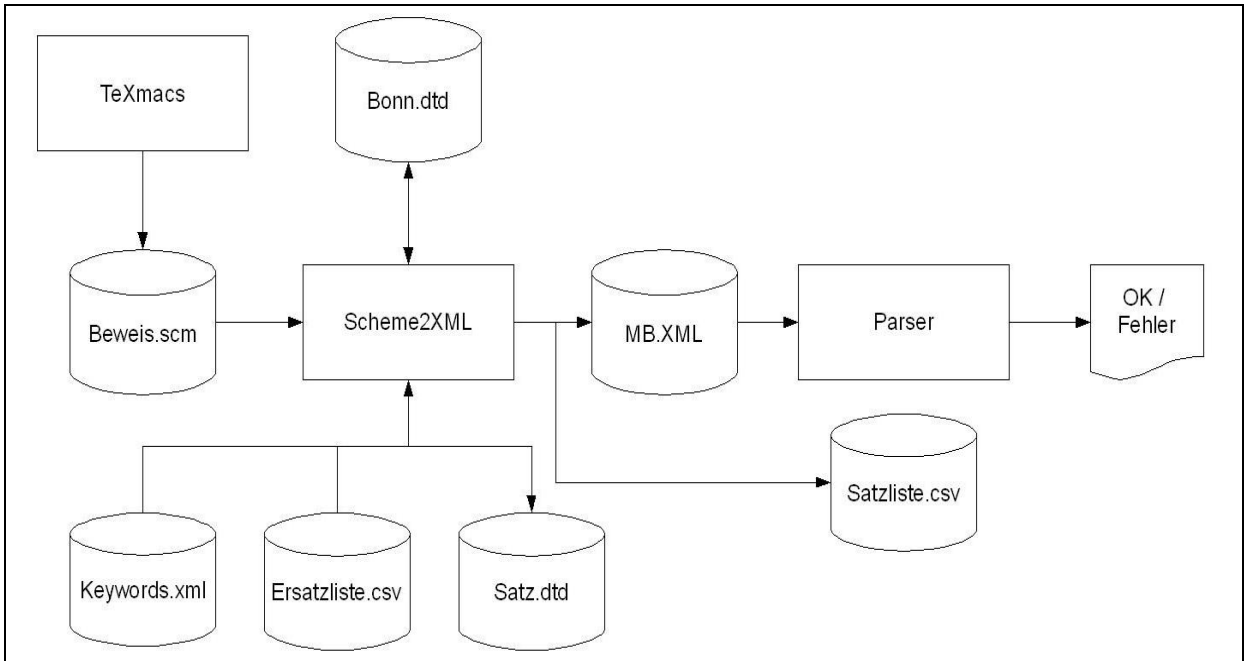
(2) Der Beweis folgt in seinem Aufbau einem Beweismodus (<BewModus>). Der Modus des Beweises von Satz 2 ist direkt (<DirBew>). Weitere Modi sind <IndBew> für Widerspruchsbeweise und <VollstIndukt> für die vollständige Induktion. Die Liste der Modi ist erweiterbar. Durch den Beweismodus wird gesteuert, ob der Beweis eine gegenüber dem direkten Beweis abweichende Substruktur hat, z.B. die Invertierung (<Invertierung>)

[13] Das Buch enthält ein nahezu vollständiges Deutsch-Englisches Vokabular, das anglophonen Studierenden ohne Deutschkenntnisse das leichte Erlernen der deutschen Sprache in mathematischen Texten ermöglichen soll.
[14] Diese DTD ist im Anhang dokumentiert.

(Annahme des logischen Gegenteils) beim indirekten Bewies oder die Induktionsverankerung bei der vollständigen Induktion.

(3) Jeder Beweis besteht aus einer Folge von Argumenten (<Arg>). Ein Argument besteht aus mindestens einer <Statementfolge>. Eine Statementfolge ist eine Kombination aus natürlich sprachlichem Text (<TEXT>) und mathematischen Formelanteilen. Die Formelanteile können einzelne mathematischen Symbole(<MATH>) oder aus mehreren Zeichen aufgebaute mathematischen Formeln sein (<MATHmodus>, <MATHreihe>, <MATHreihen>)[15]. In dem mit <TEXT> markierten XML-Element befindet sich der Hauptteil des in natürlicher Sprache verfassten Anteils der Formulierung eines Beweisschritts. Innerhalb der mathematischen Formeln (z.B. <MATHmodus> oder <MATHreihen>) können einzelne Textzeichen (z.B. „f“ als Funktionssymbol oder „a“, „x“ etc. als Formelbuchstaben) oder Wörter (z.B. „aus“, „Länge“, „in“ usw.) bzw. Wortfolgen auftreten. Diese Textelemente einer Formel werden durch das XML Element <FTEXT> verwaltet. Textsequenzen in <FTEXT> können ihrerseits neben Wörtern oder Buchstaben auch wiederum zusammengesetzte mathematische Symbole (z.B. Brüche <BRUCH>) oder einfache mathematische Symbole (z.B. das Symbol für die Menge der natürlichen Zahlen <bbb-N/>) enthalten.

(4) Mathematische Formeln (<MATHmodus>, <MATHreihe>, <MATHreihen>) können mit beliebiger Verschachtelungstiefe auftreten. Um dieses in XML rekursiv verwalten zu können wurde die ENTITY %Mathe eingeführt. Bei den XML Elementen für mathematische Symbole wurde sich an den Elementen von MathML orientiert. Z.B. wird das Symbol <bbb-R/> für das mathematische Symbol der Menge der reellen Zahlen verwendet. Enthält eine mathematische Formel eine Gleichung oder Ungleichung, wird dieses im Element <Statementtyp> notiert. Weiterhin wird dort festgehalten, ob innerhalb des Beweisschritts ein prädikatenlogischer Ausdruck auftritt. Tritt dabei ein Ausdruck mit Quantor auf, ist dieser als XML Element in der Formel ausgewiesen (Allquantor: <forall/> bzw. Existenzquantor: <exists/>).



Systemübersichtsdiagramm Scheme2XML

[15] Die Tatsache, dass es hier drei Tags zur Beschreibung mathematischer Formeln gibt, ist eine Folge unterschiedlicher Editieroptionen für komplexe Formeln in TeXmacs.

Auf Grundlage dieser DTD wurde ein Code-Transformator (Scheme2XML) spezifiziert. Hierbei wird von einem maschinenlesbaren Beweis in einem TeX Format ausgegangen. In Bezug auf das Referenzwerkzeug von NAPROCHE wurde als Eingangsformat der TeXmacs Scheme Code ausgewählt (Scheme2XML := TeXmacs Scheme Code to XML Converter). Eine erste Version des Code-Transformators wurde bereits implementiert[16]. Die Validitäts-kontrolle des Transformators wurde bisher auf einem Korpus von ca. 20 Beweisen aus der Algebra und der Analysis ausgeführt[17]. Zur Unterstützung der Beweistextsegmentierung wurde eine Schlüsselwortdatei verwendet (Keywords.xml). Hierin sind Wörter und Symbole notiert, die typischerweise Anfangs- und Endpositionen von Beweistextteilen markieren: Z.B. Beweisendesymbole: `qed.; q.e.d.; Qed.; Q.e.d.; qed;`

7. Aufbau eines DBS zu Landaus Deutsch-Englischem Glossar

Das Lehrbuch „Grundlagen der Analysis“ von Edmund Landau [LAND] enthält ein Deutsch-Englisches Glossar, das 301 mathematischen Sätzen mit zugehörigen Beweisen beinhaltet, abdeckt. Die Mehrzahl der Schlagwörter des Glossars sind deutsche Wortformen. Daneben treten als Schlagwörter auch Präfixe und Suffixe auf. Beispiele für Glossareinträge sind folgende:

aufzählen, (auf+zählen) to enumerate, to count, to list.
dann, then; **_und nur_**, if and only if.
ge-, prefix forming past participles of verbs.
heben, to lift (cog., heave).
lernend, (present participle of: lernen) learning.

Das Design der Datenbank basiert auf der syntaktischen Analyse der Glossar-Einträge in Form der Data Dictionary Notation (DDN) der Strukturierten Analyse [BALZ]. Nachfolgend ist ein Auszug aus den DDN-Produktionsregeln zu den Glossar-Einträgen gegeben:

LGEVART:=SW+(SYN_ANG)+(MORPH_ANG)+TR_LIST
TR_LIST:=TR_LIST1|TR_LIST2|…
TR_LIST1:={ENGLW+(ETYM_ANG)}
TR_LIST2:=TR_LIST1+{SUB_SW+TR_LIST1}
SW := catchword;
SYN_ANG := syntactic information;
MORPH_ANG := morphological information;
TR_LIST: list of translation;
ENGLW := English word;
ETYM_ANG := etymological information;
SUB_SW := “sub catchword”.

Das Datenbanksystem zu Landaus Glossar ist in erster Stufe erstellt. Es enthält den deutschen Schlagwortbestand (SW) mit Attributen, die Verweise auflösen (SUB_SW), Abkürzungen kennzeichnen, Schlagwörter als Partizipien ausweisen (SYN_ANG), die gfs. Angaben zum Schlagwort als Kompositum (MORPH_ANG) bzw. zur Wortherkunft machen (MORPH_ANG) und Präfixe bzw. Suffixe kennzeichnen. Weiterhin sind Tabellen zur englischen Übersetzung des deutschen Schlagworts aufgebaut, sowohl zur Verwaltung der englischen Zielphrasen als auch zur Verwaltung der Mehrdeutigkeit.

In der nächsten Ausbaustufe soll sowohl beim deutschen Schlagwortbestand als auch bei den englischen Phrasen in bestimmten Fällen notiert werden, ob es sich um Repräsentanten allgemeinsprachlicher Wörter oder um mathematische Fachwörter handelt. Bei den

[16] Sebastian Zittermann: „Entwicklung eines TeXmacs-to-XML-Parsers“, Bachelorarbeit, FH Köln, Institut für Nachrichtentechnik, September 2008.
[17] Ein Beispiel eines erzeugten XML Beweiscodes ist im Anhang gegeben.

allgemeinsprachlichen Wörtern soll in bestimmten Fällen markiert werden, ob sie logische Funktionen haben, Setzungen oder Annahmen (generell Beweisstrukturaspekte) markieren.

In der nächsten Scheme2XML-Version soll dieses DBS-Modul dann u. a. zur Generierung von XML-Tags eingesetzt werden, die bestimmte erläuternde, strukturierende oder logische Funktionen von allgemeinsprachlichen Wortfolgen in Beweisen kennzeichnen sollen.

8. Referenzen

[BALZ] Balzert, Helmut: "Lehrbuch der Software-Technik", Heidelberg [etc.] (Spektrum) 1996.

[BECK] Becker, Holger: "Semantische und lexikalische Aspekte der mathematischen Fachsprache des 19. Jahrhunderts", (Diss.) Universität Oldenburg 2005. http://docserver.bis.uni-oldenburg.de/publikationen/dissertation/2006/becsem05/becsem05.html

[CHOU] Chouaffé, Franck Edmond: „Entwicklung eines KMS (Knowledge Management System) für die Entwicklung von Mizar-Beweisen (KMS-MIZAR)" Bachelorarbeit FH Köln, Dezember 2008.

[EISE] Eisenreich, Günther: "Die neuere Fachsprache der Mathematik seit Carl Friedrich Gauß" in: L. Hoffmann, H. Kalverkämper, H. E. Wiegand et. al. (Hg.): "Fachsprachen – ein internationales Handbuch zur Fachsprachenforschung und Terminologiewissenschaft" , 1. Halbband, Berlin, New York (de Gruyter) 1998, S. 1222-1230.

[FISS] Fisseni, Bernhard: "Die Entwicklung einer Annotationssprache für natürlichsprachlich formulierte mathematische Beweise", (Magisterarbeit) Universität Bonn 2003.

[FORST] Forster, Otto: „Analysis I – Differential- und Integralrechnung einer Veränderlichen" Braunschweig/Wiesbaden (Vieweg) 1992.

[KAMP] Kamp, Hans; Reyle, U.: "From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory", Dordrecht (Kluwer) 1993.

[KOLE] Kolev, Nickolay: "Generating Proof Representation Structures for the Project NAPROCHE" (Magisterarbeit) Universität Bonn 2008.

[KÜHL] Kühlwein, Daniel: "Ein Kalkül für Proof Representation Structures" (Diplomarbeit) Universität Bonn 2008.

[LAND] Landau, Edmund: "Grundlagen der Analysis – With a Complete German-English Vocabulary", New York (Chelsea) 1965.

[MaML] Mathematical Markup Language (MathML) Version 2.0 (Second Edition), W3C Recommendation 21 October 2003: http://www.w3.org/TR/MathML2/ .

[OMDo] Kohlhaase, Michael: "OMDoc – An open markup format for mathematical documents" LNAI Nr.4180, Berlin (Springer) 2006.

[SiKl] Schielen, Michael; Klabunde, Ralf: "3.5.4 Diskursrepräsentationstheorie" in: K.-U. Carstensen et. al. (Hg.): "Computerlinguistik und Sprachtechnologie – Eine Einführung", Heidelberg (Spektrum) 2004, S.302-313.

[SrKo] Schröder, Bernhard; Koepke, Peter: „ProofML – eine Annotationssprache für natürliche Beweise". In: LDV-Forum, Nr.1,2 2003, ISSN 0175-1336, S.428-441.

6. Anhang

Auszug aus dem vom TCM2XML generierten XML Code des Beweises von Satz 2:

```
<Beweis>
 <BewModus>DirBew</BewModus>
  <BewArgFolge>
   <DirBew>
    ...
```

```
    <Arg ID= "1">
      <Statementfolge>
       <TEXT>(a) Sei</TEXT>
       <MATHmodus><FTEXT>F - G = c</FTEXT></MATHmodus>
       <Statementtyp>0;gl</Statementtyp>
       <TEXT>mit der Konstanten</TEXT>
       <MATHmodus><FTEXT>c <epsilon/> <bbb-R/></FTEXT></MATHmodus>
       <TEXT>.</TEXT>
      </Statementfolge>
    </Arg>
    <Arg ID= "2">
      <Statementfolge>
       <TEXT> Dann ist</TEXT>
       <MATHmodus><FTEXT>G'=(F-c)'=F'=f</FTEXT></MATHmodus>
       <Statementtyp>0;gl,gl,gl</Statementtyp>
       <TEXT>.</TEXT>
      </Statementfolge>
    </Arg>
   ...
  </DirBew>
 </BewArgFolge>
</Beweis>
```

Satz.DTD, eine Grammatik, die das Zielformat der Konvertierung von TeXmacs Scheme Code nach XML bestimmt(Scheme2XML):

```
<!-- DTD  : Satz.DTD                                          -->
<!-- Zweck: Eine DTD fuer einfach strukturierte               -->
<!--        mathematische Beweise                             -->
<!-- Verf.: Gregor Buechel                                    -->
<!-- Ueberarbeitung: Sebastian Zittermann                     -->
<!-- Stand: 27.11.2008                                        -->
<!-- Version: 3.x (Neue Versionszaehlung)                     -->
<!-- ######################################################## -->
<!-- Die XML-Wurzel ist: Satz.                                -->
<!--     Ein Satz braucht eine ID.                            -->
<!--     Ein Satz hat SatzInformationen.                      -->
<!--     Ein Satz kann mehrere Voraussetzungen haben.         -->
<!--     Ein Satz hat eine Behauptung.                        -->
<!--     Ein Satz hat einen Beweis.                           -->
<!-- ######################################################## -->

<!-- Tagalternativen (Wg. dynam. Ergaenzung einzelner Tags) -->
<!ENTITY % Mathe "MATH|MATHmodus|MATHreihe|MATHreihen|FTEXT">
<!ENTITY % TagsMitInhalt
"FTEXT|MSUP|MSUB|Auflistung|KLAMMERN|BRUCH|int
|uplimit|lowlimit|Ableitung|nicht|WURZEL|REIHE|CELL">
<!ENTITY % EinzelneTags "epsilon|bbb-R|gtr|bbb-N|forall|exists|geq
|bbb-Q|OR|bbb-Z|wedge|alpha|less|rightarrow|xi|leq|varepsilon
|leqslant|subset|searrow|nearrow|assign|infty|ldots|frak-a|neq
|in|thicksim|lambda|Opt-Voraussetzungen">

<!ELEMENT SATZ
(SatzInformationen,Voraussetzungen*,Behauptung,Beweis)>
<!ATTLIST SATZ
    ID CDATA #REQUIRED
>
<!-- ######################################################## -->
<!—  SatzInformationen      :                                 -->
<!-- Der Satztitel muss enthalten sein.                       -->
<!-- Das mathematische Teilgebeit kann angegeben werden.      -->
<!-- Der Autor kann angegeben werden.                         -->
<!-- Es koennen mehrere Bemerkungen hinterlegt werden.        -->
```

```
<!-- ######################################################    -->
<!ELEMENT SatzInformationen
(Satztitel,Mathemteilgebiet?,Quelle?,Bemerkung*)>
<!ELEMENT Satztitel (TEXT|%Mathe;)*>
<!ELEMENT Mathemteilgebiet (TEXT|%Mathe;)*>
<!ELEMENT Quelle (TEXT|%Mathe;)*>
<!ELEMENT Bemerkung (TEXT|%Mathe;)*>

<!-- ###################################################### -->
<!-- Voraussetzungen:                                       -->
<!-- Eine Voraussetzung besteht aus einer einzelnen         -->
<!-- Voraussetzung (Vor).                                   -->
<!-- Einer einzelnen Voraussetzung wird                     -->
<!-- eine ID zugeordnet.                                    -->
<!-- ###################################################### -->
<!ELEMENT Voraussetzungen (Vor+)>
<!ELEMENT Vor (Statementfolge)>
<!ATTLIST Vor
    ID CDATA #REQUIRED
>

<!-- ###################################################### -->
<!-- Behauptung:                                            -->
<!-- Behauptung besteht aus einer Statementfolge.           -->
<!-- ###################################################### -->
<!ELEMENT Behauptung (Statementfolge)>

<!-- ###################################################### -->
<!-- Beweis:                                                -->
<!—  Drei verschiedene Beweismodi sind derzeit spezifi-  -->
<!—  ziert. Jeder Beweismodus hat seine eigene           -->
<!—  BewArgFolge.                                        -->
<!-- ###################################################### -->
<!ELEMENT Beweis (BewModus,BewArgFolge)>
<!ELEMENT BewModus (#PCDATA)>
<!ELEMENT BewArgFolge (VollstIndukt|IndBew|DirBew)>

<!-- ###################################################### -->
<!-- direkter Beweis:                                       -->
<!-- ###################################################### -->
<!ELEMENT DirBew (Arg+)>
<!ELEMENT Arg (Statementfolge+)>
<!ATTLIST Arg
    ID CDATA #REQUIRED
>

<!-- ###################################################### -->
<!-- indirekter Beweis:                                     -->
<!-- ###################################################### -->
<!ELEMENT IndBew
(Statementfolge?,Invertierung,InvertBew,Widerspruch)>
<!-- Entspricht dem Aufbau der Behauptung                -->
<!ELEMENT Invertierung (Statementfolge)>
<!-- Entspricht dem Aufbau des Beweises                  -->
<!ELEMENT InvertBew (Arg*)>
<!ELEMENT Widerspruch (Statementfolge)>

<!-- ###################################################### -->
<!-- vollst.Induktion:                                      -->
<!-- ###################################################### -->
<!ELEMENT VollstIndukt (Induktionsverankerung,Induktionsschluss+)>
<!ELEMENT Induktionsverankerung (Statementfolge,Statementtyp?)>
```

```
<!ELEMENT Induktionsschluss
(Statementfolge?,Induktionsannahme,Induktionsbeh,Induktionsbeweis)>
<!-- Induktionsannahme entspricht Voraussetzungen        -->
<!ELEMENT Induktionsannahme (Statementfolge+)>
<!ELEMENT Induktionsbeh (Statementfolge+)>
<!ELEMENT Induktionsbeweis (Arg*)>

<!-- kann in beliebiger Reihenfolge mathematische Teile  -->
<!-- wie z.B. praedikatenlogische Ausdruecke enthalten   -->
<!ELEMENT Statementfolge
((TEXT|((MATH|MATHmodus|MATHreihen|MATHreihe),Statementtyp?))+)>

<!-- Ein Textbereich -->
<!ELEMENT TEXT (#PCDATA)>

<!-- Statementtyp: Aufbauf wie CSV -->
<!-- erste Stelle (0 oder 1) gibt an, ob es eine praedikatenlogische
Aussage beinhaltet -->
<!-- hinter Semikolon steht, ob es sich um eine Gleichung (gl) oder
Ungleichungen (ungl) handelt -->
<!ELEMENT Statementtyp (#PCDATA)>

<!-- "Inhalt" von den drei "MATH"s wird dynamisch anhand der
Fehlerexceptions bestimmt-->
<!ELEMENT MATH (%Mathe;|%TagsMitInhalt;|%EinzelneTags;)*>
<!ELEMENT MATHmodus (%Mathe;|%TagsMitInhalt;|%EinzelneTags;)*>
<!ELEMENT MATHreihe (%Mathe;|%TagsMitInhalt;|%EinzelneTags;)*>
<!ELEMENT MATHreihen (%Mathe;|%TagsMitInhalt;|%EinzelneTags;)*>
<!ELEMENT FTEXT (#PCDATA|%TagsMitInhalt;|%EinzelneTags;)*>

<!-- "verkettete" Elemente -->
<!ELEMENT MSUP (mi+)>
<!ELEMENT MSUB (mi+)>
<!ELEMENT mi (#PCDATA|%TagsMitInhalt;|%EinzelneTags;)*>

<!ELEMENT Auflistung (Item+)>
<!ELEMENT Item (#PCDATA|%TagsMitInhalt;|%EinzelneTags;)*>
<!ATTLIST Item
    ID CDATA #REQUIRED
>
<!ELEMENT KLAMMERN (#PCDATA|%TagsMitInhalt;|%EinzelneTags;)*>
<!ATTLIST KLAMMERN
    typ CDATA #REQUIRED
>
<!ELEMENT KlammerInd (#PCDATA|%TagsMitInhalt;|%EinzelneTags;)*>
<!ELEMENT KlammerExp (#PCDATA|%TagsMitInhalt;|%EinzelneTags;)*>

<!ELEMENT BRUCH (Zaehler,Nenner)>
<!ELEMENT Zaehler (#PCDATA|%TagsMitInhalt;|%EinzelneTags;)*>
<!ELEMENT Nenner (#PCDATA|%TagsMitInhalt;|%EinzelneTags;)*>

<!ELEMENT lowlimit (#PCDATA|%TagsMitInhalt;|%EinzelneTags;)*>
<!ELEMENT uplimit (#PCDATA|%TagsMitInhalt;|%EinzelneTags;)*>

<!-- einzelne Tags mit Inhalt-->
<!ELEMENT Ableitung (#PCDATA|%TagsMitInhalt;|%EinzelneTags;)*>
<!ELEMENT nicht (#PCDATA|%TagsMitInhalt;|%EinzelneTags;)*>
<!ELEMENT WURZEL (#PCDATA|%TagsMitInhalt;|%EinzelneTags;)*>
<!ELEMENT REIHE (#PCDATA|%TagsMitInhalt;|%EinzelneTags;)*>
<!ELEMENT CELL (#PCDATA|%TagsMitInhalt;|%EinzelneTags;)*>

<!-- einzelne Zeichen -->
```

```
<!-- element von -->
<!ELEMENT epsilon (#PCDATA)>
<!-- Menge reelle Zahlen -->
<!ELEMENT bbb-R (#PCDATA)>
<!-- groesser als (>) -->
<!ELEMENT gtr (#PCDATA)>
<!—- Menge der natuerlichen Zahlen -->
<!ELEMENT bbb-N (#PCDATA)>
<!-- "Fuer Alle"-Symbol -->
<!ELEMENT forall (#PCDATA)>
<!-- Existiert-Symbol -->
<!ELEMENT exists (#PCDATA)>
<!-- groesser-gleich -->
<!ELEMENT geq (#PCDATA)>
<!-- Menge der rationalen Zahlen -->
<!ELEMENT bbb-Q (#PCDATA)>
<!-- oder-Symbol  -->
<!ELEMENT OR (#PCDATA)>
<!-- Menge der ganzen Zahlen -->
<!ELEMENT bbb-Z (#PCDATA)>
<!-- /\ -Symbol (und) -->
<!ELEMENT wedge (#PCDATA)>
<!-- Alpha -->
<!ELEMENT alpha (#PCDATA)>
<!-- kleiner als -->
<!ELEMENT less (#PCDATA)>
<!-- Pfeil nach rechts -->
<!ELEMENT rightarrow (#PCDATA)>
<!-- Symbol griech. Buchstabe XSI -->
<!ELEMENT xi (#PCDATA)>
<!-- kleiner gleich -->
<!ELEMENT leq (#PCDATA)>
<!-- Symbol Epsilon -->
<!ELEMENT varepsilon (#PCDATA)>
<!-- kleiner gleich (schraeges gleich) -->
<!ELEMENT leqslant (#PCDATA)>
<!—- Teilmenge von (wie C) -->
<!ELEMENT subset (#PCDATA)>
<!-- Pfeil nach unten-rechts (Southeast) -->
<!ELEMENT searrow (#PCDATA)>
<!-- Pfeil nach oben-rechts (Northeast) -->
<!ELEMENT nearrow (#PCDATA)>
<!-- ":=" -->
<!ELEMENT assign (#PCDATA)>
<!-- Unendlichkeits-Symbol -->
<!ELEMENT infinity (#PCDATA)>
<!-- 3 Punkte (...) -->
<!ELEMENT ldots (#PCDATA)>
<!-- altdeutsches a -->
<!ELEMENT frak-a (#PCDATA)>
<!-- unlgeich (durchgestrichenes =) -->
<!ELEMENT neq (#PCDATA)>
<!-- epsilon fuer "ist Element von" -->
<!ELEMENT in (#PCDATA)>
<!-- Integralzeichen -->
<!ELEMENT int (#PCDATA)>
<!-- Tilde (~) -->
<!ELEMENT thicksim (#PCDATA)>
<!-- Lambda-Zeichen -->
<!ELEMENT lambda (#PCDATA)>
<!ELEMENT Opt-Voraussetzungen (#PCDATA)>
```

Werkzeuge zur Annotation diachroner Textkorpora

Manuel Burghardt, Christian Wolff
Institut für Information und Medien, Sprache und Kultur
Universität Regensburg
93040 Regensburg
{manuel.burghardt, christian.wolff}@sprachlit.uni-regensburg.de

Keywords: Annotationswerkzeuge, Evaluation, diachrone Korpora, historische Sprachwissenschaft, Benutzerfreundlichkeit, Funktionalität, ISO 9126, ISO 25000

Abstract

We discuss the problem of annotating syntax in diachronic corpora. A study analysing functionality as well as usability characteristics of more than 50 annotation tools is presented. For this study, we have developed a quality model based on international standards ISO/IEC 9126-1:2001 (*Software engineering – Product quality – Part 1: Quality model*) and ISO/IEC 25000:2005 (*Software Engineering – Software product Quality Requirements and Evaluation (SQuaRE) – Guide to SQuaRE*).

Zusammenfassung

Wir diskutieren zunächst die Problematik der (syntaktischen) Annotation diachroner Korpora und stellen anschließend eine Evaluationsstudie vor, bei der mehr als 50 Annotationswerkzeuge und -frameworks vor dem Hintergrund eines funktionalen und softwareergonomischen Anforderungsprofils nach dem Qualitätsmodell von ISO/IEC 9126-1:2001 (*Software engineering – Product quality – Part 1: Quality model*) und ISO/IEC 25000:2005 (*Software Engineering – Software product Quality Requirements and Evaluation (SQuaRE) – Guide to SQuaRE*) evaluiert wurden.

1 Diachrone Korpora als Herausforderung der Texttechnologie

Obwohl die Korpuslinguistik mittlerweile zum Standardrepertoire sprachwissenschaftlicher Methodik gehört (Tognini-Bonelli 2001), stellen diachrone Korpusanalysen immer noch eine Herausforderung für die menschlichen Annotatoren dar: In der Regel liegt mit steigendem Alter der Sprachentwicklungsstufe nur wenig Datenmaterial vor und Ergebnisse (sowie Analyseverfahren) synchroner Sprachbetrachtung lassen sich nicht ohne Weiteres auf ältere Sprachstufen übertragen, da bei diachronen Ansätzen in

erster Linie Wandelprozesse auf den verschiedenen sprachlichen Beschreibungsebenen (z.B. Morphologie, Phonetik, Syntax oder Semantik) entlang einer Zeitachse untersucht werden. Die explizite Annotation bestimmter sprachlicher Parameter soll dabei Entwicklungstendenzen aufdecken helfen. Problematisch sind hierbei vor allem orthografische und syntaktische Ambiguitäten, welche eine eindeutige Annotation und spätere Auswertung der Daten erschweren. Doch gerade die explizite Annotation einer Mehrzahl von Lesarten kann dabei helfen, Sprachwandelprozesse zu beschreiben. Zudem sind diachrone Annotationen zwangsläufig durch ein hohes Maß an Diskontinuität gekennzeichnet (Kroymann et al. 2004), da Texte unterschiedlicher Sprachstufen oftmals nicht direkt miteinander vergleichbar sind. Der Grund hierfür liegt in der unterschiedlich starken Ausprägung bestimmter sprachlicher Parameter wie etwa Textgattung oder syntaktische Funktion. Das Aufzeigen solcher diskontinuierlichen Entwicklungen liefert neben der Annotation von Ambiguitäten wichtige Erkenntnisse über diachrone Sprachwandelprozesse. Um diese komplexen und kontextuell schwer zu bewertenden sprachlichen Phänomene korrekt annotieren zu können, ist bei der Erstellung diachroner Korpora immer noch ein hohes Maß an manueller Arbeit nötig. Aufgabe eines geeigneten Annotationswerkzeugs muss es somit sein, den Menschen bei seiner intelligenten und kreativen Arbeit so gut wie möglich zu unterstützen und ihn auf lange Sicht zu entlasten. Dabei sollte ein Annotationswerkzeug einerseits funktional sein, und dem Benutzer immer wiederkehrende Routineaufgaben durch Automatisierungsmechanismen abnehmen, andererseits muss ein entsprechendes Tool Anforderungen an Benutzbarkeit und Softwareergonomie erfüllen. Inwiefern die derzeit verfügbaren Werkzeuge den Anforderungen einer solchen „Annotationsergonomie“ gerecht werden, soll im Folgenden untersucht werden.

2 Auswahlkriterien Werkzeuge zur Annotation diachroner Korpora

Annotation wurde bereits lange vor Anbruch des Computerzeitalters betrieben und ist seit jeher ein wichtiges Instrument um Wissen zu akkumulieren, es zu verwalten und anderen besser zugänglich zu machen. Seitdem haben sich im Bereich der Annotationspraxis zahlreiche computergestützte Techniken und Vorgehensweisen etabliert, die den menschlichen Annotator bei seiner anspruchsvollen Aufgabe unterstützen. Vor diesem Hintergrund hat sich ein breites Spektrum an Annotationswerkzeugen etabliert, das ebenso heterogen und vielschichtig ist wie die Zahl denkbarer Annotationsszenarien selbst (Ide & Brew 2000). In unserer Studie haben wir durch Systematisierung bestehender Tools sowie durch Berücksichtigung der Anforderungen, die sich aus einem diachronen Annotationsszenario ergeben, geeignete Werkzeuge für diachrone Korpora identifiziert und evaluiert.

Ein zweistufiger Selektionsprozess dient dabei der Reduktion von mehr als 50 nachweisbaren Werkzeuge und Frameworks auf eine handhabbare Menge: Die wesentlichen Kriterien für den ersten Schritt der Systematisierung der über 50 Annotationswerkzeuge sind dabei „Annotationsmodalität" und „Softwaretyp" (Wolff 2004). Obwohl die Verfügbarkeit von digitalisierten Sprach- und Videodaten in den letzten Jahren stark zugenommen hat (Dybkjær et al. 2001), sind solche multimedialen Datensätze für fundierte diachrone Untersuchungen, die sich meist auf einer mehrere Jahrhunderte umfassenden Zeitachse bewegen, (noch) nicht von allzu großer Relevanz. Tools, die für diachrone Korpusanalysen eingesetzt werden sollen, müssen in jedem Fall die Annotationsmodalität *Text* unterstützen. Beim *Softwaretyp* sind „fertig implementierte" und sofort einsetzbare Programme mit grafischer Oberfläche (i. d. R. *monolithische* Softwaresysteme bzw. „rich clients") abstrakten Klassenbibliotheken oder komplexen Frameworks vorzuziehen, da der Annotator möglichst effizient bei seiner Aufgabe unterstützt werden soll, in den meisten Annotationsszenarien jedoch weder die Zeit noch das technische *know how* zur aufwendigen Konfiguration oder Erstimplementierung eines Werkzeugs vorhanden ist (Dipper et al. 2004). Tabelle 1 gibt eine Übersicht der in der Studie untersuchten Annotationswerkzeuge:

Toolname	**Softwaretyp**
ACE Annotation Toolkit	Annotationswerkzeug (basiert auf dem AGTK)
ACT	Annotationswerkzeug
AGTK	Framework
Alembic Workbench	Framework und Annotationswerkzeug
Annotate	Annotationswerkzeug
Anvil	Annotationswerkzeug
Arboreal	Annotationswerkzeug, XML-Browser
ATLAS	Framework
CAVA	Annotationswerkzeuge
Callisto	Annotationswerkzeug (basiert auf jATLAS, Nachfolger der Alemic Workbench)
CBAS	Annotationswerkzeug
CLAN	Annotationswerkzeug für Texte eines be-stimmten Formats (CHILDES), Analysetool

Toolname	**Softwaretyp**
CLaRK	Annotationswerkzeug, Lexikonerstellung
CSLU Toolkit	Framework, Annotationswerkzeug, Analyse-tool, TTS, Sprachtrainer
DAT	Annotationswerkzeug (benutzt das DAMSL[11] Schema)
Dexter	Annotationswerkzeug
DitAT	Annotationswerkzeug
ELAN	Annotationswerkzeug
EUDICO	Framework, Workbench (Integration in GATE geplant)
EXMARaLDA	Annotationswerkzeug, Korpusmanager, Analysetool
FLEX	Annotationswerkzeug für Feldforschung, Lexikonerstellung
GATE	Framework und Annotationswerkzeug

Toolname	Softwaretyp
Interact	Annotationswerkzeug
ITE	Annotationswerkzeug
LT XML	Framework, Greptool
MATE	Framework
MediaStreams	Ikonisches Annotationswerkzeug
MMAX	Annotationswerkzeug
MMAX 2	Annotationswerkzeug (Nachfolger von MMAX)
Multext Tools	Annotationswerkzeug
MultiTool	Annotationswerkzeug, Analysetool
NITE (NXT)	Framework (Nachfolger von MATE)
Observer	Annotationswerkzeug
oXygen	XML Annotationswerkzeug
Palinka	Annotationswerkzeug (Nachfolger von Clinka)
Praat	Annotationswerkzeug, Analysetool, TTS
RST Tool	Annotationswerkzeug

Toolname	Softwaretyp
SignStream	Annotationswerkzeug, Analysetool
SmartKom	Framework (benutzen Anvil zur Annotation)
Snack	Framework
SyncWriter	Annotationswerkzeug
Synpathy	Annotationswerkzeug
Systemic Coder	Annotationswerkzeug, Analysetool
TASX	Framework und Annotationswerkzeug
Toolbox	Annotationswerkzeug für Feldforschung, Lexikonerstellung (Nachfolger von Shoebox)
Transcriber	Annotationswerkzeug
Transformer	Annotationswerkzeug
vPrism	Annotationswerkzeug
WaveSurfer	Annotationswerkzeug
UAM CorpusTool	Annotationswerkzeug (Nachfolger von Systemic Coder), Analysetool
Wordfreak	Annotationswerkzeug

Tabelle 1: Übersicht der in der Evaluationsstudie berücksichtigten Annotationswerkzeuge

Aus Darstellungsgründen sind in der Tabelle keine Links zu den Werkzeugen aufgeführt; diese sind aber unter http://www.disynde.de verfügbar, wo auch weiteres Dokumentationsmaterial zu den Tools vorhanden ist. Erfüllen die Annotationswerkzeuge im zweiten Schritt der Vorauswahl neben der geforderten Annotationsmodalität und einem entsprechenden Softwaretyp auch noch die Anforderungen *Verfügbarkeit und Aktualität der Applikation*, *Flexibilität der Annotationsschemata* sowie *Wiederverwendbarkeit des Annotationsformats*, so werden sie hinsichtlich ihrer Funktionalität und Benutzbarkeit ausführlich evaluiert. Nicht-funktionale Kriterien wie *Aktualität* und *Verfügbarkeit* einer Anwendung erscheinen als Auswahlkriterium gerechtfertigt, da Korpusaufbereitung typischerweise ein länger währender Prozess ist, bei dem die Verfügbarkeit und Unterstützung der Werkzeugumgebung eine große Rolle spielt. Bei der Evaluation gilt es, zahlreiche weitere Anforderungen wie etwa *Zeichensatz*, *Mehrebenenannotation*, *Flexibilität der Ein- und Ausgabe*, *Adaptierbarkeit der Software*, *Automatisierbarkeit der Software* sowie die *Koordination verteilter Arbeitsabläufe* (Workflow) durch ein adäquates Qualitätsmodell zu operationalisieren (Burghardt 2008). Insgesamt qualifizieren sich vier Werkzeuge für eine ausführliche Softwareevaluation: Das auf dem ATLAS-

Framework basierende Tool *Callisto*, das Framework *GATE*, das auch ein Annotationswerkzeug beinhaltet, sowie die von kleineren Teams entwickelten Programme *MMAX2* und das *UAMCorpusTool*, das Ansätze aus der systemisch-funktionalen Linguistik (Halliday & Martin 1981) aufgreift.

3 Aufbau eines standardisierten Qualitätsmodells

Die Qualitätsnormen zur Sicherung der Produktqualität von Software der ISO (*International Organization for Standardization*) und der IEC (*International Electrotechnical Commision*) (DIN 66272, 1994) sowie das Framework zur Evaluation von VNS-Software (*Software zur Verarbeitung Natürlicher Sprache*) der EAGLES-Evaluationsarbeitsgruppen (EAGLES 1999a, 1999b) dienen als Grundlage für ein weitestgehend standardisiertes und wieder verwendbares Evaluationsdesign. Dabei werden alle im vorangegangenen Kapitel allgemein formulierten Anforderungen in die beiden primären Qualitätskriterien der ISO 9126, *Funktionalität* und *Benutzbarkeit*, sowie deren Unterkriterien *Angemessenheit*, *Interoperabilität*, *Erlernbarkeit*, *Bedienbarkeit* und *Konformität* gegliedert und solange weiter aufgeteilt, bis messbare Attribute übrig bleiben. Das Ergebnis ist ein hierarchisches Qualitätsmodell, auf dessen höchster Ebene das Annotationswerkzeug als Ganzes steht und auf dessen unterster Ebene sich ein Katalog aus messbaren Attributen befindet.

<table>
<tr><td colspan="5">Softwareprodukt
(operationalisiert durch zwei Qualitätskriterien
mit insgesamt fünf Unterkriterien und 30 Attributen)</td></tr>
<tr><td colspan="2">Funktionalität
(operationalisiert durch zwei Unterkriterien
mit insgesamt 13 Attributen)</td><td colspan="3">Benutzbarkeit
(operationalisiert durch drei Unterkriterien
mit insgesamt 17 Attributen)</td></tr>
<tr><td>Angemessenheit</td><td>Interoperabilität</td><td>Erlernbarkeit</td><td>Bedienbarkeit</td><td>Konformität</td></tr>
<tr><td>9 Attribute</td><td>4 Attribute</td><td>5 Attribute</td><td>8 Attribute</td><td>4 Attribute</td></tr>
</table>

Tabelle 2: Hierarchisches Qualitätsmodell nach ISO 9126

Die Funktionalität von Annotationswerkzeugen beschreibt, in welchem Maße Funktionen zur Erfüllung einer bestimmten Aufgabe durch einen Benutzer vorhanden sind. Damit beschreibt Funktionalität das *Verhältnis zwischen Werkzeug und Aufgabe*, während Benutzbarkeit das *Verhältnis zwischen Werkzeug und Benutzer* thematisiert. Anders als beim Kriterium der Funktionalität steht bei der Benutzbarkeit der Anwender mit seinen individuellen Bedürfnissen an Interaktionsverhalten und Visualisierung der Software im Vordergrund. Ein hoher Grad an Benutzbarkeit impliziert immer auch einen möglichst geringen Aufwand zum Erlernen und zur Bedienung der Software. Den Zusammenhang von Aufgabe, Werkzeug und Benutzer veranschaulicht Abbildung 1:

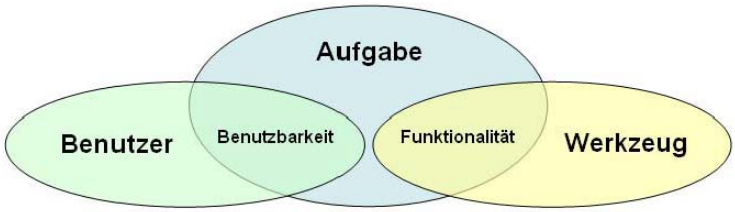


Abbildung 1: Benutzbarkeit und Funktionalität im Verhältnis zur Aufgabe

Funktionalität (I): Angemessenheit: Die Angemessenheit von Annotationswerkzeugen beschreibt deren funktionelle Eignung hinsichtlich einer bestimmten Aufgabe. Potenzielle Aufgaben, die ein angemessenes Werkzeug erfüllen muss, sind die korrekte Darstellung und Verarbeitung der Primärdaten, also der Rohtexte, und der Sekundärdaten, also der Annotationen, im Unicodeformat. Außerdem sollte ein Werkzeug in der Lage sein, über einen Tokenizer grundlegende Annotationseinheiten wie Wörter und Sätze automatisch zu erkennen. Die teilweise oder vollständige Automatisierung von immer wiederkehrenden Routineaufgaben und Annotationsphänomenen durch die Verwendung von Lexika oder regulären Ausdrücken sowie die Möglichkeit der Annotation von Relationen und Referenzen zwischen einzelnen Annotationseinheiten gehören ebenso zum Funktionsumfang eines angemessenen Annotationswerkzeugs für diachrone Sprachdaten wie die Editierbarkeit der verwendeten Annotationsschemata. Konsistenzprüfungen und Validierung der Annotation gegen das zugrunde gelegte Schema garantieren eine einheitliche Annotation.

Funktionalität (II): Interoperabilität: Mit der Interoperabilität von Annotationswerkzeugen wird die Kompatibilität der Software zu anderen bestehenden Annotations- und Korpusprogrammen beschrieben. Dabei spielen sowohl umfangreiche Import- und Exportfunktionen als auch ein wohlgeformtes und standardisiertes XML s*tand-off* Format (Dipper 2005) eine entscheidende Rolle. Idealerweise setzt das Annotationsformat auf existierenden Metastandards wie etwa dem *Syntactic Annotation Framework* (SynAF) auf (Trippel, Declerck & Heid 2005, Heilemann 2008).

Benutzbarkeit (I): Erlernbarkeit: Das Unterkriterium Erlernbarkeit beschreibt wie aufwendig es für einen Benutzer ist, sich Kenntnisse über die Funktionsweise und Bedienbarkeit der Annotationssoftware anzueignen. Dabei tragen Programmdokumentation und Tutorials zur *Lernförderlichkeit* des Werkzeugs bei, integrierte Hilfesysteme und Features wie interaktive *Tooltips* steigern die *Selbstbeschreibungsfähigkeit* des Programms. Wenn darüber hinaus noch zusätzliche Problemlösungsstrategien wie eine Mailing-Liste oder FAQs (*frequently asked questions*) angeboten werden, unterstützt das Annotationswerkzeug die *Erlernbarkeit* auf ideale Weise.

Benutzbarkeit (II): Bedienbarkeit: Ein gut bedienbares Annotationswerkzeug sollte ohne übermäßigen kognitiven Aufwand schnell und effizient gehandhabt werden können. *Bedienbarkeit* erstreckt sich von der ersten Installation und Konfiguration der Software, über die Steuerung des Programmablaufs bis zur Modifizierung der Annotationsschemata. Die Bedienbarkeit eines Tools wird durch eine angemessene Visualisierung und ein benutzerfreundliches Interaktionsdesign gesteigert. Im Idealfall können sowohl die Visualisierung als auch das Interaktionsverhalten individuell an den jeweiligen Benutzer angepasst werden.

Benutzbarkeit (III): Konformität: Konformität beschreibt, inwieweit gängige und bekannte Visualisierungs- und Interaktionskonzepte aus dem Bereich der Softwareergonomie auch im zu evaluierenden Annotationswerkzeug umgesetzt werden. Dies betrifft vor allem die Aufteilung und Gestaltung der Arbeitsfläche sowie konventionalisierte Metaphern und Funktionen zur Bearbeitung von Textdaten, wie etwa *Copy/Paste* oder *Undo/Redo*.

4 Metriken: Attribute, Werteskalen und Bewertungsregeln

Für insgesamt 30 Einzelattribute des standardisierten Qualitätsmodells werden Metriken eingesetzt, die es erlauben, dem jeweiligen Qualitätsmerkmal konkrete Werte zuzuordnen. Auf diese Weise können unterschiedliche Werkzeuge anhand ihrer individuellen Werteausprägungen verglichen und bewertet werden. Eine Metrik besteht dabei aus einem Maß und einer Messmethode, wobei die Attribute und alle potenziell zu erwartenden Werte das Maß darstellen. Die Messmethode dient dazu, einen konkreten Wert auf der Werteskala für ein Attribut zu bestimmen. Über Auswahlregeln kann definiert werden, ob mehrere Werte einer Werteskala für ein Attribut kombiniert werden dürfen. Bei Durchführung der Evaluation kann dann für jedes definierte Attribut ein konkreter Wert auf der vorgegebenen Werteskala ermittelt werden. Noch vor der Evaluationsdurchführung muss in so genannten Einstufungsniveaus festgelegt werden, welche Werte im akzeptablen, und welche Werte im nicht mehr akzeptablen Beriech liegen. Durch Aufrechnen der einzelnen Attribute läßt sich ermitteln, zu welchem Grad bestimmte Kriterien oder Subkriterien erfüllt sind. Zudem ist es möglich, Attribute unterschiedlich stark zu gewichten. Tabelle 3 zeigt messbare Attribute und entsprechende Messmethoden für ein Qualitätsmodell unter Berücksichtigung der Qualitätskriterien *Funktionalität* und *Benutzbarkeit*, welches schließlich für die Evaluation von Annotationswerkzeugen für diachrone Korpora verwendet werden soll.

Funktionalität (I): Angemessenheit	*Messmethode / Messfrage*
(01) Unicode Unterstützung	Anzeige und Speicherung als UTF-8?
(02) Alternative Zeichenkodierung	Anzeige und Speicherung als ASCII, ISO-8859 etc.?
(03) Unterstützte Dateiformate	Formate: *.txt, *.doc, *.xml, *.html, *.sgml etc.?
(04) Modifizierbarkeit der Rohtexte	Modifizierung vor und während der Annotation?
(05) Tokenisierung der Rohtexte	Interne oder externe Tokenisierung mit Möglichkeit der Parametrisierung?
(06) Automatisierbarkeit der Annotation	Automatisierbarkeit durch Lexika oder RegEx?
(07) Skopus der Annotationseinheiten	Singuläre Annotationseinheiten oder gerichtete Relationen zwischen den Einheiten?
(08) Änderung der Schemata	Nachträgliche Einfügung, Änderung und Löschung von Tags?
(09) Validierung gegen Schema	Möglichkeit der internen Validierung?
Funktionalität (II): Interoperabilität	***Messmethode / Messfrage***
(10) Im-/Export von Stand-off Formaten	Im-/Exportkompatibilität zu anderen Stand-off Formaten?
(11) Im-/Export Inline-Formaten	Im-/Exportkompatibilität zu anderen Inline-Formaten?
(12) Wohlgeformtheit der Annotation	Wohlgeformtheit des XML-basierten Annotationsformats nach Definition des W3C?
(13) Kompatibilität des Annotationsformats	Kompatibilität des Annotationsformats mit mindestens einem anderen gängigen Annotationsformat?
Benutzbarkeit (I): Erlernbarkeit	***Messmethode / Messfrage***
(14) Dokumentation	Umfang und Qualität der Dokumentation?
(15) Hilfesystem	Umfang und Qualität des Hilfesystems?
(16) Kurzinfo	Umfang und Qualität der Kurzinfos?
(17) Praktische Lernhilfen	Umfang der Lernhilfen in Form von Videos, Tutorials, Übungsdateien, Sekundärliteratur, etc.?
(18) Problemlösungsstrategien	Vorhandensein von zusätzlichen Problemlösungsstrategien wie etwa FAQs, Mailinglists, etc.?
Benutzbarkeit (II): Bedienbarkeit	***Messmethode / Messfrage***
(19) Installationsaufwand	Technischer Aufwand zur Installation der Software?
(20) Ebenenbezogene Aktionen	Löschen, Ein- und Ausblenden von Ebenen möglich?
(21) Editierung der Schemata	Technischer Aufwand bei der Editierung von Schemata?
(22) Flexibilität der Schemata	Flexibilität der Schemata bei unvorhergesehenen Annotationsphänomenen?
(23) Zugriffsrechte für Annotationsebenen	Vergabe von Zugriffsrechten für unterschiedliche Benutzer?
(24) Suchfunktion für Textdaten	Umfang und Qualität der Textsuchfunktion?
(25) Anpassung der Darstellung	Flexibilität der Darstellungsanpassung durch Skins, etc.?
(26) Anpassung der Funktionalität	Flexibilität der Funktionalitätsanpassung durch Shortcuts, etc.?

Benutzbarkeit (III): Konformität	*Messmethode / Messfrage*
(27) Aufteilung der Arbeitsfläche	Aufteilung der Arbeitsfläche in *tiled panes*, *one-widow paging* oder *multiple windows* (Tidwell 2006) ?
(28) Gestaltung der Arbeitsfläche	Stimmigkeit der Farbpalette und Größenverhältnisse sowie Aussagekraft von Piktogrammen und Ähnlichem?
(29) Standardisierte Aktionen	Implementierung von *Undo*, *Redo*, *Autosave*, etv?
(30) Selektion von Annotationseinheiten	Auswahl der Annotationseinheiten durch Doppelklick, Klicken-Markieren-Loslassen, Richtungstasten, etc.?

Tabelle 3: Übersicht zu grundlegenden Attributen und Metriken der in der Evaluationsstudie

Aus Platzgründen können an dieser Stelle die Werteskalen ebenso wenig wie die Gewichtungsregeln aufgeführt werden, vgl. dazu ausführlich Burghardt 2008. Prinzipiell können die Attribute auch für andere Evaluationsstudien im Bereich (diachroner) Annotation herangezogen, und je nach Szenario und Anforderungsprofil mit individuellen Skalen und Gewichtungen versehen werden.

5 Interpretation der Ergebnisse und Ausblick

Bei der Auswertung der Evaluationsergebnisse wird deutlich, dass jedes der vier Tools individuelle Stärken und Schwächen aufweist, insgesamt jedoch das GATE Framework durchweg die besten Ergebnisse erzielt. Unter Berücksichtigung der unterschiedlichen Gewichtung von obligatorischen und optionalen Attributen, und durch Aufrechnung der jeweils akzeptabel erfüllten Attribute für die verschiedenen Qualitätskriterien der ISO 9126 und deren Unterkriterien, soll Tabelle 4 Aufschluss darüber geben, in welchem Maße die vier evaluierten Werkzeuge der Forderung nach Softwarequalität gerecht werden. Eine hundertprozentige Erfüllung bedeutet dabei ein akzeptable Erfüllung aller Attribute des entsprechenden Bereichs.

Werkzeug / *Kriterium*	*CALLISTO*	*GATE*	*MMAX2*	*UAM*
Qualität der **Funktionalität**	57,1%	90,5%	81%	61,9%
(I) Qualität der Angemessenheit	56,3%	87,5%	87,5%	56,3%
(II) Qualität der Interoperabilität	60%	100%	60%	80%
Qualität der **Benutzbarkeit**	70,4%	77,8%	66,7%	63%
(I) Qualität der Erlernbarkeit	85,7%	85,7%	71,4%	100%
(II) Qualität der Bedienbarkeit	60%	73,3%	60%	60%
(III) Qualität der Konformität	80%	80%	80%	20%

Tabelle 4: Übersicht zur Qualität von Funktionalität (Angemessenheit, Interoperabilität) und Benutzbarkeit (Erlernbarkeit, Bedienbarkeit, Konformität)

Die Evaluation zeigt, dass das GATE-Framework in vielen Bereichen Stärken zeigt, und dabei nur wenige Schwächen aufweist. Individuelle Schwachstellen sind zwar vorhanden, können aber in allen

Fällen entweder kompensiert oder vernachlässigt werden. Ursprünglich konzipiert als komplexe Architektur für die Implementierung beliebiger Anwendungen aus dem Bereich des Text Engineering, ist die Verwendung als Annotationswerkzeug nur eines von vielen möglichen Einsatzgebieten der *General Architecture for Text Engineering*. GATE stellt für die Automatisierung von Abläufen ein elaboriertes Konzept auf Basis von endlichen Automaten und regulären Ausdrücken zur Verfügung, welches die regelbasierte Programmierung von Annotationsautomatismen erlaubt. Eine weitere Stärke von GATE besteht darin, verschiedenste Dateiformate lesen zu können. Die optional zuschaltbare Funktion *markup-awareness* erlaubt es darüber hinaus, Dokumente mit bereits bestehendem Markup hinsichtlich der verwendeten Tags und der Dokumentstruktur zu interpretieren. So können etwa teilannotierte Texte in einem beliebigen Markupformat importiert und mitsamt den Annotationen bearbeitet werden.

Bei der Evaluation wird außerdem deutlich, dass trotz des enormen Funktionsumfangs von GATE, und der durchweg guten Testergebnisse aller evaluierten Werkzeuge immer noch etliche Bereiche wie etwa die Koordination eines verteilten Workflows und die konsistente Annotation von sprachlichen Ambiguitäten existieren, die noch von keinem der getesteten Annotationswerkzeuge zufriedenstellend abgedeckt werden. Da GATE aber bereits seit 1995 (Cunningham et al. 2007) kontinuierlich weiterentwickelt wird, und die Community des *open source*-Werkzeugs stetig anwächst, scheint GATE gut geeignet, die benannten Desiderata in absehbarer Zeit zu erfüllen.

6 Literatur

Burghardt, Manuel (2008). Annotationswerkzeuge für diachrone Korpora. Klassifikation und Evaluation von Annotationswerkzeugen. Magisterarbeit, Informationswissenschaft, Universität Regensburg, online verfügbar auf dem Volltextserver der Universität Regensburg unter: urn:nbn:de:bvb:355-opus-10769.

Cunningham, Hamish et al. (2007). *Developing Language Processing Components with GATE Version 4 (a User Guide). For GATE version 4.0 (July 2007)*. [http://gate.ac.uk/sale/tao/ index.html] – Zugriff am 10.06.2008.

DIN 66272 (1994). Bewerten von Softwareprodukten. Qualitätsmerkmale und Leitfaden zu ihrer Verwendung (Identisch mit ISO/IEC 9126: 1991). In: *DIN-Taschenbuch 354 – Software-Ergonomie*. Berlin: Beuth Verlag.

Dipper, Stefanie et al. (2004). Simple Annotation Tools for Complex Annotation Tasks: An Evaluation. In: *Proceedings of the LREC Workshop on XML-based Richly Annotated Corpora*. Lisbon, S. 54–62.

Dipper Stefanie (2005). XML-based Stand-off Representation and Exploitation of Multi-level Linguistic Annotation. In: *Proceedings of Berliner XML Tage 2005 (BXML 2005)*. Berlin, S. 39–50.

Dybkjær, Laila et al. (2001). *Survey of Existing Tools, Standards and User Needs for Annotation of Natural Interaction and Multimodal Data. ISLE Natural Interactivity and Multimodality Working Group Deliverable D11.1*. [http://isle.nis.sdu.dk/reports/wp11/D11.1-14.2.2001. pdf] – Zugriff am 10.06.2008.

Halliday, M. A. K. & J. R. Martin (eds.) (1981). Readings in Systemic Linguistics. London: Batsford Academic & Educational.

Heilemann, Michael (2008). *Informationsstrukturierung für die syntaktische Annotation eines diachronen Korpus des Deutschen*. Magisterarbeit, Informationswissenschaft, Universität Regensburg, online verfügbar auf dem Volltextserver der Universität Regensburg unter: urn:nbn:de:bvb:355-opus-10778.

Ide, Nancy & Chris Brew (2000). Requirements, Tools, and Architectures for Annotated Corpora. In: *Proceedings of the EAGLES/ISLE Workshop on Data Architectures and Software Support for Large Corpora*. Paris: European Language Resources Association, S. 1–5.

Kroymann, Emil et al. (2004). Eine vergleichende Analyse von historischen und diachronen digitalen Korpora. Technischer Bericht 174 am Institut für Informatik. Berlin: Humboldt-Universität.

Tidwell, Jennifer (2006). *Designing Interfaces. Patterns for Effective Interaction Design*. Sebastopol: O'Reilly.

Tognini-Bonelli, Elena (2001). Corpus linguistics at work. Amsterdam: John Benjamins.

Trippel, Thorsten, Thierry Declerck & Ulrich Heid (2005). Sprachressourcen in der Standardisierung. LDV-Forum 20(2) (2005), 17-30.

Wolff, Christian (2004). *Systemarchitekturen. Aufbau texttechnologischer Anwendungen*. In L. Lemnitzer & H. Lobin (Eds.), Texttechnologie. Perspektiven und Anwendungen (pp. 165-192). Tübingen: Stauffenburg.

Werkzeuge zur Extraktion von signifikanten Wortpaaren als Web Service

Fabienne Fritzinger, Max Kisselew, Ulrich Heid, Andreas Madsack, Helmut Schmid

Universität Stuttgart
Institut für maschinelle Sprachverarbeitung
– Computerlinguistik –
Azenbergstr. 12
D 70174 Stuttgart
{fritzife|kisselmx|heid|madsacas|schmid}@ims.uni-stuttgart.de

1 Einleitung

Es gibt viele Techniken und Werkzeuge für die Extraktion von Daten zu signifikanten Wortkookkurrenzen aus Texten, d.h. zur Suche nach Wortverbindungen wie *eine Frage stellen* (Verb+Objektsnomen), *eine knifflige Frage* (Adjektiv+Nomen), *ein Fünkchen Hoffnung* (Nomen+Nomen). Ein Beispiel für ein kombiniertes morphosyntaktisch-statistisches Werkzeug ist die Sketch Engine (Kilgarriff et al. 2004), für ein statistisches Werkzeug ist das UCS toolkit (Evert 2005) beispielhaft. Diese Werkzeuge sind auch erhältlich (mit kommerzieller Lizenz bzw. via SourceForge[1]). Trotzdem ist es nicht trivial, diese und vergleichbare Werkzeuge zu beschaffen, zu installieren und anzuwenden, wenn ad hoc eine Fragestellung zu bearbeiten ist, bei der signifikante Wortverbindungen zu extrahieren sind.

Ziel des vorliegenden Papiers ist es, anhand eines Beispiels die Umrisse einer verteilten Ressourceninfrastruktur für die Extraktion signifikanter Wortpaare zu zeigen. Architekturen, Werkzeuge und Formate für eine solche verteilte Infrastruktur werden derzeit im D-SPIN-Projekt entwickelt[2]: die verteilte Infrastruktur soll Texte und Werkzeuge über Web Services zugänglich und kombinierbar machen. Damit entfällt für den Benutzer das Problem der Beschaffung, der Installation und z.T. der Einarbeitung in die Details der Nutzung der Werkzeuge. Wir simulieren ein solches durch Web Services unterstütztes Szenarium am Beispiel laufender Arbeiten zu Werkzeugen für die Extraktion signifikanter Wortpaare. Dabei ist einerseits die Bereitstellung der Werkzeuge in Web Services neu gegenüber ihrer bisherigen Nutzung, und zum anderen stellen die Extraktionsverfahren eine interessante Kombination von Dependenzparsing, (morpho)syntaktisch basierter Extraktion und den üblichen Signifikanzmaßen dar.

[1] http://www.sourceforge.net

[2] D-SPIN (Deutsche Sprachenressourcen-Infrastruktur) ist ein vom BMBF gefördertes Projekt (2008–2010), das Werkzeuge, Verfahrensweisen und rechtliche Rahmenbedingungen für eine nationale Ressourceninfrastruktur erarbeitet. Das Projekt wird von E. Hinrichs (Universität Tübingen) koordiniert. Die Autoren sind Werkvertragsnehmer der Universität Tübingen. D-SPIN ist das nationale deutsche Ergänzungsprojekt zum eurpäischen Projekt CLARIN, dessen Zielsetzung die Schaffung einer Sprachressourcen-Infrastruktur für ganz Europa ist. Details zu D-SPIN und CLARIN: http://www.sfs.uni-tuebingen.de/dspin/ und http://www.clarin.eu

2 Szenarium

Jemand analysiert deutsche Berichte über Medikamentenprüfungen und will sich über die wichtigsten lexikalischen Kollokationen (im Sinne von Hausmann 2004, Bartsch 2004, etc.), bzw. über die in der Textsorte ganz allgemein häufigsten Zweiwortkombinationen informieren, z.B. weil er solche Texte verfassen oder übersetzen will. Wir gehen hier beispielhaft von einem Übersetzer oder einem technischen Redakteur aus, der an den Daten interessiert ist, aber selbst keine Zeit, kein Interesse oder keine Kompetenz dafür hat, eigene computerlinguistische Werkzeuge zu entwickeln, oder auch nur die verteilt verfügbaren Werkzeuge zu modifizieren oder ihre Interaktion zu steuern.

Als Beispielkorpus in unserem Experiment dienen die Texte über Medikamentenprüfungen der EMEA-Agentur[3], die in verschiedenen Sprachen (ca. 10-14 Millionen Wörter pro Sprache) auf Tiedemanns OPUS-Website[4] zur Verfügung stehen. Aus den deutschen Texten sollen Wortpaare verschiedener grammatischer Kategorien extrahiert werden: Adjektiv+Nomen (z.B. *antagonistische Wirkung, arzneiliche Bestandteile*), Verb+Objektsnomen (*Packungsbeilage beachten, Nebenwirkungen bemerken*) und Nomen+Genitivattribut (*Abfall des Blutdrucks, (wirksame) Bestandteile des Arzneimittels*). Die Wortpaare[5] sollen alphabetisch nach den Kollokationsbasen (Hausmann 2004) sortiert werden: Adjektiv+Nomen nach Nomen, Verb+Objektsnomen nach Nomen, Nomen+Genitivattribut nach Genitivattribut. Außerdem soll eine Sortierung nach der absoluten Frequenz und nach dem mit Hilfe des LogLikelihood Ratio Tests (Dunning 1993) bestimmten Signifikanzwert erfolgen (wie typisch sind die Kombinationen, bzw. wie fest zusammengehörig? Vgl. Evert 2005).

Der Benutzer sucht bei D-SPIN nach Vorschlägen zur Lösung des Problems. Wir nehmen für das vorliegende Experiment an, dass für die allgemeine Aufgabenstellung der Suche nach Kookkurrenzen der in Abschnitt 3 skizzierte Workflow vorgeschlagen werden kann. Auf dessen Bereitstellung als Webservice gehen wir in Abschnitt 4.1 ein. In Abschnitt 4.2 stellen wir ein einfaches Format dar, in dem die Ergebnisse an den Benutzer geliefert werden könnten.

3 Extraktion von Kollokationen

Für Deutsch (anders als z.B. für Englisch) erscheint es notwendig, die Extraktion von Daten zu signifikanten Wortpaaren auf syntaktisch analysiertem Text aufsetzen zu lassen: die drei Verbstellungsmuster des Deutschen und die relativ freie Konstituentenreihenfolge im Mittelfeld (Heid/Weller 2008), Kasussynkretismus (Evert 2004) und als Auswirkung von beidem strukturelle Ambiguitäten führen andernfalls zu schlechten Ergebnissen. Experimente mit Kilgarriffs Sketch Engine (Ivanova et al. 2008) haben gezeigt, dass eine Extraktion aus lediglich getaggtem und lemmatisiertem Text zu niedriger Precision führt, oder dass nur sehr eingeschränkte Kontexte benutzt werden können, um den Preis von sehr geringem Recall. Ein Vergleich zwischen rekursivem Chunking und Dependenzparsing (cf. Heid et al. 2008) hat darüber hinaus ergeben, dass eine Extraktion auf der Grundlage von *"chunked text"* zwar eine akzeptable Precision liefern kann, aber nur ca. 40% des Recalls, der auf geparsten Texten erzielt werden kann.

Für unsere Experimente benutzen wir den Dependenzparser FSPAR (Schiehlen 2003), der grammatische Funktionen markiert und lokale Ambiguitäten annotiert, bezüglich Etiketten (Subjekt vs. Objekt) und Attachment. Der Parser integriert die anderen Schritte der Korpusvorverarbeitung: Tokenizing, Tagging und

[3] Euorpean Medicines Agency.

[4] `http://urd.let.rug.nl/tiedeman/OPUS/EMEA.php`

[5] Eigentlich Paare von Lemmata. Da viele Kollokationen Präferenzen für einen bestimmten Numerus aufweisen, zitieren wir sie hier in der bevorzugten Form.

Lemmatisierung[6]. Im vorliegenden Fall wird der Ausgangstext dennoch zunächst tokenisiert um Satzgrenzen vorab zu identifizieren (zwecks Erhöhung der Verarbeitungsgeschwindigkeit des Parsers). Der Ablauf der Kollokationsextraktion ist in Abbildung 1 schematisch dargestellt.

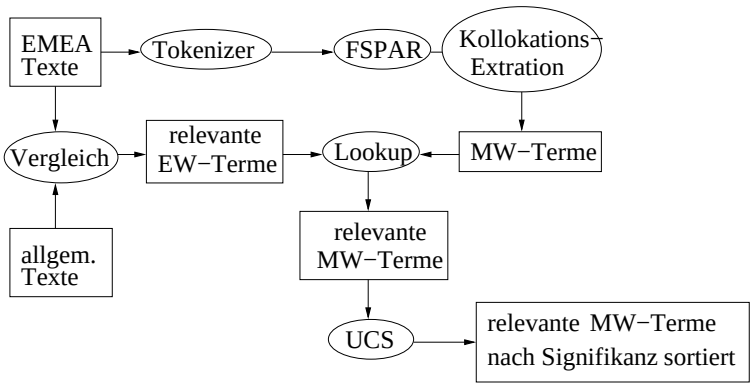


Abbildung 1: Schema der Prozessierungspipeline für die Extraktion von Mehrworttermen.

3.1 Einzelwörter

Es sollen terminologisch relevante Mehrwortausdrücke extrahiert werden, d.h. solche, die terminologisch relevante Einzelwörter enthalten. Derartige Einzelwörter werden vorab anhand des POS-getaggten und lemmatisierten Textes mit Hilfe eines Vergleichs der relativen Häufigkeit in Fach- und Nicht-Fachtexten ermittelt (vgl. Ahmad et al. 1992). Als allgemeinsprachliche Referenz für den Vergleich der relativen Frequenzen verwenden wir Texte aus Zeitungskorpora (bspw. aus der *Frankfurter Rundschau*, 1992/93, FR).

Zwei verschiedene Arten von Listen sind das Ergebnis dieses Verfahrens: eine Liste enthält diejenigen Einzelworttterme, die im Referenztext nicht vorkommen (diese Liste ist nach der absoluten Häufigkeit der Terme sortiert, Vgl. Tabelle 1(a)); die andere Liste besteht aus Termen, die im Referenztext vorkommen und für jeden Termkandidaten aus einem Quotienten, der besagt, wieviel häufiger der Term im Fachtext auftritt als im allgemeinsprachlichen Text. Letztere Liste ist nach diesem Quotienten absteigend sortiert (Tabelle 1 (b)).

(a) Nur EMEA, Nicht FR

Termkandidat	Frequenz
Durchstechflasche	5638
Injektionsstelle	3489
Pharmakokinetik	3426
Hämoglobinwert	3395
Fertigspritze	3271
Ribavirin	3234
Gebrauchsinformation	2801
Dosisanpassung	2580
Epoetin	2302
Hydrochlorothiazid	2128

(b) EMEA und FR

Termkandidat	Quotient	Frequenz
Filmtablette	25522	6389
Injektionslösung	19854	4970
Packungsbeilage	14710	7365
Niereninsuffizienz	14233	3563
Verkehrstüchtigkeit	13558	3394
Leberfunktion	8385	2099
Hypoglykämie	8353	2091
Toxizität	7957	1992
Einnehmen	7035	7045
Hypotonie	6823	1708

Tabelle 1: Die ersten 10 Nomina der beiden Einzelwort-Termlisten.

[6]In einer anderen Werkzeugumgebung müßten u.U. separate Tools für diese Funktionen vorgeschlagen werden.

Für jede Wortart werden für die beiden Listen-Typen Schwellen gesetzt, um das Fachvokabular möglichst präzise zu erfassen. Die daraus resultierenden Ergebnisse werden dann im nächsten Schritt mit ihren Kollokationspartnern versehen.

3.2 Signifikante Wortpaare

Die Mehrwortausdrücke werden aus dem geparsten Text mit Hilfe von in Perl formulierten regulären Suchanfragen extrahiert. Für Verb+Objekt-Paare wird dabei z.B. nach Verben und ihren Akkusativ-Objekten in Aktivsätzen, bzw. nach Subjekten und Verben von Passivsätzen gesucht. Wir untersuchen drei Arten von Mehrwortausdrücken: Adjektiv+Nomen-Paare, Verb+Objekt-Paare sowie Nomen+Genitiv-Nomen-Paare. Die Suche findet alle Lemmapaare, für die der Parser eine dieser drei syntaktischen Kombinationen annotiert hat. Um die häufigsten, bzw. die signifikantesten Paare zu ermitteln, wird für jedes Paar mit dem UCS-Toolkit[7] (Evert 2005) der LogLikelihood-Wert ermittelt. Die Daten werden u.a. nach absteigendem LogLikelihood-Wert sortiert. Ergebnisbeispiele sind in Tabelle 2 angegeben.

(a) Adjektiv+Nomen

Adjektiv	Nomen	Frqz	LogL
allgemeiner	Risikofaktor	64	655
starke	Nierenfunktion	108	511
potentielle	Reaktion	60	508
arzneilicher	Bestandteil	39	458
kardiotoxische	Substanz	43	457
späte	Bewegungsstörung	33	410

(b) Verb+Objektsnomen

Nomen	Verb	Frqz	LogL
Nebenwirkung	bemerken	1665	11381
Packungsbeilage	beachten	1303	9241
Apotheker	fragen	1051	8533
Maschine	bedienen	543	6973
Einnahme	vergessen	677	5584
Anwendung	empfehlen	884	4396

Tabelle 2: Wortpaare mit Angabe der Absolutfrequenz und des LogLikelihood-Wertes.

4 Einbettung in Web Services

Ziel unserer Arbeit ist es, eine verteilte Ressourcenlandschaft zu simulieren, in der die EMEA-Texte, der Parser samt Vorverarbeitung, die Extraktionsroutinen für Mehrwortkandidaten und das UCS-Toolkit jeweils an verschiedenen Orten vorliegen und durch Web Services verbunden werden müssen. Im Gegensatz zu existierenden Web Services, die sich mit linguistischen Anfragen auf Wortebene befassen (wie z.B. im Projekt "Deutscher Wortschatz" entstanden[8]), legen wir unser Hauptaugenmerk auf die Verarbeitung von umfangreichen Textsammlungen in Größenordnungen von etwa 50MB - 200MB.

Für die Realisierung dieser Web Services nehmen wir ein dreistufiges Schichtenmodell an, das eine klare Trennung der verschiedenen Anforderungen und Aufgaben darstellt (Vgl. Abbildung 2). Die äußerste Schicht, auf die wir hier nicht gesondert eingehen werden, befasst sich mit Fragen von Zugriffsberechtigungen, Authentifizierung, Abrechnung, etc. In unserer Implementierung setzen wir voraus, dass es eine solche Schicht bereits gibt[9]. Die innerste Schicht (oder auch der Kern) des Modells enthält (computer-) linguistische Werkzeuge, die bereits vorhanden sind, und die ohne prinzipielle Änderungen in die Web Service Infrastruktur eingebunden werden können. Wir konzentrieren uns hier auf das Bindeglied des Schichtenmodells, auf die mittlere Schicht. Diese befasst sich mit den Schnittstellen zwischen Werkzeugen und dem Benutzer. Es gilt dabei u.a. Fragen zu klären, die die Definition von Ein- bzw. Ausgabeformaten betreffen,

[7] Erhältlich unter `http://www.collocations.de`

[8] Siehe `http://wortschatz.uni-leipzig.de/Webservices/`

[9] Innerhalb des D-SPIN/CLARIN-Projektes gibt es Spezialisten, die sich intensiv mit den genannten Problemstellungen befassen.

welche einerseits mit den Web Services vollständig kompatibel sein sollen, andererseits aber auch mächtig genug sein müssen, die linguistisch annotierten Zwischenergebnisse adäquat zu beschreiben.

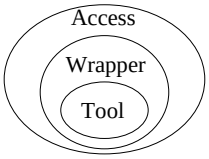


Abbildung 2: Schichtenmodell der Aufgabenteilung.

4.1 Stand der Arbeiten

Bei der Implementierung des vorgestellten Beispielszenarios handelt es sich um laufende Arbeiten, die zum Zeitpunkt der Abfassung des vorliegenden Papiers noch nicht vollständig abgeschlossen sind. Dieser Abschnitt enthält daher eine Momentaufnahme, in der wir auf Gegebenheiten und Ergebnisse auf dem Stand von Januar 2009 eingehen.

4.1.1 Technische Aspekte

Auf Grundlage der REST-Architektur (Richardson/Ruby 2007) entwickeln wir einen Web Service, der die Funktionalität einer Pipeline aus den oben genannten Werkzeugen realisiert. Wir gehen von einem übergeordneten Web Service aus, der zur Laufzeit mehrere Komponenten-Web Services (möglichst an die Module der Pipeline angepasst) aufruft. Diese Teil-Services werden in die Pipeline von Abbildung 1 eingebaut, und in Abbildung 3 (mithilfe gestrichelter Linien) skizziert.

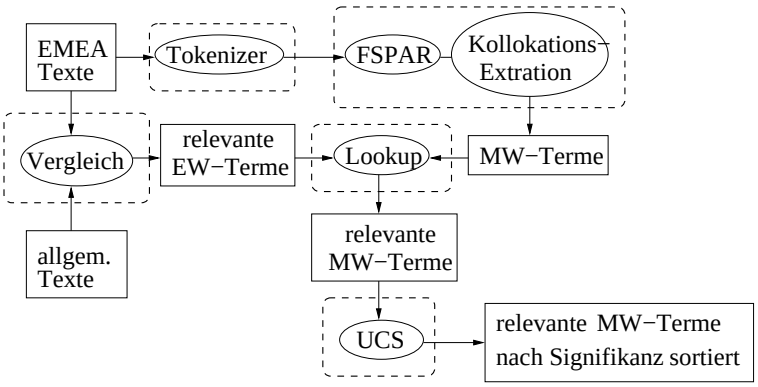


Abbildung 3: Prozessierungspipeline mit angedeuteten Webservice-Komponenten.

Als Eingabe soll der übergeordnete Service einen fachsprachlichen Text (im aktuellen Fall aus der EMEA-Domäne) und einen allgemeinsprachlichen Referenztext entgegennehmen, dann intern die Teil-Services anstoßen und schließlich als Ausgabe eine Liste mit terminologisch relevanten Wortkookkurrenzpaaren produzieren, die nach Kollokationstyp getrennt und jeweils nach Signifikanz (LogLikelihood) absteigend sortiert sind.

Problematisch sind hierbei vor allem die beiden Teil-Services “Vergleich” und “Lookup”, da diese jeweils zwei verschiedene Eingabedateien erwarten. Der Service “Vergleich” ist vor allem insofern schwierig, als

dass er vom Benutzer zwei Dateien übergeben bekommt. Eine mögliche Alternative wäre hier, den Benutzer zwischen hinterlegten allgemeinsprachlichen Texten wählen zu lassen. Beim Service “Lookup” ergibt sich zudem die Schwierigkeit, daß die beiden Eingabedateien erst produziert werden müssen (mit potentiell unterschiedlich langen Produktionszeiten) und dieser Service erst gestartet werden kann, wenn beide Dateien zur Verarbeitung vorliegen.

Unser Webservice kann in seiner momentanen Implementierung nur jeweils eine Eingabedatei bearbeiten; daher wurde zunächst nur ein Teil der Pipeline (tatsächlich) realisiert. Es handelt sich dabei um denjenigen Teil, in dem es um die Extraktion von signifikanten Mehrwortausdrücken aus einem gegebenen Text geht (Vgl. Abbildung 4 unten).

Das in Abschnitt 2 gegebene Szenarium können wir dennoch aufrecht erhalten, da die statistischen Signifikanzmaße meist ohnehin domänenspezifische Kollokationen hoch bewerten, auch ohne dass der Eingabetext vorher mit einem allgemeinsprachlichen Referenztext abgeglichen wurde (siehe hierzu auch die Beispiele aus Tabelle 2, die sich ohne Abgleich genauso verändert hätten.).

4.1.2 Implementierung

Zu Testzwecken wurde zunächst ein auf Python basierender Webserver implementiert. Dieser ermöglicht die Realisierung der Grundfunktionalitäten der Pipeline ohne auf HTTP-Anforderungen eingehen zu müssen. Der aktuelle Stand der Implementierung ist in Abbildung 4 angedeutet. Dort sind die realisierten Teile in schwarzen Linien dargestellt, während die noch nicht implementierten Teile der Architektur aus Abbildung 3 grau gedruckt sind.

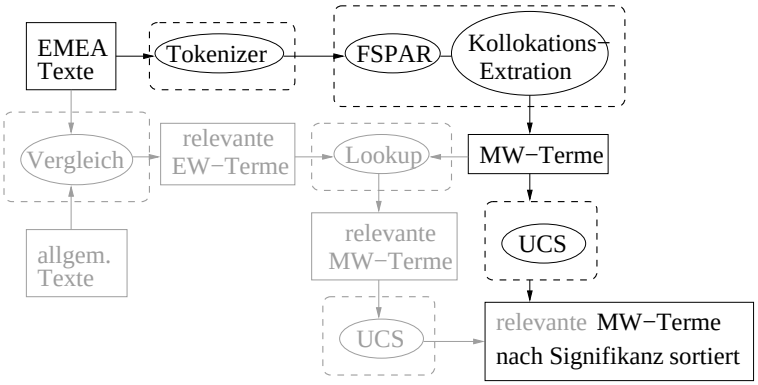


Abbildung 4: Prozessierungspipeline mit bisher realisierten Webservice-Komponenten.

In Kürze soll die Pipeline jedoch auf eine Kombination von Python und einem Apache-Webserver umgestellt werden, damit der Web Service (aus technischer Sicht[10]) auch extern zur Verfügung gestellt werden kann. Diese Umstellung ist für zwei Komponenten der Pipeline (“Tokenizer” und “UCS”) bereits realisiert.

Die Modifizierbarkeit der Pipeline durch den Benutzer ist im momentanen Aufbau noch eingeschränkt: im vorliegenden Szenario (Vgl. Abschnitt 2) gehen wir von einem computerlinguistischen Laien aus, der über die Zwischenschritte und auch die Zwischenergebnisse der Pipeline-Komponenten keine Auskunft erhält. In Zukunft soll dem Benutzer jedoch die Möglichkeit gegeben werden, die Zusammensetzung der Pipeline und auch die Präsentation der Ergebnisse beeinflussen zu können (Vgl. hierzu Abschnitt 4.2.1).

[10] Auf Fragen bezüglich Authentifizierung und Zugriffsberechtigungen gehen wir hier nicht gesondert ein.

Momentan ist es neben einer Anfrage an den übergeordneten Service möglich, auch Anfragen an die drei implementierten Teil-Services (nämlich “Tokenizer”, “FSPAR+Kollokationsextraktion” und “UCS”) zu stellen, jedoch ohne die Möglichkeit, Komponenten durch gleichwertige Werkzeuge (die sich in einer verteilten Ressourcenlandschaft u.U. an verschiedenen Orten befinden) auszutauschen. Hierfür müssen in Zukunft vor allem noch Fragen des Eingabe- und Ausgabeformats sowie der Konvertierung in derartige Formate geklärt werden. Ein erster möglicher Ansatz in dieser Richtung wird in Abschnit 4.2.2 beschrieben.

Da der auf einer Dependenz-Grammatik basierende Parser einen für den computerlinguistischen Laien eher unverständlichen Output liefert, kann dieser Service im Moment nur in Kombination mit anschließender Kollokationsextraktion aufgerufen werden. Aus rein technischer Sicht ist es jedoch auch problemlos möglich, dem Benutzer die Parser-Ausgabe direkt abzuliefern.

4.1.3 Ergebnisse

Die Pipeline greift unter anderem auf statistische Assoziationsmaße (in UCS: LogLikelihood) zurück, daher ist es notwendig, eine ausreichend große Textmenge (mindestens 5-10 Millionen Wörter) in die Prozessierungspipeline einzuspeisen. Das EMEA-Korpus, welches in der Beispiel-Implementierung als Testkorpus verwendet wurde, besteht aus etwa 10 Millionen Wortformen (das entspricht im aktuellen Fall 67MB in ASCII-Format) und lässt sich von sämtlichen Komponenten problemlos verarbeiten. Insgesamt dauert ein vollständiger Durchlauf durch die Pipeline (auf einem Dual AMD Opteron Rechner mit 2 x 2,2 GHz und 16GB RAM) etwa 35 Minuten, wovon die meiste Zeit (etwa 29 Minuten) auf das Parsing entfällt.

Bevor unser Web Service nach außen hin bereitgestellt wird, müsste noch untersucht werden, ob es eine Obergrenze an Daten gibt, die die einzelnen Komponenten maximal verarbeiten können (in Bezug auf Speicherbedarf und Rechenzeitkapazität). Außerdem bleibt noch zu klären, wie mit simultanen Anfragen umgegangen wird.

Die Ausgaberoutine liegt momentan in zwei verschiedenen Varianten vor. Zum einen hat der Benutzer die Möglichkeit, die Ergebnisse seiner Anfrage auf dem Bildschirm angezeigt zu bekommen (Standard Output), zum anderen kann der Web Service den Benutzer über einen eindeutigen Link zum Download der Ergebnisdatei führen, sobald diese fertig gestellt ist. Obwohl untypisch für einen Webservice, wäre in Anbetracht der vom Umfang der Anfrage abhängigen Laufzeiten das Zusenden einer E-Mail mit entsprechendem Download-Link durchaus eine denkbare Alternative.

4.2 Geplante Arbeiten

Die aktuelle Implementierung stellt einen im Vorhinein festgelegten Durchgang durch die Pipeline der Kollokationsextraktion dar. Der Benutzer kann nur zu Beginn in die Prozedur eingreifen: durch Auswahl des zu bearbeitenden Textes. In Anbetracht verschiedener Nutzergruppen (bspw. Linguisten, Fachleute der Historiker) erscheint es jedoch sinnvoll, die Pipeline weiter zu modularisieren, damit der Benutzer sie optimal an seine Vorkenntnisse und Bedürfnisse anpassen kann. Dazu müssen zum einen Interaktionspunkte zum Service definiert werden, an welchen der Benutzer die weitere Verarbeitung (u.U. basierend auf Zwischenergebnissen, die er bis dahin erhalten hat) beeinflussen kann, zum anderen müssen Formate gefunden werden, die ohne Informationsverlust die Kombination verschiedener Werkzeuge ermöglichen.

4.2.1 Interaktivität

In Zukunft soll an verschiedenen Stellen in die Prozedur eingegriffen werden können. Die Definition eines geeigneten Web Interfaces für die Interaktion des Benutzers mit den Webservices ist dabei unumgänglich. Die Interaktion selbst könnte dann über eine übergeordnete Web Application stattfinden, die Eingaben des Benutzers entgegennimmt und daraufhin Web Services aufruft.

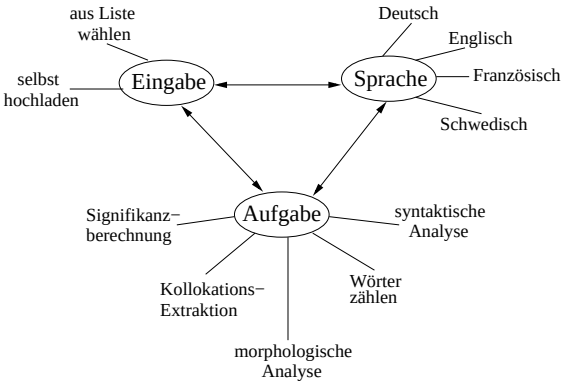


Abbildung 5: Auswahlparameter vor Beginn des eigentlichen Web Services.

Bevor eine Verarbeitungspipeline gestartet wird, müssen im Dialog mit dem Benutzer zunächst grundlegende Fragen, etwa nach der Sprache des zu verarbeitenden Textes, nach der Art der Übergabe und auch nach der Verarbeitung der Eingabe geklärt werden. Diese drei Parameter sind stark voneinander abhängig (Vgl. hierzu auch Abbildung 5), da bspw. nicht für jede Sprache die selbe Anzahl an Werkzeugen (und damit Bearbeitungsmöglichkeiten) gegeben sind. Weiterhin ist von der Wahl der Aufgabe, die der Web Service bearbeiten soll, das Eingabeformat abhängig (z.B. Tokenisieren, Parsing: Text, Signifikanzberechnung: Wortpaare).

Einem Benutzer mit Vorkenntnissen in der linguistischen Datenverarbeitung soll ermöglicht werden, für die einzelnen Verarbeitungsschritte der von ihm ausgewählten Pipeline selbst zu bestimmen, welche Werkzeuge verwendet werden sollen (z.B. Wahl des Formalimus bzw. der Technologie, die für das Parsing benutzt wird). Außerdem soll der Benutzer (auf Wunsch) Zwischenergebnisse der einzelnen Komponenten erhalten können. Hierfür müssen für jedes Modul der Pipeline Interaktionspunkte gesetzt werden, an welchen der Benutzer in den laufenden Prozess eingreifen kann (Vgl. Graphik in Abbildung 6). Um eine derartige Modularität der Pipeline-Komponenten zu realisieren, ist es unumgänglich, für alle (alternativen) Verarbeitungskomponenten nutzbare Ein- und Ausgabeformate zu entwickeln (Vgl. hierzu auch Abschnitt 4.2.2).

Neben einem Einfluss auf die Zusammensetzung der Prozessierungspipeline sollte der Benutzer zudem auswählen können, in welchem Format ihm die Ergebnisse präsentiert werden sollen: z.B. eine Sortierung der Kollokationskandidaten nicht nur absteigend nach Signifikanz, sondern u.U. auch alphabetisch nach Substantiven oder nach den Kollokatoren. Manche Benutzer möchten vielleicht auch Beispielsätze zu den Kollokationen sehen, oder eine Verteilung über Singular und Plural (d.h. eventuelle morphosyntaktische Präferenzen)[11]. Abbildung 6 zeigt beispielhaft, an welchen Stellen der Benutzer in die Pipeline eingreifen könnte, und was für Auswahlmöglichkeiten er dabei hat. Da diese Pipeline nicht schon zu Beginn festgelegt ist, könnten optionale Komponenten (wie etwa die Signifikanzberechnung, in Abb. 6 gestrichelt dargestellt) auch weggelassen werden.

4.2.2 Eingabe und Ausgabe-Formate

Im folgenden Abschnitt beschreiben wir einige Vorschläge für geeignete Ein- und Ausgabeformate in XML, wie man sie in einer modular aufgebauten Pipeline, bei der jede Komponente als ein untergeordneter Web Service realisiert ist, verwenden könnte. Die vorgestellten Formate sind von den in unserer Studie verwende-

[11] Die Ausgabe von Beispielsätzen und morphosyntaktischen Informationen ist in der aktuellen Pipeline noch nicht vorgesehen, könnte aber bereitgestellt werden, indem eine weitere Extraktionskomponente auf der Grundlage des geparsten Textes hinzugefügt wird.

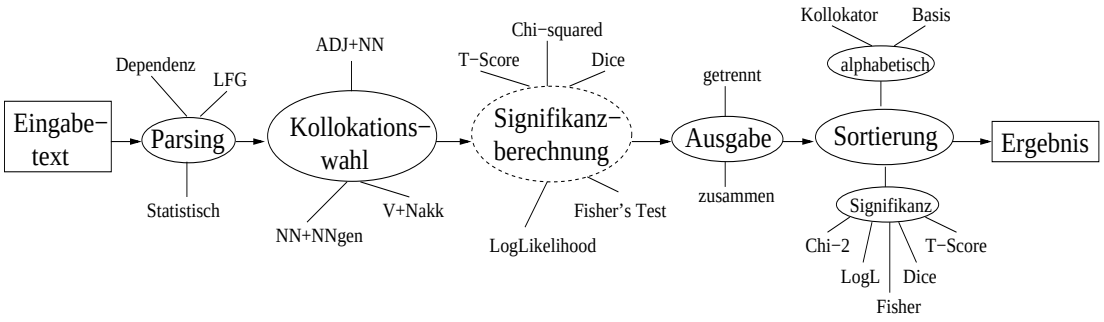


Abbildung 6: Pipeline zur Kollokationsextraktion inklusive Auswahlmöglichkeiten einzelner Module.

ten Werkzeugen inspiriert, sollten aber an Anforderungen anderer Werkzeuge angepasst werden können[12]. In der oben skizzierten aktuellen Implementierung unserer Pipeline (Vgl. Abschnitt 4.1.2) haben wir die Formatvorschläge noch nicht berücksichtigt. Das Format wird anhand eines Beispielsatzes aus dem EMEA-Korpus illustriert: *"Die zweite Studie lieferte ähnliche Ergebnisse."*. Aus Gründen der Übersichtlichkeit werden die XML-Strukturen sukzessive skizziert und vorgestellt.

Allgemeine Struktur Nach einem Durchlauf der oben vorgestellten Pipeline zur Extraktion von Kollokationen soll nicht nur eine Liste der signifikanten Kollokationen selbst zurückgegeben werden, es soll vielmehr alle Information, die während des Durchlaufs gesammelt wurde, abgespeichert werden (bspw. Zwischenergebnisse von einzelnen Komponenten). Ein Ausgabedokument, das als Ergebnis aus der Verarbeitungspipeline zurückgeliefert wird, könnte daher mit der in Abbildung 7 gegebenen Spezifikation beginnen, die u.a. Informationen über den Analysierungsgrad des eingegebenen Ursprungstextes, sowie über diesen Text selbst enthält und damit implizit die durchlaufenen Komponenten widerspiegelt (z.B. in den Belegungen der Attribute "parsing" und "mweExtraction").

```
<metadata>
  <source>IMS, Universität Stuttgart</source>
</metadata>
  <TextCorpus language=''de'' encoding=''utf8'' tokenisation=''yes'' POStagging=''STTS''
     lemmatisation=''yes'' parsing=''FSPAR'' mweExtraction=''yes''
     source=''ftp://www.ims.uni-stuttgart.de/pub/D-Spin/emea_fsparsed.xml''>
  <text>
     Die<B/>zweite<B/>Studie<B/>lieferte<B/>ähnliche<B/>Ergebnisse.<NL/>
  </text>
```

Abbildung 7: Vorschlag für den XML-Header der Ausgabedatei mit Repräsentation des eingegebenen Textes.

Repräsentationen auf Wortebene Die Ausgabe des Parsers enthält neben syntaktischen Relationen auch Informationen zu Wortgrenzen (Tokenisierung), zu den Grundformen der Wörter (Lemmatisierung) und zu den Wortarten (Tagging). Die entsprechenden Repräsentationen[13] dieser Informationen sind in Abbil-

[12] Im Rahmen des D-SPIN Projektes sollen einheitliche Formate entwickelt werden, die so allgemein wie möglich gehalten sind, damit sie für die Ein- und Ausgabe von möglichst vielen Werkzeugen verwendbar sind, und damit die Werkzeuge untereinander dadurch einen möglichst hohen Grad an Kompatibilität erreichen. Zum Zeitpunkt der Fertigstellung dieses Papiers waren in der Diskussion der Projektteilnehmer über das Format noch keine endgültigen Festlegungen getroffen.

[13] Hier zugunsten der Übersichtlichkeit leicht abgekürzt.

```
<tokens>
   <token id=''t1'' start=''1'' end=''3''>Die</token>
   <token id=''t2'' start=''5'' end=''11''>zweite</token>
   <token id=''t3'' start=''13'' end=''19''>Studie</token>
   ...
   <token id=''t7'' start=''52'' end=''52''>.</token>
</tokens>
<sentences>
   <sentence id=''s1'' start=''1'' end=''52''/>
</sentences>
<POStags>
   <tag tokID=''t1'' cat=''ART''/>
   <tag tokID=''t2'' cat=''ADJA''/>
   <tag tokID=''t3'' cat=''NN''/>
   ...
   <tag tokID=''t7'' cat=''$.''/>
</POStags>
<lemmas>
   <lemma tokID=''t1''>d</lemma>
   <lemma tokID=''t2''>2.</lemma>
   <lemma tokID=''t3''>Studie</lemma>
   ...
   <lemma tokID=''t7''>.</lemma>
</lemmas>
```

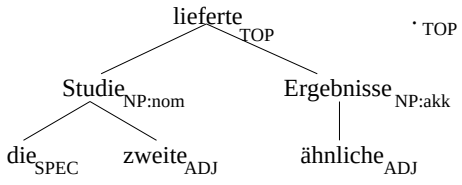
Abbildung 8: XML-Formatvorschlag für Tokenisierung, Tagging und Lemmatisierung.

dung 8 dargestellt. Zunächst wird jedes Wort (Token) mit einer Identifikationsnummer (TOKID) versehen. Außerdem wird festgehalten, aus welchen Zeichen (*"characters"*) ein Wort besteht (Vgl. "start"- und "end"-Attribute in Abb. 8). Bei der Darstellung von Wortarten (in Form von POS-Tags) und auch bei der Lemmatisierung wird auf diese TOKIDs referenziert.

Repräsentation auf Satzebene Das syntaktische Analyse stellt die komplexeste darzustellende Komponente dar. Die in Abbildung 9(a) gezeigte XML-Struktur enthält nur die wichtigsten syntaktischen Informationen, nämlich die Abhängigkeiten (Dependenzen) der Satzglieder untereinander[14].

```
<depparsing>
   <dependency function=''SPEC'' govID=''t3'' depID=''t1''/>
   <dependency function=''ADJ'' govID=''t3'' depID=''t2''/>
   <dependency function=''NP:nom'' govID=''t4'' depID=''t3''/>
   <dependency function=''ADJ'' govID-=''t6'' depID=''t5''/>
   <dependency function=''NP:akk'' govID=''t6'' depID=''t6''/>
<depparsing>
```

(a) XML-Format



(b) Baumrepräsentation

Abbildung 9: Repräsentation der syntaktischen Analyse (Dependenzstruktur).

Der XML-Code für dieses Format kann problemlos automatisch aus der Parsing-Ausgabe generiert werden, jedoch enthält die Ausgabe des verwendeten Parsers einige zusätzliche Informationen, die in diesem Format

[14]Zur Verdeutlichung der Dependenz-Struktur ist daneben in Abbildung 9(b) eine Baumrepräsentation des Satzes gegeben.

verloren gehen. Diese Art von Information sollte in einer zukünftigen (erweiterten) Version des Formates jedoch unbedingt berücksichtigt werden. In Zukunft muss noch untersucht werden, inwiefern die syntaktischen Repräsentationen in diesem XML-Format von der Art des Parsings abhängig sind (z.B. Dependenzparsing vs. Konstituentenstruktur-Parsing) und ob es Möglichkeiten gibt, die Ergebnisse unterschiedlicher Parser (in Form von XML-Repräsentationen) ineinander zu überführen.

Repräsentation von Wortkookkurrenzen Für die signifikanten Mehrwortterme, die das Ergebnis unserer Extraktionspipeline darstellen, soll ebenfalls eine geeignete XML-Struktur gefunden werden. Abbildung 10 zeigt ein mögliches XML-Format für Mehrwortterme ("mwe" = *Multiword Expressions*), in dem die errechneten Signifikanzmaße (hier: "loglik") und die absolute Frequenz ("absfreq") der Terme mitberücksichtigt sind (siehe auch Abb. 10(b) für eine Darstellung im Tabellenformat).

```
<mwes>
   <mwe type=''v_nobj''>
      <k1 refid=''l6''/>
      <k2 refid=''l4''/>
      <loglik value=''22.82''/>
      <absfreq value=''7''/>
   </mwe>
   <mwe type=''adj_nn''>
      <k1 refid=''l2''/>
      <k2 refid=''l3''/>
      <loglik value=''273.44''/>
      <absfreq value=''183''/>
   </mwe>
   <mwe type=''adj_nn''>
      <k1 refid=''l5''/>
      <k2 refid=''l6''/>
      <loglik value=''357.20''/>
      <absfreq value=''64''/>
   </mwe>
</mwes>
```

(a) XML-Format

Nomen	Verb	Frqz	LogL
Ergebnis	liefern	7	22.82

Adjektiv	Nomen	Frqz	LogL
zweite	Studie	183	273.44
ähnlich	Ergebnis	64	357.20

(b) Tabellen-Format

Abbildung 10: Repräsentation von Kollokationen.

5 Zusammenfassung

Das vorliegende Papier ist eine Momentaufnahme unserer laufenden Arbeiten zur Bereitstellung computerlinguistischer Werkzeuge und Prozessierungspipelines durch Web Services. Anhand eines Beispielszenarios wurde eine übliche Verarbeitungspipeline vorgestellt und teilweise realisiert. Dabei sind wir auf einige grundlegenden Fragen gestoßen, die es in Zukunft noch zu untersuchen gilt. Ein zentrales Problem bei der Einbindung von NLP-Werkzeugen in Web Services stellen die für Web Services unüblichen langen Antwortzeiten der Werkzeuge dar (Vgl. Abschnitt 4.1.3, FSPAR: 30 Minuten für 10 Millionen Wortformen). Weiterhin sind die (Zwischen-) Ergebnisse z.T. sehr umfangreich (die FSPAR-Ausgabe eines Textes mit etwa 10 Millionen Wortformen hat ein Volumen von 414MB) und sollten daher in Zukunft komprimiert an den Benutzer zurückgegeben werden. In Bezug auf Möglichkeiten der Benutzerinteraktion mit dem Web Service gibt es ebenfalls noch einige offenen Fragen zu klären. Damit der Benutzer an möglichst vielen Stellen in den Verarbeitungsprozess eingreifen kann, müssen die einzelnen Komponenten einer NLP-Pipeline jedoch zunächst stärker modularisiert werden. Wir werden uns in Zukunft verstärkt auf die Definition geeigneter Ein- und Ausgabeformate konzentrieren um eine solche Modularisierung zu ermöglichen.

Literatur

[Ahmad et al. 1992] Khurshid Ahmad, Andrea Davies, Heather Fulford and Margaret Rogers (1992): "What is a term? The semi-automatic extraction of terms from text", in: Mary Snell-Hornby et al.: *Translation Studies – an interdiscipline*, John Benjamins Publishing Company (Amsterdam/Philadelphia)

[Bartsch 2004] Sabine Bartsch (2004): "Structural and functional properties of collocations in English", in: A corpus study of lexical and pragmatic constraints on lexical co-occurence, Tübingen, Narrr

[Dunning 1993] Ted Dunning (1993): "Accurate Methods for the Statistics of Surprise and Coincidence", in: Computational Linguistics, 19/1.

[Evert 2004] tefan Evert (2004): "The Statistical Analysis of Morphosyntactic Distributions", in: *Proceedings of LREC 2004 Lisbon*, Portugal, 2004, pp. 1539-1542

[Evert 2005] Stefan Evert (2005): "The Statistics of Word Cooccurrences: Word Pairs and Collocations", PhD. Thesis, Universität Stuttgart.

[Hausmann 2004] Franz Josef Hausmann (2004): "Was sind eigentlich Kollokationen?", in: Wortverbindungen – mehr oder weniger fest, DeGruyter, Berlin

[Heid et al. 2008] Ulrich Heid, Fabienne Fritzinger, Susanne Hauptmann, Julia Weidenkaff and Marion Weller (2008): "Providing corpus data for a dictionary of German juridical phraseology", in: Angelika Storrer, Alexander Geyken, Alexander Siebert and Kay-Michael Würzner: *Text Resources and Lexical Knowledge* (= Proceedings of the 9th Conference on Natural Language Processing, KONVENS 2008).

[Heid/Weller 2008] Ulrich Heid and Marion Weller (2008): "Tools for Collocation Extraction: Preferences for Active vs. Passive", in: *Proceedings of LREC-2008, Marrakesh, Morocco*

[Ivanova et al. 2008] Kremena Ivanova, Ulrich Heid, Sabine Schulte im Walde, Adam Kilgarriff and Jan Pomikálek (2008): "Evaluating a German Sketch Grammar: A Case Study on Noun Phrase Case", in: *Proceedings of LREC-2008, Linguistic Resources and Evaluation Conference, Marrakesh, Morocco*,

[Kilgarriff et al. 2004] Adam Kilgarriff, Pavel Rychlý, Pavel Smrz and David Tugwell (2004): "The Sketch Engine", in: Proceedings of EURALEX-2004, Lorient, France

[Richardson/Ruby 2007] Leonard Richardson and Sam Ruby (2007): "RESTful Web Services", O'Reilly

[Schiehlen 2003] Michael Schiehlen (2003): "A Cascaded Finite-State Parser for German", in: Proceedings of the Research Note Sessions of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL'03), Budapest, April, 2003.

[Schmid et al. 2004] Helmut Schmid, Arne Fitschen and Ulrich Heid (2004): "A German Computational Morphology Covering Derivation, Composition and Inflection", in: Proceedings of the IVth International Conference on Language Resources and Evaluation (LREC 2004).

Annotating Question Types in Social Q&A Sites

Kateryna IGNATOVA, Cigdem TOPRAK [1], Delphine BERNHARD and Iryna GUREVYCH

Ubiquitous Knowledge Processing (UKP) Lab, Computer Science Department
Technische Universität Darmstadt, Germany
www.ukp.tu-darmstadt.de

Abstract. In all domains, including eHumanities, it is crucial to understand how people seek information and what kinds of questions they ask. In this paper, we present an annotation study of domain-specific questions collected from the current leading social Question and Answer site, namely Yahoo! Answers. We define an annotation scheme with 9 question types and additional attributes to identify unclear and opinion questions. We show that annotating questions extracted from social media content is a difficult task due to errors and ambiguities in question formulations. However, we obtain good to very good inter-annotator agreement on all but one of the defined question types.

Keywords. question type, annotation study, social Q&A

1. Introduction

Information search and the ways users express their information needs in the form of questions is a fundamental issue in all domains, including eHumanities. Indeed, information search is an important tool for eHumanities researchers looking for information in their discipline-specific information repositories. It is therefore crucial to understand how people ask questions when seeking information on a given topic.

This is also highly relevant for automatic Question Answering (QA) systems. Typical QA systems rely on a question type classification which circumscribes the kind of questions that the system is able to answer. Most of the existing open-domain QA systems utilize question type classification schemes tuned to answer a restricted set of factoid, definition or list questions from the TREC (Text REtrieval Conference) or CLEF (Cross Language Evaluation Forum) QA evaluation campaigns [1]. QA systems aiming to cope with more complex user questions necessitate broader question type classifications, based on real user questions collected in an authentic setting.

In this article, we propose to use the wealth of questions available in the Yahoo! Answers (YA) social Question and Answer (Q&A) site[2] to perform a thorough study of

[1]Corresponding Author: Cigdem Toprak, UKP Lab, Technical University of Darmstadt, Hochschulstr. 10, 64289 Darmstadt, Germany; E-mail: c_toprak@tk.informatik.tu-darmstadt.de

[2]According to [2], Yahoo! Answers is the current leader in the social Q&A market

the types of questions users ask online, getting closer to realistic use cases for automatic QA systems. To this aim, we introduce a question type annotation scheme, enhanced with attributes to identify unclear and opinion questions, aiming at answering the following research questions: (i) what types of information seeking questions are actually asked in social Q&A sites, (ii) what is the proportion of domain-specific questions that entail opinionated answers, and (iii) how well written are questions asked on social Q&A sites.

2. Annotation Scheme

The goal of our annotation study is to capture several kinds of information about user questions, as shown in Figure 1.[3]

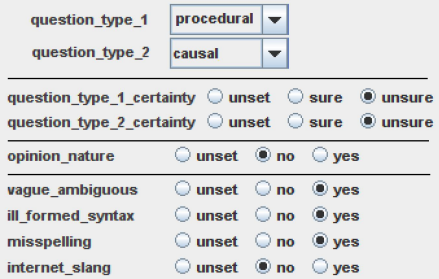


Figure 1. Example annotation of the question: *in Excel, regression problem, "Input data conatins non-numeric data"?*

As a basis for question type annotation we used the scheme developed by [3]. Our slightly modified scheme is detailed in Table 1. To account for ambiguous and multiple sentence questions, we allow the annotators to assign two question type labels to each question.

A better understanding of opinion questions is required by multi-perspective QA systems [4]. For this reason, we additionally define the binary opinion attribute to assess the user's request for opinions or suggestions on the question's topic (e.g., *what can we do for quality and improvement in higher or lower education?*)

To identify questions problematic for manual and automatic analysis, we use additional attributes to assess clarity on the semantic (*ambiguity*), syntactic (*ill-formedness*), and lexical (*slang*, *misspellings*) levels.

3. Annotation Study

The follow-up experiment involved three annotators: two students of English linguistics (later, A1 and A2) and a co-author of the paper (A3). The annotation was performed using MMAX2 [5].

[3] We deliberately kept spelling errors in the examples.

Proposed Type	Examples from Yahoo! Answers
Concept Completion	*which r websites to learn web design?*
Definition	*what is KPO (knowledge processing and out sourcing)?*
Procedural	*How do I create a risk management database?*
Comparison	*what is the difference between retesting and regression testing?*
Disjunctive	*Which one is better to use for speech recognition and image processing..C,C++,VC++ or Matlab?*
Verification	*Does the linear regression of a data set pass through the centroid of the data set?*
Quantification	*how many bytes of storage are available just using the 6800's data registers?*
Causal	*why is 0.05 used as s significant value in data analysis?*
General Information Need	*i have a hard time dealing with database management can anyone please help me?*

Table 1. The proposed question type classification scheme based on [3].

3.1. Experimental Setup

We compiled a dataset by extracting questions from the YA website using the Yahoo! Answers API[4] focusing on the domains of Data Mining, Natural Language Processing (NLP), and eLearning. The resulting dataset contained 805 questions, 50 of which appeared twice since they occurred in several categories. We did not exclude repeated questions to use them later for assessing intra-annotator agreement. We divided the dataset of 805 questions into 50 training questions and 755 questions for the annotation study.

To measure inter- and intra-annotator agreement, we use the Kappa statistic[5] [7] and, basing upon the ideas presented in [8], define two basic ways to assess agreement. **Partial Overlap (PO)** requires the agreement of the annotators on *at least one* label, i.e. partial agreement is counted as agreement. **Complete Overlap (CO)** requires the agreement on *both* labels. Furthermore, we are interested in how well the annotators agree on the question type in those cases when they are confident about their choices. Thus, we additionally calculate agreement when the *certainty* attribute is labeled as "sure" ($\mathbf{PO}_{sure}$ and $\mathbf{CO}_{sure}$).

3.2. Experimental Results

Table 2 displays the distribution and the distinguishability of the question types[6]. Almost half of the questions (46.3%) were classified as *Concept Completion*. Such a large proportion shows that it might be relevant to refine the *Concept Completion* type in fu-

[4] `http://developer.yahoo.com/answers/`

[5] $\kappa = \frac{P(A)-P(E)}{1-P(E)}$, where $P(A)$ is the observed, and $P(E)$ is the expected probability. According to [6], $\kappa > 0.8$ indicates good reliability, and $0.67 < \kappa < 0.8$ is marginally reliable.

[6] Based on the questions labeled with a single type on which all three annotators agreed (434 questions in total).

ture work. The *Definition*, *Procedural*, and *Comparison* types constituted another significant group of questions (45.9%). Surprisingly, the *Causal* why-questions proved to be quite infrequent. To assess type distinguishability, we study agreement on individual question types following the procedure proposed by [9]. The lowest κ value is obtained for the *General Information Need* type. *General Information Need* type questions are underspecified since most of them are formulated as search queries using just a set of keywords, e.g. *"Mobile database management - design?"*

Question Type	**Frequency**	**Distinguishability (κ_{PO})**
Concept Completion	46.3%	.745
Definition	20.3%	.856
Procedural	17.1%	.803
Comparison	8.5%	.911
Causal	3.2%	.638
Disjunctive	1.8%	.702
Verification	1.4%	.756
Quantification	0.9%	.747
General Information Need	0.5%	.154

Table 2. Distribution and distinguishability of question types.

	PO	**PO**$_{sure}$	**CO**	**CO**$_{sure}$
A1-A3	.852	**.875**	.617	.780
A2-A3	.837	**.874**	.617	.789
A1-A2	.775	**.800**	.683	.738
A1-A1	**.947**	.900	.679	.867
A2-A2	.878	**.900**	.878	**.900**
A3-A3	.949	**1.0**	.772	**1.0**

Table 3. Inter-/intra-annotator κ on the question type.

Agreement on the question type annotation over all categories can be found in Table 3. The kappa value is reported for the four setups defined in Section 3.1. The upper part of the table presents inter-annotator agreement for 755 questions, the lower part corresponds to intra-annotator agreement for the 50 repeated questions. The examination of inter-annotator agreement shows that all methods of assessing kappa, apart from *CO*, yield reliable or marginally reliable agreement while the best results are obtained with *PO*$_{sure}$. The intra-annotator agreement shows that the annotation is stable, i.e. annotation results do not considerably vary over time.

Only 3.8% of all questions have been marked as opinion questions by all three annotators. We obtain low inter-annotator agreement[7] for this task which is caused by two major reasons: (i) correct decisions occasionally necessitate deep domain knowledge;

[7] Values for the pairwise agreement in opinion attribute annotation: κ_{A1-A3}=0.493, κ_{A2-A3}=0.396, κ_{A1-A2}=0.267

(ii) implicit requests for opinions can be too subtle to recognize. The opinion questions identified by all annotators are all explicit requests for opinions such as: *is it desirable to use technology to support teaching and learning in campuse-based courses?* We believe that a deeper study of opinion questions is needed in order to gain a better understanding of their properties.

The analysis of the question clarity attributes shows that about 1/5 of the questions are lexically, syntactically or semantically ill-formed[8]. In order to better understand the influence of question clarity on the manual question type classification, we measured inter-annotator agreement after removal of questions labelled with at least one question clarity attribute. The best inter-annotator agreement was obtained when ambiguous and syntactically ill-formed questions were removed. The surface-level ill-formedness caused by misspellings or Internet slang proved to be less detrimental to the question type annotation than ambiguity on the semantic level.

4. Conclusions

In this paper, we presented a question type classification scheme developed to gain a better understanding of the kinds of questions people ask on social Q&A sites. We used this scheme to annotate a sample of user questions and obtained good to very good inter-annotator agreement on this task. The annotation of opinion questions proved to be more difficult and hence necessitates further investigation. Around 1/5 of the questions were lexically, syntactically or semantically ill-formed. This observation has practical consequences for automatic QA systems aiming to deal with real and complex user questions: first, they have to integrate pre-processing components to handle surface level (lexical and syntactic) errors; second, they have to help users formulate better questions in order to get better answers.

Acknowledgements

This work has been supported by the Emmy Noether Program of the German Research Foundation (DFG) under the grant No. GU 798/3-1, and by the Volkswagen Foundation as part of the Lichtenberg-Professorship Program under the grant No. I/82806.

References

[1] H. T. Dang, J. Lin, and D. Kelly, "Overview of the TREC 2006 Question Answering Track," in *Proceedings of TREC 2006*, 2006.

[2] M. Tatham, "U.S. Visits to Question and Answer Websites Increased 118 Percent Year-over-Year." [Online; visited March 26, 2008], March 19 2008. `http://www.hitwise.com/press-center/hitwiseHS2004/question-and-answer-websites.php`.

[3] A. Graesser, C. McMahen, and B. Johnson, "Question asking and answering," in *Handbook of psycholinguistics* (M. Gernsbacher, ed.), ch. 15, pp. 517–538, San Diego: Academic Press, 1994.

[4] V. Stoyanov, C. Cardie, and J. Wiebe, "Multi-Perspective Question Answering Using the OpQA Corpus," in *Proceedings of HTL-EMNLP 2005*, pp. 923–930, 2005.

[8] 16.8% are ambiguous, 20.1% are syntactically ill-formed, 8.7% contain Internet slang, 18.3% are misspelled.

[5] C. Müller and M. Strube, "Multi-level annotation of linguistic data with MMAX2," in *Corpus Technology and Language Pedagogy: New Resources, New Tools, New Methods*, pp. 197–214, 2006.
[6] K. Krippendorff, *Content Analysis: An Introduction to Methodology*. Sage Publications, Inc., 1980.
[7] J. Cohen, "A Coefficient of Agreement for Nominal Scales," *Educational and Psychological Measurement*, vol. 20, no. 1, pp. 37–46, 1960.
[8] A. Rosenberg and E. Binkowski, "Augmenting the kappa statistic to determine interannotator reliability for multiply labeled data points," in *Proceedings of HLT-NAACL 2004*, pp. 77–80, 2004.
[9] S. Teufel, A. Siddharthan, and D. Tidhar, "An annotation scheme for citation function," in *Proceedings of SIGDIAL-06*, (Sydney, Australia), 2006.

Information Extraction with RapidMiner

Felix Jungermann

Artificial Intelligence Group, TU Dortmund, http://www-ai.cs.tu-dortmund.de

Abstract. In this paper we present the Information Extraction (IE)-plugin for the open source Data Mining (DM) software *RapidMiner*[1] [Mierswa et al., 2006]. The IE-plugin can be seen as an interface between natural language and IE- or DM-methods, because it converts documents containing natural language texts into machine-readable form in order to extract interesting information like special entities and relations between those. The plugin is very modular and easy to use, which makes it more applicable for different domains and tasks.

Key words: Natural Language Processing, Information Extraction, Information Extraction System, Named Entity Recognition, Relation Extraction, Structured Methods

1 Introduction

Nowadays more and more information is available spread all over the internet. The information is present on websites – containing pure text on the one hand and html-code on the other hand –, in documents (pdf-documents for instance), or in log-files and so on. To process this daily growing huge amount of information manually is impossible.
Therefore IE-techniques are used for *the automatic identification of selected types of entities, relations, or events in free text*, as [Grishman, 2003] says.
While some IE-systems process IE-tasks like for instance Named Entity Recognition (NER) in a somehow black-boxed way, we present a very modular system, which can easily be adjusted and extended for already known or new tasks.
In the following section we give a short overview of the history of IE. In section 3 we present the state-of-the-art DM software *RapidMiner*. In section 4 the special aspects of natural language processing and how they are respected in *RapidMiner* will be explained. Some principles concerning automatic IE are defined in section 5. The IE-plugin and an exemplary text-handling task will be presented in section 6. Finally section 7 will give a conclusion about current efforts and future work.

2 Information Extraction

Beside the definition of [Grishman, 2003], [Cardie, 1997] defines IE as a kind of summarization of texts according to some (predefined) topic or domain. Both

[1] http://www.Rapid-I.com

definitions aim at a deeper text understanding.
But before one gets to know how to gain more knowledge from unknown texts, one should have a look at the history of IE.

The field of IE was born together with the Message Understanding Conference (MUC) [Grishman and Sundheim, 1996] which were scientific, shared tasks, offering texts of a special domain. Participants in these tasks had to answer specific questions concerning several special mentions in the text. One of the first domains for example was 'terrorist acts'. The questions to answer were 'Who was the actor?', 'Who was the victim?', 'What happened?', and so on.
Answering these questions automatically is a non-trivial task. Because of that, many systems were handcrafted and contained a huge amount of manually designed rules, resulting in systems biased towards a specific domain.
Using these systems for other domains was impossible or very unattractive, because much human effort had to be invested to tune all the rules manually.
The understanding of the semantic of texts was not very deep, and therefore tasks were defined that need to be processed in every domain for a deeper understanding: namely *co-reference analysis*, *word sense disambiguation* and *predicate-argument structure*.
Especially a new task of the sixth MUC – namely NER – got very popular. In its original form, the task was to extract predefined (very basic) named entities like person names, locations and organizations.
The goal to create a new task was successful, but NER is in no way domain-independent, because every special domain needs special entities to be extracted.
A very popular application area of NER is the biological domain which uses NER for identifying genes, proteins and so on. That task is called 'bio-entity recognition', like in the BioNLP of 2004 ([Kim et al., 2004]).

Those entities are the first semantic milestones on the way to a better, automatic text understanding.
But, effective – domain- and language-independent – systems should not be based on handcrafted linguistic rules. Although manually tuned systems are sometimes better in terms of precision and recall (see section 5.3), machine-learning or statistical methods are used, because they are more flexible and do not rely on heavy human efforts.
Machine-learning systems used for NER are hidden Markov models [Rabiner, 1989], maximum entropy Markov models [McCallum et al., 2000], conditional random fields [Lafferty et al., 2001] and structured support vector machines [Tsochantaridis et al., 2005].

3 RapidMiner

In this section we will give a short overview about the concepts of traditional DM. The opensource DM-software *RapidMiner* – one of the best DM tools [2] –

[2] according to the `www.KDnuggets.com` poll 2007

is presented as an exemplary environment for DM-tasks.

One can say that Knowledge Discovery in Databases (KDD) is the whole field of research of extracting new and interesting knowledge out of (masses of) information, which in addition might be very unstructured. The term KDD is often used equivalently in one context with the term of DM, which describes the techniques used to extract that knowledge. Because the amount of information is often huge, the techniques must be scalable to work efficiently.
Traditionally this extraction was done on databases containing related entities. These relations are decomposed and put into a flat form in order just to have one table to learn from.
As it is the same convention in *RapidMiner* from now on this table is called exampleset, and this exampleset, again, contains examples. In addition, examples consist of attributes. The special *label*-attribute is the one to predict from unlabeled new examples. Table 1 shows such an exampleset. In this exampleset the label is a binary one.

Table 1. Possible exampleset in *RapidMiner*

$Example$	$Attribute_1$	$Attribute_2$	...	$Attribute_m$	$Label$
...	...	...	...	...	...
$example_{n-1}$	0.1	true	...	'B'	1
$example_n$	0.2	false	...	'Re'	-1
$example_{n+1}$	0.2	true	...	'Bc'	1
$example_{n+2}$	0.05	true	...	'BB'	1
...	...	...	...	...	...

3.1 Process-view

The process-view (see Figure 1) of *RapidMiner* offers a modular and pipelined view on the processed experiment. The process normally consists of four stages:

- the input-stage,
- the preprocessing-stage,
- the learning-stage,
- and the evaluation-stage.

These stages correspond to the CRISP-model ([Chapman et al., 1999]) which is a well-known standard for DM processes.

4 Information Extraction aspects

Remembering section 2, one sees that IE can be seen as a special case of traditional data mining.

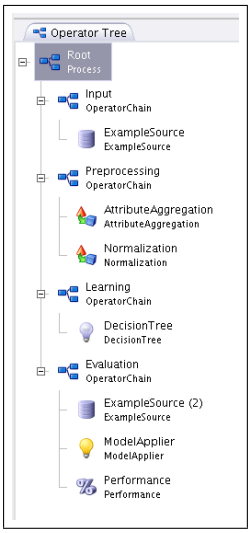


Fig. 1. The *RapidMiner* process-view

Examples will contain the tokens of the text , whereas the whole document(-collection) can be seen as the exampleset. The attributes are syntactic or semantic pieces of the tokens, and the label is the special semantic information one likes to predict.
One of the most important aspects in IE is the sequential character of texts. For the categorization of documents it is often sufficient to just use the so called *bag of words*, which is a simple index list of the contained words, to classify an unknown document – having seen enough classified documents during the training phase. But for extracting relevant information, IE needs the words itself and its contexts, e.g. to predict if it is an entity or not.
[McDonald, 1996] describes external and internal evidence as necessary to extract the correct semantics for a given word. The fact that *RapidMiner* is a DM software is remarkable because in 'traditional' DM a special example (and its attributes) originally is independent of the other examples – although they might likely share patterns. For IE all documents have to be converted to proper examples first. During conversion it is necessary to maintain the sequential character of documents and especially of sentences. Standard ML operators like sampling have to be adapted to meet the needs of the plugin in order to not destroy the structural information of the language constructs.
One way to maintain such structural characteristics is to associate words and sentences with identifiers, usually by using counter variables. Table 2 shows a possible exampleset for IE-tasks. For a better understanding the reader should have a look at section 6 which shows an example process for IE.

Table 2. Possible IE-exampleset in *RapidMiner*

Token	*Doc.No.*	*Sent.No.*	*TokenNo.*	$Attribute_1$	$Attribute_2$	...	$Attribute_m$	*Label*
...	...	...	...	...	...	...	...	...
goes	2	4	2	'g'	's'	...	aaaa	O
to	2	4	3	't'	'o'	...	aa	O
Hamburg	2	4	4	'H'	'g'	...	Aaaaaaa	LOC
because	2	4	5	'b'	'e'	...	aaaaaaa	O
...	...	...	...	...	...	...	...	...

5 Automatic-Information Extraction

In this section we explain two of the most important tasks in IE and their techniques that are present in the IE-plugin.

5.1 Named Entity Recognition (NER)

Traditionally, NER is the task to predict the best label-sequence Y for a given observation-sequence X, having seen a sufficient amount of training-pairs

$$< x^{(i)}; y^{(i)} >$$

There are some unsupervised approaches for NER (which do not need masses of training-data) ([Nadeau et al., 2006], [Collins and Singer, 1999],
[Etzioni et al., 2005]) but they heavily rely on seed-examples and knowledge about the wanted Named Entity (NE)s.
The current state-of-the-art technique for NER are CRF we will explain now in more detail.

Conditional Random Fields (CRF) CRFs are undirected graphical models and have first been proposed by [Lafferty et al., 2001].

Definition 1. *Let $G = (V, E)$ be a graph such that $Y = (Y_v)_{v \in V}$, so that Y is indexed by the vertices of G. Then (X, Y) is a CRF, when conditioned on X, the random variables Y_v obey the Markov property with respect to the graph:*

$$p(Y_v|X, Y_w, w \neq v) = p(Y_v|X, Y_w, w \sim v)$$

where $w \sim v$ means that w and v are neighbors in G.

The generated label-sequence is calculated on the whole graph conditioned on the complete observation-sequence. The probability of the graph is defined via two kinds of features:

- transition-features: $f(e, y|_e, x)$ defined on the edges of the graph and
- state-features: $g(v, y|_v, x)$ defined on the vertices.

In the simplest and most important example for modeling sequences, G is a simple chain or line:

$$G = (V = \{1, 2, ..., m\}, E = \{(i, i+1)\})$$

If the graph is a tree one can write the conditional distribution of Y and X as follows:

$$p_\theta(y|x) \propto exp\left(\sum_{e \in E,k} \{\lambda_k f_k(e, y|_e, x)\} + \sum_{v \in V,k} \{\mu_k g_k(v, y|_v, x)\}\right)$$

Using the log-likelihood we can derive the following optimization function $\mathcal{L}(\lambda)$ as:

$$\mathcal{L}(\lambda) = \sum_i \left(log\ \frac{1}{Z(x^{(i)})} + \sum_k \lambda_k F_k(y^{(i)}, x^{(i)})\right)$$

The goal is to optimize λ which is the weight-vector for the CRF-features such that $\mathcal{L}(\lambda)$ is maximized. This optimization can be done by using quasi-newton optimization (e.g. L-BFGS, [Liu and Nocedal, 1989]).

5.2 Relation Extraction (RE)

If the task of finding the entities is sufficiently done, one can look for relations between these entities. The scientific field of finding relations between entities has become popular since the ACE-tasks[3]. Especially the relation detection and classification task in 2004 [LDC, 2004] has heavily been worked on. If all entities in a sentence have been found, every possible pair of two entities is combined to a relation-candidate in order to find out wether there is a relation and to predict the corresponding relation type.

Composite Kernel The state-of-the-art techniques in this field of research are treekernel-methods based on the research done by [Haussler, 1999] and [Collins and Duffy, 2001]. A recent extension is the combination of treekernels on the one hand with linear kernels on the other hand, resulting in a so called composite kernel. The composite kernels have shown to achieve better results than with just one of these kernels ([Zhang et al., 2006], [Zhang et al., 2007]). These techniques are heavily based on structural information (parse-trees). Therefore the flat data-structure of *RapidMiner* is not well-suited for this task. Nevertheless treekernels (and composite kernels) are available in the IE-plugin.
A treekernel proposed by [Collins and Duffy, 2001] is an efficient way to compare

[3] `http://projects.ldc.upenn.edu/ace/`

two trees:

$$K(T_1, T_2) = \sum_i (\sum_{n_1 \in N_1} I_{subtree_i}(n_1))(\sum_{n_2 \in N_2} I_{subtree_i}(n_2))$$
$$= \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} \Delta(n_1, n_2)$$

where T_1 and T_2 are trees and $I_{subtree_i}(n)$ is an indicator-function that returns 1 if the root of subtree i is at node n. $\Delta(n_1, n_2)$ represents the number of common subtrees at node n_1 and n_2. The number of common subtrees finally represents a syntactic similarity measure and can be calculated recursively starting at the leaf nodes in $O(|N_1||N_2|)$.
The composite kernel combines a treekernel and a linear kernel.
[Zhang et al., 2006] therefore present linear combination for instance:

$$K_1(R_1, R_2) = \alpha \hat{K_L}(R_1, R_2) + (1 - \alpha)\hat{K}(T_1, T_2)$$

where R_1 and R_2 are relation-candidates. Tuning the α-value changes the influence of the specific kernels.

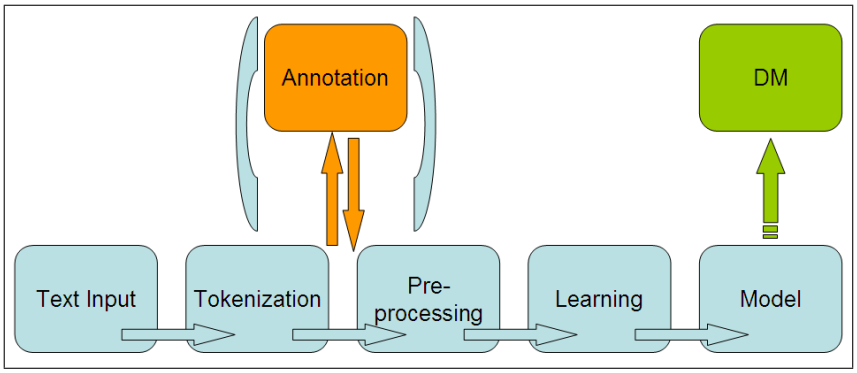


Fig. 2. Schema of the IE-plugin processing

5.3 Evaluation measures

For the evaluation of results for IE-tasks, precision, recall and f-measure are widely used. They are defined as follows:

$$precision = \frac{|\text{correctly labeled tokens}|}{|tokens|}$$
$$recall = \frac{|\text{correctly labeled tokens}|}{|\text{labeled tokens}|}$$
$$f - measure_n = (1 + n)\frac{precision * recall}{n * precision + recall}$$

A special label is reserved for tokens which are 'not labeled'. Labeled tokens have another label but not this special one.

6 Plugin

The IE-task in the plugin is following a simple scheme (Figure 2). The major steps are described in detail in the following part.

Text input and tokenization Texts or documents are available in various forms like plain ascii-files, xml- or html-files and pdf-files for example. Some text-formats are easy to process and some need additional parsing effort. The plugin offers an input-operator which can handle ascii-, html- and pdf-files.
After loading a document, it is present as as a continuous block of text in one table cell, which is not accessible in a profitable way. As a first step tokenizers have to be used to split up the text. Up to now, a trivial sentence- and word-tokenizer is available. Figure 3 shows a html-file in *RapidMiner* after being loaded and tokenized in sentences. While tokenizing splits the texts into smaller parts the original structure is still kept available. So, if one wants to process documents on the word-level, the sentence-level (the level above) is still present (see Figure 5) and can be used for processing the features. These features basically can be extracted from the current position of the document (sentence, word, ...) and from circumfluent positions (see 6).

Annotation and visualization A visualization operator allows to view the documents and to annotate parts of them. One can select attributes which shall be visualized, in order to highlight different aspects of the text. Figure 4 shows a document with some annotated words.

Preprocessing The most important part of the IE-plugin is the preprocessing. The preprocessing-operators are used to enrich the document and its tokens with syntactical information which will later be used to extract semantic information. Tokens can be enriched with contextual information as well as they can deliver inner-token information. One should keep in mind that different tasks need different preprocessing. To use machine learning algorithms for IE, one has to enrich the documents, sentences or words with attributes extracted from themselves (internal evidence) and from their context (external evidence). The plugin offers preprocessing-operators which can be used in a very modular way. There are operators working on document-, sentence- and word-level.
Figure 5 shows a document after tokenization and preprocessing. Every word of the document is represented by one row (example) in the exampleset, so the whole document is contained in the exampleset.
The columns represent the attributes for each example (word). Here, in addition to the word itself (word_0), the prefix of the current word of length 3 (prefix_0_3) is used as an attribute.

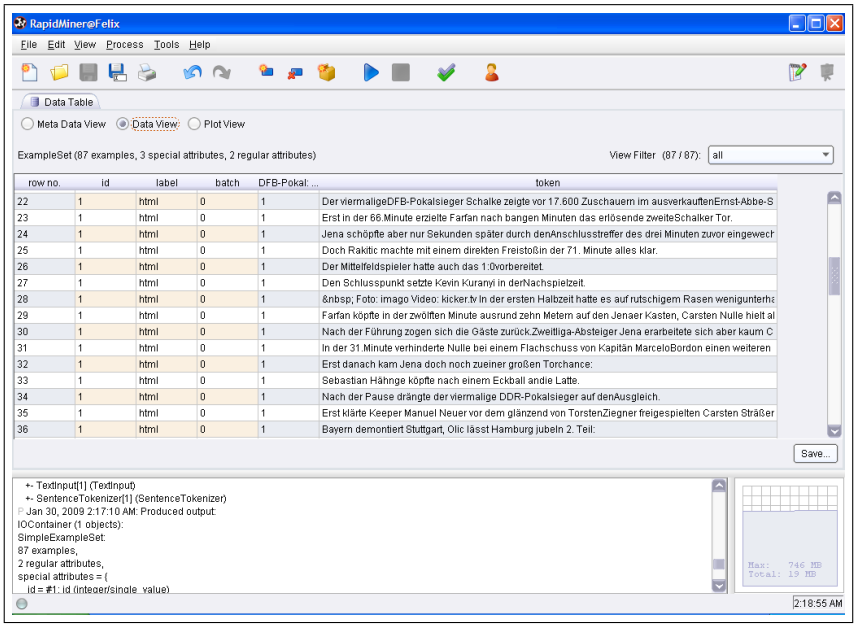


Fig. 3. The html-file after being loaded and tokenized in *RapidMiner*

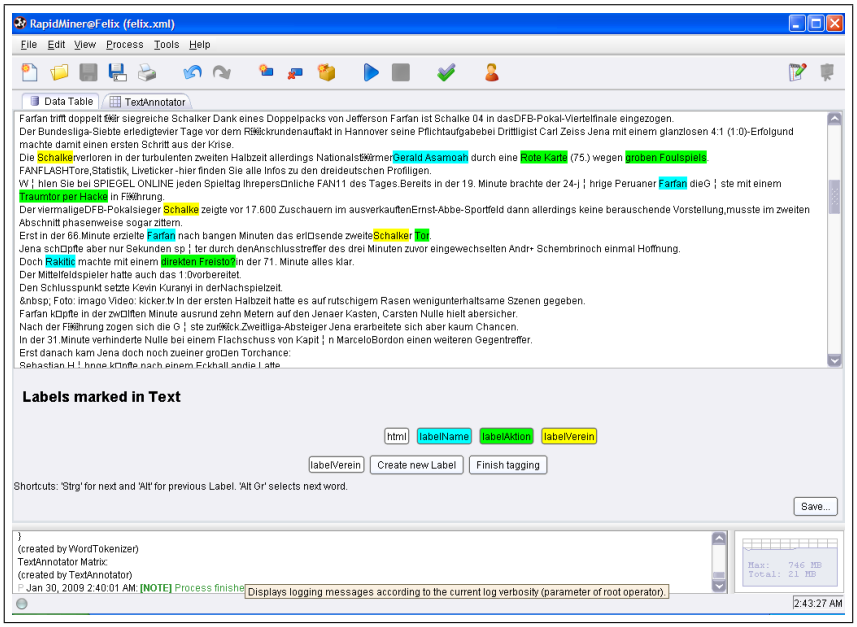
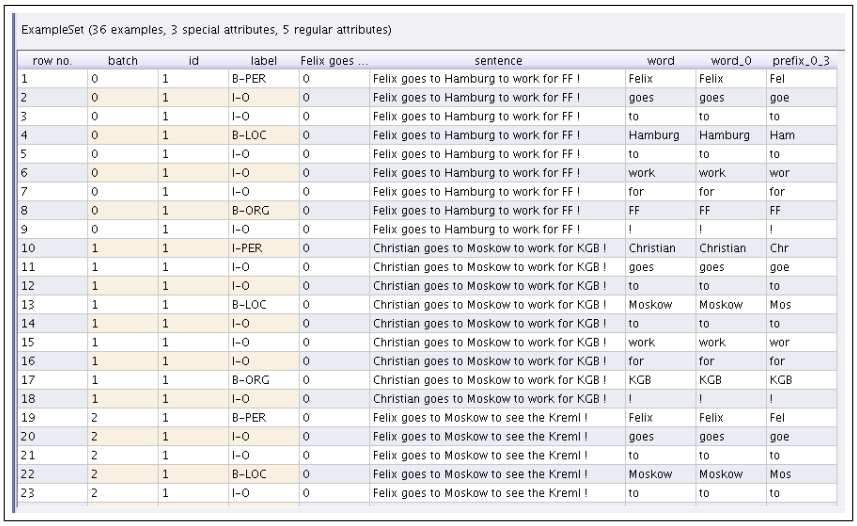


Fig. 4. The html-file after being tokenized in words and annotated



ExampleSet (36 examples, 3 special attributes, 5 regular attributes)

row no.	batch	id	label	Felix goes ...	sentence	word	word_0	prefix_0_3
1	0	1	B-PER	0	Felix goes to Hamburg to work for FF !	Felix	Felix	Fel
2	0	1	I-O	0	Felix goes to Hamburg to work for FF !	goes	goes	goe
3	0	1	I-O	0	Felix goes to Hamburg to work for FF !	to	to	to
4	0	1	B-LOC	0	Felix goes to Hamburg to work for FF !	Hamburg	Hamburg	Ham
5	0	1	I-O	0	Felix goes to Hamburg to work for FF !	to	to	to
6	0	1	I-O	0	Felix goes to Hamburg to work for FF !	work	work	wor
7	0	1	I-O	0	Felix goes to Hamburg to work for FF !	for	for	for
8	0	1	B-ORG	0	Felix goes to Hamburg to work for FF !	FF	FF	FF
9	0	1	I-O	0	Felix goes to Hamburg to work for FF !	!	!	!
10	1	1	I-PER	0	Christian goes to Moskow to work for KGB !	Christian	Christian	Chr
11	1	1	I-O	0	Christian goes to Moskow to work for KGB !	goes	goes	goe
12	1	1	I-O	0	Christian goes to Moskow to work for KGB !	to	to	to
13	1	1	B-LOC	0	Christian goes to Moskow to work for KGB !	Moskow	Moskow	Mos
14	1	1	I-O	0	Christian goes to Moskow to work for KGB !	to	to	to
15	1	1	I-O	0	Christian goes to Moskow to work for KGB !	work	work	wor
16	1	1	I-O	0	Christian goes to Moskow to work for KGB !	for	for	for
17	1	1	B-ORG	0	Christian goes to Moskow to work for KGB !	KGB	KGB	KGB
18	1	1	I-O	0	Christian goes to Moskow to work for KGB !	!	!	!
19	2	1	B-PER	0	Felix goes to Moskow to see the Kreml !	Felix	Felix	Fel
20	2	1	I-O	0	Felix goes to Moskow to see the Kreml !	goes	goes	goe
21	2	1	I-O	0	Felix goes to Moskow to see the Kreml !	to	to	to
22	2	1	B-LOC	0	Felix goes to Moskow to see the Kreml !	Moskow	Moskow	Mos
23	2	1	I-O	0	Felix goes to Moskow to see the Kreml !	to	to	to

Fig. 5. Exampleset after preprocessing

Learning The IE-plugin offers operators for NER and for RE. *RapidMiner* learners deliver so called models which equate to a function and can be applied to an (unlabeled) exampleset. Until now, for NER, the Conditional Random Fields (CRF) operator (see section 5.1) can be used. For RE the treekernel operator (see section 5.2) should be used. The underlying techniques have been described before. The implementation of these learning algorithms has been kept modular to allow the combination of various methods. Due to this modularization the CRF-learner can be combined with various optimization-methods such as quasi-newton-methods or evolutionary algorithms(particle swarm optimization for example).

DM-usage The calculated model can – of course – be used for the extraction of interesting information from unseen documents, but after having processed entities or relations or every other information one can easily build up a new exampleset containing the extracted information to gain additional knowledge.

7 Conclusion and Future Work

We presented the IE-plugin[4] for the opensource datamining software *RapidMiner*. The IE-plugin enriches the functionalities of *RapidMiner* with IE-related techniques.
The flat internal database-like data-format has the advantage of easily allowing flat and simple syntactical features to be added or removed. A very broad and

[4] the plugin should be available at `http://rapid-i.com/content/view/55/85/`

deep access to the data and the learners is possible. Because of the modular architecture it is possible to change many settings – in contrast to black-boxed systems.
A minor disadvantage of the current implementation is, that structured information can not be handled in an easy and adequate way, yet.Therefore the next steps will be to develop new internal datatypes to handle structural information in a better way. Additionally, to evaluate the performance of the implemented IE operators currently tests are running on well known datasets for RE to compare the results of the plugin with other systems.

References

[LDC, 2004] (2004). *The ACE 2004 Evaluation Plan*. Linguistic Data Consortium.

[Cardie, 1997] Cardie, C. (1997). Empirical methods in information extraction. *AI Magazine*, 18.

[Chapman et al., 1999] Chapman, P., Clinton, J., Khabaza, T., Reinartz, T., and Wirth, R. (1999). The crisp–dm process model. Technical report, The CRIP–DM Consortium NCR Systems Engineering Copenhagen, DaimlerChrysler AG, Integral Solutions Ltd., and OHRA Verzekeringen en Bank Groep B.V. This Project (24959) is partially funded by the European Commission under the ESPRIT Program.

[Collins and Duffy, 2001] Collins, M. and Duffy, N. (2001). Convolution kernels for natural language. In *Proceedings of Neural Information Processing Systems, NIPS 2001*.

[Collins and Singer, 1999] Collins, M. and Singer, Y. (1999). Unsupervised models for named entity classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

[Etzioni et al., 2005] Etzioni, O., Cafarella, M., Downey, D., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D. S., and Yates, A. (2005). Unsupervised named-entity extraction from the web: An experimental study. In *Artificial Intelligence*, volume 165, pages 91 – 134. Elsevier Science Publishers Ltd. Essex, UK.

[Grishman, 2003] Grishman, R. (2003). Information extraction. In *Handbook of Computational Linguistics Information Extraction*, chapter 30. Oxford University Press, USA.

[Grishman and Sundheim, 1996] Grishman, R. and Sundheim, B. (1996). Message understanding conference - 6: A brief history. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*.

[Haussler, 1999] Haussler, D. (1999). Convolution kernels on discrete structures. Technical report, University of California at Santa Cruz, Department of Computer Science, Santa Cruz, CA 95064, USA.

[Kim et al., 2004] Kim, J.-D., Otha, T., Yoshimasa, T., Yuka, T., and Collier, N. (2004). Introduction to the bio-entity recognition task at JNLPBA. In *Proceedings of the Joint Workshop on Natural Language Processing in Biomedicine and its Applications(JNLPBA-2004)*, Geneva, Switzerland.

[Lafferty et al., 2001] Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA.

[Liu and Nocedal, 1989] Liu, D. C. and Nocedal, J. (1989). On the limited memory method for large scale optimization. In *Mathematical Programming*, volume 45, pages 503–528. Springer Berlin / Heidelberg.

[McCallum et al., 2000] McCallum, A., Freitag, D., and Pereira, F. (2000). Maximum Entropy Markov Models for information extraction and segmentation. In *Proc. 17th International Conf. on Machine Learning*, pages 591–598. Morgan Kaufmann, San Francisco, CA.

[McDonald, 1996] McDonald, D. (1996). Internal and external evidence in the identification and semantic categorization of proper names. In Boguraev, B. and Pustejovsky, J., editors, *Corpus Processing for Lexical Acquisition*, pages 21–39. MIT Press, Cambridge, MA.

[Mierswa et al., 2006] Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., and Euler, T. (2006). YALE: Rapid Prototyping for Complex Data Mining Tasks. In Eliassi-Rad, T., Ungar, L. H., Craven, M., and Gunopulos, D., editors, *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006)*, pages 935–940, New York, USA. ACM Press.

[Nadeau et al., 2006] Nadeau, D., Turney, P. D., and Matwin, S. (2006). Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity.

[Rabiner, 1989] Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77:257–286.

[Tsochantaridis et al., 2005] Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. (2005). Large Margin Methods for Structured and Interdependent Output Variables. *Journal of Machine Learning Research*, 6:1453–1484.

[Zhang et al., 2007] Zhang, M., Che, W., Aw, A. T., Tan, C. L., Zhou, G., Liu, T., and Li, S. (2007). A grammar-driven convolution tree kernel for semantic role classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 200–207. Association for Computational Linguistics.

[Zhang et al., 2006] Zhang, M., Zhang, J., Su, J., and Zhou, G. (2006). A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings 44th Annual Meeting of ACL*, pages 825–832.

Abbreviations

CRF Conditional Random Fields
DM Data Mining
IE Information Extraction
KDD Knowledge Discovery in Databases
ML Machine Learning
MUC Message Understanding Conference
NE Named Entity
NER Named Entity Recognition
RE Relation Extraction

Computer- und korpuslinguistische Verfahren für die Analyse massenmedialer politischer Kommunikation: Humanitäre und militärische Interventionen im Spiegel der Presse

Peter Kolb[§], Amelie Kutter[§§], Cathleen Kantner[§§], Manfred Stede[§]

Zusammenfassung. *Dieser Beitrag zeigt am Beispiel eines aktuellen Forschungsprojekts auf, welche Potenziale computer- und korpuslinguistische Verfahren für eine multilinguale politikwissenschaftliche Medientext-Analyse bieten. Wir führen in die Forschungsfragen ein, die in der an der FU Berlin angesiedelten Medienanalyse zu militärischen und humanitären Interventionen bearbeitet werden, erläutern die Erfahrungen, die dabei mit vorhandenen computerlinguistischen Verfahren bisher gemacht wurden und diskutieren die Möglichkeiten, die innovative Werkzeuge der Computerlinguisten der Universität Potsdam erschließen.*

1. Einleitung

Mit der Verfügbarkeit umfangreicher elektronischer Daten eröffnet sich für Sozialwissenschaftler die Möglichkeit, ihre Fragestellungen auf größere Datenmengen anzuwenden, die Erfassung und Analyse teilweise zu automatisieren und den Aussagen damit größere (quantifizierte) Validität zu verleihen. Zugleich ergeben sich neue technische Herausforderungen für Datenmanagement und -analyse, die sich nur mit Hilfe der Computerlinguistik lösen lassen. Dieser Beitrag zeigt am Beispiel eines aktuellen Forschungsprojekts auf, welche Potenziale computer- und korpuslinguistische Verfahren für eine multilinguale politikwissenschaftliche *Medientext-Analyse* bieten. Wir führen in die Forschungsfragen ein, die in der an der FU Berlin angesiedelten Medienanalyse zu militärischen und humanitären Interventionen bearbeitet werden, erläutern die Erfahrungen, die dabei mit vorhandenen computerlinguistischen Verfahren bisher gemacht wurden und diskutieren die Möglichkeiten, die innovative Werkzeuge der Computerlinguisten der Universität Potsdam erschließen.

2. Die sozialwissenschaftliche Fragestellung

Im Zentrum der an der FU Berlin durchgeführten Medienanalyse zu militärischen und humanitären Interventionen steht die Frage, ob die unterschiedlichen nationalen Medienöffentlichkeiten in der Europäischen Union (EU) *Problemsichten* teilen, ähnliche politische Akteure – insbesondere die EU – als *Handlungsträger* sehen und ähnliche *normative Kriterien* zur Beurteilung von politischen Problemen heranziehen. Eine solche „Problemlösungsgemeinschaft“, so die grundlegende Annahme, würde die grenzübergreifende Diskussion und Entscheidungsfindung zu politischen Problemen erleichtern, die sich zunehmend auf EU-Ebene verlagern. Auch im Bereich der äußeren Sicherheit, darunter der militärischen und humanitären Interventionen, hat sich die EU als (zusätzliches) Steuerungszentrum etabliert. Auf welchen Grundlagen sie aber in Aktion treten soll ist umstritten und bedarf der politischen Kommunikation und Selbstverständigung innerhalb und zwischen den nationalen Öffentlichkeiten. Mediendebatten zu militärischen Interventionen eignen sich daher als Gegenstand, anhand dessen die mögliche Herausbildung einer transnationalen Problemlösungsgemeinschaft untersucht werden kann. Die Dekade nach dem

[§] Universität Potsdam, Institut für Linguistik, AG Angewandte Computerlinguistik, Karl-Liebknecht-Str. 24-25, 14476 Golm. {kolb|stede}@ling.uni-potsdam.de

[§] [§] Freie Universität Berlin, FB Politik und Sozialwissenschaften, Otto-Suhr-Institut für Politikwissenschaft, Arbeitsstelle Europäische Integration, Ihnestr. 22, 14195 Berlin. {kantnerc|akutter}@zedat.fu-berlin.de

Ende des Kalten Kriegs bietet sich als Untersuchungszeitraum an, da in dieser Zeit auch auf dem europäischen Kontinent gewaltsame Konflikte ausbrachen und die EU Instrumente für ein gemeinsames Management dieser Konflikte entwickelte.

Grundlage der ländervergleichenden Längsschnittanalyse ist ein bereinigtes Vollsample von 489.500 Zeitungsartikeln, die in den Jahren 1990-2006 in je zwei großen Tageszeitungen in acht Ländern erschienen. Sie wurden mit Hilfe einer komplexen Suchanfrage aus Medienarchiven wie LexisNexis und aus den Archiven einzelner Zeitungen gewonnen. Die Untersuchungsländer sind: Deutschland, Frankreich, Großbritannien, Irland, Niederlande, Österreich, Polen und – als außereuropäischer Vergleichsfall – die USA. Angesichts dieser Datenmenge ergaben sich zwei wesentliche Herausforderungen: einerseits eine automatisierte Aufbereitung der Rohdaten für ein auf die Forschungsfragen zugeschnittenes dynamisches Datenmanagement; andererseits die automatische Erfassung von semantischen Teilmengen, die entweder entfernt oder aber für nähere Analysen herangezogen werden sollten (z.B. alle fälschlicherweise erfassten Duplikate und Samplingfehler, Artikel zu Interventionen im engeren Sinne, Artikel mit EU-Referenzen etc.). Computerlinguistische Verfahren waren hierfür unerlässlich.

3. Unterstützung durch computerlinguistische Verfahren

Zwei grundlegende Methoden kamen in der ersten Phase zum Einsatz. Für die Datenaufbereitung nutzten wir die *automatische Musterkennung in (linguistisch nicht spezifizierten) Zeichenketten*, so etwa beim Parsen von Datenbank-relevanten Informationen aus den Rohdaten, für ein Vektorraum-Verfahren, das Dubletten identifizierte, sowie für die Extraktion und Kalkulation von Artikeln, die spezifische Suchworte enthielten. Dies geschah mit Hilfe der Software SPSS Clementine[1]. Zur Bestimmung inhaltlich relevanter Zeichenketten führten wir zusätzlich eine *Konkordanzanalyse* mit WordSmith[2] durch. Durch das manuelle Überprüfen der signifikanten Kollokate konnten wir bspw. bestimmen, in welcher Kombination mehrdeutige Begriffe wie „Europa“, „europäisch“, „Europäer“ eindeutig der EU zuzurechnen waren. Alle diese Verfahren erforderten jedoch viel Programmierung und inhaltliche manuelle Arbeit – sei es bei der Bestimmung, wann ein Artikel eine Dublette sei und wann eine zu berücksichtigende Wiederveröffentlichung, sei es bei der Disambiguierung von Wortgruppen oder bei der Bestimmung sämtlicher Flexionen eines Wortes. Zudem erforderten sie profunde Sprach- und Grammatikkenntnisse und manuellen Sprachvergleich in fünf Sprachen bzw. acht länderspezifischen Lexiken. Die genutzte Software bot teilweise Erleichterung (bspw. SPSS Clementine mit einem Tool zur Erkennung von „Konzepten“ – Lemmata) war aber so intransparent, dass sie für die Bedürfnisse des Projekts nicht angepasst werden konnte.

3.1 Das Text Mining Tool

In der zweiten Phase wurde dann ein erstes an der Universität Potsdam entstandenes Werkzeug für die Textmenge angepasst und erprobt: das Distributionsanalyse-System DISCO. DISCO ist Bestandteil eines „Text Mining Tools“ (TMT), das unterschiedliche Zugangsweisen zu einem Korpus in einer Anwendung integriert. TMT weist eine Client-Server-Architektur auf. Die gesamten Daten befinden sich auf einem Server-Rechner, der eine Webschnittstelle bereitstellt, über die mittels eines beliebigen Webbrowsers auf TMT zugegriffen werden kann. Dies bietet drei Vorteile: Erstens kann ein Client-Rechner von einem beliebigen Standort über das Internet auf das Tool zugreifen, zweitens spielt das jeweilige Betriebssystem, unter dem der Client läuft, keine Rolle, und drittens muss auf dem Client außer einem Webbrowser keine weitere Software installiert werden.
Das Text-Mining-System TMT umfasst drei Teilkomponenten: die Suchmaschine LiSCo, das Distributionsanalysewerkzeug DISCO und eine sogenannte „Themenmaschine“, die sich noch in der Entwicklungsphase befindet. Diese Bausteine werden im Folgenden beschrieben.

[1] http://www.spss.com/de/clementine/
[2] http://www.lexically.net/wordsmith/version4/

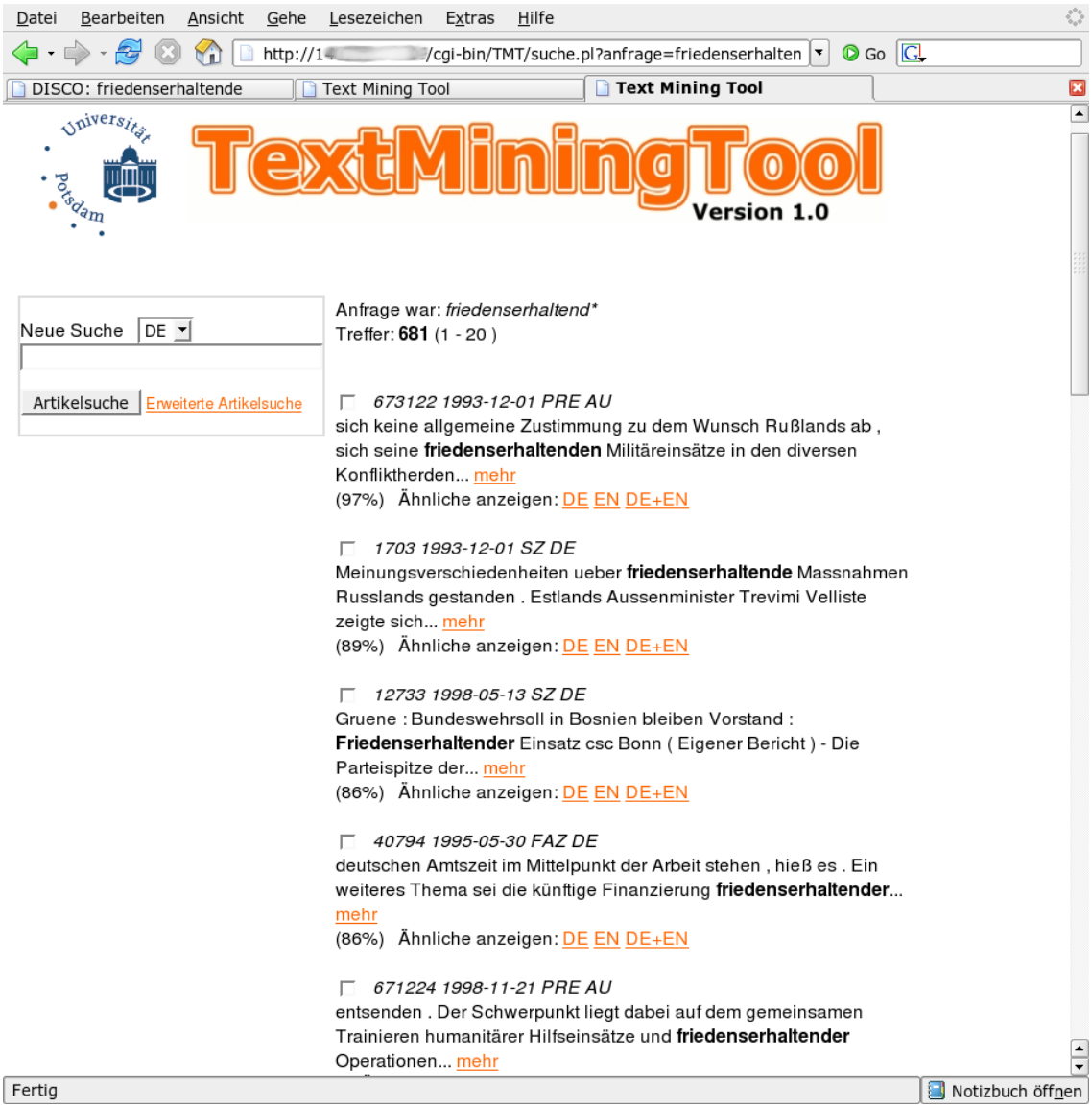


Abbildung 1: Trefferanzeige zur Suchanfrage 'friedenserhaltend'*

3.1.1 LiSCo

Die Suchmaschine LiSCo (Linguistische Suche in Corpora) indexiert das gesamte Korpus und stellt diverse Suchwerkzeuge bereit. LiSCo basiert auf dem Lucene-Index[3], einem in Java implementierten leistungsfähigen Volltextindex, der frei verfügbar ist.
Den ersten Schritt der Indexierung bildet die Aufbereitung und Vorverarbeitung des Korpus. Die Zeitungsartikel wurden mit Metadaten wie Quelle, Datum, Ursprungsland usw. versehen und in einem XML-Format gespeichert. Anschließend wurde mit Hilfe des Tree-Taggers [Schmid1995] ein PoS-Tagging und eine Lemmatisierung durchgeführt. Zum Schluss wurden alle Texte ggf. nach Unicode (UTF-8) konvertiert.
Die Zeitungsartikel wurden dann in den Lucene-Index eingelesen, wobei verschiedene durchsuchbare Felder für jedes Dokument gespeichert wurden. So kann sowohl nach den ursprünglichen Wortformen, als auch nach den Lemmata gesucht werden, außerdem nach Metadaten wie Datum, Land, Quelle usw. Der gesamte Volltext jedes Artikels wurde ebenfalls in den Index aufgenommen.

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

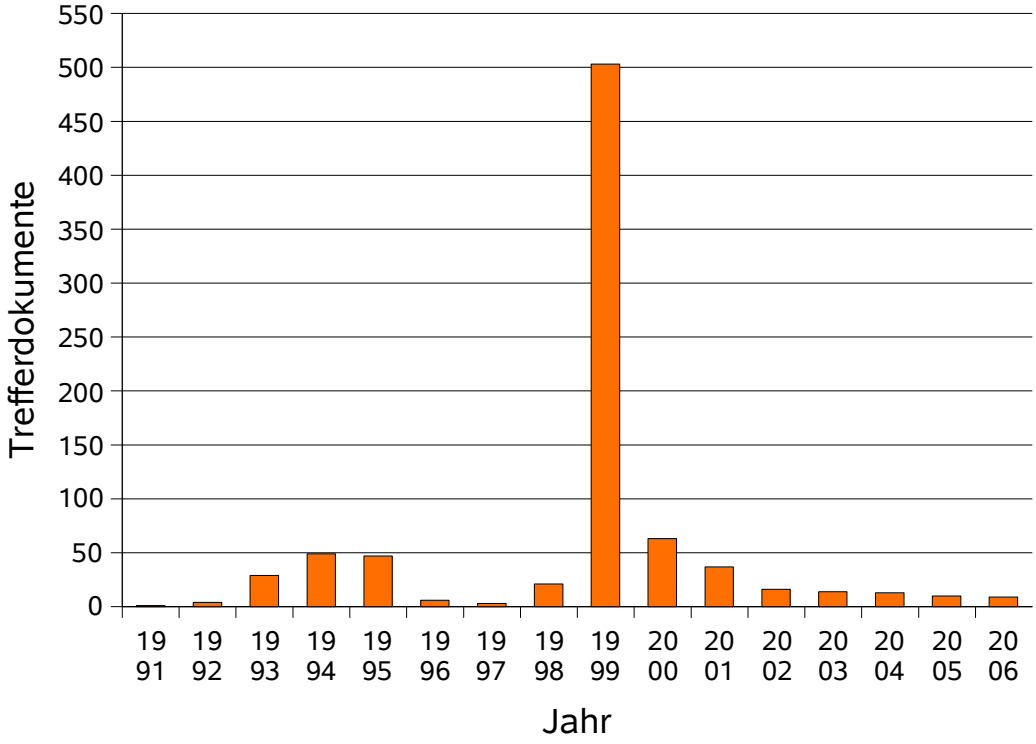


Abbildung 2: Anfrage: (Luftkrieg OR Luftangriffe OR Bombardierung) AND Nato AND (Serbien OR Serbiens)

Zur Suche im Index stellt Lucene eine leistungsfähige Abfrage-Syntax bereit. Eine Boolesche Suche mit den Operatoren AND, OR und NOT ist ebenso möglich wie eine Suche mit Wildcards, eine trunkierte Suche (*Tor** findet *Tor*, *Tore*, *Tors*, *Torpfosten*, ...), oder eine exakte Phrasensuche, bei der die Suchanfrage (wie bei Google) in doppelte Anführungszeichen eingeschlossen werden muss.
Lucene bietet außerdem die Möglichkeit, Treffer in ihrem Kontext anzuzeigen und hervorzuheben. Damit lässt sich bereits eine einfache Konkordanzanzeige bereitstellen.
Abbildung 1 zeigt die Trefferanzeige zur Suchanfrage *friedenserhaltend**. Durch einen Klick auf „mehr“ wird der Volltext des jeweiligen Artikels angezeigt.
Da zu jedem Artikel das Datum im Index gespeichert und somit abfragbar ist, konnte in einfacher Weise eine „Zeitleisten“-Funktion implementiert werden, mit deren Hilfe die zeitliche Entwicklung eines Themas in Form von Balkendiagrammen veranschaulicht werden kann. Dazu muss vom Benutzer lediglich der gewünschte Zeitraum und die zeitliche Auflösung (Monate, Jahre) ausgewählt werden. TMT führt dann automatisch eine Reihe von Anfragen aus, in der die eigentliche Suchanfrage mit den Abschnitten des gewünschten Zeitraums kombiniert und die Anzahl der Treffer protokolliert und als Tabelle gespeichert wird. Das Ergebnis kann per Mausklick entweder als Excel- oder Latex-Datei gespeichert werden. Abbildung 2 zeigt ein Beispielergebnis für den Zeitraum 1991-2006 mit jährlicher Auflösung.
Lucene implementiert neben einem Standard-Volltextindex auch das sogenannte Vektormodell [Salton1971] des Information Retrievals. Dieses Modell folgt der Annahme, dass Dokumente durch die Häufigkeiten der enthaltenen Terme charakterisiert werden. Im Gegensatz zur Booleschen Volltextsuche, bei der nur geprüft wird, ob ein Dokument ein Suchwort enthält oder nicht, wird beim Vektormodell zusätzlich die Wichtigkeit oder Relevanz des Wortes im jeweiligen Dokument berücksichtigt. Die Treffermenge kann dadurch nach Relevanz sortiert werden. Dazu speichert Lucene zu jedem Term die Auftretenshäufigkeit im jeweiligen

Dokument („Termfrequenz“ TF) sowie seine Häufigkeit in der gesamten Dokumentensammlung („Dokumentenfrequenz“ DF). Aus diesen Angaben kann das bekannte TF-IDF-Maß zur Bestimmung der Relevanz eines Terms berechnet werden. Ein Term ist dabei umso wichtiger, je häufiger er im jeweiligen Dokument vorkommt und je seltener er insgesamt in der Dokumentensammlung auftaucht. Auf dieser Grundlage haben wir ein Relevanzfeedback [Rocchio1971] und eine Suche nach inhaltlich ähnlichen Dokumenten implementiert.
Beim Relevanzfeedback kann der Benutzer die zu einem Suchergebnis angezeigten Trefferdokumente durch Anklicken als relevant oder nicht relevant bewerten, und die Anfrage dann per Mausklick wiederholen. Die Suchanfrage wird automatisch um die relevantesten Terme aus den vom Nutzer als relevant bewerteten Dokumenten erweitert. Terme aus den als irrelevant bewerteten Dokumenten werden aus der Suchanfrage entfernt. Die Terme der automatisch erzeugten Suchanfrage können ausgegeben werden.
Per Mausklick kann auch nach inhaltlich ähnlichen Dokumenten zu einem gegebenen Dokument gesucht werden (siehe Abbildung 1). Dabei wird das Ausgangsdokument durch einen Vektor seiner relevantesten Terme repräsentiert, die mit ihrem TF-IDF-Wert gewichtet sind. Dieser Vektor kann als Suchanfrage an den Lucene-Index geschickt werden, der dann die ähnlichsten Dokumente als Treffer ausgibt. Dieses Verfahren arbeitet für den einsprachigen Fall bereits sehr zufriedenstellend. Für den crosslingualen Fall, also z.B. die Suche nach englischen Dokumenten zu einem gegebenen deutschen Dokument, muss der Vektor der relevantesten Terme erst in die Zielsprache übersetzt werden. Unser erster Ansatz, einfach alle im Wörterbuch aufgeführten Übersetzungsmöglichkeiten in den Zielvektor aufzunehmen, führte zu inakzeptablen Ergebnissen, was sicher auch an der großen inhaltlichen Homogenität des Korpus liegen mag. Hier muss noch ein geeignetes Verfahren zur Disambiguierung der Übersetzungsmöglichkeiten gefunden werden. Wir planen, dafür die Kookkurrenz- bzw. distributionellen Ähnlichkeitswerte zwischen Wörtern, wie sie von DISCO geliefert werden, einzusetzen. Außerdem haben wir bereits vielversprechende erste Experimente zur automatischen Extraktion neuer Wort-Übersetzungen aus den multilingualen Korpora durchgeführt. Dabei werden die signifikanten Kookkurrenten eines Ausgangsworts der Quellsprache mit dem vorhandenen Wörterbuch in die Zielsprache übersetzt und dann mit den Kookkurrenzprofilen aller Wörter der Zielsprache verglichen. Das am Ähnlichsten verwendete Wort der Zielsprache wird dann als Übersetzung des Ausgangswortes vorgeschlagen [Rapp1999].
Ein weiteres in TMT implementiertes Suchverfahren bildet die automatische Kategorisierung von Dokumenten in ein vom Benutzer vorgegebenes Kategorienmodell. Zuerst muss vom Benutzer ein Kategorienmodell (eine Hierarchie in Form eines Baums) erstellt werden. Jede Kategorie wird durch eine Anzahl manuell ausgewählter Dokumente (sogenannter Prototypen) definiert. Eine Kategorie kann bereits durch ein einzelnes Dokument definiert werden. Die automatische Einordnung neuer Dokumente in das Kategorienmodell erfolgt über die zuvor beschriebene Ähnlichkeitssuche. Das neue Dokument wird mit allen prototypischen Dokumenten im Kategorienmodell auf Ähnlichkeit verglichen und in die Kategorie mit den ähnlichsten Dokumenten eingeordnet. Dazu wird das sogenannte *k-nearest-neighbour*-Verfahren [Sebastiani2002] eingesetzt (kNN-Verfahren). Es arbeitet folgendermaßen: das neue Dokument x wird mit allen prototypischen Dokumenten im Kategorienmodell auf Ähnlichkeit verglichen. Die k ähnlichsten Dokumente werden ausgewählt (z.B. k = 20, in Abhängigkeit von der Anzahl der Kategorien). Jedes dieser k Dokumente erhält ein "Stimmrecht", dessen Wert gleich seinem Ähnlichkeitsgrad zum Dokument x geteilt durch seinen Rangplatz in der Ergebnisliste ist. Die Stimmen von Dokumenten, die aus der gleichen Kategorie stammen, werden addiert. Diejenige Kategorie gewinnt, die die meisten Stimmen erhält. Für den Fall k = 1 gewinnt die Kategorie, in der sich das zu x ähnlichste Dokument befindet. Das kNN-Verfahren zeichnet sich durch Robustheit, Geschwindigkeit und eine gute Skalierbarkeit hinsichtlich der Kategorienanzahl aus. Ein großer Vorteil unseres Kategorisierungsverfahrens besteht darin, dass kein eigener

Trainingsschritt erforderlich ist. Dadurch können prototypische Dokumente nach Belieben zu einer Kategorie hinzugefügt oder daraus entfernt werden, und die Auswirkungen werden sofort sichtbar. Prototypen können auch zwischen Kategorien verschoben werden. Dem Benutzer kann beim Aufbau eines Kategorienmodells Hilfestellung gegeben werden. Wenn z.B. ein neues Dokument als Prototyp einer Kategorie hinzugefügt werden soll, kann sofort angezeigt werden, wenn die Ähnlichkeit zu einem prototypischen Dokument einer anderen Kategorie größer ist als zu den übrigen Dokumenten derselben Kategorie. Durch dieses unmittelbare Feedback können qualitativ wesentlich bessere Kategorienmodelle aufgebaut werden.

3.1.2 DISCO

Mit dem Distributionsanalyse-System DISCO [Kolb2008] lassen sich zu einem Suchwort die signifikanten Kookkurrenzen und die distributionell ähnlichen Wörter anzeigen (Abbildung 5). Zudem können auf Grundlage der distributionellen Ähnlichkeit Wortcluster berechnet und graphisch dargestellt werden (Abbildung 3). Die Kookkurrenzen vermitteln einen ersten Eindruck, in welchen Zusammenhängen das Suchwort im Korpus verwendet wird. Auf Basis der Kookkurrenzen berechnet DISCO die Wörter, die im Korpus eine ähnliche Distribution

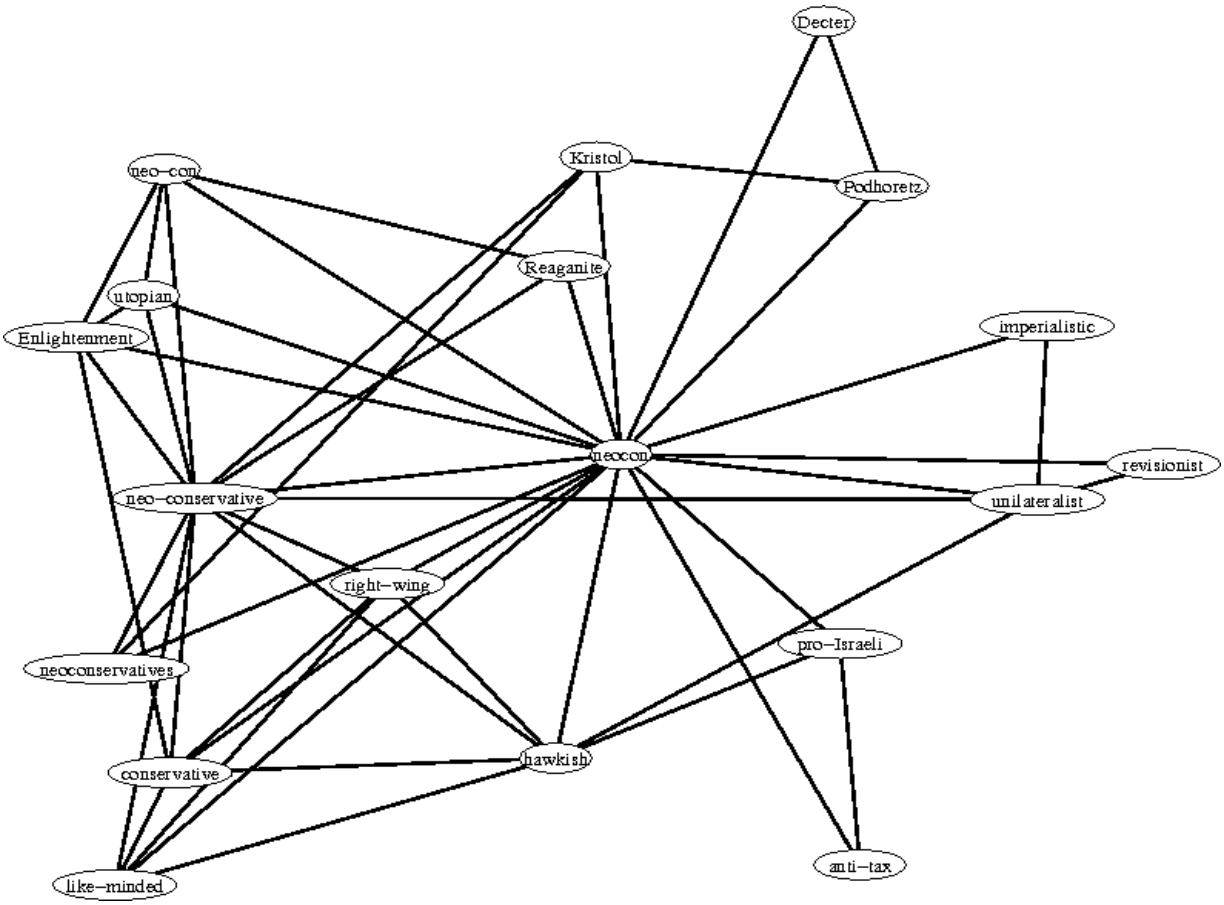


Abbildung 3: Automatisch erzeugter Graph zu 'neocon'

aufweisen. Besonders im Falle abstrakter Nomen erhält man hier semantisch ähnliche Wörter zum Ausgangswort, teilweise das ganze semantische Spektrum. Für das vorliegende, thematisch sehr spezifische Korpus eignete sich DISCO zur schnellen Identifizierung von Wortfeldern von bestimmten Abstrakta (z.B. “Intervention”, “Massenmord” etc.), und zwar in einem Umfang und Tempo, das über WordSmith nie herzustellen gewesen wäre.

3.1.3 Themenmaschine

Um die Tokenebene von DISCO zu verlassen ist geplant, im Korpus automatisch *Themen* in Form relevanter Phrasen zu identifizieren. Diese sollen mit Hilfe linguistischer Analysen zu einer Taxonomie aus Ober- und Unterthemen verknüpft werden, die graphisch als Baum dargestellt wird und navigierbar ist. Die Dokumente, in denen die Themen gefunden wurden, bilden die Blätter des Baums. Zur Unterstützung der Themenvernetzung sollen mit Hilfe syntaktischer Muster aus dem Korpus Wortkandidaten für bestimmte semantische Relationen wie Hyponymie, Meronymie etc. extrahiert werden [Hearst1992]. Außerdem können die von DISCO gefundenen ähnlichen Wörter zur weiteren Vernetzung „assoziierter" Themen genutzt werden.
Das Verfahren zur Themenextraktion arbeitet wie folgt. Durch einen auf dem vorhergehenden PoS-Tagging aufbauenden Analyseschritt werden zunächst Nominalphrasen erkannt. Zum Beispiel werden aus dem Satz

Außenminister Joschka Fischer fordert militärische Intervention

die Phrasen *Außenminister Joschka Fischer* und *militärische Intervention* extrahiert. Im anschließenden Normalisierungsschritt werden verschiedenartige Phrasen auf eine gemeinsame Form gebracht. Dadurch können inhaltliche Übereinstimmungen zwischen unterschiedlich formulierten Textabschnitten erkannt werden. Beispielsweise würden die drei Phrasen

militärische Intervention, Intervention des Militärs, Militärintervention

alle zu *Intervention Militär* normalisiert. Dies erfordert (zumindest im Deutschen) den zusätzlichen Einsatz eines morphologischen Analyseschrittes zur Zerlegung von Komposita in ihre Konstituenten. Als nächstes werden Phrasen und Teilphrasen zusammengefasst. Hierbei würde eine Phrase wie *Bundesaußenminister Fischer* mit der oben aufgeführten Phrase *Außenminister Joschka Fischer* identifiziert werden. Durch diese Art der Zusammenfassung ähnlich formulierter Ausdrücke über den ganzen Text hinweg wird erreicht, dass sich die anschließende Relevanzberechnung deutlich verbessert. In diesem folgenden Arbeitsschritt werden mittels statistischer Auswertung aus Phrasen Themen. Nur diejenigen Phrasen, die einen bestimmten Relevanzwert erreichen, gelangen in den Themenindex. Die Relevanz einer Phrase ist umso größer, je öfter sie oder ihre Teilphrasen im aktuellen Dokument vorgekommen sind, und je niedriger, je öfter die Phrase insgesamt in der indexierten Dokumentensammlung vorgekommen ist. Das Modul Themensuche ermöglicht, dass die Dokumente thematisch erfasst werden. Die Informationen werden dadurch thematisch verknüpft, wie nachfolgende Darstellung veranschaulicht.

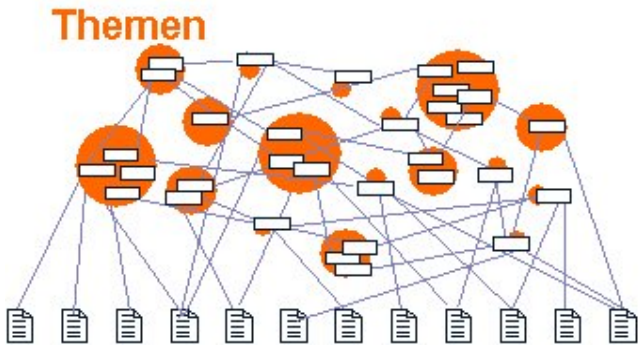


Abbildung 4: Verknüpfung von Themen und Dokumenten

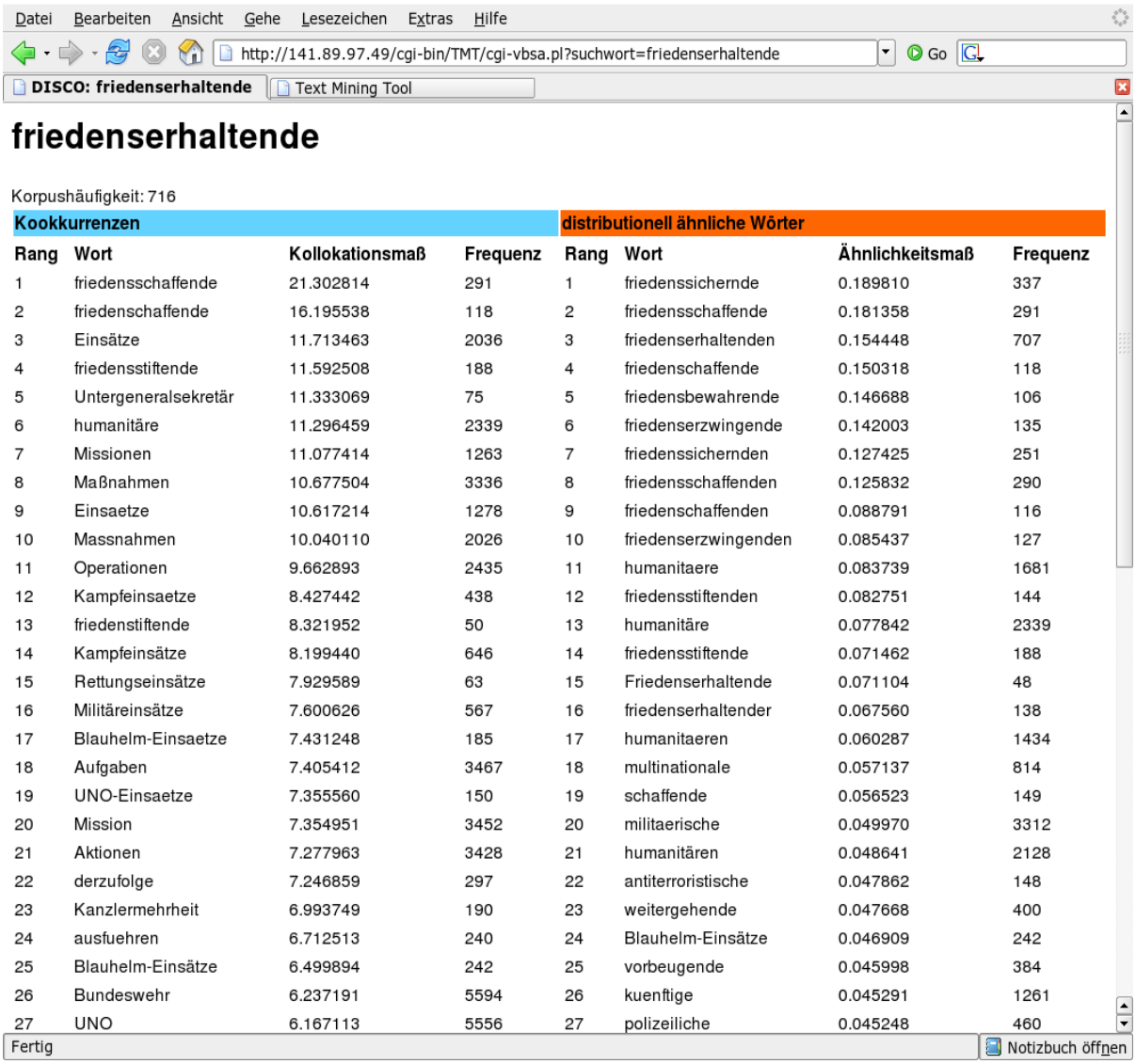
Dieser Prozess vollzieht sich kontinuierlich. Jedes eingehende Dokument wird automatisch nach seinen Themen analysiert und die Verknüpfungen zu den bereits vorhandenen Themen hergestellt. Neue Themen werden erkannt und aufgenommen.

Wir hoffen, dass die Navigation im Themenbaum die mühsame Inspektion von Konkordanzen zur Validierung der jeweils aktualisierten Bedeutung ersetzen können wird.

3.2 Beispielhafte Anwendungsfälle

3.2.1 Erstellung komplexer Suchanfragen mittels semantischer Felder

Eine zu lösende Aufgabe ist die Erstellung von Suchanfragen, mit denen aus dem Korpus diejenigen Zeitungsartikel abgerufen werden können, die sich mit bestimmten, interessierenden Fragestellungen oder Themen befassen. Hierbei kommt es auf eine gleichzeitige Maximierung von Präzision und Vollständigkeit („Recall“) der Suchergebnisse an, damit anschließende quantitative Aussagen auf einer gesicherten Grundlage ruhen. Am Beispiel des Konzepts 'friedenserhaltende Intervention' soll gezeigt werden, wie DISCO bei der Formulierung einer komplexen Suchanfrage helfen kann.
Als erste Suchanfrage dient der Name des Konzepts selbst, also *friedenserhaltende Intervention*.



friedenserhaltende

Korpushäufigkeit: 716

Kookkurrenzen				distributionell ähnliche Wörter			
Rang	Wort	Kollokationsmaß	Frequenz	Rang	Wort	Ähnlichkeitsmaß	Frequenz
1	friedensschaffende	21.302814	291	1	friedenssichernde	0.189810	337
2	friedenschaffende	16.195538	118	2	friedensschaffende	0.181358	291
3	Einsätze	11.713463	2036	3	friedenserhaltenden	0.154448	707
4	friedensstiftende	11.592508	188	4	friedenschaffende	0.150318	118
5	Untergeneralsekretär	11.333069	75	5	friedensbewahrende	0.146688	106
6	humanitäre	11.296459	2339	6	friedenserzwingende	0.142003	135
7	Missionen	11.077414	1263	7	friedenssichernden	0.127425	251
8	Maßnahmen	10.677504	3336	8	friedensschaffenden	0.125832	290
9	Einsaetze	10.617214	1278	9	friedenschaffenden	0.088791	116
10	Massnahmen	10.040110	2026	10	friedenserzwingenden	0.085437	127
11	Operationen	9.662893	2435	11	humanitaere	0.083739	1681
12	Kampfeinsaetze	8.427442	438	12	friedensstiftenden	0.082751	144
13	friedenstiftende	8.321952	50	13	humanitäre	0.077842	2339
14	Kampfeinsätze	8.199440	646	14	friedensstiftende	0.071462	188
15	Rettungseinsätze	7.929589	63	15	Friedenserhaltende	0.071104	48
16	Militäreinsätze	7.600626	567	16	friedenserhaltender	0.067560	138
17	Blauhelm-Einsaetze	7.431248	185	17	humanitaeren	0.060287	1434
18	Aufgaben	7.405412	3467	18	multinationale	0.057137	814
19	UNO-Einsaetze	7.355560	150	19	schaffende	0.056523	149
20	Mission	7.354951	3452	20	militaerische	0.049970	3312
21	Aktionen	7.277963	3428	21	humanitären	0.048641	2128
22	derzufolge	7.246859	297	22	antiterroristische	0.047862	148
23	Kanzlermehrheit	6.993749	190	23	weitergehende	0.047668	400
24	ausfuehren	6.712513	240	24	Blauhelm-Einsätze	0.046909	242
25	Blauhelm-Einsätze	6.499894	242	25	vorbeugende	0.045998	384
26	Bundeswehr	6.237191	5594	26	kuenftige	0.045291	1261
27	UNO	6.167113	5556	27	polizeiliche	0.045248	460

Abbildung 5: Ausgabe von DISCO zum Suchwort "friedenserhaltende"

Um die Präzision hoch zu halten, wird nach der exakten Phrase gesucht, indem (wie bei Google) die Suchwörter in doppelte Anführungszeichen gesetzt werden. Textstellen mit *friedenserhaltenden Mission* oder *friedenserhaltende militärische Mission* werden also nicht

gefunden. Die erste Suchanfrage, „*friedenserhaltende Mission*“ ergibt 14 Treffer auf dem deutschsprachigen Datenbestand. Um die Vollständigkeit der Treffermenge zu erhöhen, wird in DISCO jetzt nach ähnlich gebrauchten Wörtern zum Term *friedenserhaltende* gesucht. Das Ergebnis der DISCO-Anfrage ist in Abbildung 5 dargestellt. Die distributionell ähnlichen Wörter wie *friedenssichernde*, *friedensschaffende*, *humanitäre* usw. können in die Anfrage aufgenommen werden, entsprechend wird mit dem zweiten Suchterm verfahren. Zum Suchwort *Intervention* liefert DISCO die folgenden Wörter unter den ersten 30 distributionell ähnlichsten Wörtern:

> *Eingreifen Militäraktion Invasion Militärintervention Einmischung Einsatz Militärschlag Angriff Mission Vorgehen Operation Gewaltanwendung Einmarsch Engagement ...*

Damit lässt sich durch eine Disjunktion (per booleschem Operator OR) aller möglichen Kombinationen der Suchterme eine komplexe Suchanfrage mit stark erhöhtem Recall der Treffermenge formulieren:

> *„friedenserhaltende Intervention“ OR „friedenssichernde Intervention“ OR „friedensschaffende Intervention“ OR ... OR „humanitäre Mission“*

3.2.2 Schreibvarianten von Namen

Ein häufiges Problem, das die Vollständigkeit von Treffermengen negativ beeinflusst, ist die große Varianz bei der Schreibung insbesondere fremdsprachiger Namen. So finden wir in unserem englischsprachigen Teilkorpus zur Anfrage *Arafat* 12.610 Trefferdokumente. Um die Präzision sicherzustellen, soll aber nach dem vollständigen Namen gesucht werden. Die DISCO-Anfrage mit dem Suchwort *Arafat* liefert den richtigen Vornamen, inklusive mehrerer im Korpus verwendeter Schreibvarianten:

> Kookkurrenzen: *Fatah exiling Yasser PLO YASSIR Palestinian Barak Qureia Kanafani Yasir Yassar Peres ...*
> distributionell ähnliche: *Yasser Yasir Abbas Yassir PLO Dahlan Fatah Netanyahu Peres Rabin Erekat Qureia Mazen Barak ...*

Während die Suche nach „*Yasser Arafat*“ lediglich zu 6.607 Trefferdokumenten führt, ergibt die komplexe Suchanfrage

> *„Yasser Arafat“ OR „Yasir Arafat“ OR „Yassir Arafat“ OR „Yassar Arafat“ OR (Arafat AND (PLO OR Palestine OR Palestinian))*

12.444 Treffer und ist somit hinsichtlich Präzision und Vollständigkeit optimal.
Zudem kann man mittels DISCO schnell erkennen, dass es für die Schreibung des Nachnamens *Arafat* keine gängigen Varianten gibt – unter den 50 distributionell ähnlichsten Wörtern zu *Arafat* findet sich nämlich kein ähnlich geschriebener Kandidat. Eine Suchanfrage nach der am plausibelsten erscheinenden Variante *Arrafat* ergibt in der Tat nur einen einzigen Treffer (in ca. 300.000 Dokumenten).
Eine weitergehende Automatisierung des Aufdeckens von Namensvarianten wäre durch eine Filterung der Ähnlichkeitslisten mit einem String-Ähnlichkeitsmaß wie z.B. dem Levenshtein-Editierabstand in einfacher Weise realisierbar. Solche Äquivalenzklassen würden dann den ersten Schritt zu einem automatischen Ontologieaufbau [Cimiano et al. 2006] oder zur Erweiterung einer vorhandenen Ontologie bilden. Die durch die Äquivalenzklassen identifizierten Konzepte würden mit Hilfe der in Abschnitt 3.1.3 beschriebenen Verfahren in eine Taxonomie oder Ontologie eingeordnet werden.

4. Zwischenbilanz und Ausblick

Für die Erfassung von semantischen Feldern zu bestimmten inhaltlichen Konzepten (institutionelle Akteure, Europa), die aus politikwissenschaftlicher Sicht interessierten, eignete sich die gegenwärtige Version von DISCO nur bedingt, weil lediglich ähnlich verwendete und kookkurrierende Wörter (zu "Deutschland" bspw. "Frankreich", zu "Europa" "Asien") erfasst werden, nicht aber semantische Untergruppen wie "Bundesregierung", "Auswärtiges Amt" etc. Eine weitere Einschränkung von DISCO ist, dass es lediglich tokenbasiert arbeitet und daher keine Mehrwortausdrücke gesucht werden können. Auch ist mit DISCO nicht unmittelbar festzustellen, ob durch ein Wort im Korpus stets der gerade gesuchte Akteur benannt wird. Beispielsweise bezieht sich ein Vorkommen von "Bundeskanzler" nicht immer auf Deutschland als außenpolitischen Akteur. Um sicherzustellen, dass sich bestimmte Vorkommen einzelner Wörter regelmäßig auf das gesuchte semantische Feld bzw. den dadurch bezeichneten Akteur beziehen, ist eine Untersuchung der einzelnen Belegstellen notwendig, was derzeit nur durch eine manuelle Konkordanzanalyse geleistet werden kann.

Im Text Mining wie bei der Inhaltsanalyse besteht eine grundsätzliche Kluft zwischen der konzeptuellen Ebene einerseits und der textlich-lexikalischen Ebene andererseits. Existiert eine manuell erstellte Ontologie aus interessierenden Konzepten, d.h. abstrakten semantischen Einheiten, müssen diese in konkret formulierten sprachlichen Ausdrücken wiedergefunden werden. Texte müssen also erst einmal mit Konzepten annotiert werden. Dabei treten die schon aus dem Information Retrieval bekannten Probleme der Lesartenambiguität und Paraphrase auf, d.h. ein Korpus mit einer vorhandenen Ontologie zu annotieren ist keineswegs trivial. Wir erwarten, mit einem korpusgetriebenen Ontologieaufbau mittels DISCO dieses Problem entschärfen zu können.

Ein alternativer Ansatz beseht darin, textbasiert vorzugehen und aus relevanten Phrasen im Text einen "Themenbaum" aufzubauen. Hier wird versucht, von der konkreten Formulierung im Text zu einer semantischen, möglichst abstrakten Darstellung zu gelangen. Die Schwierigkeit ist dabei, einen hinreichenden Abstraktionsgrad zu erreichen, um gleichbedeutende, aber unterschiedlich ausgedrückte Inhalte überhaupt aufeinander beziehen zu können. Außerdem ist zu erwarten, dass es sich bei den automatisch extrahierten Themen nicht unbedingt um Konzepte der Art handelt, die einer intellektuell erstellten Ontologie entsprechen. Wir hoffen, durch die kombinierte Verwendung der verschiedenen Ansätze die Kluft zwischen konzeptueller und textueller Ebene überbrücken zu können.

Literatur

P. Cimiano, J. Völker und R. Studer (2006): Ontologies on Demand? - A Description of the State-of-the-Art, Applications, Challenges and Trends for Ontology Learning from Text. *Information, Wissenschaft und Praxis*, 57(6-7), S. 315-320.

M. Hearst (1992): Automatic acquisition of hyponyms from large text corpora. *COLING 1992*, Nantes, Frankreich, S. 539—545.

P. Kolb (2008): DISCO: A Multilingual Database of Distributionally Similar Words. *Tagungsband der 9. Konferenz zur Verarbeitung natürlicher Sprache – KONVENS 2008*, Berlin.

R. Rapp (1999): Automatic Identification of Word Translations from Unrelated English and German Corpora. *Proceedings of ACL*, College Park, Maryland, S. 519–526

J.J. Rocchio (1971): Relevance Feedback in Information Retrieval. In G. Salton (Hrsg.): *The SMART Retrieval System – Experiments in Automatic Document Processing*. Prentice-Hall.

G. Salton (1971): *The SMART Retrieval System – Experiments in Automatic Document Processing*. Prentice-Hall.

H. Schmid (1995): Improvements in Part-of-Speech Tagging with an Application to German. *Proceedings of the ACL SIGDAT-Workshop,* S. 47—50.

F. Sebastiani (2002): Machine learning in automated text categorization. *ACM Computing Surveys*, 34, S. 1—47.

eHumanities Desktop — eine webbasierte Arbeitsumgebung für die geisteswissenschaftliche Fachinformatik

Alexander Mehler[1], Rüdiger Gleim[1], Ulli Waltinger[2], Alexandra Ernst[2], Dietmar Esch[2] & Tobias Feith[2]

[1] Goethe–Universität Frankfurt am Main
[2] Universität Bielefeld

Zusammenfassung In diesem Beitrag beschreiben wir den *eHumanities Desktop*[3]. Es handelt sich dabei um eine rein webbasierte Umgebung für die texttechnologische Arbeit mit Korpora, welche von der standardisierten Repräsentation textueller Einheiten über deren computerlinguistische Vorverarbeitung bis hin zu *Text Mining*–Funktionalitäten eine große Zahl von Werkzeugen integriert. Diese Integrationsleistung betrifft neben den Textkorpora und den hierauf operierenden texttechnologischen Werkzeugen auch die je zum Einsatz kommenden lexikalischen Ressourcen. Aus dem Blickwinkel der geisteswissenschaftlichen Fachinformatik gesprochen fokussiert der Desktop somit darauf, eine Vielzahl heterogener sprachlicher Ressourcen mit grundlegenden texttechnologischen Methoden zu integrieren, und zwar so, dass das Integrationsresultat auch in den Händen von Nicht–Texttechnologen handhabbar bleibt. Wir exemplifizieren diese Handhabung an einem Beispiel aus der historischen Semantik, und damit an einem Bereich, der erst in jüngerer Zeit durch die Texttechnologie erschlossen wird.

1 Einleitung

Die Abzählbarkeit sprachlicher Einheiten, ob nun auf der Ausdrucks- oder Inhaltsseite, bildet eine der grundlegenden Annahmen aller quantitativen Ansätze in der Linguistik [1, 2, 28, 35]. Dies schließt die angewandte Computerlinguistik ebenso ein wie die quantitative Linguistik und die sprachorientierte Fachinformatik. Die Analyse zeitlich geschichteter Daten in der computergestützten historischen Linguistik [13, 21] stellt diesen quantitativen Ansatz vor besondere Herausforderungen [3]. Der Grund besteht darin, dass die Instanzen einzelner Zähleinheiten zeitlich variieren, was eine Vorverarbeitung der zu analysierenden Korpora in einem Maße erforderlich macht, wie es für die in der Computerlinguistik üblicherweise untersuchten Korpora unüblich ist [25]. Die quantitativ arbeitende historische Linguistik ist daher auf den Einsatz genreübergreifender Korpora einer großen zeitlichen Bandbreite ebenso angewiesen wie auf die Verwendung möglichst vieler sprachlicher Ressourcen in Form von Thesauri und

[3] Siehe http://hudesktop.hucompute.org/.

historischen Wörterbüchern, welche die Analyse letzterer Korpora unterstützen. Als ein Beispiel für ein genreübergreifendes Korpus historischer Texte sei die *Patrologia Latina* [33] genannt, die lateinische Texte aus einem Zeitraum von über 1.000 Jahren umfasst. In diesem Zusammenhang bildet wiederum der *Thesaurus Linguae Latinae* [4] ein Musterbeispiel für eine lexikalische Ressource, deren Integration in die korpusbasierte Arbeit an der Patrologia Latina erhebliche texttechnologische Mehrwerte verspricht.

Aus dem Blickwinkel der geisteswissenschaftlichen Fachinformatik stehen wir damit vor der Aufgabe, eine Vielzahl heterogener sprachlicher Ressourcen auf der einen Seite mit grundlegenden texttechnologischen Methoden auf der anderen Seite zu integrieren, und zwar so, dass das Integrationsresultat auch in den Händen von Nicht–Texttechnologen handhabbar bleibt. *Genau dieser Aufgabe stellt sich der eHumanities Desktop.* Am Beispiel der Patrologia Latina und einer zugehörigen lexikalischen Ressource demonstrieren wir den Aufbau und die Gestaltung einer rein webbasierten Arbeitsumgebung für die textbasierte geisteswissenschaftliche Fachinformatik. Hierzu erläutern wir zunächst die Software–Architektur des *eHumanities Desktop*s (in Sektion 2). Ausgehend von der Korpusbildung (Sektion 3) und der generischen Nutzbarmachung lexikalischer Ressourcen (Sektion 4) demonstrieren wir im Anschluss hieran die Korpusanalyse (Sektion 5) und die Visualisierung lexikalischer Strukturen (Sektion 6) mit Hilfe des Desktops ebenso wie einen Brückenschlag zum Bereich des *Text Mining* (Sektion 7). Wir zeigen damit die Nutzbarkeit des Desktops auch für Wissenschaftler ohne texttechnologisches Basiswissen und ermöglichen damit eine Nutzbarmachung texttechnologischer Ressourcen in solchen Bereichen, die bislang nur zögerlich von Methoden der computerbasierten Textanalyse Gebrauch gemacht haben. Sektion 8 gibt schließlich einen Ausblick auf anvisierte Erweiterungen des *eHumanities Desktop*s.

2 Systemarchitektur und Rechteverwaltung

Der hohe Aufwand der Erstellung, Verwaltung sowie Vor- und Weiterverarbeitung von Korpora wird oft nicht von einem Forscher/einer Forscherin allein, sondern kollaborativ von einer Arbeitsgruppe erbracht. Der *eHumanities Desktop* unterstützt diese Arbeitsschritte unter besonderer Berücksichtigung von Arbeitsgruppen durch ein Korpus- und Ressourcen–Managementsystems, durch einen texttechnologischen wie auch *Text Mining*–orientierten Werkzeugkasten für die explorative Korpusanalyse sowie durch ein feingliedriges Rechtemanagement, das den Zugriff auf sämtliche Ressourcen und die darauf operierenden Methoden regelt. In dieser Sektion wird die Software–Architektur des *eHumanities Desktop*s erläutert.

Die Architektur des *eHumanities Desktop*s zielt auf ein breit gefächertes Anwendungsspektrum in den Geisteswissenschaften. Daher verfolgt der gewählte Ansatz eine bestmögliche Abstraktion der verwalteten Ressourcen, der sie bearbeitenden Entitäten und der verwendeten Methoden. Dadurch ist auch bei zukünftigen Anforderungen an das System eine nahtlose Erweiterbarkeit sicher-

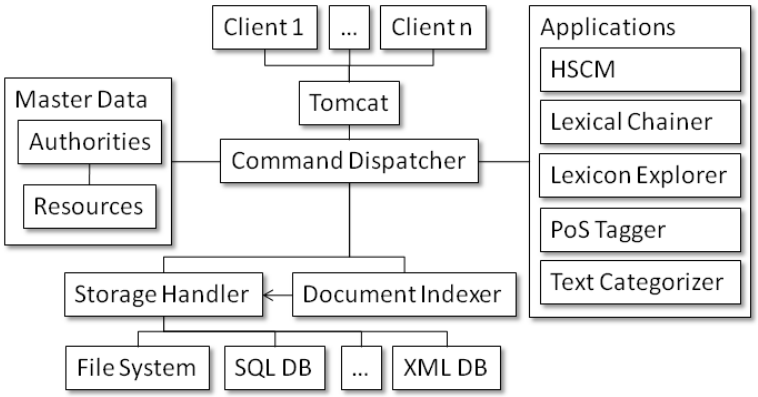


Abbildung 1. Die Architektur des *eHumanities Desktop*s.

zustellen. Gleichwohl muss die Konzeption des HCI die Komplexität des Systems auch für Benutzer mit nur grundlegendem Computerwissen beherrschbar machen. Dieser Anforderung stellt sich das Design des *eHumanities Desktop* durch eine konsequente Weborientierung der gesamten Applikation. Im Folgenden wird die Architektur des Desktops mit Blick auf die Ressourcen- und Rechteverwaltung dargestellt.

Der *eHumanities Desktop* ist als Client/Server System auf der Basis von Java EE Technologien realisiert (siehe Abbildung 1). Benutzer können sich plattformunabhängig über einen Browser anmelden, um Zugriff auf die Ressourcen und die Funktionalität des Desktops zu erhalten. Im Mittelpunkt steht der *CommandDispatcher*, der Anfragen der *Clients* entgegen nimmt und mit Rückgriff auf die Stammdatenverwaltung (*Master Data*), den *Storage Handler* sowie die Applikationsschnittstelle bearbeitet. Es können beliebige Formate im System verwaltet werden (z.B. auch Multimediadateien), die durch den *Storage Manager* im jeweils bestgeeigneten *Storage Backend* transparent für den Benutzer abgelegt werden.

Eine beispielhafte Benutzeranfrage könnte etwa das Tagging eines Textdokuments beinhalten: Der entsprechende Befehl wird von der *Client*–Anwendung an den Server geschickt und dort vom *CommandDispatcher* entgegen genommen. Zunächst wird auf Basis der Stammdaten geprüft, ob der Benutzer die Berechtigung dazu besitzt, das PoS–Tagging durchzuführen und das zugrundeliegende Textdokument lesen zu dürfen. Im positiven Fall wird durch eine Anfrage an den *Storage Handler* das Dokument ausgelesen und als Eingabe für den PoS–Tagger verwendet. Dieser erstellt nun eine aufbereitete und getaggte Version des Dokuments im TEI P5–Format, welches schließlich über den *Storage Manager* in einem dafür geeigneten *Storage Backend* abgelegt wird — in diesem konkreten Fall in einer nativen XML Datenbank. Nun wird auch in der Stammdatenverwaltung das neue Dokument sowie die Information, aus welchem Dokument es

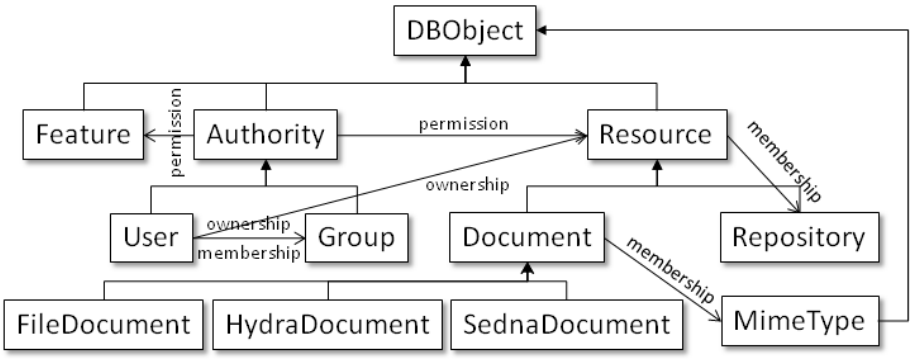


Abbildung 2. Das UML–Klassendiagramm der Stammdatenverwaltung des *eHumanities Desktop*s.

abgeleitet wurde, gespeichert. Schlussendlich wird die *Client*–Anwendung über die erfolgreiche Bearbeitung informiert.

Der Kern des *eHumanities Desktop* besteht in der Stammdatenmodellierung (siehe Abbildung 2), die auf die Erfassung kleiner Arbeitsgruppen ebenso ausgerichtet ist wie auf komplexe Verbundprojekte, für die eine feingliedrige Zugriffs- und Dokumentverwaltung erforderlich ist. Das Stammdatenmodell basiert genauer auf der Unterscheidung von *Autoritäten*, *Ressourcen*, *Systemfunktionen* (bzw. *Features*) und deren Relationen. Eine Autorität wird in diesem Kontext als ein abstraktes Konzept verstanden, welches durch *Benutzer* und *Gruppen* instanziiert wird. Beiden ist gemeinsam, dass ihnen Zugriffsrechte auf Ressourcen und Features zugewiesen werden können. Diese werden unterschieden nach Lese-, Schreib- und Löschrechten sowie nach dem Recht, selbst Zugriffsrechte vergeben zu dürfen. Benutzer können beliebig vielen (Arbeits-)Gruppen angehören und erhalten dadurch — sozusagen über ihre persönlichen Berechtigungen hinaus — die der Gruppen zugewiesenen Rechte. Jeder Ressource und jeder Gruppe ist ferner ein eindeutiger Benutzer als Besitzer zugeordnet. Ressourcen werden wiederum danach unterschieden, ob es sich um *Dokumente* oder *Repositories* handelt. Die Menge aller im System erfassten Dokumente (der so genannte *Dokumentraum*) ist zunächst aus Benutzersicht unstrukturiert. Diese Sicht kann jedoch durch *Repositories* strukturiert werden, und zwar durch die Zuordnung von Ressourcen zu (beliebig vielen) Repositories. Auf diese Weise ist auch eine Unterordnung von Repositories realisierbar wie sie z.B. zum Zwecke der Korpusbildung benötigt wird: Ein Spezialkorpus für eine Teilmenge von Nutzern kann beispielsweise dadurch gebildet werden, dass ein neues Repository angelegt wird und durch die Rechtevergabe für den gewünschten Nutzerkreis freigegeben wird. Im Anschluss können nun alle Dokumente des zu bildenden Korpus diesem Repository zugewiesen werden. Durch die Möglichkeit der Unterordnung von Repositories können auf diese Weise leicht Teilkorpora angelegt und verwaltet werden.

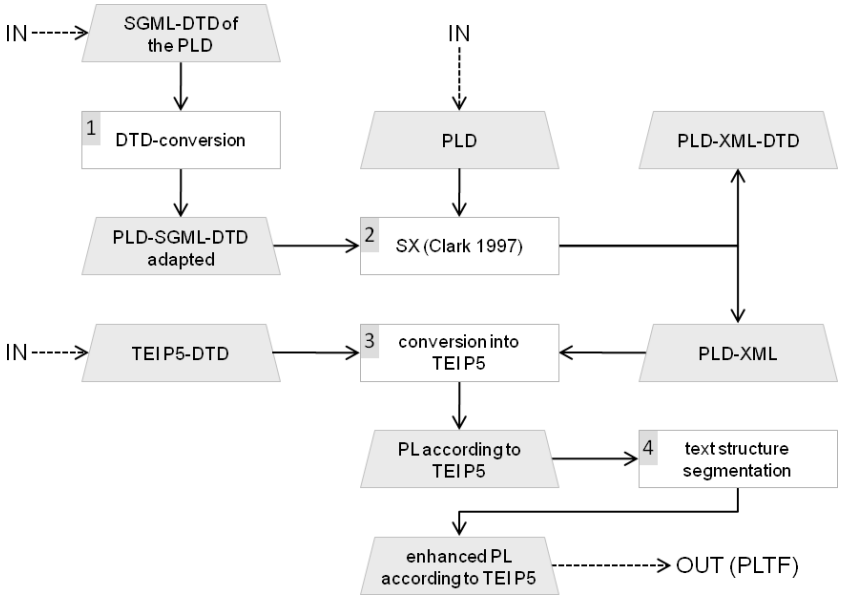


Abbildung 3. Überführung der *Patrologia Latina DB* (PLD) [33] in das Zielformat der TEI P5 unter expliziter Annotation von Textstrukturelementen.

Analog zur Vergabe von Rechten auf Ressourcen kann auch der Zugriff auf die Programmfunktionalität feingliedrig geregelt werden. Nicht alle Benutzer sollen etwa in der Lage sein, Gruppen anzulegen oder Dokumente hochzuladen. Anstatt dies nun statisch über Attribute festzulegen, findet auch hier das Prinzip der dynamischen Rechtevergabe Verwendung. Auf diese Weise können neue Programmfunktionen gezielt für bestimmte Nutzerkreise freigegeben oder gesperrt werden. Die nachfolgende Sektion behandelt Grundlagen für den Einsatz dieses Systems im Rahmen der Korpusbildung.

3 Korpusbildung

Die Gewährleistung einer texttechnologisch versatilen Arbeitsumgebung für die geisteswissenschaftliche Fachinformatik im Bereich der historischen Linguistik ist an die effiziente Verarbeitung des zugrundeliegenden Textmaterials gebunden. Dieser effiziente Umgang in Form entsprechender Operationen auf einer geeigneten Textdatenbank [14, 31] erfordert seinerseits die Aufbereitung des Korpusmaterials auf der Basis fachüblicher Standards [19, 39]. Am Beispiel der *Patrologia Latina* demonstrieren wir nun diesen Aufbereitungsschritt mit dem Ziel der Korpusbildung für die historische Semantik [21]. Zu diesem Zweck veranschaulichen wir nachfolgend die Ergebnisse einer vollständigen Transformation der *Patrologia Latina* in das TEI P5-Format [5].

3.1 Das Beispiel der Patrologia Latina

Die Überführung der PL in ein Format, welches jener Art von Korpusanalyse zugänglich ist, die der Desktop unterstützt, erfolgt im Wesentlichen in vier Schritten (vgl. Abbildung 3): Aus dem proprietären und dokumentstrukturell wenig expliziten Format des PL–Originals, welches auf einer SGML–DTD basiert, ist eine XML–Entsprechung zu erzeugen. Dies setzt jedoch die vorangehende Übersetzung der SGML–DTD dieses Originals in eine entsprechende XML–DTD voraus. In einem dritten Schritt ist die XML–basierte Entsprechung der PL in das gleichermaßen XML–basierte Format der *Text Encoding Initiative* (TEI P5) [5] zu überführen. Schließlich ist in einem vierten Schritt die TEI P5-basierte Repräsentation der PL dahingehend zu erweitern, dass eine Vielzahl von Dokumentstrukturen, welche im digitalen PL–Original nicht annotiert sind, repräsentiert und damit für nachfolgende Analyseschritte zugänglich gemacht werden. Dies betrifft im Wesentlichen die Annotation von Satzgrenzen, deren Bestimmung für die satzsensitive Kollokationsanalyse unabdingbar ist. Im Folgenden skizzieren wir kurz diese vier in Abbildung 3 zusammengefassten Vorverarbeitungsschritte gemäß ihrem exemplarischen Charakter für die Vorverarbeitung von Korpora.

Die *Patrologia Latina DB* (PLD) [33] beinhaltet eine digitale Fassung der 221 Bände der ersten Edition der Patrologia Latina von Jacques–Paul Migne, und zwar in Form ebenso vieler SGML–Dateien. Da für die Weiterverarbeitung von SGML–Dateien kaum brauchbare Werkzeuge verfügbar sind, ist eine Transformation der PLD für die Zwecke der Korpusanalyse, wie sie der *eHumanities Desktop* anstrebt, unabdingbar. Als Zielsprache dieser Transformation dient XML, da der *eHumanities Desktop* standardmäßig Dokumentstrukturen auf TEI P5 abbildet. Als Konversionswerkzeug wiederum wurde das SX-Tool von [6] eingesetzt, das eine entsprechend eingeschränkte Input–DTD voraussetzt. Für die Bereitstellung dieser DTD wurde projektintern eine modifizierte Fassung der DTD der PLD erarbeitet, und zwar unter weitgehender Auflösung von Parameter–Entitäten. Einige wenige Zeichen–Entities, die [6] nicht unterstützt, wurden entsprechend angepasst oder ersetzt.[4]

Die Bände der PL verfügen über eine einheitliche Dokumentstruktur. Auf oberster Dokumentstrukturebene enthalten sie drei Elemente: `volfront`, `volbody` und das optionale Element `volback`. Das `volfront`-Element umfasst das Titelblatt und das Inhaltsverzeichnis des jeweiligen Bandes, das `volbody`-Element die zugehörigen Dokumente und das `volback`-Element gegebenenfalls einen Index

[4] Es sei darauf hingewiesen, dass für die Zwecke der Transformation unter anderen folgende Korrekturarbeiten vollzogen wurden: Da XML Groß- und Kleinschreibung unterscheidet, mussten mehrere Attribute und Attributwerte in Kleinschreibung überführt werden. Einige Textteile sind im Original mittels `hi`-Tags hervorgehoben, die für die weitergehende Verarbeitung irrelevant sind und daher entfernt wurden. Der Transformationsprozess selbst produzierte eine Reihe von Fehlermeldungen, die Nachbearbeitungsbedarf erzeugten. So sind beispielsweise im Original Dokumentpositionen mit Text gefüllt, an denen laut PLD–DTD nichts stehen darf. Aber auch ein nicht geschlossenes Tag und ein falsch beendetes Entity erzeugten weiteren Korrekturbedarf.

samt abschließender Kommentare. Für die korpusanalytische Verarbeitung der PL innerhalb des *eHumanities Desktop*s wurden nur die Inhalte der `volbody`-Elemente berücksichtigt. Diese wurden aus den Bänden extrahiert und als separate Dokumente mit fortlaufender ID erfasst. Dabei wurde das `doc`-Element um die Attribute `doc_id` und `vol_id` erweitert, um eine spätere eindeutige Bandzuordnung zu gewährleisten.

Aus der Sicht von Kookkurrenzanalysen, wie sie der *eHumanities Desktop* unterstützt, ist es nötig, die Struktur von Dokumenten zumindest bis hinunter zur Satzebene zu annotieren. Zu diesem Zweck wurden die aus der PL extrahierten Dokumente einer Analyse zur Extraktion von Abkürzungen unterzogen, welche in einem zweiten Schritt dem Satzerkenner (vgl. [31]) des Desktops verfügbar gemacht wurden.[5] Es sei angemerkt, das die PLD Satzgrenzen nicht annotiert, dies also eine Leistung des Desktops ist. Dabei wurden Gedichte und Kapitelüberschriften von der Satzerkennung ausgenommen. So stehen beispielsweise der zeilenorientierten Annotation von Gedichten Enjambments entgegen, so dass zur Vermeidung von Überlappungen die zeilenorientierte Gedichtannotation vorgezogen wurde.

Die Konversion in das TEI P5-Format erfolgt schließlich mittels eines speziell entwickelten `Java`-Programms, das eine Vielzahl struktureller Besonderheiten des PLD–Formats an das TEI P5-Format anpasst. So erlaubt das PLD–Format beispielsweise das Anlegen von Dokumentverweisen mittels `id-` und `rid`-Attributen, die auf `xml:id`-Attribute des TEI P5-Formats abgebildet bzw. durch ein `<listRef><ptr target="#rid"/><listRef/>`–Konstrukt ersetzt wurden. Im Anschluss hieran wurden leere Tags entfernt und alle übrigen Elemente mit einer eindeutigen `xml:id` versehen. Schließlich wurden alle Sätze und höhergeordneten Textsegmente indexiert und mit zusätzlichen Informationen attribuiert. Diese betreffen unter anderem den Textsegmenttyp (etwa in Form von Notizen, Gedichten, Listen, Überschriften, Tabellen). Die aus diesen Konversionsschritten hervorgehende Instanz der TEI–DTD in Form der PL bezeichnen wir in der Form, in der sie der *eHumanities Desktop* bereitstellt, als *Patrologia Latina according to the TEI P5 Format* (PLTF).

Tabelle 1 gibt abschließend einen Überblick über die Häufigkeitsverteilung von Textstrukturelementen der TEI P5 in der PLTF. Tabelle 1 weist als Vergleichsmaßstab die entsprechenden Werte der deutschsprachigen Wikipedia aus (vgl. [31]). Es wird ersichtlich, dass obzwar die Wikipedia der Textmenge nach deutlich größer ist, die Tokenmenge der PL jedoch bis zu einem Drittel an die entsprechende Menge der Wikipedia heranreicht. Das solcherart vorverarbeitete und annotierte Korpus kann nun mittels des *eHumanities Desktop* zum Gegenstand korpusanalytischer Operationen gemacht werden [21], von denen wir hier die Kollokationsanalyse exemplifizieren (siehe Sektion 5).

[5] Dieser Analyseschritt identifizierte 8.904 Abkürzungskandidaten.

Element / Objekt	Anzahl
Autor	2.024
Text	8.508
Paragraph	870.509
Satz	9.464.285
Token	119.632.281
Wortform	1.111.420

Element / Objekt	Anzahl
Autor	3.586.131
Text	875.404
Paragraph	10.431.961
Satz	56,677,686
Token	436.439.087
Wortform	4.592.145

Tabelle 1. Einzelne Elementtypen bzw. Objekttypen und deren Anzahl in der PLTF (links) im Vergleich zu der deutschsprachigen Wikipedia (rechts) (vgl. [31]). Die Berechnung der Anzahl der Wikipedia–Autoren erfolgt mittels Zählung von Benutzernamen bzw. IP–Adressen (und zwar in Fällen, in denen kein Benutzername, sondern nur die IP–Adresse vorliegt). Für die Zählung wurde ein Dump von Juni 2008 verwendet.

4 Lexikalische Ressourcen

Eine wichtige Voraussetzung für die geisteswissenschaftliche Fachinformatik bildet die Verfügbarkeit einer möglichst generischen Schnittstelle für den Einsatz lexikalischer Ressourcen. Es geht darum, für die unterschiedlichen Aufgaben der (teil-)automatischen oder auch nur computergestützten Textanalyse lexikalische Ressourcen bereitzustellen. Hierfür wird ein Datenmodell benötigt [38], das sämtliche dieser Ressourcen generisch repräsentiert und durch entsprechende Datenbankoperationen (etwa der Lexikonsuche und des -updates) flankiert. In dieser Sektion skizzieren wir ein solches Modell auf der Basis eines relationalen, datenorientierten Datenbankschemas.

Wegen der Vielfalt lexikalischer Ressourcen, welche über den Desktop verfügbar zu machen sind, ist die Datenmodellierung in diesem Bereich vor besondere Herausforderungen gestellt. Unterschiedlich strukturierte Inputlexika sind unter Erhalt ihrer Struktur auf eine Weise generisch zu repräsentieren, die ihren effizienten Zugriff gewährleistet. Diese Anforderung erfüllen wir durch ein Datenmodell (siehe Abbildung 4), das einen gerichteten Hypergraphen mit mehrfach benannten Knoten und Hyperkanten modelliert. Ergänzt wird dieses Modell durch eine Ordnungsrelation über der Menge jener Knoten, welche durch die jeweils *gerichtete* Hyperkante verbundenen sind. Die Implementierung dieses Datenmodells in MySQL ermöglicht einen nahtlosen Zugriff auf alle im Desktop verwalteten lexikalischen Ressourcen. Darüber hinaus stellen wir eine `C++`-basierte Active–Record–Implementierung [12] dieses Datenmodells bereit, welche ein Mapping aller Datenbankinhalte auf die Objekte der dem Desktop zugrundeliegenden Klassenbibliothek leistet.

Im Folgenden wird die Verwendung des Datenmodells zur Integration eines lateinischen Vollformenlexikons in den *eHumanities Desktop* kurz umrissen. Dieses Lexikon wurde aus der *Grammatica Latina* extrahiert, einem Parser für klassisches Latein, der zu dem NLP–System *Affix Grammars over a Finite Lattice* (AGFL) [23] gehört. Das auf die Datenbank abgebildete lateinische Vollformen–Lexikon umfasst im Wesentlichen vier Klassen von Informationsobjekten: (i)

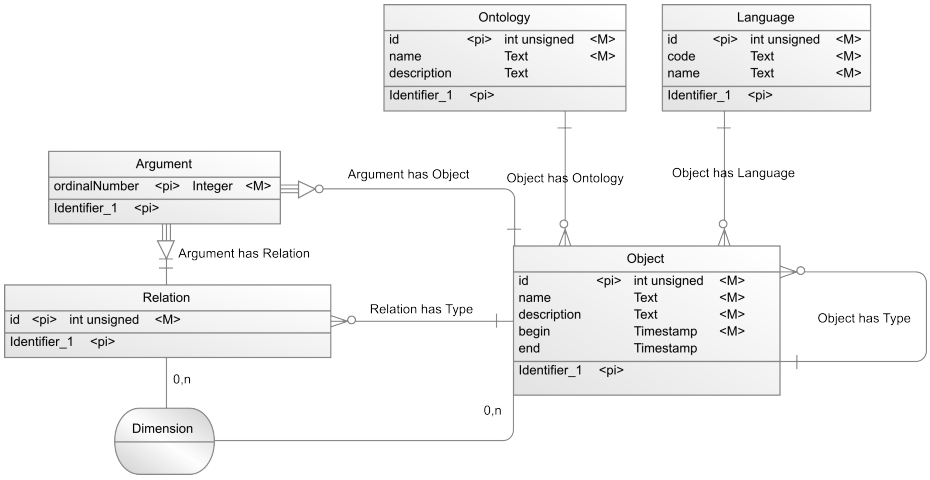


Abbildung 4. Das Datenbankschema der Lexikonkomponente des *eHumanities Desktop*s.

Lemmata mit Angabe ihrer Wortart, (ii) Wortformen, (iii) Zuordnungen zwischen Lemmata und Wortformen unter Angabe grammatischer Informationen sowie (iv) Derivationsrelationen, die zwischen Lemmata bestehen. Lemmata, Wortformen und grammatische Informationen werden als benannte Knoten eines Hypergraphen abgebildet, alle relationalen Informationsbestandteile hingegen als Hyperkanten. Eine Klasse von Kanten bilden Instanziierungsbeziehungen zwischen Lemmata und Wortformen, wobei grammatische Informationen, welche diese Beziehungen spezifizieren, den Kanten als Beschriftungen zugeordnet sind. Derivationsrelationen werden ebenfalls als entsprechend benannte Hyperkanten abgebildet.

Der *eHumanities Desktop* macht eine Vielzahl lexikalischer Ressourcen verfügbar. Tabelle 2 zählt diese Ressourcen im Einzelnen auf; Tabelle 3 gibt Informationen über deren Umfang. Neben Vollformenlexika und terminologischen Ontologien, die jeweils vollständig auf das in Abbildung 4 dargestellte Datenbankschema abgebildet wurden, macht der *eHumanities Desktop* auch rein webbasierte lexikalische Ressourcen verfügbar. Diese Ressourcen werden je nach Bedarf aus dem WWW extrahiert. Hierzu zählen insbesondere *Social Tagging*-Systeme (z.B. Delicious oder Flickr) wie sie für Anwendungen des Web 2.0 charakteristisch sind. Ihrer Größe und Dynamik wegen stehen sie einer statischen Datenbankabbildung entgegen, so dass sich eine dynamische Einbindung empfiehlt wie sie der Desktop realisiert. Der Desktop integriert hierzu eine einheitliche Schnittstelle zur Extraktion und Nutzbarmachung von Ressourcen dieser Art (siehe Abbildung 5).

Klasse von lexikalischer Ressource	im Desktop verfügbare Instanz der Klasse	Art der Einbindung
terminologische Ontologie	- WordNet - GermaNet	DB–basiert DB–basiert
Kollokationsnetzwerk	- Leipziger Wortschatz (*nicht* lemmatisiert) - Kollokationsnetzwerk (lemmatisiert) basierend auf der Wikipedia, der Wochenzeitung Die Zeit, der TAZ und der Süddeutschen Zeitung	dynamisch DB–basiert
soziale Ontologie	- de.Wikipedia - en.Wikipedia - de.Wiktionary	DB–basiert DB–basiert dynamisch
Social Tagging-basierte Ressource	- Delicious - Flickr - Amazon - Mister Wong	dynamisch dynamisch dynamisch dynamisch
Vollformenlexikon	- AGFL–basiertes lateinisches Vollformenlexikon	DB–basiert

Tabelle 2. Auflistung der im Desktop verfügbaren lexikalischen Ressourcen. DB–basiert steht für eine datenbankbasierte Einbindung, dynamisch für eine Form der Einbindung, welche die Extraktion der erforderlichen Daten zum Verwendungszeitpunkt aus dem WWW beinhaltet. Der Begriff terminologische Ontologie wird hier im Sinne von [37] verwendet. Eine terminologische Anmerkung: Ein Kollokationsnetzwerk (vgl. [30]) ist ein Netzwerk, dessen Knoten Wortformen oder Lemmata entsprechen, und dessen Kanten solche Kookkurrenzbeziehungen abbilden, die aufgrund eines zugehörigen wahrscheinlichkeitstheoretischen Modells als überzufällig gelten.

5 Korpusanalyse

Die Korpusanalyse im Stile der Korpuslinguistik [10, 27] bildet eines der Hauptanwendungsgebiete des *eHumanities Desktop*s aus der Sicht seiner geisteswissenschaftlichen Nutzer. Sie gilt auch im Bereich der historischen Semantik als vielversprechender Zugang für die Exploration sprachlich manifestierter sozial–geschichtlicher Prozesse [21]. Aus diesem Grunde und mit Blick auf historische Korpora wie die Patrologia Latina (siehe Sektion 3) integriert der Desktop das *Historical Semantics Corpus Management System* (HSCMS). Die Verfahrensweise des HSCM soll nun am Beispiel der PLTF demonstriert werden.

HSCMS arbeitet mit dem in Sektion 4 skizzierten Vollformenlexikon der Lateinischen Sprache sowie mit einem Index der PLTF. Hierzu wird eine Datenbank bereitgehalten, welche alle in der PLTF vorkommenden Wortformen umfasst. Diese Wortformen sind zum Teil mit ihrem zugehörigen Lemma verknüpft,

Name	abgebildete Informationen	#Knoten	#Kanten	#Kantenlabels
WordNet	Wörter, Synsets, Wort–Synset-Zuordnungen unter Angabe von Sense–Number und Häufigkeit Synset–Synset–Relationen	475.012	499.351	905.645
GermaNet	Wörter, Synsets Wort–Synset–Relationen, Synset–Synset–Relationen	227.393	505.186	—
lemmatisiertes Kookkurrenznetz	Wörter, Sätze, Quellen, Wort–Satz–Zuordnungen unter Angabe der Wortposition im Satz, Satz–Quellen Zuordnungen, Bigramme mit Häufigkeit und log–likelihood, satzbasierte Kookkurrenzen mit Häufigkeit und log–likelihood	3.036.864	8.107.811	6.982.954
de.Wikipedia	Namensräume, Seiten, Links zwischen Seiten, entsprechend der Namensräume der verbundenen Seiten typisiert.	1.468.080	19.015.018	—
en.Wikipedia	siehe de.Wikipedia	4.461.898	54.722.527	—
AGFL–basiertes lateinisches Vollformenlexikon	Lemmata, Wortformen, Wortform–Lemma–Zuordnungen mit zusätzlichen grammatischen Informationen, Lemma–Lemma-Zuordnungen	311.630	610.370	—

Tabelle 3. Auflistung der verfügbaren Ressourcen. Anmerkung: Alle Kanten und Knoten des modellierten Hypergraphen sind typisiert. In der Tabelle werden jedoch nur solche weiteren Beschriftungen explizit als Kantenlabel ausgegeben, die über die obligatorische einfache Typisierung hinaus abgelegt sind.

wobei die Lemmazuordnung kontinuierlich vervollständigt wird. Eine besondere Anforderung der historischen Semantik besteht in der Berücksichtigung von Lexemverbänden, die ebenfalls durch das HSCMS definiert und verwaltet werden können. Basierend auf der Wortformendatenbank sowie der Abbildung von Wortformen auf Lemmata bzw. Lexemgruppen lässt sich eine mehrstufige Expansion von Suchtermen realisieren. Diese funktioniert auf der Ebene einzelner Suchterme ebenso wie auf der Ebene zusammengesetzter Suchanfragen. Ferner ist zu beachten, dass das HSCMS die geisteswissenschaftliche Arbeit durch den Einsatz von vielfältigen Korpusfiltern, die Teilkorpora der PLTF erzeugen, unterstützt.

Eine der Hauptfunktionen des HSCMS besteht in der Berechnung von Konkordanzen. Hierzu erlaubt es das HSCMS, variable Satzkontexte mit bis zu 9 Nachbarsätzen in der links- oder rechtsseitigen Umgebung des jeweiligen Suchtreffers auszuwerten. Die resultierenden Ergebnislisten können anschließend lemmatisiert und weiterverarbeitet werden. Dies betrifft unter anderem die Nutzung mehrerer Zielformate für den Datenexport. Darüber hinaus verfügt das HSCMS

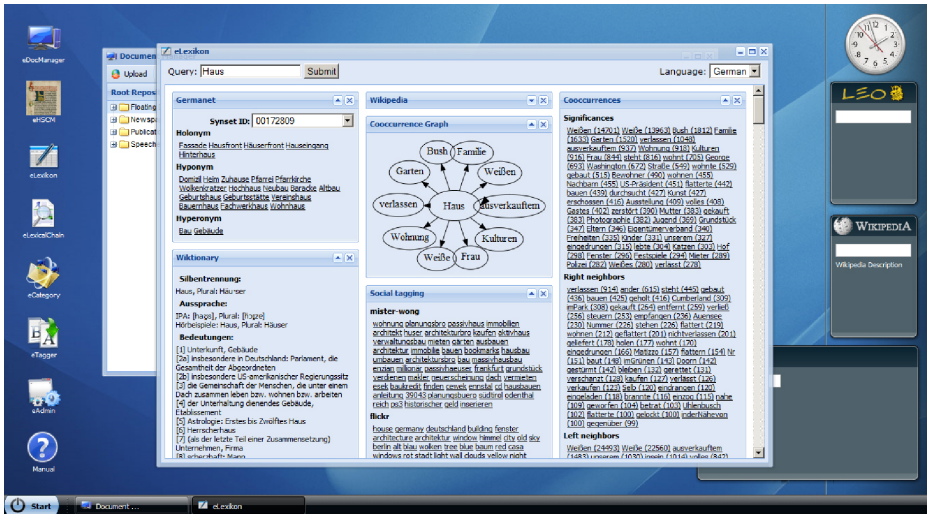


Abbildung 5. Ausschnitt des HCI der Lexikonkomponente des *eHumanities Desktop*s unter gleichzeitiger Einbindung von GermaNet [24], der deutschsprachigen Wikipedia, des deutschsprachigen Wiktionaries, mehrerer *Social Tagging*-Ressourcen sowie eines kookkurrenzbasierten Lexikons [16].

über ein Modul für den mengenorientierten Vergleich von Ergebnislisten. Dies erlaubt es wiederum, ganze Texte und Textkollektionen nach signifikanten Abweichungen in den Kollokationsbeziehungen ihrer lexikalischen Konstituenten zu untersuchen.

Das HSCM–Modul unterstützt weiterhin die Überführung von Teilkorpora in Term–Dokument–Matrizen wie sie für lexikalische Dokumentrepräsentationsmodelle [36] unabdingbar sind. Auf diese Weise schlägt der Desktop eine Brücke zwischen historischen Korpora und den im Information Retrieval gängigen Verfahren der Dokumentrepräsentation und -verarbeitung. Ein besonderer Mehrwert des Systems besteht in diesem Zusammenhang darin, dass für die Auswahl der lexikalischen Dimensionen der zu erstellenden Term–Dokument–Matrizen frei zwischen Wortformen, Lemmata und Lexemgruppen gewählt werden kann. Hierdurch eröffnet der Desktop eine operationale Freiheit, wie sie für vergleichbare Systeme oft eingefordert, seltener jedoch erbracht wird.

6 Visualisierung

Die Visualisierung der Vernetzungsregularitäten sprachlicher Ressourcen bildet einen weiteren Mehrwert des *eHumanities Desktop*s. Es geht dabei um die generische Modellierung lexikalischer Relationen bezogen auf deren Visualisierung. Einen grundlegenden Aspekt dieser Aufgabe bildet die interaktive Gestaltung der Visualisierungsschnittstelle als Mittel zur Steuerung des Desktops selbst.

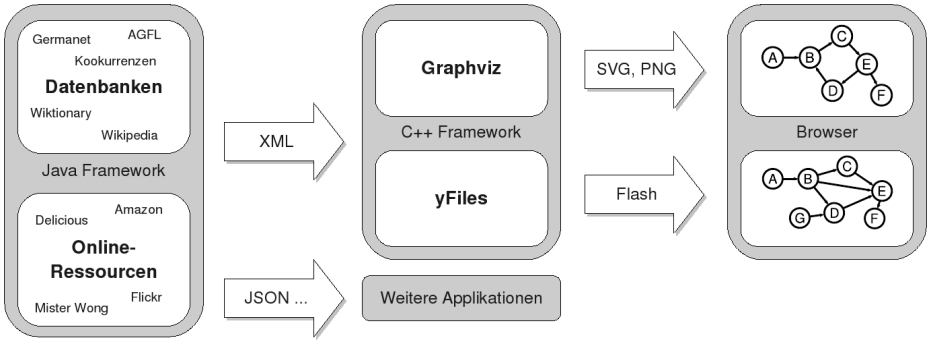


Abbildung 6. Übersicht über die der Visualisierungskomponente des Desktops zugrunde liegende Software–Architektur.

Im Folgenden skizzieren wir die hierbei einschlägigen Modellierungsschritte und verweisen auf entsprechende HCI–orientierte Erweiterungsmaßnahmen.

Damit alle Ressourcen des Desktops über eine einheitliche Schnittstelle zugreifbar sind, wurden diese in ein generisches `Java`-Framework integriert (siehe Abbildung 6). Dieses Framework erlaubt die Integration heterogener Daten und stellt eine einheitliche Schnittstelle für hierauf aufsetzende Visualisierungskomponenten bereit, die im vorliegenden Fall `C++`–basiert sind. Die Architektur dieser Schnittstelle folgt dem Paradigma der *serviceorientierten Architektur* (SOA) [9]. Das bedeutet, dass sämtliche Systemkomponenten als Dienste bereitgestellt werden, die über wohldefinierte Schnittstellen kommunizieren. Auf diese Weise entsteht eine Trennung zwischen der Modellierung lexikalischer Relationen als Informationsobjekte einerseits und ihrer Visualisierung andererseits.

Der Vorteil dieses Ansatzes besteht in seiner Platform- und Programmiersprachenunabhängigkeit; sie ermöglichen die Anbindung der hier genutzten `C++`-basierten Visualisierungskomponente (siehe Abbildung 6). SOA bietet darüber hinaus eine große Flexibilität im Hinblick auf die Wahl von Austauschformaten. Für die Visualisierung kommt ein eigens entwickeltes XML–Schema namens eGraphML zum Einsatz (siehe Tabelle 4 für eine entsprechende Schema-Instanz [6]), der neben Knoten- und Kanten–bezogenen Informationen weitere visualisierungsrelevante Informationen bereithält. Dies betrifft unter anderem Kantengewichte und Formatierungsanweisungen für elementare Graphelemente. Über dieses XML–basierte Austauschformat hinaus stehen anzubindenden Applikationen eine JSON- [7] sowie eine SOAP–Schnittstelle zur Verfügung.

Als Visualisierungssoftware nutzt der Desktop unter anderem den in Graphviz [8] enthaltenen Graph–Layouter Neato. Dieser verteilt Knoten mithilfe des Algorithmus von [22] und erzeugt auch bei großen bzw. breiten Graphen gut lesbare Ansichten. Die Menge der lexikalischen Relationen ist jedoch so groß, dass selbst Neato daran scheitert, diese in einem sichtbaren Fenster des Desktops

[6] Das Schema liegt unter http://hucompute.org/resources/eGraph/eGraph.dtd, eine Instanz unter http://hucompute.org/resources/eGraph/eGraph_instance.xml.

```
<!DOCTYPE graph SYSTEM "eGraph.dtd">
<graph>
  <parser>eHumanities Desktop</parser>
  <author>eHumanities Desktop</author>
  <subGraph id="c1">
  <graphAttribute name="label" value="significant collocates of house" />
    <path id="p1">
      <sourceNode id="sn1">
        <nodeAttribute name="label" value="Haus" />
        <nodeAttribute name="fontsize" value="20" />
      </sourceNode>
      <targetNode id="pn1">
        <nodeAttribute name="label" value="Weißen" />
      </targetNode>
      <edge id="e1" step="1" sourceId="sn1" targetId="pn1">
        <edgeAttribute name="significance" value="14701" />
      </edge>
    </path>
  </subGraph>
</graph>
```

Tabelle 4. Eine Beispielinstanz von eGraphML für die Visualisierung lexikalischer Netzwerke.

darzustellen. Daher wird der angezeigte Graph in seiner Breite stark begrenzt; gleichzeitig werden Anstrengungen unternommen, eine intuitive und performante Steuerung der Visualisierungsschnittstelle zu bewerkstelligen. Dies betrifft beispielsweise anklickbare Knoten und eine damit verbundene Aktualisierung des darzustellenden Graphausschnitts.

7 Text Mining mittels Lexical Chaining

Neben der Bereitstellung und Verfügbarmachung sprachlicher Korpora und Ressourcen bildet die Einbindung von automatisierten Verfahren der explorativen Textanalyse bzw. des *Text Mining* [32] ein weiteres Expansionsgebiet des *eHumanities Desktop*s. Eine wesentliche Grundlage für Text Mining–Module bilden wiederum Komponenten für die syntaktische bzw. strukturelle Vorverarbeitung natürlichsprachlicher Texte. Aus diesem Grunde integriert der Desktop eine Reihe von Funktionsobjekten (vgl. [31, 42]) für die Spracherkennung, die Satzgrenzenerkennung, das Stemming, die Lemmatisierung, das Part–of–Speech–Tagging und die Eigennamenerkennung sowie für die automatische Segmentierung von Dokumentstrukturen und deren Abbildung auf die TEI P5 [5] bzw. den CES [20]. Diese Module, deren Testergebnisse Tabelle 5 ausschnittweise wiedergibt, sind über eine erweiterbare Tool–API in den *eHumanities Desktop* integriert.

Modul	Sprache	Parameter	F–Score	Test Corpus
PoS–Tagging	de	3.000 Sätze	0,975	NEGRA Korpus [40]
	de–gesprochen	3.900 Sätze	0,982	Tübinger Spoken Language Corpus [17]
	en	5.000 Sätze	0,956	Penn Treebank [26]
Lemmatisierung	de	888.573 Wortformen	0,921	NEGRA Korpus
Sprachenidentifikation	21 Sprachen	50 Chars	0,956	Wikipedia
	21 Sprachen	100 Chars	0,970	Wikipedia

Tabelle 5. Ergebnisse der Evaluation der im Desktop integrierten Komponenten für die syntaktische bzw. textstrukturelle Vorverarbeitung (vgl. [42] und [41]).

Zur semantischen Anreicherung von Dokumentinstanzen wiederum rekurriert der Desktop auf das Prinzip der lexikalischen Kohäsion [15]. Das so genannte *Lexical Chaining* [18, 29, 43, 44] dient in diesem Zusammenhang der Extraktion von Themensträngen auf der Basis terminologischer (z.B. WordNet [11]) oder sozialer Ontologien (z.B. Wikipedia). Semantisch verwandte Wortformen werden dabei automatisch zu lexikalischen Ketten verknüpft — Abbildung 7 verdeutlicht dies anhand eines Beispiels. In einem zweiten Schritt erfolgt die

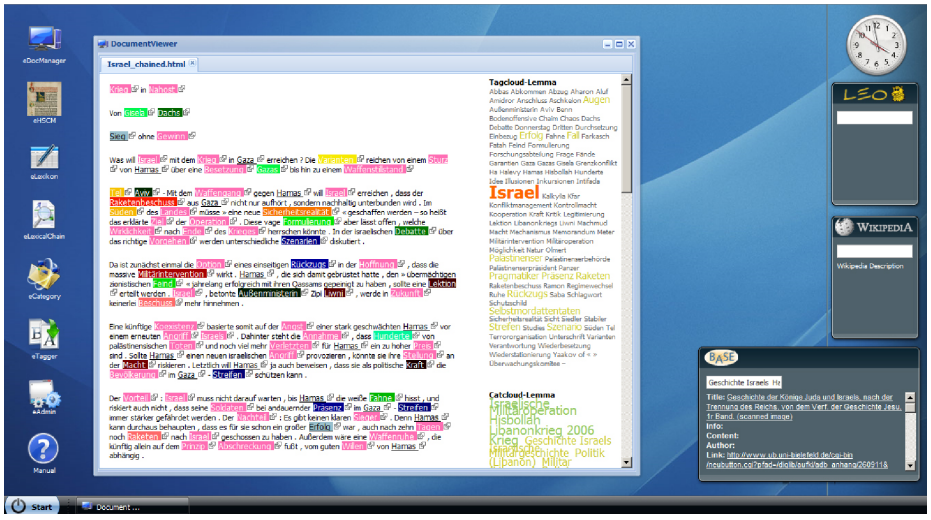


Abbildung 7. Ein Beispiel für das *Lexical Chaining* eines Zeitungsartikels der Wochenzeitung Die Zeit im Rahmen des *eHumanities Desktop*s. Das BASE–Fenster im rechten unteren Bildschirmausschnitt demonstriert eine semantische Suche.

wiederum automatische Abbildung der lexikalischen Ketten auf das Kategoriensystem der Wikipedia, und damit auf eine soziale Ontologie. Das bedeutet, dass

die Repräsentanten von Themensträngen in Form lexikalischer Ketten im Einzelnen auf Knoten des Kategoriensystems der Wikipedia abgebildet werden, welche diese Themenstränge namentlich bezeichnen bzw. als semantische Metainformation charakterisieren. Durch die Kombination von Themensträngen und -namen werden schließlich semantische Suchanfragen generierbar. Dies geschieht unter Nutzung der *Bielefeld Academic Search Engine* (BASE) [34]. Gerade die Anreicherung der Textketten mittels kategorialer bzw. konzeptueller Informationen ermöglicht es, Terme für Suchanfragen zu verwenden, welche *nicht* im Inputtext vorkommen müssen und dennoch dessen Inhalt charakterisieren. In diesem Sinne ist von einer semantischen Suche zu sprechen, welche letztlich die im *eHumanities Desktop* verwalteten Dokumente mit dem im WWW verfügbaren Bestand digitaler Bibliotheken verbindet, und zwar über BASE.

8 Schlussfolgerung und Ausblick

Es ist davon auszugehen, dass schon in naher Zukunft jene Art von Texttechnologie, welche der *eHumanities Desktop* ermöglicht, einem breiten Anwenderkreis zugänglich und gleichermaßen zuhanden sein wird. Damit stehen wir vor der Aufgabe, mehr und mehr texttechnologische bzw. *Text Mining*–orientierte Funktionalitäten, die bislang noch immer ausschließlich in den Händen ihrer Entwickler funktionieren, so zu integrieren, dass sie die geisteswissenschaftlichen Werkzeugkästen bereichern. Aus dieser Sicht ist zu fragen, welche Richtung die Weiterentwicklung des *eHumanities Desktop*s nehmen wird. Eine dieser Richtungen besteht in der konsequenten Weiterentwicklung und Integration von Text-Mining-Technologien. Noch immer können der Bereich des sprachorientierten *Machine Learning* einerseits und sein potenziellen Anwenderkreis in den Geisteswissenschaften als streng separiert gelten. Hier eine Brücke zu schlagen, wird denn auch eine der wichtigsten Aufgaben des Desktops bleiben. Eine zweite grundsätzliche Richtung für die Weiterentwicklung des Desktops betrifft die Interface–Gestaltung. Denn nur ein wirklich versatiler und zugleich transparenter Desktop wird Geisteswissenschaftler davon überzeugen, Methoden der quantitativen, explorativen Datenanalyse zu übernehmen, um über bloß nominal- oder ordinalskalierte Messungen hinaus auch zu verhältnisskalierten Messresultaten zu gelangen [2]. Mit der graphbasierten Darstellung und Manipulation ist bereits ein Weg beschritten, in stärkerem Umfang HCI–orientierte Systemsteuerungselemente bzw. *cognitive interaction technologies* in den Desktop zu integrieren. So könnten beispielsweise zukünftige Versionen des Desktops die Navigation in Graphrepräsentationen sprachlicher Ressourcen mittels *eye tracking* ermöglichen. Auf diese Weise werden perspektivische Darstellungen und Manipulationsmöglichkeiten eröffnet wie sie in herkömmlichen webbasierten Informationssystemen noch immer nicht genutzt werden. Unter dieser Perspektive zeigt sich ein vielversprechender Ansatz für die Verbindung von Texttechnologie und kognitiver Informatik, dem zukünftig auch diese Arbeitsgruppe folgen wird.

Literaturverzeichnis

[1] G. Altmann. *Wiederholungen in Texten.* Brockmeyer, Bochum, 1988.

[2] G. Altmann. Science and linguistics. In R. Köhler and B. B. Rieger, editors, *Contributions to Quantitative Linguistics*, pages 3–10, Dordrecht, 1993. Kluwer.

[3] M. V. Arapov and M. M. Cherc. *Mathematische Methoden in der historischen Linguistik.* Brockmeyer, Bochum, 1983.

[4] Bayerische Akademie der Wissenschaften. *Thesaurus linguae Latinae. Vol. I–IX.* Teubner, (Stuttgart u. Leipzig (bis 1999); KG Saur-Verlag, München u. Leipzig (bis 2006); Walter de Gruyter, Berlin, New York (ab 2007), 2007.

[5] L. Burnard. New tricks from an old dog: An overview of TEI P5. In L. Burnard, M. Dobreva, N. Fuhr, and A. Lüdeling, editors, *Digital Historical Corpora- Architecture, Annotation, and Retrieval*, number 06491 in Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2007.

[6] J. Clark. SX — An SGML system conforming to the international standard ISO 8879 — Standard Generalized Markup Language. http://www.jclark.com/sp/sx.htm, 1997.

[7] D. Crockford. The application/json media type for javascript object notation (JSON). http://www.ietf.org/rfc/rfc4627.txt?number=4627, 2006.

[8] J. Ellson, E. Gansner, L. Koutsofios, S. C. North, and G. Woodhull. *Graphviz — Open Source Graph Drawing Tools*, pages 594–597. Springer Berlin / Heidelberg, 2002.

[9] T. Erl. *Service-Oriented Architecture. Concepts, Technology, and Design.* Prentice Hall, Upper Saddle River, 2004.

[10] S. Evert. Corpora and collocations. In A. Lüdeling and M. Kytö, editors, *Corpus Linguistics. An International Handbook of the Science of Language and Society.* Mouton de Gruyter, Berlin/New York, 2008.

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

[12] M. Fowler. *Patterns of Enterprise Application Architecture.* Addison-Wesley Longman, Amsterdam, 2002.

[13] J. Gippert. TITUS — Alte und neue Perspektiven eines indogermanistischen Thesaurus. *Studia Iranica, Mesopotamica et Anatolica*, 2:46–76, 2001.

[14] R. Gleim, A. Mehler, and H.-J. Eikmeyer. Representing and maintaining large corpora. In *Proceedings of the Corpus Linguistics 2007 Conference, Birmingham (UK)*, 2007.

[15] M. A. K. Halliday and R. Hasan. *Cohesion in English.* Longman, London, 1976.

[16] G. Heyer, U. Quasthoff, and T. Wittig. *Text Mining: Wissensrohstoff Text.* W3L, Herdecke, 2006.

[17] E. Hinrichs, J. Bartels, Y. Kawata, V. Kordoni, and H. Telljohann. The Tübingen treebanks for spoken german, english, and japanese. *Verbmobil: Foundations of Speech-to-Speech Translation*, pages 552–576, 2000.

[18] G. Hirst and D. St-Onge. Lexical chains as representations of context for the detection and correction of malaproprisms. In C. Fellbaum, editor, *WordNet — An Electronic Lexical Database*. MIT Press, Cambridge, Massachusetts, 1998.

[19] N. Ide. Linguistic annotation framework. Technical report, ISO/TC 37/SC4 N311, 2006.

[20] N. Ide and G. Priest-Dorman. Corpus encoding standard. http://www.cs.vassar.edu/CES/, 1998.

[21] B. Jussen, A. Mehler, and A. Ernst. A corpus management system for historical semantics. *Sprache und Datenverarbeitung. International Journal for Language Data Processing*, 31(1-2):81–89, 2007.

[22] T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Inf. Process. Lett.*, 31(1):7–15, 1989.

[23] C. H. A. Koster and E. Verbruggen. The AGFL grammar work lab. In *Proceedings FREENIX/Usenix 2002*, pages 13–18, 2002.

[24] L. Lemnitzer and C. Kunze. Adapting GermaNet for the Web. In *Proceedings of the First Global Wordnet Conference*, pages 174–181, Central Institute of Indian Languages, Mysore, India, 2002.

[25] A. Lüdeling, T. Poschenrieder, and L. C. Faulstich. DeutschDiachronDigital — Ein diachrones Korpus des Deutschen. *Jahrbuch für Computerphilologie*, pages 119–136, 2005.

[26] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. Building a large annotated corpus of english: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[27] O. Mason. Parameters of collocation: The word in the centre of gravity. In J. M. Kirk, editor, *Corpora Galore: Analyses and Techniques in Describing English*, pages 267–280. Rodopoi, Amsterdam, 1999.

[28] A. Mehler. Eigenschaften der textuellen Einheiten und Systeme / Properties of Textual Units and Systems. In R. Köhler, G. Altmann, and R. G. Piotrowski, editors, *Quantitative Linguistik. Ein internationales Handbuch / Quantitative Linguistics. An International Handbook*, pages 325–348. De Gruyter, Berlin/New York, 2005.

[29] A. Mehler. Preliminaries to an algebraic treatment of lexical associations. In C. Biemann and G. Paaß, editors, *Learning and Extending Lexical Ontologies. Proceedings of the Workshop at the 22nd International Conference on Machine Learning (ICML '05), August 7-11, 2005, Universität Bonn, Germany*, pages 41–47, 2005.

[30] A. Mehler. Large text networks as an object of corpus linguistic studies. In A. Lüdeling and M. Kytö, editors, *Corpus Linguistics. An International Handbook of the Science of Language and Society*, pages 328–382. De Gruyter, Berlin/New York, 2008.

[31] A. Mehler, R. Gleim, A. Ernst, and U. Waltinger. WikiDB: Building interoperable wiki-based knowledge resources for semantic databases. *Sprache*

und Datenverarbeitung. International Journal for Language Data Processing, 32(1):47–70, 2008.

[32] A. Mehler and C. Wolff. Einleitung: Perspektiven und Positionen des Text Mining. *LDV Forum – Zeitschrift für Computerlinguistik und Sprachtechnologie*, 20(1):1–18, 2005.

[33] J.-P. Migne, editor. *Patrologiae cursus completus: Series latina*, volume 1–221. Chadwyck-Healey, Cambridge, 1844–1855.

[34] D. Pieper and F. Summann. Bielefeld academic search engine (base): An end-user oriented institutional repository search service. *Library Hi Tech*, 24(4):614–619, 2006.

[35] B. Rieger. Warum fuzzy Linguistik? Überlegungen und Ansätze zu einer computerlinguistischen Neuorientierung. In D. Krallmann and H. W. Schmitz, editors, *Perspektiven einer Kommunikationswissenschaft. Internationales Gerold Ungeheuer Symposium, Essen 1995*, pages 153–183. Nodus, Münster, 1998.

[36] G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Addison Wesley, Reading, Massachusetts, 1989.

[37] J. F. Sowa. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks/Cole, Pacific Grove, 2000.

[38] T. Trippel. *The Lexicon Graph Model: A generic Model for multimodal lexicon development*. AQ-Verlag, Saarbrücken, 2006.

[39] T. Trippel, T. Declerck, and N. Ide. Interoperable language resource. *Sprache und Datenverarbetiung – International Journal for Language Data Processing*, 31(1-2):101–113, 2007.

[40] H. Uszkoreit, T. Brants, S. Brants, and C. Foeldesi. NEGRA Corpus. http://www.coli.uni-saarland.de/projects/sfb378/negra-corpus/, 2006.

[41] U. Waltinger and A. Mehler. Web as preprocessed corpus: Building large annotated corpora from heterogeneous web document data. In preparation, 2008.

[42] U. Waltinger and A. Mehler. Who is it? context sensitive named entity and instance recognition by means of Wikipedia. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence (WI-2008)*. 2008.

[43] U. Waltinger, A. Mehler, and G. Heyer. Towards automatic content tagging: Enhanced web services in digital libraries using lexical chaining. In *4th Int. Conf. on Web Information Systems and Technologies (WEBIST '08), 4-7 May, Funchal, Portugal*. Barcelona, 2008.

[44] U. Waltinger, A. Mehler, and M. Stührenberg. An integrated model of lexical chaining: Application, resources and its format. In A. Storrer, A. Geyken, A. Siebert, and K.-M. Würzner, editors, *Proceedings of KONVENS 2008 — Ergänzungsband Textressourcen und lexikalisches Wissen*, pages 59–70, 2008.

Relationserkennung auf deutschen Fließtexten

Maren Scheffel
Rheinische Friedrich-Wilhelms-Universität Bonn
maren.scheffel@uni-bonn.de

1 Einleitung

> Most of the web's content today is designed for humans to read, not for computer programs to manipulate meaningfully. Computers can adeptly parse Web pages for layout - here a header, there a link to another page - but in general, computers have no reliable way to process the semantics. [1]

Diese Aussage von Tim Berners-Lee über den Inhalt von Internetseiten lässt sich auf alle großen digitalisierten Dokumentsammlungen, wie zum Beispiel digitalisierte Archive von Zeitungen, übertragen: Ein Mensch erkennt, worum es in einem Text geht; ein Computerprogramm verarbeitet lediglich eine Reihenfolge von Zeichen. Um dieses Problem zu lösen, muss der semantische Gehalt jedes einzelnen Satzes maschinenlesbar gemacht werden. Dieser lässt sich besonders gut in Form von Relationen zwischen den in Texten erwähnten Personen oder Dingen, also den Entitäten, ausdrücken. Können diese Relationen aus einem Text extrahiert werden, können Aussagen zum Inhalt eines Textes getroffen werden.

Zur Erschließung eines Textes auf seiner Bedeutungsebene dient in der Linguistik die semantische Analyse. Die daraus gewonnenen semantischen Informationen ermöglichen z.B. die Suche nach Inhalten und Zusammenhängen. Werden aus einem Dokument oder einer Textsammlung inhaltliche Informationen zu komplexen Ereignissen extrahiert, reicht eine Suche nach einzelnen Wörtern oder Sätzen meist nicht aus. Konzentriert man sich jedoch auf die im Text befindlichen Relationen, ist dies sehr wohl möglich. Als Träger semantischer Informationen sind Relationen somit wichtig für die Erschließung der Bedeutung eines Textes.

Mit dieser Art der Problemstellung beschäftigt sich das Forschungsgebiet der Relationserkennung, welches im Rahmen der *Message Understanding Conferences* (MUCs) [2] als einer der fünf zentralen Aufgabenbereiche der Informationsextraktion festgelegt wurde [3]. Das Hauptaugenmerk bereits existierender Ansätze und Systeme liegt jedoch beinahe ausschließlich auf englischen Texten. Deutsche Texte finden kaum Beachtung.

Vor diesem Hintergrund stellt diese Arbeit heraus, wie Relationserkennung auf deutschen Fließtexten realisiert werden kann und inwieweit das Einbeziehen linguistischer Informationen in die Analyse die Qualität der Ergebnisse verändert.

2 Related Work

Auf Grund des Aufwandes werden nur noch wenige Systeme vollkommen manuell erstellt. Da auch überwachte Systeme noch ein relativ großes Maß an manueller Vorarbeit erfordern, sind die meisten Systeme zur automatischen Relationserkennung halb oder nicht überwacht. Viele dieser

Systeme werden auf Zeitungskorpora trainiert und getestet, was auf die MUCs zurückzuführen ist. Typische Beispiele für extrahierte Relationen sind Firmenfusionierungen oder -übernahmen, Firmenhauptsitze oder Staatsoberhäupter. Eine zweite große Domäne der Relationserkennung ist die Biologie. Aus Korpora medizinischer Zeitschriften oder anderer medizinischer Texten werden dabei vor allem Relationen gesucht, die Wechselwirkungen zwischen Proteinen oder Medikamenten beschreiben.

2.1 Regelbasierte Systeme

Das System von Blaschke u.a. [4] beschäftigt sich mit Proteinwechselwirkungen. Nach der Festlegung der gesuchten Proteinnamen, werden die Wechselwirkungen bestimmt, nach denen gesucht werden soll. Textabschnitte werden untersucht und dadurch manuell Regeln erstellt, die als Beispiele für weitere Texte dienen. Eine Erweiterung des Systems ist das *SUISEKI Information Extraction System* [5], bei dem die Regeln ebenfalls manuell erstellt, die Proteine aber automatisch aus dem Text gefiltert werden.

Ein weiteres regelbasiertes System ist DIPRE - *Dual Iterative Pattern Relation Expansion* - von Brin [6], welches auf dem *bootstrapping*-Prinzip basiert. Ziel der Anwendung ist es, Autor-Titel-Paare von Büchern im *World Wide Web* zu finden. Zu fünf manuell gewählten Paaren werden Vorkommen im Internet gesucht; aus den Erwähnungen werden Muster generiert, nach denen dann wieder gesucht wird. Dadurch werden neue Paare gefunden, die wiederum zum Finden neuer Muster verwendet werden.

Mit ihrem *Snowball*-System übernehmen Agichtein und Gravano [7] die Grundidee von DIPRE, fügen aber noch weitere Techniken hinzu. So werden die Dokumente von einem *Named-Entity*-Tagger bearbeitet. Auch die Pattern enthalten NE-Tags. Außerdem kann der Kontext der Regeln gewichtet werden, was eine Evaluation der Pattern und Tupel ermöglicht.

Ein ebenfalls auf *bootstrapping* und Patternverarbeitung basierendes System zur Extraktion von Relationen ist URES - *Unsupervised Web Relation Extraction System* - von Rosenfeld und Feldman [8]. Mit einer kleinen Anzahl an Beispielmustern werden Dokumente im Internet nach den gesuchten Relationen durchsucht. Die daraus erstellten Muster werden gelernt und schließlich die Relationen nach den Mustern aus den Sätzen extrahiert. Eine Erweiterung des Systems bietet SRES - *Self-supervised Relation Extraction System* [9].

2.2 Kernelbasierte Systeme

Viele Relationserkennungssysteme verwenden Kernelmethoden an Stelle von Regeln und *pattern-matching*-Algorithmen. Der Ansatz von Zelenko u.a. [10, 11] ist ein Beispiel für die Verwendung solcher Methoden. Texte werden von einem *shallow parser* analysiert und anschließend danach klassifiziert, ob eine gesuchte Relation enthalten ist oder nicht. Aus den Ergebnissen wird ein Modell generiert. Die zu vergleichenden Objekte bestehen aus Zeichenfolgen und eine Ähnlichkeitsfunktion berechnet die Anzahl der gemeinsamen Teilfolgen zweier Zeichenfolgen.

Auch Culotta und Sorensen [12] verwenden bei ihrem Relationsextraktionssystem Kernelmethoden. Da es sich um ein überwachtes Verfahren handelt, werden Parsebäume zuvor annotiert und mit einem Label markiert, wenn sie eine der später zu extrahierenden Relationen enthalten. Die Ergebnisse verschiedener *Tree*-Kernels schneiden bei der Evaluation besser ab als die Ergebnisse eines *bag-of-words*-Kernels, welcher keinerlei Informationen über die Struktur berücksichtigt.

Ein weiteres Beispiel einer Kernel-basierten Methode stammt von Zhao und Grishman [13]. Ihr Ansatz kombiniert mehrere Kernel miteinander, so dass Informationen aus der Satztokenisierung, dem Parsing und der Tiefenstrukturananlyse in die Analyse mit einbezogen werden können.

2.3 Weitere Systeme

Im Gegensatz zu den bisher genannten Systemen benötigt das von Hasegawa u.a. [14] entwickelte Verfahren keine Relationsbeispiele und ist somit ein nicht überwachtes Verfahren. Das Verfahren beruht auf der Annahme, dass Entitätenpaare, die in ähnlichen Kontexten auftreten, in Relationsclustern zusammengefasst werden können. Mit Hilfe eines Vektorraummodells und der Kosinusähnlichkeit wird überprüft, ob sich die Kontexte zwischen den Entitäten ähnlich genug sind, um geclustert werden zu können. Nach Abschluss des Clustering erhalten die Cluster ein Label, das aus den im Kontext eines Clusters am häufigsten vorkommenden Wörtern hervorgeht.

Diesen Ansatz greifen Zhang u.a. [15] auf, erweitern ihn aber um Parsebaumrepräsentationen der Relationen. Sie beziehen somit die Struktur des Kontextes zwischen den Entitäten bei der Ähnlichkeitsberechnung mit ein.

Mit URIES - *Unsupervised Relation Identification and Extraction System* - haben Rozenfeld und Feldman [16] ein System entwickelt, welches das *pattern learning* ihrer anderen Systeme mit der nicht überwachten Relationserkennung von Hasegawa u.a. verbindet.

2.4 Das WIKINGER-Projekt

Das Projekt WIKINGER[1] (*Wiki Next Generation Enhanced Repository*) ist ein Anwendungsbeispiel für Relationserkennung auf deutschen Texten, bei dem es sich um eine Zusammenarbeit des Fraunhofer-Instituts für Intelligente Analyse- und Informationssysteme IAIS[2], der Computerlinguistik der Universität Duisburg-Essen[3] und der Kommission für Zeitgeschichte KfZG[4] in Bonn handelt. Ziel ist es, nach der Relationserkennung mit Hilfe der extrahierten Relationen Ontologien zu erstellen. Als Korpus dienen etwa 10000 Kurzbiogramme in Fußnoten und Registerinformationen von 150 Veröffentlichungen der KfZG. Es werden alle Sätze weiterverarbeitet, die mindestens zwei Entitäten enthalten. Jeder Satz wird durch die in ihm enthaltenen Entitätenkategorien repräsentiert, die daraus entstehenden Elementmengen werden von einem *apriori*-Algorithmus bearbeitet, der wiederum Assoziationsregeln erstellt, welche dann nach verschiedenen Parametern kategorisiert werden können [17]. Die Assoziationsregeln dienen dem Cluster-Schritt als Eingabe. Die Wortvektoren der Sätze werden nach Kosinusähnlichkeit geclustert. Im letzten Schritt können dann die Entitäten und ihre Relationen in eine Ontologiesprache übersetzt werden, wodurch ein semantisches Netz entsteht. Der Teil des WIKINGER-Systems, der sich mit dem Erkennen und Clustern von Relationen beschäftigt, dient dem hier verfolgten Ansatz der Relationserkennung auf deutschen Fließtexten als Grundlage.

3 Ansatz

Eine statistische und für englische Texte erfolgreich verwendete Methode besteht darin, jeweils nur den Bereich eines Satzes in die Analyse miteinzubeziehen, der zwischen den Entitäten steht, und die daraus entstehenden *bags of words* nach ausgewählten Methoden zu clustern. Auf Grund der englischen Syntax ist die Wahrscheinlichkeit, dass ein Verb zwischen zwei Entitäten steht, relativ hoch. Im Deutschen hingegen ist dies nicht immer der Fall, vor allem nicht in Tempora, die mit Partizipien gebildet werden. Die folgenden Beispielsätze machen den Unterschied deutlich:

[1] `http://www.wikinger-escience.de`
[2] `http://www.iais.fraunhofer.de`
[3] `http://www.uni-due.de/computerlinguistik/`
[4] `http://www.kfzg.de`

(1) `Firma A hat Firma B gekauft.` $\Longrightarrow$ `hat(FirmaA, FirmaB)`

(2) `Company A has bought company B.` $\Longrightarrow$ `hasbought(companyA, companyB)`

Enthält der Kontextvektor nur die Wörter zwischen den Entitäten, könnte aus dem deutschen Satz nur die oben angegebene Relation extrahiert werden. Ist zusätzlich noch ein Datum oder Zeitpunkt angegeben, stehen im Deutschen noch weniger Informationen zur Verfügung, wie Satz (3) zeigt:

(3) `Gestern hat Firma A Firma B gekauft.` $\Longrightarrow$ `__(FirmaA, FirmaB)`

(4) `Yesterday company A has bought company B.` $\Longrightarrow$ `hasbought(companyA, companyB)`

Im Englischen steht das Verb in beiden Fällen zwischen Subjekt und Objekt, im Deutschen jedoch nicht. Zur Relationserkennung auf deutschen Texten sollte daher der ganze Satz in die Analyse mit einbezogen werden. Der im Folgenden beschriebene Ansatz basiert in seinen Grundzügen auf der Vorgehensweise und den Algorithmen der im WIKINGER-Projekt durchgeführten Relationserkennung.

Da es sich bei den dort analysierten Texten um Biogrammfußnoten, also keine Fließtexte handelt, ist es das Ziel, zu untersuchen, ob und wie sich die Algorithmen der Relationserkennung des WIKINGER-Projektes übernehmen, verändern und erweitern lassen, so dass eine automatische Relationserkennung auf deutschen Fließtexten möglich ist. Dabei wird, auf Grund der oben beschriebenen möglichen Problematik im Deutschen, nicht nur der Kontext zwischen den Entitäten, sondern der vollständige Satz in die Relationserkennung mit einbezogen. Neben den rein statistischen Analyseverfahren werden auch linguistische Aspekte berücksichtigt. So wird überprüft, ob sich die Ergebnisse der Relationserkennungs- und Clustervorgänge verbessern lassen, wenn die Wortvektoren beispielsweise lemmatisiert oder die Gewichtung einzelner Wortklassen darin erhöht werden.

3.1 Beschreibung des Ansatzes

Abbildung 1 zeigt den schematischen Arbeitsablauf des zu erstellenden Systems, welches sich in zwei Hauptteile gliedert: zunächst muss das Korpus ausgewählt und vorverarbeitet werden, anschließend können die Relationen erkannt und geclustert werden. Da es sich beim zweiten Teil um einen dynamischen Prozess handelt, der mit verschiedenen Einstellungen durchlaufen werden kann, wird an dieser Stelle eine Benutzeroberfläche zur Verfügung gestellt, was in der Abbildung durch den Rahmen um diesen Teil dargestellt ist.

Als Korpus dienen Zeitungsartikel. Diese sind thematisch relativ breit gefächert und sollen den Leser darüber informieren, wer was wann wie und wo getan hat. Da linguistische Aspekte in die Analyse mit einbezogen werden, wird das Korpus einer Vorverarbeitung unterzogen. Relationen sollen immer nur zwischen den Entitäten eines Satzes gefunden werden. Das Korpus wird daher zunächst in einzelne Sätze geteilt, so dass jeder Satz als unabhängig von den ihn umgebenden Sätzen zu sehen ist. Um einzelnen Wörtern innerhalb eines Satzes linguistische Informationen hinzufügen zu können, werden die Sätze tokenisiert. Dadurch kann jedes Token einzeln betrachtet und annotiert werden. Zusätzlich zu den Token werden noch Lemma und Wortart annotiert. Danach erfolgt die Markierung der Entitäten.

Anschließend können die Sätze des Korpus zunächst die Algorithmen zur Erstellung von Assoziationsregeln und anschließend die Clusteralgorithmen durchlaufen. Um Regelmäßigkeiten in Häufigkeiten gemeinsamen Auftretens von Entitäten zu ermitteln, kommt der *apriori*-Algorithmus von Agrawal und Srikant [18] zum Einsatz. Für diesen Algorithmus werden nur jene Sätze beachtet, die mindestens zwei Entitäten enthalten. Alle anderen können ignoriert werden. Jeder noch

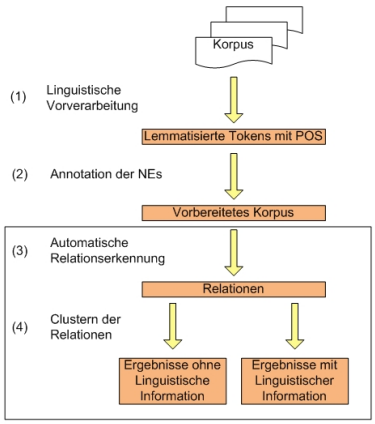


Abbildung 1: Schematischer Arbeitsablauf des Systems

verbleibende Satz wird durch die in ihm vorkommenden Entitäten, welche Items genannt werden, repräsentiert. Die daraus entstehenden so genannten Itemsets dienen dem *apriori*-Algorithmus als Eingabe, um Assoziationsregeln der Form $x \Rightarrow y$ für die Items zu erstellen. Da auch mehrstellige Relationen erkannt werden sollen, kann der Konsequent einer Assoziationsregel aus mehr als einem Element bestehen.

Um herauszufinden, wieviele und welche Relationen in den von einer Assoziationsregel abgedeckten Sätzen enthalten sind, wird die Ähnlichkeit der Sätze zueinander berechnet, so dass Cluster gebildet werden können. Dies wird dadurch erreicht, dass die Kontexte der Entitäten, also die übrigen im Satz enthaltenen Wörter, miteinander verglichen werden. In jedem Satz werden die Instanzen der Entitäten durch ihre Entitätsklasse ersetzt. Anschließend wird für jeden Satz ein Wortvektor mit den entsprechenden Gewichten der Wörter erstellt. Die Gewichte werden mit Hilfe des $tf * idf$-Algorithmus errechnet, bilden also eine Kombination aus Termhäufigkeit und inverser Dokumenthäufigkeit.

Bei dem in diesem Ansatz verwendeten Clusterverfahren handelt es sich um das so genannte agglomerative Clustern. Zu Beginn des Clustervorgangs bildet jeder Wortvektor ein eigenes Cluster. Die zwei Cluster, die sich am meisten ähneln, werden zu einem zusammengefasst. Dieser Prozess wird so lange wiederholt, bis das Abbruchkriterium erfüllt oder nur noch ein einziges Cluster übrig ist. Es gibt mehrere Methoden, den Abstand zweier Cluster zu berechnen. Für den hier vorgestellten Ansatz werden zwei Methoden verwendet: *single linkage* und *complete linkage*.

$$\text{single linkage: } d(A,B) = min\left\{d(a_j, b_k) : a_j \in A, b_k \in B\right\}$$
$$\text{complete linkage: } d(A,B) = max\left\{d(a_j, b_k) : a_j \in A, b_k \in B\right\}$$

Als Distanzfunktion d, die zwei Wordvektoren miteinander vergleicht, dient die Kosinusähnlichkeit $\frac{A \cdot B}{|A||B|}$. Dies ergibt sich aus dem Skalarprodukt zweier normalisierter Vektoren.
Für zwei Vektoren $\vec{a_j} = (w_{1,j}, w_{2,j}, w_{3,j}..., w_{n,j})$ und $\vec{b_k} = (w_{1,k}, w_{2,k}, w_{3,k}, ..., w_{n,k})$ gilt:

$$sim(\vec{a_j}, \vec{b_k}) = \vec{a_j} \cdot \vec{b_k} = \sum_{i=1}^{N} w_{i,j} \times w_{i,k}$$

Ein Aspekt dieser Arbeit besteht darin, zu überprüfen, inwiefern sich die Ergebnisse der oben beschriebenen automatischen Relationserkennungs- und Clustervorgänge verändern, wenn linguistische Informationen über die einzelnen Token bei der Erstellung der Wortvektoren berücksichtigt

werden. Dass eine Änderung der Wortvektoren Einfluss auf das Ergebnis haben wird, ist anzunehmen. Die Frage ist, ob die Auswirkung positiver oder negativer Art ist. Da ein Mehr an Information im Rahmen von Analyseprozessen im Allgemeinen zu detaillierteren Aussagen führt, wird erwartet, dass die Einbindung sprachlicher Aspekte ebenfalls eher dazu beiträgt, bessere und somit angemessenere Relationscluster zu generieren. Es werden drei verschiedene Ansätze verfolgt: Stoppwortentfernung, Lemmatisierung der Wortvektoren und Boosten der Vollverben.

Neben der einzelnen Anwendung dieser Ansätze, besteht die Möglichkeit, sie zu kombinieren, um die Ergebnisse der automatischen Relationserkennung noch weiter zu verbessern. Es ergeben sich dadurch insgesamt acht Kombinationsmöglichkeiten für die Erstellung der Wortvektoren. Dadurch dass es zwei verschiedene Clusteralgorithmen gibt (*single linkage* und *complete linkage*), ergeben sich insgesamt 16 mögliche Kombinationen von Clustermethoden und linguistischen Zusatzpaketen:

Clustermethode	**Basis plus möglicher Zusatz**	**Kürzel**
single linkage	W/L	W/L_SL
	W/L mit Stoppwortentfernung	W/L_ST_SL
	W/L mit Stoppwortentfernung und Verb-Boost	W/L_ST_V_SL
	W/L mit Verb-Boost	W/L_V_SL
complete linkage	W/L	W/L_CL
	W/L mit Stoppwortentfernung	W/L_ST_CL
	W/L mit Stoppwortentfernung und Verb-Boost	W/L_ST_V_CL
	W/L mit Verb-Boost	W/L_V_CL

Tabelle 1: Kombinationsmöglichkeiten beim Durchlaufen des Programms; W=Wort, L=Lemma

3.2 Umsetzung des Ansatzes

Das verwendete Korpus entstammt dem *TIGER*-Projekt[5] und ist eine so genannte Baumbank, also ein syntaktisch vollständig annotiertes Korpus. Bei den ausgewählten Texten handelt es sich um ganze Artikel aus der *Frankfurter Rundschau*, welche verschiedenen Domänen entnommen wurden. POS-Tags und Lemmata der einzelnen Wörter waren somit schon vorhanden. Da die Art und Weise der Eigennamenannotation im *TIGER*-Korpus diesem Ansatz jedoch nicht genügt, müssen die Entitäten nachträglich annotiert werden. Um den Zeitaufwand in Grenzen zu halten, wurden die ersten 5000 von über 50000 Sätzen des Korpus ausgewählt, wodurch die Abdeckung einer genügend großen Menge an Relationen weiterhin gewährleistet sein sollte. Folgende Entitätsklassen wurden verwendet: PERSON, ORGANISATION, EINRICHTUNG, GPE, ORT, ROLLE, EREIGNIS, DATUMZEIT.

Über eine GUI können die verschiedenen Parametereinstellungen und Schwellwerte zu den Algorithmen eingegeben und die einzelnen Schritte somit verfolgt werden. Schließlich erhält man so eine Liste der erstelleten Cluster mit den darin enthaltenen Relationsbeispielen.

4 Evaluierung

Um die Performanz des Relationserkennungssystems zu testen, werden die Ergebnisse mit einem manuell erstellten Goldstandard verglichen. Für die Evaluierung wurden zwei Regeln ausgewählt. Die erste Wahl fiel auf die Regel `Rolle` ⇒ `[Person]`. Weiterhin wurde die Regel `Rolle` ⇒ `[Gpe,`

[5] `http://www.ims.uni-stuttgart.de/projekte/TIGER/`

`Person]` gewählt, da es sich um eine dreistellige Regel handelt, die gleichzeitig eine Spezialisierung der ersten Regel ist. Es ist zu vermuten, dass speziellere Regeln auch speziellere Relationen enthalten als allgemeinere Regeln.

Zur Ermittlung des Goldstandards wurden die Sätze der beiden ausgewählten Regeln manuell geclustert. Da auch das System nur die Cluster ausgeben soll, die mehr als zwei Sätze beinhalten, musste ein Relationscluster auch beim Goldstandard aus wenigstens zwei Sätzen bestehen. Blieben am Ende einzelne Sätze übrig, da sie keinem Cluster ähnlich genug waren und somit ein eigenständiges Cluster hätten bilden müssen, wurden diese verworfen und für den weiteren Verlauf der Evaluierung ignoriert. Für die erste Regel wurden 28 Cluster gebildet, für die zweite Regel waren es 14.

Um die Qualität der Ergebnisse zu evaluieren, werden die Maße Recall, Precision und F-Maß der einzelnen Kombinationen ermittelt. Einmal geschieht dies auf globaler Ebene und einmal auf der Ebene einer einzelnen Relation.

4.1 Evaluierung auf globaler Ebene

Bei der Evaluierung auf globaler Ebene wird die Performanz des Systems insgesamt analysiert. Dazu durchlaufen beide gewählten Regeln mit jeder möglichen Einstellung das System. In den ausgegebenen Relationsclusterlisten wird jeweils zunächst gezählt, wie viele Cluster erstellt wurden. Anschließend wird mittels des manuell erstellten Goldstandards überprüft, wie viele der gefundenen Cluster für die aktive Regel korrekt ermittelt wurden. Für den Fall, dass das System mehrere Cluster zu einer Relation des Goldstandards bildet, wird das Cluster als korrekt gebildet erkannt. Zur Anzahl der korrekt gebildeten Cluster zählt jedoch nur eins dieser Cluster. Für die Evaluationsmaße bedeutet dies:

$$\text{Recall } R = \frac{\#\text{ korrekter Cluster}}{\#\text{ Cluster in Goldstandard}} \qquad \text{Precision } P = \frac{\#\text{ korrekter Cluster}}{\#\text{ ermittelter Cluster}} \qquad \text{F-Maß } F = \frac{2PR}{P+R}$$

Tabelle 2 zeigt die Ergebnisse für die erste Regel, `Rolle` ⇒ `[Person]`. Die jeweils höchsten Werte für Recall, Precision und F-Maß sind hervorgehoben. Demnach werden für diese Relation die besten Resultate erreicht, wenn auf der Basis von Wörtern geclustert und den Verben zusätzliches Gewicht gegeben wird. Vergleicht man die beiden linkage-Methoden miteinander, fällt auf, dass mit *complete linkage* deutlich bessere Werte erziehlt werden.

Kombination	**R**	**P**	**F**	**Kombination**	**R**	**P**	**F**
Wort_SL	0,36	0,38	0,37	Wort_CL	0,43	0,27	0,33
Wort_ST_SL	0,36	0,36	0,36	Wort_ST_CL	0,32	0,24	0,27
Wort_ST_V_SL	0,64	0,53	0,58	Wort_ST_V_CL	0,68	0,45	0,54
Wort_V_SL	**0,68**	**0,58**	**0,63**	Wort_V_CL	**0,71**	**0,53**	**0,61**
Lemma_SL	0,25	0,28	0,26	Lemma_CL	0,43	0,23	0,30
Lemma_ST_SL	0,29	0,32	0,30	Lemma_ST_CL	0,39	0,23	0,29
Lemma_ST_V_SL	0,25	0,33	0,28	Lemma_ST_V_CL	0,61	0,36	0,45
Lemma_V_SL	0,25	0,33	0,28	Lemma_V_CL	0,61	0,36	0,45

Tabelle 2: Recall, Precision und F-Maß aller möglichen Kombinationen für die Regel `Rolle` ⇒ `[Person]`

Insbesondere ist anzumerken, dass der Recall beim *complete linkage* immer dann höher ausfällt, wenn der Verb-Boost aktiviert wird. Beim *single linkage* wirkt sich dieser nur auf eine Wort-, nicht jedoch auf eine Lemmabasis aus. Die generell relativ niedrigen Werte der Lemma-SL-Kombinationen sind darauf zurückzuführen, dass sich beim *single linkage* nur die zwei einander am nächsten stehenden Elemente zweier Cluster ähnlich genug sein müssen, um aus den beiden

Clustern ein neues zu bilden. Alle Lemma-SL-Kombinationen dieser Regel enthalten mindestens ein Cluster, in dem über 30, manchmal sogar 50 Sätze in einer Relation zusammengefasst wurden. Es lässt sich daher festhalten, dass *single linkage* im Schnitt weniger geeignet zu sein scheint, als *complete linkage*.

Das Entfernen von Stoppwörtern hat kaum positiven Einfluss auf die Ergebnisse. Bei der Verwendung von *complete linkage* wirkt es sich sogar negativ aus, da Recall, Precision und F-Maß im Vergleich zu den Werten der reinen Wort- oder Lemmabasis abfallen. Beim *single linkage* auf Wortbasis bewirkt die Stoppwortentfernung kaum eine Änderung der Ergebnisse; auf Lemmabasis heben sich die Werte nicht stark genug, um zu zeigen, dass eine Stoppwortentfernung die Ergebnisse zusätzlich verbessert.

Die Resultate für die zweite Regel, `Rolle` ⇒ `[Gpe, Person]`, sind in Tabelle 3 aufgeführt. Sie unterscheiden sich in mehreren Aspekten von denen der ersten Regel. Auch hier ist der beste Wert des jeweiligen Maßes hervorgehoben.

Kombination	**R**	**P**	**F**	**Kombination**	**R**	**P**	**F**
Wort_SL	0,36	0,45	0,40	Wort_CL	0,43	0,50	0,46
Wort_ST_SL	0,36	0,50	0,42	Wort_ST_CL	0,36	0,42	0,39
Wort_ST_V_SL	0,29	0,44	0,35	Wort_ST_V_CL	0,28	0,36	0,32
Wort_V_SL	0,29	0,44	0,35	Wort_V_CL	0,29	0,40	0,34
Lemma_SL	**0,50**	0,63	0,56	Lemma_CL	**0,57**	**0,53**	**0,55**
Lemma_ST_SL	**0,50**	**0,70**	**0,58**	Lemma_ST_CL	0,43	0,46	0,44
Lemma_ST_V_SL	**0,50**	0,47	0,48	Lemma_ST_V_CL	0,50	0,33	0,40
Lemma_V_SL	**0,50**	0,47	0,48	Lemma_V_CL	0,50	0,33	0,40

Tabelle 3: Recall, Precision und F-Maß aller möglichen Kombinationen für die Regel `Rolle` ⇒ `[Gpe, Person]`

Hier erzielten sowohl beim *single* als auch beim *complete linkage* Lemma-Kombinationen die besseren Ergebnisse: beim *single linkage* ist das beste Resultat die Kombination aus Lemmabasis und Stoppwortentfernung, beim *complete linkage* die Lemmabasis ohne Erweiterung. Besonders der positive Effekt der Stoppwortentfernung steht im Gegensatz zu den Ergebnissen der ersten Regel. Da es sich bei der zweiten Regel um eine Spezialisierung der ersten handelt und somit zu vermuten ist, dass auch die darin enthaltenen Relationen spezieller sind, könnte sich der positive Effekt dadurch erklären, dass die speziellen Relationen durch die Stoppwortentfernung in Kombination mit der Lemmatisierung auf ihre wesentlichen Merkmale reduziert werden: Sätze, die sich vorher nicht ähnlich genug waren, um korrekt geclustert zu werden, sind nun speziell genug, so dass keine oder nur noch sehr wenige falsche Sätze zusammen mit ihnen in ein Cluster sortiert werden.

Ebenfalls bemerkenswert ist, dass innerhalb der beiden Clustermethoden kein Unterschied sichtbar wird, wenn neben der Erhöhung der Verbgewichte bei einer Lemmabasis auch die Stoppwortentfernung angewendet wird. An dieser Stelle bietet die Stoppwortentfernung also keinen Vorteil. Durch das Boosten der Verben ist ihr neues Gewicht so stark, dass die Gewichte der Stoppwörter auch dann keine unmittelbaren Auswirkungen haben, wenn sie sich noch im Satz befinden. Da dieses Phänomen auch bei der ersten Regel zu beobachten war, lässt sich daraus schließen, dass eine Stoppwortentfernung bei geboosteten Verben und Lemmabasis nicht in jedem Fall von Nöten ist.

Des Weiteren fällt im Vergleich zu den Ergebnissen der ersten Regel auf, dass die Kombinationen, die dort gute Resultate lieferten, also Wort-Verb-Boost-Kombinationen, bei der zweiten Regel die schlechtesten Ergebnisse liefern, unabhängig von der Clustermethode. Grund hierfür könnte einmal mehr der speziellere Charakter der Relationen einer spezielleren Regel sein. Zu vermuten ist, dass schwächere Distanzmaße beim Clusteralgorithmus bessere Ergebnisse erzielen.

Vor allem beim *complete linkage* steigt die Zahl der erstellten Cluster stark an, je niedriger der Maximalabstand gesetzt wird.

4.2 Evaluierung auf Relationsebene

Für die Evaluierung auf Relationsebene wurde zunächst für beide Regeln das Cluster ausgewählt, welches im Goldstandard die meisten Sätze enthielt. In beiden Fällen war dies das mit SAGEN betitelte Relationscluster. Die Systemausgaben aller möglichen Kombinationen wurden dahingehend überprüft, ob Cluster enthalten waren, welche mit dem Label SAGEN versehen werden konnten. Zunächst wurden alle Sätze in jedem SAGEN-Cluster gezählt; daran anschließend wurde ermittelt, wieviele dieser Sätze sich im Goldstandard dieses Clusters bei der jeweiligen Regel befanden. Diese Sätze konnten dann als korrekte Sätze markiert werden. Für die Berechnung der Evaluationsmaße ergibt sich daraus Folgendes:

$$\text{Recall } R = \frac{\#\text{ korrekte Sätze}}{\#\text{ Sätze in Goldstandard}} \qquad \text{Precision } P = \frac{\#\text{ korrekte Sätze}}{\#\text{ ermittelte Sätze}} \qquad \text{F-Maß } F = \frac{2PR}{P+R}$$

Die Tabellen 4 und 5 listen die Ergebnisse dieser Evaluierung auf. Die besten Werte jedes Maßes sind auch hier hervorgehoben. Im Vergleich mit den Ergebnissen aus der Gesamtevaluierung fällt sofort auf, dass auf der Relationsebene teilweise eindeutig höhere Werte für Recall, Precision und F-Maß erreicht werden.

Kombination	**R**	**P**	**F**	**Kombination**	**R**	**P**	**F**
Wort_SL **(2)	0,38	0,18	0,24	Wort_CL **(5)	0,13	**1**	0,23
Wort_ST_SL **(1)	0,25	0,40	0,31	Wort_ST_CL *(6)	0,06	0,5	0,12
Wort_ST_V_SL	**0,88**	0,74	0,80	Wort_ST_V_CL	0,81	**1**	0,90
Wort_V_SL	**0,88**	**0,93**	**0,90**	Wort_V_CL	0,81	**1**	0,90
Lemma_SL	0,44	0,13	0,20	Lemma_CL *(5)	0,06	0,5	0,12
Lemma_ST_SL	0,31	0,15	0,20	Lemma_ST_CL *(5)	0,06	0,5	0,12
Lemma_ST_V_SL	0,63	0,19	0,29	Lemma_ST_V_CL	**0,88**	**1**	**0,94**
Lemma_V_SL	0,81	0,27	0,41	Lemma_V_CL	**0,88**	**1**	**0,94**

Tabelle 4: Recall, Precision und F-Maß aller möglichen Kombinationen für das Relationscluster SAGEN der Regel `Rolle` ⇒ `[Person]`; in den mit *(X) markierten Reihen gab es jeweils X Cluster zu dieser Relation, die alle die angegebenen Werte hatten; in den mit **(X) markierten Reihe gab es neben den angegeben Werten zusätzlich X Cluster zu dieser Relation mit den Werten R=0,06 - P=0,5 - F=0,12

Für das Relationscluster SAGEN der ersten Regel (Tabelle 4) variieren die Werte sehr stark. Die besten Ergebnisse mit *single linkage* liefert die Kombination aus Wortbasis und Verb-Boost, gefolgt von Verb-Boost mit Stoppwortentfernung auf Wortbasis. Die gleichen Einstellungen erzielen auch beim *complete linkage* sehr gute Ergebnisse, dort beträgt die Precision sogar 100%. Dieses Verhalten ist mit den Ergebnissen aus der Evaluierung der Regel auf globaler Ebene vergleichbar. Noch besser sind die Resultate beim *complete linkage* bei den Lemma-Verb-Kombinationen, da dort ein F-Maß von 0,94 erreicht wurde. Die Lemma-Verb-Kombinationen beim *single linkage* hingegen haben zwar relativ hohe Recall-Werte, jedoch eine sehr schlechte Precision. Wie auch schon bei der Gesamtevaluierung führt *single linkage* hier zu sehr großen Clustern von 30, 40 oder gar 50 Sätzen. Dass in einer so großen Ansammlung von lemmatisierten Sätzen, in denen die Verben zusätzlich höher gewichtet wurden, auch viele Sätze der SAGEN-Relation enthalten sind, ist nicht verwunderlich und erklärt den großen Unterschied zwischen Precision und Recall. Auch die niedrigen Werte der anderen beiden Lemma-Kombinationen mit *single linkage* sind auf sehr große Cluster zurückzuführen.

Die sehr niedrigen Werte, vor allem bei Wort- und Lemmabasen ohne Zusatz oder mit Stoppwortentfernung, kommen dadurch zu Stande, dass nicht ein einziges, sondern viele kleine SAGEN-Cluster erstellt wurden (in der Tabelle durch * und ** markiert).

Für das Relationscluster SAGEN der zweiten Regel (Tabelle 5) lässt sich festhalten, dass nur dann gute Ergebnisse erzielt werden, wenn ein Verb-Boost stattfindet. In allen anderen Fällen sind die Resultate entweder relativ schlecht oder aber gar nicht vorhanden. Keine der Wortbasis-Kombinationen ohne Verb-Boost, unabhängig von der Clustermethode, war in der Lage, das SAGEN-Cluster zu bilden.

Kombination	**R**	**P**	**F**	**Kombination**	**R**	**P**	**F**
Wort_SL	/	/	/	Wort_CL	/	/	/
Wort_ST_SL	/	/	/	Wort_ST_CL	/	/	/
Wort_ST_V_SL	0,83	**0,83**	0,83	Wort_ST_V_CL	0,75	**1**	0,86
Wort_V_SL	0,83	**0,83**	0,83	Wort_V_CL	0,75	**1**	0,86
Lemma_SL	0,17	0,18	0,17	Lemma_CL	0,17	**1**	0,29
Lemma_ST_SL	0,17	0,25	0,2	Lemma_ST_CL *(5)	0,17	**1**	0,29
Lemma_ST_V_SL	**1**	0,75	**0,86**	Lemma_ST_V_CL	**0,83**	**1**	**0,91**
Lemma_V_SL	**1**	0,75	**0,86**	Lemma_V_CL	**0,83**	**1**	**0,91**

Tabelle 5: Recall, Precision und F-Maß aller möglichen Kombinationen für das Relationscluster SAGEN der Regel `Rolle` ⇒ `[Gpe, Person]`; in der mit *(5) markierten Reihe gab es 5 Cluster zu dieser Relation, die alle die angegebenen Werte hatten

Wie schon bei den Ergebnissen der Gesamtanalyse, so zeigt sich auch hier, dass das Hinzuziehen der Stoppwortentfernung keinerlei Einfluss auf die Resultate einer Lemma-Verb-Boost-Kombination hat. Im Falle des zweiten SAGEN-Clusters trifft dies sogar auf die Wort-Verb-Boost-Kombinationen zu. Daraus lässt sich schließen, dass die Stoppwortentfernung lediglich dann Vorteile bringen kann, wenn sie unabhängig vom Verb-Boost eingesetzt wird.

Des Weiteren zeigt sich hier ebenfalls deutlich, dass das System durchaus in der Lage ist, bei einzelnen Clustern sehr hohe F-Maße zu erreichen. Einige kleine Cluster, die sehr gleichmäßig strukurierte Sätze beinhalten, werden meistens richtig erkannt und haben daher oft ein hohes F-Maß. Dass dies auch bei größeren Clustern, wie dem analysierten SAGEN-Cluster gelingt, zeigt, dass das Boosten der Verben eine Möglichkeit darstellt, die Resultate zu verbessern.

4.3 Ergebnis der Evaluierung

Die Evaluierung des Ansatzes hat gezeigt, dass eine Umsetzung des hier vorgestellten Ansatzes zu einem funktionierenden Relationserkennungssystem grundsätzlich möglich ist. Die aus dem WIKINGER-Projekt übernommenen Algorithmen wurden erfolgreich verändert, angepasst und ergänzt, so dass Relationen in deutschen Fließtexten erkannt und in Clustern zusammengefasst werden konnten.

Als weiteres Ergebnis der Evaluierung ist festzuhalten, dass der Einsatz linguistischer Verfahren während des Relationserkennungsprozesses die Qualität der Ergebnisse deutlich steigert. Dabei ist es jedoch nicht möglich, eines dieser Verfahren als 'Sieger' der Evaluierung herauszustellen, da sein erfolgreicher Einsatz auch von der Art der für die Analyse gewählten Assoziationsregel abhängt. Ein deutlich zu bevorzugendes Verfahren gibt es somit nicht.

Die Analyse der Ergebnisse der allgemeineren Regel zeigen, dass weder Lemmatisierung, noch Stoppwortentfernung ein gutes Resultat liefern, da sich hier das F-Maß nicht signifikant steigern lässt. Werden jedoch die Verben geboostet, verbessert sich das F-Maß im Vergleich zur reinen Wortbasis um fast 90% von 37% auf 63% bei *single linkage* und von 33% auf 61% mit *complete linkage*. Bei der zweiten, spezielleren Regel sind es wiederum gerade Lemmatisierung und zum

Teil die Stoppwortentfernung, die das Ergebnis verbessern. Bei *single linkage* steigt das F-Maß von 40% auf 58%, bei *complete linkage* von 46% auf 55%. Die Verbesserung beträgt dort somit um die 50% bzw 25%. Betrachtet man einzelne Relationscluster, wird deutlich, dass sowohl Cluster von allgemeineren als auch solche von spezielleren Regeln F-Maße erreichen können, die über 90% liegen. Auch hier wird dies durch das Verb-Boosten erreicht.

Diese Beobachtungen können als Indiz dafür gesehen werden, dass bei allgemeineren Regeln diejenigen Verfahren besser funktionieren, die dem Träger des semantischen Gehalts eines Satzes gezielt mehr Bedeutung zukommen lassen. Dazu müssen diese jedoch dem System zuvor bekannt sein. Bei spezielleren Regeln sind hingegen solche Verfahren wie Lemmatisierung und Stoppwortentfernung hilfreich, die den Satz generalisieren und somit ein Clustern überhaupt erst ermöglichen.

5 Schluss

Im Rahmen der Evaluierung wurde deutlich, dass eine Relationserkennung auf deutschen Fließtexten mit statistisch arbeitenden Algorithmen möglich ist. Es konnte zudem gezeigt werden, dass die Qualität der Ergebnisse durch die Berücksichtigung linguistischer Aspekte bei der Analyse der Texte deutlich gesteigert werden kann. Dabei wurde festgestellt, dass zwischen verschiedenen Regeltypen zu unterscheiden ist, da allgemeiner formulierte Regeln andere Verfahren fordern, um gute Ergebnisse zu erzielen, als spezieller formulierte Regeln.

Interessanter Gegenstand einer weiteren wissenschaftlichen Auseinandersetzung könnte genau dieser Unterschied im Verhalten verschiedener Regeltypen auf verschiedene linguistische Verfahren sein. Es wäre zu überprüfen, ob sich das beobachtete Verhalten verallgemeinern lässt, und den verschiedenen Regeltypen feste linguistische Verfahren zugewiesen werden können.

Auch eine Untersuchung zur Verwendung weiterer linguistischer Verfahren könnte die Ergebnisse noch weiter verbessern. So ist anzunehmen, dass besonders das Einbeziehen der Satzstruktur, wie es bei einigen Verfahren auf englischen Texten bereits getestet wurde, auch bei deutschen Texten zu einer Steigerung der Qualität der Ergebnisse führen wird. Eine weitere Möglichkeit zur Erweiterung des Systems besteht im Zusammenfassen synonymer Verben, da die Zahl der Relationscluster dadurch reduziert würde, was in einer Verbesserung der Ergebnisse resultieren sollte.

Abschließend lässt sich festhalten, dass die Kombination aus informationstechnologischen und linguistischen Verfahren für Relationserkennungssysteme von Vorteil ist und die Forschung auf dem Gebiet der Relationserkennung interdisziplinäre Ansätze weiter verfolgen sollte.

Literatur

[1] Berners-Lee, Tim; Hendeler, James; Lassila, Ora: „The Semantic Web“. In: *Scientific American*. May, 2001.

[2] SAIC (Hrsg.): *Proceedings of the Seventh Message Understanding Conference MUC-7*. `http://www.itl.nist.gov/iaui/894.02/related_projects/muc/index.html`. 1998. - Zugriffsdatum: 30.01.2009

[3] Cunningham, Hamish: „Information Extraction, Automatic“. In: *Encyclopedia of Language and Linguistics*. 2nd Edition. Amsterdam [u.a.]: Elsevier, 2006, S. 665-677

[4] Blaschke, Christian u.a.: „Automatic Extraction of Biological Information from Scientific Text: Protein-Protein Interactions“. In: *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*. Menlo Park: AAAI Press, 1999, S. 60-67

[5] Blaschke, Christian; Valencia, Alfonso: „The Frame-Based Module of the SUISEKI Information Extraction System“. In: *IEEE Intelligent Systems* 17 (2002), Nr. 2, S. 14-20

[6] Brin, Sergey: „Extracting Patterns and Relations from the World Wide Web“. In: *WebDB '98: Selected papers from the International Workshop on The World Wide Web and Databases*. London, UK: Springer-Verlag, 1999, S. 172-183

[7] Agichtein, Eugene; Gravano, Luis: “SNOWBALL: Extracting Relations from Large Plain-Text Collections“. In: *DL '00: Proceedings of the fifth ACM Conference on Digital libraries*. New York, NY, USA: ACM, 2000, S. 85-94

[8] Rosenfeld, Benjamin; Feldman, Ronen: „URES: an Unsupervised Web Relation Extraction System“. In: *Proceedings of the COLING/ACL Main Conference Poster Sessions*. Morristown, NJ, USA: Association for Computational Linguistics, 2006, S. 667-674

[9] Feldman, Ronen u.a.: „Self-supervised Relation Extraction from the Web“. In: Esposito, Floriana u.a. (Hrsg.): *ISMIS* Bd. 4203, Springer, 2006, S. 755-764

[10] Zelenko, Dmitry; Aone, Chinatsu; Richardella, Anthony: „Kernel Methods for Relation Extraction“. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Morristown, NJ, USA: Association for Computational Linguistics, July 2002, S. 71-78

[11] Zelenko, Dmitry; Aone, Chinatsu; Richardella, Anthony: „Kernel Methods for Relation Extraction“. In: *Journal of Machine Learning Research* 3 (2003), S. 1083-1106

[12] Culotta, Aron; Sorensen, Jeffrey: „Dependency Tree Kernels for Relation Extraction“. In: *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 2004, S. 423-429

[13] Zhao, Shubin; Grishman, Ralph: „Extracting Relations with Integrated Information Using Kernel Methods“. In: *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 2005, S. 419-426

[14] Hasegawa, Takaaki; Sekine, Satoshi; Grishman, Ralph: „Discovering Relations Among Named Entities From Large Corpora“. In: *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 2004, S. 415-422

[15] Zhang, Min u.a.: „Discovering Relations between Named Entities from a Large Raw Corpus Using Tree Similarity-based Clustering“. In: *Proceedings of the 2nd International Joint Conference on Natural Language Processing (IJCNLP'05)* Bd. 3651, Springer, 2005, S. 378-389

[16] Rozenfeld, Binjamin; Feldman, Ronen: „High-Performance Unsupervised Relation Extraction from Large Corpora“. In: *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*. Washington, DC, USA : IEEE Computer Society, 2006, S. 1032-1037

[17] Bröcker, Lars; Rössler, Marc; Wagner, Andreas: „Knowledge Capturing Tools for Domain Experts - Exploiting Named Entity Recognition and n-ary Relation Discovery for Knowledge Capturing in E-Science“. In: *Proceedings of Semantic Authoring, Annotation and Knowledge Markup Workshop (SAAKM)*. Whistler, Canada, October 2007

[18] Agrawal, Rakesh; Srikant, Ramakrishnan: „Fast Algorithms for Mining Association Rules in Large Databases“. In: *VLDB '94: Proceedings of the 20th International Conference on Very Large Data Bases*. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 1994, S. 487-499

Extracting Professional Preferences of Users from Natural Language Essays

Cigdem TOPRAK [1], Christof MÜLLER and Iryna GUREVYCH
Ubiquitous Knowledge Processing (UKP) Lab, Computer Science Department
Technical University of Darmstadt, Germany
www.ukp.tu-darmstadt.de

Abstract. This paper presents an unsupervised sentiment analysis approach for extracting professional preferences of users from natural language essays in a career recommendation scenario. Our system first extracts terms facilitating career recommendation such as objects, activities, hobbies, and places from the essays. Then, it applies a lexicon-based sentiment analysis approach to assign polarities representing user preferences.

Keywords. unsupervised sentiment analysis, information extraction

Introduction

Subjectivity and sentiment analysis are computational linguistics tasks focusing on the automatic analysis of subjective content in text. Subjectivity analysis aims at automatically distinguishing subjective content (opinions) from objective content (factual information). Sentiment analysis, on the other hand, involves additional subtasks such as: *(i)* determining the emotional orientation (polarity) of the subjective content, i.e., determining whether the analysed content conveys a positive, negative or neutral attitude towards its target, and *(ii)* determining the targets of the opinions.

Recently, subjectivity and sentiment analysis gained an increasing importance as they support information retrieval (IR) and information extraction (IE) applications especially in the domains containing a vast amount of subjective content such as humanities. For instance, subjectivity and sentiment analysis can support IR in two possible ways: *(i)* in query preprocessing where the user is allowed to enter complex natural language queries, such as `negative criticism about the works of A`, subjectivity and sentiment analysis components can help the system to classify this query as opinionated with negative polarity towards the target `works of A`; *(ii)* in opinion-oriented information retrieval, knowing that the query is opinionated and negative, the IR system can retrieve the documents containing opinionated snippets with negative polarity towards the target in the query.

In this paper, we present an unsupervised sentiment analysis component, and its intrinsic evaluation for facilitating query preprocessing in a semantic IR system for career

[1]Corresponding Author: Cigdem Toprak, UKP Lab, Technical University of Darmstadt, Hochschulstr. 10, 64289 Darmstadt, Germany; E-mail: c_toprak@tk.informatik.tu-darmstadt.de

recommendation [1]. The overall system allows users to describe their interests in short essays, called *professional profiles*, treated as queries in IR. Example 1 presents a sample profile.

> **Example 1.** I would like to work with animals, to treat and look after them, but I cannot stand the sight of blood and take too much pity on the sick animals. On the other hand, I like to work with computer, can program in C, Python and VB and so I could consider working in software development. I cannot imagine working in a kindergarden, as a social worker or as a teacher, as I am not very good at asserting myself.

Professional profiles contain words or phrases describing objects `(computer)`, places `(kindergarden)`, activities `(program)`, and profession names `(teacher)` which help the IR system to pinpoint suitable professions for the user based on textual descriptions of professions contained in a database. However, as the example illustrates, professional profiles contain both preferred `(computer, program)` and dispreferred `(kindergarden, social worker)` items. Our sentiment analysis component aims at extracting *professional preferences* which are modelled as *(target, polarity)* tuples where *polarity* reflects user's preference regarding the *target*. This way terms with negative polarities can be excluded from the actual IR query.

The remainder of this paper is structured as follows: We describe the corpus of professional profiles and the evaluation gold standards in Section 2. Sections 3 and 4 present our approaches to target extraction and sentiment analysis as well as the result analysis of both tasks. Finally, we draw some conclusions in Section 5.

1. Annotation Study

The professional profiles corpus contains 45 profiles consisting of 311 sentences and 4668 words in total. We collected 30 profiles from university students and the rest from high school students. We used 30 profiles as the development set and 15 profiles for testing purposes. We manually annotated *professional preferences* using the annotation scheme presented in the next subsection.

1.1. Annotating Professional Preferences

The manual annotation of the *professional preferences* requires annotators to first mark *targets* of the *professional preferences*, i.e., mark spans of words without any part-of-speech restrictions, and then, assign the *category* and *polarity* values to the marked *target*.

While defining the *professional preference* notion, we considered the fact that the extracted *professional preferences* were going to be used by an IR system, not a human. For instance, consider the sentence in Example 2 where the user prefers a challenging job.

> **Example 2.** In any case my future job should challenge me and should not be boring, and I want to be able to realize my own ideas.

While humans can make sense of this sentence in the career recommendation process, it is not informative enough for an IR system as it describes meta-characteristics of a desired profession rather than concrete objects, places, or activities involved in the profession. Therefore, we define the *targets* of *professional preferences* as *objects*, *activities*, *places*, and *profession names* or *areas*, i.e, as concrete clues which define a job or differentiate one job from another. The *category* attribute captures the types mentioned in our definition with the possible values of *object, activity, place, profession*, or *other*. *Other* value is used to mark the targets not fitting within any given category.

The *polarity* attribute represents the user's preference regarding the marked *target* with the possible values of *positive, negative* or *neutral*. Table 1 shows example annotations of the *professional preferences* for the profile presented in Example 1.

Professional preference	Category	Polarity
sick animals, blood	object	negative
animals, computer	object	positive
C, Python, VB	object	positive
program	activity	positive
software development	profession	positive
kindergarden	place	negative
social worker, teacher	profession	negative

Table 1. Example professional preference annotations

The polarity attribute is assigned the *positive* or *negative* value if there is an explicit mention of a preference or dispreference regarding the target. For instance, `like to work with`, and `cannot imagine working` in Example 1 illustrate explicit mentions of preference and dispreference respectively. The *neutral* value is used for the cases where the user mentions a target without indicating an explicit preference. For instance, the polarity for the target `photo laboratory` in the sentence, `I worked in a photo laboratory once`, would be annotated as *neutral* as it does not contain an explicit cue for a specific preference.

1.2. Inter-Annotator Agreement Study

Two linguistics students annotated 28 profiles from the development corpus according to the described scheme. They were given two profiles for training. The annotation process requires marking word spans. Therefore, the annotations exhibit variations in word length. We considered two annotations to be matches if one is a subset of the other in terms of word spans and they intend to mean the same *target*. For instance, in Example 3 the spans `testing improvements` and `testing` are counted as matches since they refer to the same activity.

Example 3. I enjoy transferring knowledge to machines, and then [[testing]$_{AnnA}$ improvements].[2] $_{AnnB}$

The number of the *professional preferences* identified by two annotators differs. For instance, consider the sentence in Example 4 where annotator A marked 5, and annotator B marked 3 targets.

[2]Es macht mir Spaß einer Machine Wissen zu vermitteln, und dann die Fortschritte zu testen.

Example 4. Maybe I could work [in the [industry]$_{AnnA}$ in a [big corporation]$_{AnnA}$ in the [executive board]$_{AnnA}$]$_{AnnB}$, for example at [Daimler]$_{AnnA,AnnB}$ (I like [cars]$_{AnnA,AnnB}$).[3]

For *expression level* agreement calculation we used the directional metric *agr* as proposed in [2]. Let A and B be two sets of annotations marked by two annotators A and B, agreement of annotator B to annotator A is measured as:

$$agr(A||B) = \frac{|A\, matching\, B|}{|A|} \tag{1}$$

The directional metric *agr* measures what proportion of A was also annotated by the annotator B. In other words, $agr(A\|B)$ corresponds to recall if B is being evaluated and A is the gold standard, and to precision if B is the gold standard and A is being evaluated. We obtained an $agr(A\|B)$ of 0.76 and $agr(B\|A)$ of 0.83 where $|A\, matching\, B|$ was 256, i.e., we considered 256 annotations from both sets as refering to the same targets according to the previously mentioned matching criteria. For the 256 matching *professional preferences* we reach a kappa of 0.87 and 0.68 for the *category* and the *polarity* attributes respectively. Based on the sufficient agreement in marking *targets*, *category* and *polarity*, only one annotator labelled the test corpus.

2. Target Extraction

The majority of the *targets* of *professional preferences* consist of nouns, noun phrases and verbs. However, extracting all nouns and verbs as *targets*, despite prunning efforts via a stop list, results in an overgeneration of *professional preferences* as reported by [3]. Therefore, we apply a two-stage approach for extracting the *targets* of the *professional preferences*. We first mark the words belonging to a *target* using an automatically generated lexicon, hereafter called a *target constituent lexicon*. Then, we apply a set of manually defined extraction patterns on the marked words *(target constituents)* to finalize the extraction.

Marking target constituents: *Target constituents* are words which make up the *target*. For instance, the words `English` and `teacher` are the *target constituents* of the *target* `English teacher`. We populate a *target constituent lexicon* from GermaNet [4] using a seed term list of 57 words, whereby 41 words were collected from the tagset of the BERUFEnet portal[4] and 16 words were artificial concepts[5] from GermaNet. The tagset contains three categories of keywords describing *objects* (e.g. `media, foreign language, food`), *places* (e.g. `zoo, garden, manufacturing plant`), and *activities* (e.g. `plant, paint, build`). We retrieved the synsets for

[3] Vielleicht könnte ich in der Wirtschaft bei einem grossen Unternehmen im Vorstand arbeiten, zum Beispiel bei Daimler (ich mag nämlich Autos)

[4] http://interesse-beruf.de provided by the German Federal Labor Office

[5] Artificial concepts represent unlexicalized concepts in the language. For example, *selbständiger_Mensch* and *angestellter_Mensch* are artificial concepts which are co-hyponyms of the *Mensch* concept, however they do not represent real lexical items. Our list of artificial concepts include *Schultyplehrer, hierarchischer_Lehrer, funktionaler_Lehrer, berufstätiger_Mensch, selbstständiger_Mensch, angestellter_Mensch, hausangestellter_Mensch, Strassenberufler, Heilberufler, Sozialberufler, Medienberufler, Lebensmittelverarbeiter, verbeamteter_Mensch, professioneller_Mensch, ausgebildeter_Mensch, abgeordneter_Mensch*

each seed term from GermaNet, and then, recursively queried each sense for its hyponyms in GermaNet. Thereby, we preserved the category of the seed term and assigned it as the category of the terms generated based on the seed term. We apply this approach until we observe no change in the size of the resulting lexicon. Furthermore, we used the 16 seed terms from artificial concepts for populating terms of the *profession* category. As a result, we obtained a lexicon of approximately 9000 terms with category assignments.[6]. We utilized the *target constituent lexicon* to mark the *target constituents*. Additionally, we marked named entities, words tagged with *NE* tag by the POS tagger, as the *target constituents*.

Extraction step: We perform POS tagging and chunking using TreeTagger [5]. Then, we apply the following extraction rules based on POS and chunk tags:

1. **Base noun phrase pattern** extracts base noun phrases containing the *target constituents*. Phrases conform to the POS pattern $(ADJ\ NN^* \| NN^* \| NE^*)$ where NN and NE are the noun and named entity *target constituents*. Example targets extracted according to this pattern include `sick animals`, `electrical equipment`.
2. **Infinite verb pattern** analyses consecutive NC (noun phrase) and VC (verb phrase) chunks where the NC chunk contains a *target constituent* and the VC chunk contains a construction (*to*[7] *infinite verb*). We first apply the *base noun phrase pattern* to the NC chunk and then add the infinite verb from the VC chunk to the base noun phrase. Example *target* extractions based on this pattern include `transfer knowledge to machines`[8], `see other countries`[9].
3. **Coordination pattern** analyses coordinated NC chunks which are the constituents of the *and* and *or* coordinations. According to this pattern, we extract the base noun phrase in a coordinated chunk as a *target*, if the other coordinated chunk already contains a *target*. Example *targets* extracted based on this pattern include `IT Sector` from the sentence `I'm interested in languages and IT Sector` where `languages` was already extracted as a target.
4. **Prepositional phrase pattern** extracts *targets* from PC chunks (prepositional phrases) headed by *with, in*, and *at*[10]. We apply the *base noun phrase pattern* to such PC chunks even if they do not include any *target constituents*. An example *target* extraction based on this pattern includes `software development` from the sentence `I see my future in software development.`

We evaluate our system against a human annotator who marked 336 targets in the development and 96 targets in the test corpus. We used the same matching strategy applied in the expression-level inter-annotator agreement study. We obtain a precision of 0.41, and a recall of 0.66 on the development, and a precision of 0.40, and a recall of 0.55 on the test corpus. Except for the *prepositional phrase pattern* which is lexicon independent, all extraction patterns rely on lexicon look-ups. Therefore, both the coverage and the quality of the *target constituent lexicon* play a crucial role in our extraction approach.

[6]The feature lexicon contains duplicates due to the fact that different seed terms from different categories occasionally populated the same terms. We kept both terms with different category assignments

[7]German: zu

[8]German: Machine Wissen vermitteln

[9]German: andere Länder sehen

[10]with: (German) mit, in: (German) in, at: (German) bei

However, the quality of an automatically generated lexicon is suboptimal as it contains too many noisy terms.

We observe that most of the unextracted *target constituents* were due to the poor coverage of the lexicon, whereas spurious extractions resulted from the overgeneration. For instance, the *targets* `wine` and `nutritional science` in the sentence `By the way, my interests include`[11] `wine and nutritional science` were not extracted due to insufficient coverage. On the other hand, we extracted `interests`[12] as a target due to the noise in lexicon.

In the profiles, users describe their professional interests, dislikes and expectations without any restrictions. We observe that the *targets* of the *professional preferences* are not restricted to those included in a lexicon created from a small set of seed terms.

3. Sentiment Analysis

In the sentiment analysis stage, we assign polarities to the extracted targets, in other words, we focus on the preference part. We utilize the *opinion lexicon* from [3] which contains unigrams with manually assigned polarities. The *opinion lexicon* is generated from GermaNet using the hyponyms of the concepts *feeling (Gefühl)*; *to feel, to care (empfinden)*, and the artificial concepts *evaluation specific (Bewertungsspezifisch)*, *feeling specific (Gefühlspezifisch)*. We utilize the lexicon to mark the *sentiment cues*.

We perform full parsing with BitPar[13]. BitPar delivers parse trees with nodes representing syntactic categories (e.g. S: sentence, NP: noun phrase, VP: verb phrase, PP: prepositional phrase), and edges representing functional units (e.g. SB: subject, HD: head, PD: predicate) and modification relations (e.g. MO: modifier, NG: negation).

We analyse each clause as a separate unit based on the assumption that a clause represents a unit of thought. We assign polarities on the clause level. A clause can be identified as an S node headed by a finite verb in a parse, or a VP node as a coordinate constituent in a parse. Starting from the leaves, we assign each node a polarity based on the polarity of its immediate children following the approach:

if (children contain at least one negative polarity)
  update node polarity as negative
else if (children contain positive polarity)
  update node polarity as positive
else
  update node polarity as neutral
if (node has a child connected with NG (negation) edge)
  reverse node polarity (neutral polarity is switched to negative)

Finally, we assign the polarity of the clause to the targets occurring within the clause.

We evaluate the polarity assignment for the correctly extracted 225 targets in the development and 53 correctly extracted targets in the test corpus against the polarity decisions of a human annotator. Precision (P) reported for a polarity value indicates the

[11] ich habe großes Intresse an
[12] großes Interesse
[13] http://www.ims.uni-stuttgart.de/tcl/SOFTWARE/BitPar.html

proportion of the correctly identified polarity instances for this value to all instances identified with this polarity value by the system. For instance, P for positive polarity is $\frac{correctly\ classified\ positive\ targets}{all\ targets\ classified\ as\ positive}$. Recall (R) of a polarity value is the proportion of the correctly identified polarity instances for this value to the actual number of the polarity instances for this value in the gold standards. For instance, R for positive polarity is $\frac{correctly\ classified\ positive\ targets}{number\ of\ positive\ targets\ in\ gold\ standards}$. Table 3 and Table 2 present the results and the polarity distribution for the correctly extracted targets. The error analysis shows that

Total	Positive	Negative	Neutral
Development (225)	158	23	44
Test (53)	40	3	10

Table 2. Polarity distribution among the correctly extracted targets

Corpus	Positive		Negative		Neutral	
	P	R	P	R	P	R
Development	0.95	0.60	0.73	0.82	0.40	0.90
Test	0.85	0.45	0.50	0.33	0.23	0.70

Table 3. Polarity assignment evaluation for the correctly extracted targets

our opinion lexicon performs satisfactorily at marking the positive sentiment cues most of the time. However, high precision against low recall in positive polarity assignments and the reverse situation in neutral assignments (low precision against high recall) reveal problems with the polarity assignments despite the good coverage of the lexicon.

A major source of errors is the clause-level granularity of analysis. We loose sentiments in the subordinate clauses which refer to targets in the main clauses and vice versa. For instance, we assign neutral polarity to the professional feature `numbers` in the sentence `I'm impressed with the fact that one can explain everything with numbers` as the subordinate clause does not contain any sentiment cue. Furthermore, parsing errors constitute an additional problem in polarity assignments especially when the clause boundaries were not marked correctly in the parse tree. In such cases, we were unable to assign the correct polarity even though we were able to detect the sentiment cue.

We observe relatively good results on the negative polarity assignments in the development corpus compared to the test set. We cannot be very conclusive regarding this on the test corpus due to a small number of the respective negative polarity instances. Again, in negation detection, we see that our approach cannot detect the long distance negation, i.e., the negation spanning the subordinate clauses.

4. Conclusions

We presented an unsupervised lexicon based sentiment analysis component and its intrinsic evaluation in a career recommendation scenario. We extracted *professional preferences* defined as *(target, polarity)* tuples, where *targets* are the terms and phrases facilitating automatic career recommendation and *polarities* are user's preferences regarding the targets. In target extraction, we utilized GermaNet and syntactic patterns over the

results of shallow parsing. The results of *target* extraction show that the lexicon based approach is tied to the coverage and the quality of the lexicon.

We also applied a lexicon based approach to sentiment analysis. The approach assumes that each clause represents an individual unit. Hence, we assigned the polarity of a clause to the *targets* within the clause. The performance of our approach, on one hand, relies on the accuracy of the parser which is sometimes erroneous, and on the other hand on the number of sentences, in which we fail to associate the sentiment cue with the target due to long distance negations or modifications in the subordiante clauses. Nevertheless, unlike statistical methods currently dominating the field of sentiment analysis our approach does not require any training data which is very expensive to obtain for new domains.

During the manual annotation study, we observed that discourse level analysis and coreference resolution play important roles in the correct interpretation of one's preferences. Users tend to express their opinions in a sequence of sentences, where first sentence contains the target, and subsequent sentences contain preferences regarding the target using references to the target. We plan to incorporate these aspects in our future work.

Acknowledgements

This work was supported by the German Research Foundation (DFG) under the grant *GU 798/1-2*, *Semantic Information Retrieval from Texts in the Example Domain Electronic Career Guidance*, and by the Volkswagen Foundation as part of the Lichtenberg-Professorship Program under grant No. I/82806.

References

[1] I. Gurevych, C. Müller, and T. Zesch, "What to be? - electronic career guidance based on semantic relatedness," in *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, (Prague, Czech Republic), pp. 1032–1039, June 2007.

[2] J. Wiebe, T. Wilson, and C. Cardie, "Annotating expressions of opinions and emotions in language," *Language Resources and Evaluation*, vol. 39, pp. 165–210, 2005.

[3] V. Cvoro, "Recognition of emotional preferences for professional features," Master's thesis, Ruprecht-Karls University, Department of Computational Linguistics, Heidelberg, August 2005.

[4] C. Kunze, *Lexikalisch-semantische Wortnetze*, ch. Computerlinguistik und Sprachtechnologie, pp. 423–431. Spektrum Akademischer Verlag, 2004.

[5] H. Schmid, "Probabilistic part-of-speech tagging using decision trees," in *Proceedings of Conference on New Methods in Language Processing*, (Manchester, UK), pp. 44–49, September 1994.

"Multilinguality in an on-line platform for classical philology - beyond localisation of the user interface"

Cristina Vertan
University of Hamburg, Insitute for Greek and Latin Philology
cristina.vertan@uni-hamburg.de

1. Introduction

Classical philology is using increasingly repositories and adequate tools for making available to its researchers, but also to a broader public, valuable materials otherwise only available in libraries spread all over the world, and usually with restricted access. The advantages of such digital libraries are enormous, however one may consider the challenges imposed by the specificity of the domain (classical philology) to the design both of the user interface as well as to the functionality.

In comparison with digital libraries archiving modern documents, the objects in classical philology have particularities like:

- Are quite often only partially described (i.e. for one class of objects one can find one or two fields constantly mentioned for all objects.). This is mainly due to the lack of information researchers have about manuscripts
- It is almost impossible to define relations between objects which are valid for all elements inside a class (e.g. it is very often the case that not both the object description as well as object's digital form exist)
- One object contains text in several languages (Greek, Latin, at least one modern language)

Due to the above mentioned complexity up to now in classical philology we deal only with one object-type-repositories, which means that either it is a collection of manuscripts or a collection of watermarks, or collection of digitalized books, in the very best case with their descriptions (the most well known example is the Perseus digital Library [1]) We will refer to other initiatives more details in the following sections.

It is without doubt that for researcher the navigation and search possibility among various objects would be of great help in establishing interconnections, and helping decisions like e.g. establishing the date of a manuscript by means of the use of a certain watermark. Additionally such a platform is a real help for the user community, if it enables active participation of the researchers through comments, contributions, critics related to the documents on the platform. One particularity of communities in humanities, particularly in classical philology is that they have usually at least two general accepted communication languages, not only English. In classical philology, five languages (English, German, French, Spanish, Italian) are commonly accepted at conferences, journals, as well as communication means between researchers. Therefore a dedicated on-line platform should not only support all these languages (i.e. localization of the user interface) but also manage internally multilingual services. Methods from language technology and semantic web have to be used to ensure such a functionality

In this article we will present a flexible architecture that tries to embed various types of objects a classical philologist would work with, link them and offer to the users cross-lingual services.
Section 2 is giving an overview of types of data objects to be represented within the platform. Section 3 is describing the functionality of the system while section 4 refers to multilingual problems and solutions.

2. Data objects inside Teuchos platform

The Teuchos Center for Manuscript and text research [2] was set-up in 2007 at the Institute for Greek and Latin Philology at the university of Hamburg in cooperation with the Aristoteles –Archive at the Free University Berlin. It is a long-term infrastructure project, which is financed in a first phase until 2010 by the German Research Foundation in the frame of the programme „Theme-oriented information networks".

There are two main directions in which efforts of classical philologists concentrated in relation with IT. One is the construction of watermark collections, the other large repositories of digitalized manuscripts. Unfortunately up to now, to our knowledge there was no attempt to unify these collections. Our system proposes a model and encoding schema, which allows interoperability between the watermark and the manuscript collections.

Following types of documents have to be made available through the on-line research platform:

1) Manuscript descriptions: descriptions of medieval codices that have different complexity degrees from basic inventories up to deep descriptions (as for e.g. the Aristoteles Graecus ([3]). These descriptions point to parts of the corresponding manuscript but also to parts of related manuscripts.
2) Digitized versions of the manuscripts, these are image data acquired with high-resolution (sometimes also multispectral) techniques. These images have to be aligned with transcriptions (when available) and with other data.
3) Additional research data like digitized versions of watermarks graphics, biographical and bibliographical data
4) Transcriptions, and text variants for part of the manuscripts
5) Research articles, and comments created within the forum functionality

To illustrate the approach we too for the object encoding we describe below the data models for watermarks and manuscripts.

2.1. Watermarks

All medieval manuscripts referring to ancient Greek texts have watermarks. The availability of collections of watermarks is extremely important for researches as these watermarks allow often a more precise dating or establishment of the origin of the manuscript

Over the time the watermarks were catalogued on paper in various formats, but always containing a graphic (the watermark form) and a short description. On-line catalogues for medieval watermarks are already available [4]. In these on-line catalogues, the search is possible following hierarchies of watermarks motifs (graphical representation of the watermark) or following either different physical features like: size, paper structure, or characteristics of the manuscript in which they were found.

The innovative aspect of our approach is the introducing of links between different watermark instances, links that model different graphical similarities. As a consequence of the technological process for their generation, watermarks are not isolated but appear usually in pairs or even close related group of three or four. A group of several watermarks sharing the same main motif constitutes a Watermark-Motif –Object.

We are using here an Object Oriented approach, with the Watermark-Motif in the place of super-class from which each watermark instance object (unique realization of a watermark) is derived. As one motif can appear in different manuscripts we maintain a list of identical motifs. In this case the watermark was produced from the same sieve. Opposite, motifs can be very similar but produced from different sieves. In this case we will refer these motifs in a list of “similar motifs”. The realization of one motif on different pages is called instance object. In figure 1 we present the motif object while in figure 2 the motif instance object is described.

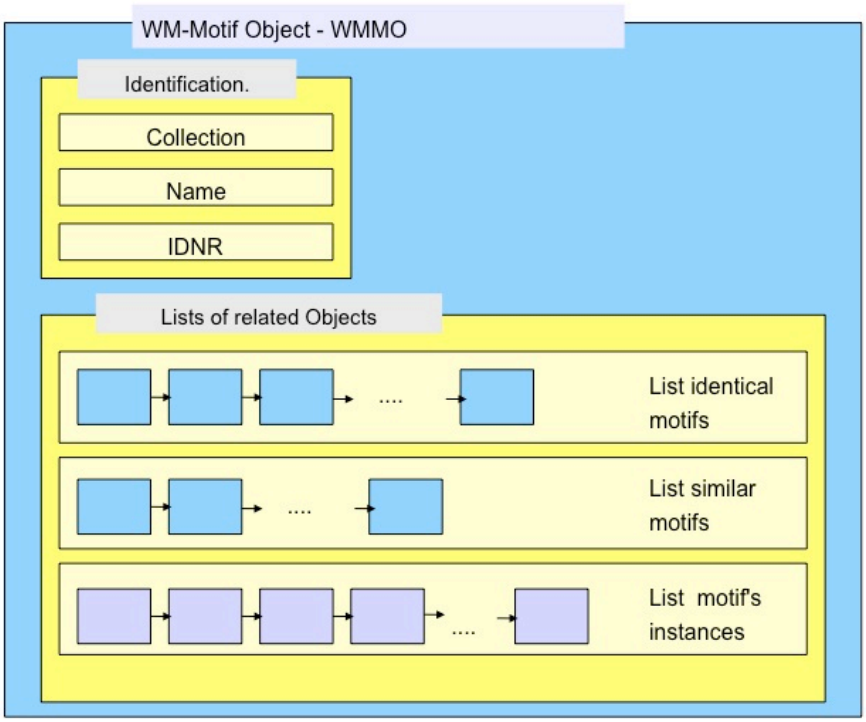


Figure 1. Watermark Motif Object

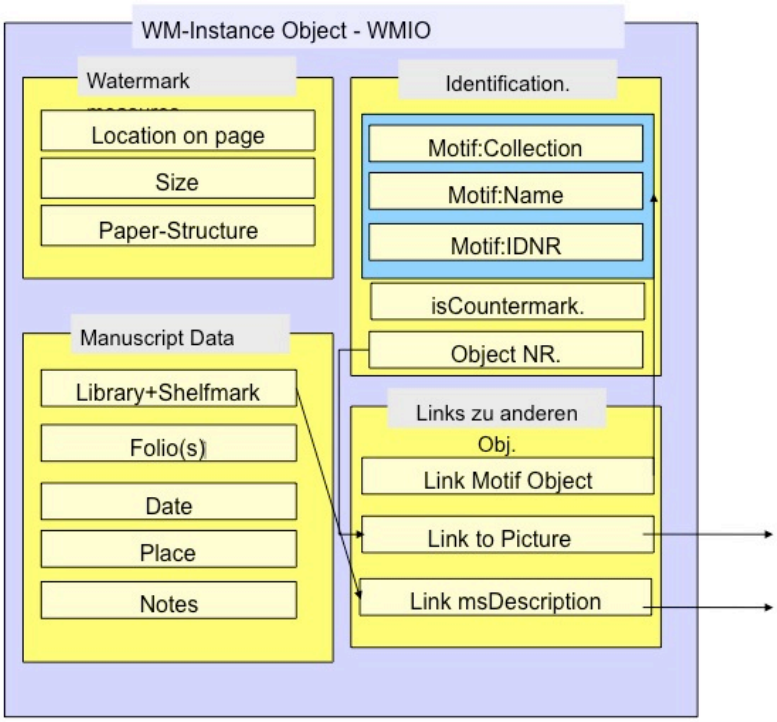


Figure 2. Watermark Motif-Instance Object

The encoding is realized in XML. We developed an XML schema, as the element “watermark” from TEI-P5 does not offer enough flexibility to record all above-mentioned information. Below we present one example of watermark instance object:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="wmobject.xslt"?>
  <teuwmo:teuwmObj    xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation='WatermarkObject.xsd'
   xmlns:teuwmm="http://teuchosObjects.com/watermarks/WMMotif">
  <teuwmo:wmIdent wmIsCountermark="false">

<teuwmo:wmObjId> TEU_WMDesc_Aiglem2-21.xml</teuwmo:wmObjId>
<teuwmm:wmIdentification>
        <teuwmm:wmIdnr> 21 </teuwmm:wmIdnr>
        <teuwmm:wmCollection> Harlfinger </teuwmm:wmCollection>
        <teuwmm:wmName>
               <wmNameLanguage wmLang="fr"> Aigle</wmNameLanguage>
               <wmNameLanguage wmLang="de"> Adler</wmNameLanguage>
         </teuwmm:wmName>
      </teuwmm:wmIdentification>
    </teuwmo:wmIdent>
 <teuwmo:wmManuscriptData>
    <teuwmo:msName> Vatic.1469 </teuwmo:msName>
    <teuwmo:msFolio> ff.1-72 </teuwmo:msFolio>
    <teuwmo:msDate> 1495 </teuwmo:msDate>
<teuwmo:wmManuscriptData>
<teuwmo:wmLinks>
    <teuwmo:pictureLink> Aigle-21m2.jpg </teuwmo:pictureLink>
    <teuwmo:msDescLink> Aigle-21.xml </teuwmo:msDescLink>
    <teuwmo:motifLink> Aigle.xml </teuwmo:motifLink>
 </teuwmo:wmLinks>
</teuwmo:teuwmObj>
```

As it can be observed from this example it is a common practice to record names in at least two languages for a watermark motif. The two languages are however not predefined, so any combination of two of the five languages mentioned in section 1 is possible. An ontological based approach is required in order to ensure consistency between watermark names. We will refer to this in section 3.

2.2. Manuscripts

We consider both the results of the digitalization process as well as the manuscript descriptions. We have a tolerant model in which not every manuscript description has automatically attached the collection of images for that manuscript, as well as the reverse situation where the manuscript description is missing.

A manuscript description is structured in several parts marked by keywords. These keywords (e.g. “Reklamanten”, “Kopisten”, Reklamanten”, “Inhalt”[1]) were used as identifiers for different sections of the manuscript description in the annotation process. The descriptions were encoded following a modified version of the TEI-P5 Manuscript Description Module[5]. This module was extended in order to serve to our purposes.

For example the “watermark” element as defined in TEI-P5 was inadequate for annotating the watermarks mentions in our documents. References to watermarks in manuscript descriptions are more than simple mentions of the watermark motif but include details to the place of such watermark, remarks to the respective motif etc. The annotation process was done for the moment for 100 manuscript descriptions from “Aristotle Graecus” completely automatic. Additionally we annotated automatically all remarks referring folio numbers as well as all Watermark motifs already present in our watermark collection. In this way we realized the connection between the manuscript description collection and the watermark collection. This is to our knowledge the first attempt to link different types of on-line description collections of classical philology.

[1] in our case we work with German texts. However the automatic annotation process can be easily adapted to other languages or keywords.

The annotation of folio references in the descriptions allows us as well to link digitalized versions of these folios, as soon as they become available.

3. Teuchos functionality

The system we describe intends to offer to the classical philologist a powerful tool to editing and searching and publishing materials. In this section we will describe the user scenarios we have in mind, and the system architecture that ensures the realization of these scenarios. The system architecture we refer is presented in figure 3.

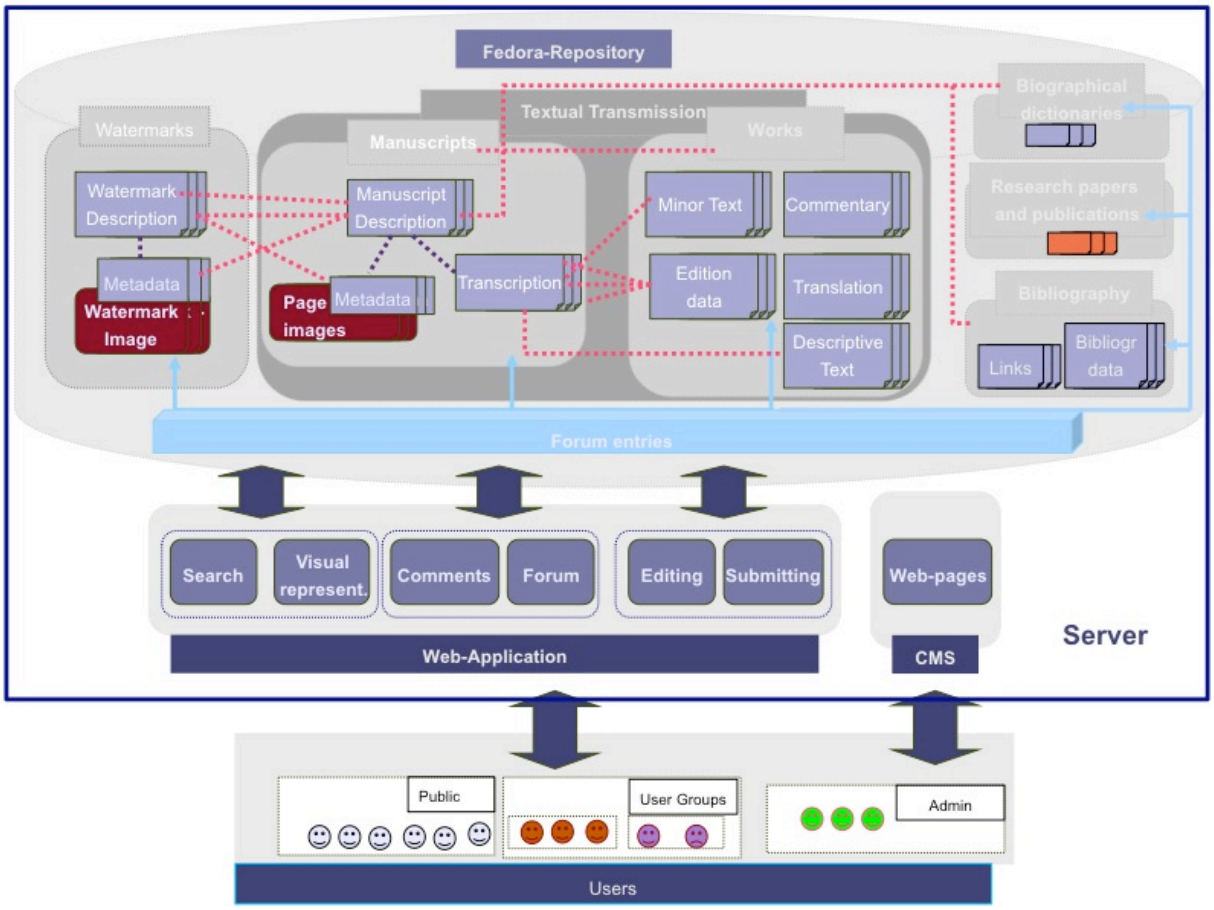


Figure 3. System Architecture

All our objects are stored on a Fedora-repository[2]- the choice of Fedora is grounded not only by the fact that it is an open source software but mainly because the stored objects can be linked following the RDF-model, i.e. gives the possibility to define semantic relations between objects.

The user interacts with the repository via a Web application that manages the editing, searching, and uploading processes. We have 3 categories of users:

- The system administrators who have full access to al parts of the system and control the content made available by the other users
- Registered users that have access to view all materials on the system can upload materials, write forum entries and upon their interest edit different material types. We envisage a hierarchy of such users having different access types to parts of the system.
- Normal users, who can only view materials declared as public.

[2] http://www.fedora.info/

The digital objects stored on the Fedora-Repository built six groups. Technically these groups are collections of Fedora-Objects.

- *Watermarks:* here are stored the watermark descriptions as well as the watermark image. Watermark images are digital graphical representations of the watermarks as they were collected in paper catalogues. Each watermark image has associated metadata in XML format, containing Dublin Core[3]-like information about the data. Each watermark description is linked to these metadata.
- The "*Textual Transmission*" group is divided in two subgroups: the manuscripts and the works
- Through "*manuscript*" we refer to one manuscript description to which we can have associated a transcription and/or a list of page images with their metadata.
- Through "*Work"* we refer to all other materials referring to one manuscript that appeared over time like: translation, different edition data, commentaries, texts about that particular manuscript. We link all these objects with the translation object.
- Two other groups the "*Biographical dictionaries*" and the "Bibliography" refer to collections of persons or woks cited in one or other manuscript description.
- A special group is dedicated *to research papers* that can refer different manuscripts.

Separating objects in such groups gives us the possibility to model two relation types:

Intra-group relations, labeled with "correspondTo", and various inter-group relations "isCitedIn", "isreferedBy", "usedFor". These relations are described as RDF-triples.

With help of these relations we are enabling a search functionality across groups, i.e. the user can for example search *"which manuscript in Collection X used the watermark Y".* To our knowledge this is the first approach in this sense, at least for classical philology.

Apart from these static objects we are implementing a forum that will give the possibility to registered users to comment any of the documents available on the repository.

4. Multilingual Aspects inside of Teuchos platform

As we mentioned in section 1 users of the Teuchos platform are speakers (or at least understand) one of five languages used inside of he community. At a first glance the straightforward consequence is the localization of the user interface in the envisaged languages. However, we claim in the following that there are deeper multilingual aspects which have to be handled and we illustrate how language technology can help.

We define three types of multilingual phenomena, occurring in our platform:
1) "Macro-document "-Multilinguality at the level of users and the uploaded multilingual documents: Therefore the platform is required not only to support uploading of documents in all these languages but also to manage their relations to one or more manuscripts in a consistent way.
2***.) "Micro-document" - Multilinguality*** at the lever of primary data to be analysed. As we already mentioned manuscripts are accompanied by modern descriptions, critical texts, which although written in modern languages (see 1) are containing often passages from the manuscript, or Latin citations. This is a real challenge when trying to process the documents automatically.
3) ***"Terminological "- Multilinguality***, as we mentioned in section 2.1. related to watermarks. Even watermarks descriptions written in one language, may declare watermarks-motifs in a variety of languages. We have to ensure that watermarks are then classified as belonging to the correct class.

To handle these three types of multilinguality we propose an ontology based approach, integrating different ontologies related to components of the system. In each of the system main components (manuscripts, watermarks, etc.) a domain specific language independent ontology ensures the correct mapping of documents on the right concept(s). Links between components are realized between the

[3] http://dublincore.org/

nodes of the ontology and not the particulars instance objects (namely the documents). The approach is represented in figure 4.

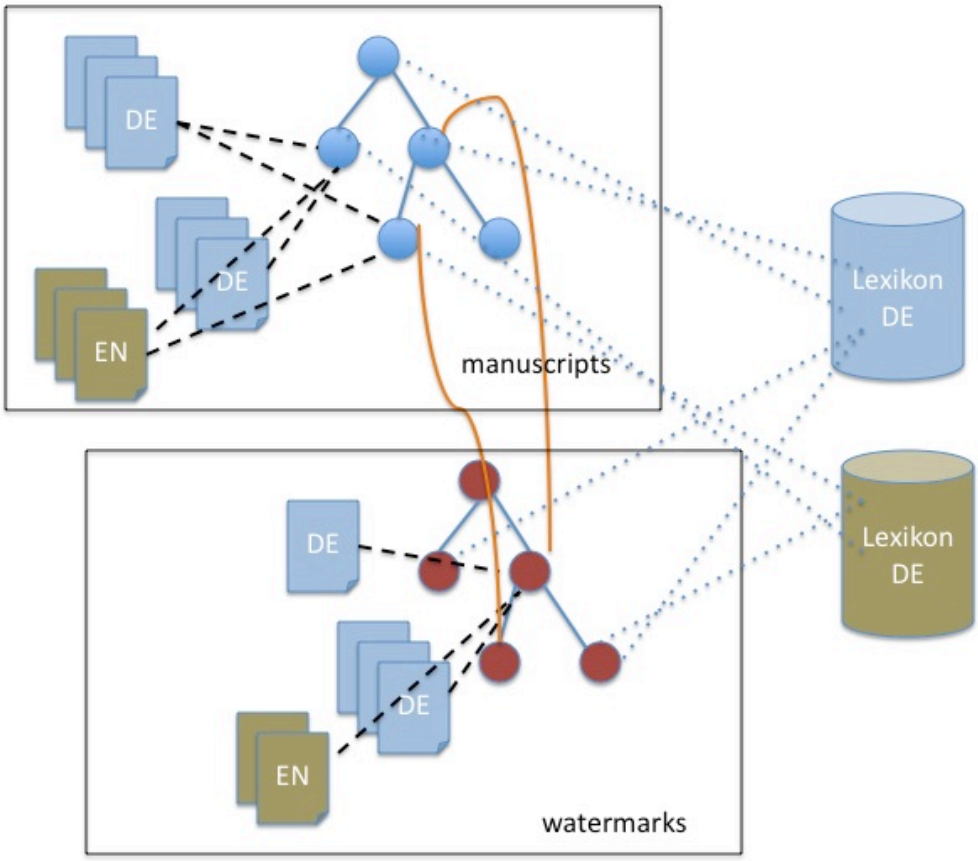


Figure 4. Ontological representation of multilingual documents.

For the mapping between the lexical materials and the ontology, the annotation of documents with concepts and the cross-lingual semantic search we intend to use the approach followed in [6].

5. Conclusions and further work

In this paper we presented an architecture for storing and accessing objects for classical philology. We introduced the main objects we manipulate, their particularities and the way are stored in our system.
We argue that the representation of multilingual objects inside the platform can be done only via an ontological approach. This will ensure both consistency for the management of various data and also enable cross-lingual retrieval.

For the moment we modeled and stored watermark- and manuscript-objects, and we interconnected these two.
We are working now on a deeper annotation of manuscript description that will allow us a more refined interconnection between objects. Through deeper annotation we understand automatic recognition and annotation of person names, indications of time (year, century etc.), titles of works.
First version of the system will be released by the end of the month. This will enable real users to post comments related to different objects and this enable us to experiment the ontological setting.

References

[1] Perseus, digital library, http://www.perseus.tufts.edu/hopper/
[2] Teuchos platform , http://www.teuchos.uni-hamburg.de/
[3] P. Moraux, Aristoteles Graecus- d. griech. Ms. d. Aristoteles, 1976, De Gruyter (Berlin, New York)
[4] on-line Watermark collections, http://www.ksbm.oeaw.ac.at/wz/wzma.php, http://watermark.kb.nl, http://www.ksbm.oeaw.ac.at/wies/
[5] Manuscript Description Model TEI-P5, http://www.tei-c.org/release/doc/tei-p5-doc/en/html/MS.html
[6] C. Vertan, K. Simov, P. Osenova, L. Lemnitzer, A. Killing, D. Evans and P. Monachesi, Crosslingual Retrieval in an eLearning Environment, Lecture Notes in Computer Science, AI*IA 2007: Artificial Intelligence and Human-Oriented Computing, Volume 4733/2007, Springer Berlin / Heidelberg, 2007

Bausteine eines *Literary Memory Information System* (LiMeS) am Beispiel der Kafka-Forschung

Benno Wagner[1], Alexander Mehler[2], Christian Wolff[3], Bernhard Dotzler[3]

[1]Fachbereich 3 – Sprach-, Literatur- und Medienwissenschaften
Universität Siegen
wagner@lit-wiss.uni-siegen.de

[2]Geisteswissenschaftliche Fachinformatik
Universität Frankfurt / Main
mehler@em.uni-frankfurt.de

[3]Institut für Information und Medien, Sprache und Kultur
Universität Regensburg
{bernhard.dotzler, christian.wolff}
@sprachlit.uni-regensburg.de

> Ich rede von denen, die je nach der verschiedenen Stufe ihrer Kenntnisse ganz verschiedene Bücher lesen, ohne bestimmten Plan, unaufhörlich wechselnd, selten in einem Buch lang ausruhend, getrieben von einer unausgesetzten, nie recht gestillten Sehnsucht. [...] sie suchen ja von Buch zu Buch, was der Inhalt keines ihrer tausend Bücher ihnen geben kann: sie suchen etwas, was zwischen den Inhalten aller einzelnen Bücher schwebt, was diese Inhalte in eins zu verknüpfen vermöchte. Sie schlingen die realsten, die entseeltesten aller Literaturen hinunter und suchen etwas höchst Seelenhaftes. [...]. Aber die Sehnsucht [...] geht durchaus nicht auf den Dichter. Es ist der Mann der Wissenschaft, der diese Sehnsucht zu stillen vermag.
> *Hugo von Hofmannsthal (1907)*

> Letztlich reicht es nicht aus, auf Seiten der Objektbasis unablässig neu digitales Material zu akkumulieren. Parallel dazu müsste auf Seiten der Forschung die Bereitschaft zum aktiven Einsatz technologisch und methodisch innovativer Verfahren gefördert werden. Die Digitalisierung allein ohne eine begleitende Theoriedebatte und ohne ein verfeinertes methodisches Rüstzeug betreiben zu wollen dürfte zu verkürzten Ergebnissen führen. Beide Seiten, das Material und die Forschung, die Technik und die Methodologie, sind aufs engste miteinander verbunden.
> *Michael Embach/Andrea Rapp (2008)*

1 Einleitung

In dem Paper beschreiben wir Bausteine eines *Literary Memory Information System* (LiMeS), das die literaturwissenschaftliche Erforschung von so genannten *Matrixtexten* – das sind Primärtexte eines bestimmten literarischen Gesamtwerks – unter dem Blickwinkel großer Mengen so genannter *Echotexte* (Topia 1984; Wagner/Reinhard 2007) – das sind Subtexte im Sinne eines literaturwissenschaftlichen Intertextualitätsbegriffs – ermöglicht. Den Ausgangspunkt dieses computerphilologischen Informationssystems bildet ein Text-Mining-Modell basierend auf dem Intertextualitätsbegriff in Verbindung mit dem Begriff des *Semantic Web* (Mehler, 2004b, 2005a, b, Wolff 2005). Wir zeigen, inwiefern dieses Modell über bestehende Informationssystemarchitekturen hinausgeht und schließen einen Brückenschlag zur derzeitigen Entwicklung von Arbeitsumgebungen in der geisteswissenschaftlichen Fachinformatik in Form eines *eHumanities Desktop*.

2 Literaturwissenschaftliche Perspektive: Transbiblionome Räume

Moderne Literatur funktioniert essentiell intertextuell und intermedial. Mit dem Heraufkommen der neuen ‚Konkurrenz'-Medien (Photographie und Film, Telegraph und Telephon) und mit der Ablösung der Referenzinstitution Bibliothek durch globale Informationssysteme (Rayward

2008) sowie das alle Lebensbereiche durchdringende Wissen moderner Verwaltungen (Wagner 2006a) existiert Schrift nurmehr im Spannungsverhältnis zwischen Buchgebundenheit und einem zunehmend transbiblionom organisierten kulturellen Kontext. Der literarische Text gerät auf diese Weise einerseits zum „geometrischen Ort eines hors-texte", zu einem „Kreuzungspunkt von Schichten, die Myriaden von Horizonten entspringen" (Topia 1984, 103). Anderseits wird Literatur unter diesen Bedingungen zu einer besonderen Instanz des kulturellen Gedächtnisses. Sie lässt sich als komplexer „Spurenkörper" (Pêcheux 1983, 55) beschreiben, oder – jedenfalls in ihren raffiniertesten und reflektiertesten Schreibweisen – auch als „hypermnemische Maschine" (Derrida 1984, 147), als dynamischer Erinnerungsapparat, dessen virtuelles Verweispotential auf andere Texte die Ordnungsraster realer Wissensspeicher (Enzyklopädien, Bibliotheken, Archive) durchkreuzt.

So schrieb Derrida in der *Grammatologie* zunächst: „Es geht [...] nicht darum, der Buchhülle noch nie dagewesene Schriften einzuverleiben, sondern endlich das zu lesen, was in den vorhandenen Bänden schon immer zwischen den Zeilen geschrieben stand. Mit dem Beginn einer zeilenlosen Schrift wird man auch die vergangene Schrift unter einem veränderten Organisationsprinzip lesen. [...] Was es heute zu denken gilt, kann in Form der Zeile oder des Buches nicht niedergeschrieben werden; ein derartiges Unterfangen käme dem Versuch gleich, die moderne Mathematik mit Hilfe einer Rechenschiebermaschine zu bewältigen." Stattdessen avisiert er, diesmal mit Leroi-Gourhan, „eine andere[n], bereits vorstellbare[n] Art der Speicherung [...], deren rasche Verfügbarkeit der des Buches überlegen sein wird: die große ‚Magnetothek' mit elektronischer Auswahl wird in naher Zukunft vorselektierte und sofort verfügbare Informationen liefern" (Derrida 1967, 154f.).

Knapp zwei Jahrzehnte später hingegen, in einem Aufsatz aus dem Jahre 1984, konfrontiert uns Derrida mit einem ganz anderen und scheinbar diametral entgegengesetzten Szenario. Im Bezug auf den *Ulysses* von Joyce heißt es nun: „for there be no simple confusion between him [Joyce] and a sadistic demiurge, setting up a hypermnesiac machine, there in advance, decades in advance, to compute you, control you, forbid you the slightest inaugural syllable because you can say nothing that is not programmed on this 1000th generation computer [...] beside which the current technology of our computers and micro-computerfied archives and translating machines remains a bricolage of a prehistoric child's toys" (Derrida 1984, 147). Hier hat sich offenbar das Komplexitätsgefälle zwischen Druckschrift und elektronischem Speicher verkehrt. Die lineare Schrift ist nicht länger Komplexitäts-Engpass, sondern sie fungiert selbst als Quelle einer überbordenden Komplexität. *Sie* ist nun der "Computer der 1000. Generation", im Vergleich zu dem die elektronischen Speichermedien als Problem erscheinen, als eine dem Gegenstand der Druckschrift unangemessene, prähistorische Spielerei.

Betrachtet man nun den Einsatz von Computern zu Zwecken der Literaturforschung seit den 1990er Jahren, so drängt sich der Eindruck auf, als habe jedes der beiden Zitate eine Arbeitsperspektive eröffnet, die von der jeweils anderen nichts zu wissen scheint. So haben Autoren wie George Landow und Jay Bolter, in einer eigentümlichen Einebnung des Unterschieds zwischen der syntagmatischen und der paradigmatischen Text-Dimension, den elektronischen Hypertext kurzerhand zu jenem Medium deklariert, mit dessen Hilfe sich das intertextuelle Verweispotential eines literarischen Textes restlos implementieren, der vieldimensionale literarische Text sich aus dem

Zwang der Zeile befreien ließe.1 Dass freilich Hypertext im besten Falle als funktional limitiertes Instrument zur Darstellung von Intertextualitätsbeziehungen dienen kann, belegen ex negativo die an diese kurzschlüssigen Verheißungen anschließenden Forschungsprojekte. Als paradigmatischer Gegenstand dient hier in erster Linie das Werk von James Joyce, wobei sich die *Hypermedia Joyce Studies* (http://www.geocities.com/hypermedia_joyce/) mit einer Vielzahl von text*theoretisch* anregenden Joyce-Lektüren als dynamisches Zentrum etabliert haben. Ein anderes Bild ergibt sich, wenn man sich die *HJS*-Liste der „hypermedia projects" ansieht (http://www.geocities.com/hypermedia_joyce/biblio_ht.html). Die dort aufgelisteten *praktischen* Versuche, den intertextuellen Raum etwa des *Ulysses* mittels Hypertext darzustellen und nutzbar zu machen, erinnerten deutlich an Derridas „prehistoric child's toy"-Verdikt, bevor sie mehrheitlich durch eine file-not-found-Meldung ersetzt wurden.

Sehr viel erfolgreicher gestaltet sich computergestützte Literaturforschung immer dann, wenn sie die transbiblionome Dimension der Intertextualität von vorherein aus ihrem Gegenstandsbereich ausschließt. Dies geschieht zumeist implizit, ohne weitere Erörterung und Reflexion, bisweilen aber auch mit programmatischem Nachdruck, wenn sich etwa die in Deutschland etablierte Computerphilologie explizit auf die Befassung mit „traditionellen philologischen Gegenständen" und damit das Potential des Computer auf die Optimierung biblionomer Funktionen sowie auf die Herstellung dezentraler Forschungskollaborationen beschränkt.

Eine *intertextuell-transbiblionome Computerphilologie* bleibt unter diesen Bedingungen *Desiderat*. Ihre methodische und technische Entwicklung hätte sich vor dem skizzierten Erfahrungshintergrund an drei Leitlinien zu orientieren:

(1) *Zielsetzung*: Transbiblionome Computerphilologie zielt auf die computergestützte Erschließung und Darstellung der intertextuellen und intermedialen Dimension literarischer Texte jenseits einer Beschränkung auf einen biblionomen Forschungshorizont und der Fixierung auf das Potential von Hypertext. Als technische Grundlage hierfür hätte eine gegenstandsspezifische digitale Arbeitsumgebung zu dienen, die zugleich die dezentrale Kollaboration von Experten(gruppen) und eine nutzerspezifisch differenzierte Aufbereitung der Forschungsergebnisse ermöglicht.

(2) *Theorie*: Unter den genannten Bedingungen kann und muss sich das zugrundeliegende Intertextualitäts-Modell der methodischen Alternative entziehen, mittels derer die printorientierte Methodendiskussion (insbesondere der 70er und 80er Jahre) sich um eine „Zähmung" (Lachmann 1984, 137) des transbiblionomen Potentials ‚moderner' Intertextualität bemüht hatte: hier eine ‚geschlossene', durch unterstellte Verknüpfungsabsichten des Autors oder faktische Verknüpfungsoperationen des Lesers begrenzte, dort eine unmittelbar auf das System der langue bezogene ‚offene' und daher, zumal unter Bedingungen einer printfixierten Forschung, forschungspraktisch niemals einholbare Intertextualität. Baßlers Entwurf eines ‚archivimmanenten Strukturalismus' (Baßler

1 Hierzu konstatiert Baßler: „Landows Parallelisierung von Hypertext mit jenem poststrukturalistischen Textbegriff, den Barthes in S/Z entwickelt, setzt sich über den elementaren Unterschied von syntagmatischer und paradigmatischer Textdimension großzügig hinweg. [...] Dabei handelt es sich jedoch um zwei vollkommen verschiedene Dinge, denn ein Hypertext mag so nonlinear sein wie er will – das betrifft doch immer nur die Sequenz, in seiner paradigmatischen Dimension dagegen unterscheidet er sich nicht vom normalen Text" (Baßler 2005, 307f.). Bei Bolter scheint dieses ebenso banale wie fatale Missverständnis zu der Auffassung zu führen, dass der von Joyce ‚stark' bewirtschaftete intertextuelle Raum des *Ulysses*, dass also das vielschichtige Paradigma des Romans (seine „several layers of allusions") eine wundersame Vermehrung seines Textsyntagmas (seiner „simple storyline") bewirkt. Das geht so weit, dass am Ende der schrille Rückkopplungseffekt vieler amerikanischer Hypertext-Panels hier einmal in Druckschrift gebannt wird: „[Joseph] Frank's characterization of James Joyce remains appropriate for the hyperfictions of Michael Joyce" (Bolter 2001, 174).

2005), der den intertextuellen Raum (das ‚Paragrammaire' nach J. Kristeva) eines Bezugstextes auf eine historische Positivität von Kontext-Dokumenten bezieht, die er ‚Archiv' nennt, kann hier als fundierte und konstruktive Alternative dienen, deren Begrifflichkeit sich unmittelbar auf die Konzepte und Leistungen einer digitalen Texttechnologie beziehen lässt. Präzisierungen und Erweiterungen der Theorie werden dort anzustreben sein, wo das 'Archiv' nicht nur den biblionomen Raum, sondern zugleich den der Textualität überschreitet, indem es sich multimedial konstituiert.

(3*) Methode*: Intertextuelle Computerphilologie dieser transbiblionomen Art zielt nicht auf die *Implementierung* literarischer Intertextualität: ihre ‚Befreiung' aus dem ‚Gefängnis' der Druckzeile und ‚vollständige Entfaltung' im barrierefreien digitalen Schreibraum, sondern auf ihre *Supplementierung*: auf die forschungstechnische Unterstützung der selbstverständlich2 stets selektiven und (projekt- und methodenspezifisch) perspektivischen Erschließung des intertextuellen Potentials eines je gegebenen literarischen Texts. Bei der Entwicklung einer zweckmäßigen Arbeitsumgebung hätte die Kooperation zwischen Philologie, Medienwissenschaft, Texttechnologie und Informatik einer Logik *pragmatischer Schnittstellenbildung* zu folgen. Statt entweder digitale Lösungsmöglichkeiten mit philologischen Problemstellungen zu überfordern, oder umgekehrt von vornherein die philologischen Problemstellungen an das Leistungsvermögen digitaler tools anzupassen, wären für jede Teilaufgabe die Schnittstellen zwischen humaner Intelligenz und künstlicher Intelligenz präzise zu definieren, um die Leistungsvermögen von Menschen und Rechnern möglichst effizient miteinander zu verschalten.

Ausgehend von diesen Überlegungen projektieren wir die Entwicklung eines *Literary Memory Information System* (LiMeS) als einer literaturwissenschaftlichen Forschungsumgebung, die *literarische Texte* nicht einfach als Gegenstände, sondern *als Medien des kulturellen Gedächtnisses* behandelt, indem sie ihr intertextuelles Verweispotential erschließbar, darstellbar und für unterschiedliche Verwertungszusammenhänge nutzbar macht. Die texttechnologischen Entwicklungen der letzten Jahre bieten u. E. eine tragfähige Basis für die Konzeption und Implementierung einer digitalen Arbeitsumgebung für eine solche intertextuell und transbiblionom orientierte Literaturforschung.

3 Kafka als Paradigma

Das Werk Franz Kafkas bietet einen idealen Testgegenstand für ein solches Vorhaben. Benjamins Bemerkung, nach der „Kafkas ganzes Werk einen Kodex von Gesten darstellt" (Benjamin 1981, 18), lässt sich nämlich ohne weiteres von Kafkas Protagonisten auf seinen Text selbst übertragen. Dies ist im Weiteren näher auszuführen.

3.1 Kafkas literarische Gesten

Wenn sich die sekundäre Sprache der Literatur (J. Lotman) aus einem Ensemble *„literarischer Gesten"* („gestes littéraires") zusammen, die wiederum auf einen Horizont von Archetypen verweisen, den sie beständig reproduzieren, transformieren und überschreiten (Jenny 1976, 257), dann wäre diese Definition für Kafkas literarische Gesten zunächst zu modifizieren. Ihr Zuschnitt und ihre Verweis-Funktion auf den kulturellen Horizont beruhen bereits auf der Erkenntnis, dass dieser Horizont im Zeitalter der Hyperliterarisierung und der aufkommenden elektronischen Medien längst zerfallen ist.

2 Oder nicht ganz so selbstverständlichen, wie Stephan Porombka (2001, 104; 127ff.) in seiner Rekonstruktion der mit dem Hypertext verknüpften Vollständigkeits- und Totalitäts-Phantasien zeigt...

Zunächst sind es die vielgestaltigen, bereits von Massenmedien getragenen Nationalismen der Donaumonarchie, die die Zerfallsmasse dieses Horizonts als Steinbruch fragmentierter „Archaismen" ausbeuten, denen sie, je nach Standpunkt und Anlass, „einen ‚aktuellen Sinn' zu geben versuchen" (Deleuze/Guattari 1976, 35). Das intertextuelle Resonanzfeld der literarischen Gesten Kafkas – seine mauerbauenden Chinesen, nachahmenden Affen, forschenden Hunde, etc. – erstreckt sich durch alle diese Steinbrüche hindurch und setzt so okzidentale und orientalische Religionen und Mythologie, historische Narrative, die Philosophie und die modernen Wissenschaften, und nicht zuletzt das schier unüberschaubare Verbreitungsfeld massenmedialer Stereotypen. Dabei beschränken sich Kafkas literarische Gesten keineswegs darauf, die Versatzstücke ihres kulturellen Kontexts zu speichern und zu rearrangieren (zu letzterem v.a. Neumann 1996; Wagner 2006a). Sie implizieren vielmehr eine dauernd mitlaufende Reflektion der medialen und institutionellen Anschlusspunkte ihrer Operationen. Wo immer Kafka über Telefonzentralen (*Der Verschollene*), Schreibapparate (*In der Strafkolonie*), oder über bürokratische Aktenzirkulation schreibt (*Das Schloß*), transkribiert er die Medientechniken und -dispositive seiner Zeit in selbstreferentielle Protokolle seiner eigenen, nicht auf Sinnbildung oder -zerstörung, sondern auf intermediale Verknüpfung heterogener Kontexte abstellenden Schreibweise. So statten diese Gesten die Schrift mit einer monomedialen Intermedialität (also: Hypermedialität) aus, indem sie zum einen Inhalte aller denkbaren medialen Träger kopieren wie zum anderen diesen Vorgang literaler Inkorporation als mediale Operation schlechthin – nämlich die der Konnektivierung – explorieren (Dotzler 2008). Aufgrund des immensen Reichtums seines intertextuellen Resonanzfelds und der hohen Reflexionsschärfe für den Zusammenhang von Diskursivität, Medialität und Poetizität stellt Kafkas Werk eine ideale Herausforderung für die Entwicklung einer digitalen Arbeitsumgebung für transbiblionome Literaturforschung dar.

3.2 *Literarische Gesten* als Schnittstelle von Intertextualitätstheorie und Texttechnologie

Unter den skizzierten Voraussetzungen liegt es auf der Hand, dass substantialistische Konzeptionen von Intertextualität (wie etwa die der ‚Montage' oder des ‚Palimpsests') durch eine zugleich dynamische und relationale Konzeption zu ersetzen sind. So ließen sich die eingangs beschriebenen literarischen Gesten auch als Ensemble „kybernetischer Schlüssel" beschreiben, die jeweils „ganze Serien von Referenzen, Reminiszenzen, Konnotationen, Echos, Zitate, Pseudo-Zitate, Parallelen und Reaktivierungen" auslösen und rearrangieren (Topia 1984, 103). Während solche kybernetischen Effekte für alle Ebenen des literarischen Texts beschrieben werden können (Topik, Stil, Narrativik, Genre), hat die Kafkaforschung früh bemerkt, was Kafka in verschiedenen Formulierungen auch selbst bekundet: dass es in diesem Falle eine besondere Schicht intertextueller Referenzen gibt, die offenbar auf ein durchgängiges auktoriales Kalkül zurückgeht. Mit anderen Worten, Kafkas Schreibweise ist mit einer ungewöhnlich systematischen Bewirtschaftung des intertextuellen Raums verbunden. Sie basiert auf einem Satz von literarischen Gesten, die zwischen den Polen der Tmesis und des Stereotyps oszillieren und so auf die methodisch grundlegende, wenn auch stets unscharfe Unterscheidung zwischen einem *auktorialen Feld* und einem *historischen Emergenzfeld* verweisen (Wagner/Reinhard 2007, 102f.).3

3 Zum Verhältnis von Tmesis, Stereotyp und intertextueller Katalysis vgl. Baßler 2005, 212ff. Als Beispiel kann hier die Strafmaschine in Kafkas *Strafkolonie* dienen. Sie schließt einerseits an einen hochgradig stereotypisierten Gebrauch der Maschinen-Metapher für die Organisation und Funktion von Staat und Verwaltung an, andererseits schneidet sie sich durch die Einführung des Menschen als Schreibfläche (des Verurteilten, der in die Strafmaschine hineingelegt wird, die ihm dann sein Urteil in die Haut ritzt) von diesem weiten Resonanzfeld ab. Dieses Spannungsverhältnis

Hier liegt nun die entscheidende Schnittstelle zwischen einer literaturwissenschaftlichen und einer texttechnologischen Untersuchung von Intertextualität. Das grundlegende (besser: bodenlose) analytische Problem besteht ja gerade darin, dass die topischen Einheiten, die wir hier *literarische Gesten* nennen, *keine Bestandteile des literarischen Matrix- Textes* sind, sondern dass sie aus der komplexen Differenz zwischen dem Matrix-Text und dem Feld der Echo-Texte allererst hervorgehen. Sie sind mithin *zugleich Ausgangspunkt und Ziel einer empirischen Intertextualitätsforschung*. Was immer von menschlichen Forschern – sei es aufgrund der literarischen Tradition, sei es aufgrund im Regelfalle vereinzelter und jedenfalls selektiver Text-Kontext-Hypothesen – als Ausgangspunkt einer oder mehrerer intertextueller Verweise betrachtet wird, kann mithilfe texttechnologischer Suchfunktionen auf sein intertextuelles Verweispotential zumindest jener Bestandteile des Archivs überprüft werden, der für einen solchen digitalen Zugriff zur Verfügung steht.4 Vor allem aber soll die von uns projektierte Arbeitsumgebung auch umgekehrt die Möglichkeit bieten, bestimmte Wortzusammenstellungen aus dem Syntagma des Matrix-Textes mittels der durch LiMeS bereitgestellten Text Mining-Funktionen den menschlichen Forschern als Kandidaten für literarische Gesten sichtbar zu machen (s.u., 4.1.).

4 Entwicklung eines *Literary Memory Information System* als eHumanities-Vorhaben

Ziel des Vorhabens, das den Brückenschlag zwischen Literatur- und Medienwissenschaft auf der einen Seite und geisteswissenschaftlicher Fachinformatik und Medieninformatik auf der anderen Seite anstrebt, ist die Entwicklung eines *Literary Memory Information System* (LiMeS), dessen Leistungen sich auf drei zentrale Desiderate gegenwärtiger und künftiger Literaturforschung richten:

1. Die informatische Rekonzeption der essentiell konnektiven Logik moderner Printliteratur, sowie die Entwicklung angemessener digitaler Techniken und Werkzeuge zu ihrer Erschließung und Nutzung im Rahmen eines dezentralen und kollaborativen Forschernetzwerks (mit differenzierbaren Zugangsoptionen für ein prinzipiell unbegrenzten Nutzerkreis),

---

zwischen einer technisch-deskriptiven oder metaphorisch-stereotypen Kookkurrenz von Zeichen auf der einen Seite und einer ‚unerhörten' Kookkurrenz auf der anderen wird nun durch eine begrenzte Serie ‚starker' Intertexte vermittelt, von denen man annehmen kann, dass sie Teil eines kalkulierten Spiels seitens des Autors sind – von Nietzsches *Genealogie der Moral* über einen Aufsatz zur Funktionsweise der elektrischen Lochkarten-Zählmaschine bis zu Texten über elektrotherapeutische Apparate, auf deren Ähnlichkeit die Geschichte sogar selbst verweist (dazu ausführlich Wagner 2006b; Dotzler 2006). Kafka selbst hat dieses grundlegende Verhältnis zwischen Autorschaft, intertextuellem Dialog und intertextuellem Rauschen einmal in einer Beschreibung seines Schreibtisches zusammengefasst: „Ich kenne beiläufig nur das, was obenauf liegt, unten ahne ich bloß Fürchterliches" (Kafka 1976, 153). Einer digitale Arbeitsumgebung wie LiMeS soll es erstmals ermöglichen, auch das ‚Unten' der Intertexts und insbesondere seine Beziehungen zum ‚Obenauf' als positiven Untersuchungsgegenstand zu behandeln.

4 Als kompaktes und forschungsstrategisch hoch relevantes Beispiel kann hier die historische Tagespresse dienen, deren umfassende Digitalisierung gegenwärtig große Fortschritte macht. So hat die germanistische Kafkaforschung in Zeitungen wie dem *Prager Tagblatt* oder der *Bohemia* vereinzelt prägnante intertextuelle Beziehungen zu Stellen in Kafkas literarischem Werk ‚entdeckt', konnte deren weiterreichende Signifikanz aber niemals durch eine umfassende Korpusanalyse verifizieren. „Sie hat keinen schlechten Blick, aber zu konzentriert, sie sieht nur den Kern; den Ausstrahlungen zu folgen, die ja eben den Kern fliehen, ist ihr zu mühsam", hat Kafka (1958, 162) in diesem Zusammenhang einmal hellsichtig als Manko erkannt, was hier als zentrale Aufgabenstellung einer digitalen Literaturforschung zu entwickeln ist.

2. die Erschließung und Aufbereitung literarischer Texte als dynamische Archive für kultur- und mediengeschichtliche Forschung, sowie als Folien für eine Zeitdiagnose unserer heutige Situation und
3. die Bereitstellung einer eHumanities-Forschungsumgebung, die auf die zunehmende Verfügbarkeit von Print-Literatur im digitalen Raum reagiert, indem sie der digitalen Erfassung „eine begleitende Theoriedebatte und [...] ein verfeinertes methodisches Rüstzeug" (so die Forderung von Embach/Rapp 2008) zur Seite stellt.

Das Ziel, innovative rechnergestützte Arbeitsplätze zu schaffen, ist dabei alles andere als neu und begleitet die geisteswissenschaftliche Fachinformatik (Texttechnologie, Computerlinguistik, automatische Sprachverarbeitung etc.) als zentrales Desiderat seit vielen Jahren (vgl. etwa Barrow 1997).

Qualitativ neu an dem hier vorgestellten Ansatz ist dagegen die Kombination der folgenden Merkmale eines solchen Arbeitsplatzes:

- weitestgehende Integration digitaler Ausgangsmaterialien (Primär- und Sekundärtexte, Integration anderer Medien, v.a. Karikatur, Photographie, Film, (vgl. Wolff 2008))
- Aufbau auf den heute etablierten texttechnologischen Standards (die XML-Familie als Basisstandard sowie ihre spezifischen Anwendungen in der Texttechnologie wie etwa TEI P5 (Lobin & Lemnitzer 2004)
- Nutzung von Text Mining-Diensten (Mehler & Wolff 2005), die auf den (hier: Kafkaschen) Matrix-Text ebenso bezogen sein können wie auf das zeitlich und material begrenzte Reservoir seiner Echotexte oder, noch weitergehend, einen beliebig konfigurierbaren Texthorizont als Vergleichsmaterial
- der Aufbau auf aktuellen Konzepten der Softwarearchitektur (Reussner & Hasselbring 2006)
- die Bereitstellung als webbasierte Oberfläche, die Kollaboration und damit auch die gemeinsame Texterstellung und -kommentierung unterstützt.

4.1 Texttechnologische Perspektive

Wir gehen davon aus, dass Text Mining-Verfahren (Mehler & Wolff 2005) mittlerweile einen Reifegrad erreicht haben, der sie jenseits rein experimenteller Kontexte (der Texttechnologie-Forschung) als für die Integration in den philologischen Arbeitskontext geeignet erscheinen lässt. Damit geht ein solcher Arbeitsplatz funktional erheblich über die reine Durchsuchbarkeit digitaler Textressourcen (z. B. mit Hilfe von KWIC-Darstellungen, Konkordanzen, Volltextsuche) hinaus.

Texttechnologische Informationssysteme wie der rein webbasierte eHumanities Desktop (Mehler, Gleim et al. 2009) verfügen mittlerweile über eine Vielzahl texttechnologischer Kernfunktionen, die weit über elementare Operationen der Korpuserstellung und -verwaltung hinausgehen. Dies betrifft grundlegende Funktionen der automatischen Annotation textueller Aggregate wie die automatische Spracherkennung, die Satzgrenzenerkennung, das Stemming, die Lemmatisierung, das Part-of-Speech-Tagging, die Eigennamenerkennung oder auch die automatische Segmentierung von Dokumentstrukturen und deren Abbildung auf geeignete Standards wie die TEI P5 (Burnard 2007) bzw. den CES (Ide & Priest-Dorman 1998). Aber auch *Text Mining*-Funktionen wie das *Lexical Chaining* und die hierauf basierende unüberwachte Themenklassifikation sowie die semantische Suche bilden bereits Kernbestandteile des eHumanities Desktops (Gleim, Waltinger, e.a. 2009; Mehler, Gleim, e.a. 2008). Im Folgenden skizzieren wir in Ergänzung zu diesen Kernfunktionalitäten texttechnologische Bausteine als *notwendige* Einheiten

eines *Literary Memory Information System*, welches die Aufdeckung, Verwaltung, Suchbarmachung und literaturwissenschaftliche Erforschung dieser Art von intertextuellen Beziehungen zwischen literarischen Matrixtexten und genreübergreifenden Echotexten ermöglichen soll. Dabei gehen wir auf den Umstand ein, dass das Instrumentarium der automatischen Textanalyse im Bereich literarischer Texte auf ein Datenbeschaffungsproblem stößt, für das es in der bisherigen *Text Mining*-Forschung kaum Lösungsansätze gibt. Den Ausgangspunkt der im Folgenden zu beschreibenden LiMeS-Komponenten bildet ein *Text Mining*-Modell basierend auf dem Intertextualitätsbegriff in Verbindung mit dem Begriff des *Semantic Web* (Mehler 2004; 2005a; 2005b), für den die folgenden Überlegungen maßgeblich sind:

- Anders als bei der klassischen Textkategorisierung (Sebastiani 2002) geht es nicht darum, Texte auf vorgegebene Mengen von Inhaltskategorien abzubilden. In dem anvisierten Informationssystem stellen diese Inhaltskategorien vielmehr jene Erkenntniseinheiten dar, welche seine Anwender *mittels* der LiMeS-seitig zu explorierenden Vernetzung von Matrix- und Echotexten herausarbeiten.
- Anders als bei klassischen Methoden der Textkonversion geht es ferner nicht darum, vorgefundene intertextuelle Beziehungen zwischen Texten hypertextuell zu explizieren. Vielmehr wird die Auffassung zugrundegelegt, dass das Netzwerk der intertextuellen Beziehungen ein semantisches Netzwerk referenzieller und typologischer Verweisbeziehungen von Inhaltskategorien widerspiegelt, welche die Texte manifestieren. Dieser Lesart gemäß kann ein Text A auch dann auf einen Text B intertextuell bezogen sein, wenn der Autor von Text A den Text B nicht kannte, dieser aber zur literaturwissenschaftlichen Klärung der Semantik von A beiträgt, und zwar vor dem Hintergrund einer beiden Texten gemeinsamen oder verwandten Topik.

Die hierfür erforderlichen texttechnologischen Komponenten werden nachfolgend beschrieben. LiMeS als Beispiel für eine Klasse von eHumanities-Informationssystemen erfordert demnach folgende drei Kernkomponenten:

- Ein System für die Verschränkung von literaturwissenschaftlicher Interpretation und maschinellem Lernen (siehe Sektion 4.1.1)
- Social Software für die literaturwissenschaftliche Fachinformatik (siehe Sektion 4.1.2) sowie
- ein System für die ereignisorientierte Exploration intermedialer Relationen.

Nachfolgend beschreiben wir die ersten beiden dieser Komponenten. Abbildung 1 zeigt schematisch die Bandbreite von Systemen und Ansätzen zur Annotation literarischer Daten wie sie in einem LiMeS benötigt werden. Sektion 1.2 greift den wichtigen Aspekt der HCI-Gestaltung auf, dem im Falle von LiMeS besondere Beachtung gilt, da dessen Nutzer gerade keine Texttechnologen oder Informatiker, sondern Literaturwissenschaftler sind.

Abbildung 1: Drei Quellen von Annotationen literarischer Basisdaten und Korpora: (1) klassische texttechnologische und Text Mining-Verfahren; (2) Verfahren, welche aus der Verschränkung von menschlicher Expertise von *Text Mining* hervorgehen sowie (3) Verfahren, die auf *Human Computation* beruhen.

4.1.1 Mining-Algorithmen für die Literaturwissenschaft

Die automatische inhaltsorientierte Annotation literarischer Daten kann nicht durch die einfache Übertragung herkömmlicher *Text Mining*-Ansätze gelingen. Die Interpretationsleistung eines Literaturwissenschaftlers lässt sich nicht in das Gerüst einer vorgegebenen Menge von Inhalts- oder Strukturkategorien zwängen. Auch ist hinsichtlich der Merkmalsselektion, die jedem *Text Mining* vorangeht, zu bedenken, dass das anvisierte LiMeS auf literarische Texte *und* auf Gebrauchstexte zielt. Das bedeutet nicht nur, dass Texte völlig verschiedener Genres zu verarbeiten sind. Vielmehr ist auch zu bedenken, dass *Text Mining*-Verfahren für Gebrauchstexte, nicht aber für literarische Texte konzipiert wurden, deren Sprache im Vergleich zu ersteren eine völlig andere Funktion hat. Zu dieser horizontalen, funktionalen Heterogenität — die auch das Problem der Multilingualität der Zielkorpora einschließt — tritt die vertikale, entstehungsgeschichtliche Vielfalt der Quellentexte hinzu: Denn Matrix- und Echotexte können zeitgeschichtlich verschiedenen Sprachstufen angehören. Vor dem Hintergrund dieses komplexen Gegenstands muss das *Text Mining* neue Wege beschreiten, wenn es im Bereich literarischer Texte vergleichbare Erfolge erzielen will wie im Bereich herkömmlicher Gebrauchstextsorten (Rolf 1993).

Ein solcher Ansatz soll im Folgenden am Beispiel der oben (3.1., 3.2.) beschriebenen *literarischen Gesten* skizziert werden. Seine Grundidee besteht in der Entwicklung eines Bootstrapping-Algorithmus, der das unüberwachte Lernen mit der teilüberwachten Interpretationsleistung von Literaturwissenschaftlern integriert:

1. Ausgehend von einem klassischen Lernszenario zur Identifikation von *literarischen Gesten* wird in einem ersten Schritt ein unüberwachter Klassifikator entwickelt, der potentielle *literarische Gesten* in Inputtexten identifiziert und dem beteiligten Literaturwissenschaftler vorlegt.
2. Der menschliche Experte hat dann die Gelegenheit, die ihm vorgelegten Kandidaten zu bewerten. Doch anders als bei klassischen Feedback-Algorithmen soll der Experte die Möglichkeit erhalten, auf der Basis wohldefinierter Operationen über dem operativen Repräsentationsmodell, gestaltcharakterisierende Transpositionen und vergleichbare Modifikationen der vorgelegten Kandidaten zu identifizieren. Beruht das Repräsentationsmodell beispielsweise auf

einem *bag of words*-Modell, so besteht eine Transposition etwa in dem Austausch oder dem Löschen merkmalbildender Wörter.

3. Die Transpositionen werden im nächsten Schritt dem Klassifikationsalgorithmus zugänglich gemacht, der sie für die neuerliche Identifikation von Relais-Instanzen nutzt. Durch die Iteration dieses Verfahrens besteht die Möglichkeit einer Konvergenz der Klassifikationsleistung bei einer feingliedrigen Abstimmung des zugrundeliegenden Repräsentationsmodells aufgrund der Interpretationsleistungen des Experten.

Dieses Szenario einer fortgesetzten Verschränkung von unüberwachtem maschinellem Lernen einerseits und Rückkoppelung an die Interpretationsleistung des Experten andererseits weicht insofern von klassischen Lernszenarien ab, als hier auch auf Seiten des Experten nicht die Kenntnis des Kategoriensystems als Bezugssystem für seine Transpositionsleistung vorausgesetzt wird. Dieses wird gewissermaßen erst im Zuge der Arbeit mit dem Algorithmus (Stichwort 'dynamische Kategorisierung') erschlossen und strukturiert. Im Extremfall koppeln sich zwei unüberwachte Agenten, ein künstlicher und ein menschlicher, wobei letzterer aufgrund seiner maschinell nicht einholbaren Intuition dem dynamischen Prozess der Exploration und Kategorisierung von Relais eine Richtung gibt. Zielgröße dieses Algorithmus sind Relais, über deren Existenz oder Tragweite der Wissenschaftler im Voraus keine (hinreichende) Kenntnis besitzt. Dabei handelt es sich um Relais, die erst im Zuge der Arbeit mit LiMeS identifiziert werden und daher möglicherweise einer enormen Fluktuation ausgesetzt sind. Der Schwerpunkt dieses Verfahrens liegt daher auf der Exploration zuvor unbekannter, überraschender, neuartiger literarischer Strukturen. In diesem Sinne ist von einem Literatur-Mining zu sprechen.

Auch wenn ein solcher Ansatz vielversprechend und wegen der Offenheit literarischer Interpretationsleistungen unabdingbar ist, basiert er letztlich auf der Koordination der Klassifikationsleistung eines menschlichen und eines künstlichen Agenten. Es ist also stets ein Literaturwissenschaftler, der in Kooperation mit dem Text-Mining-System tritt. Soll nun die Annotationsleistung einer solchen Paarung vergrößert werden, müssen weitere Literaturwissenschaftler hinzutreten. Das bedeutet aber, dass für ein Mehr an annotierten Daten der Einsatz menschlicher Experten linear zu vergrößern ist. Gerade für den vorliegenden Gegenstandsbereich mit seinen großen Mengen an multimedialen Daten, auf die LiMeS fokussiert, ist dies ein schwieriges Szenario. Zwar bedeutet diese Einschätzung keine prinzipielle Abkehr von Text-Mining-Algorithmen. Wenn es jedoch darum geht, eine Vielzahl hochwertiger Annotationen zu erzeugen, welche letztlich den Mehrwert eines LiMeS ausmachen, dann sind in Ergänzung hierzu alternative Wege zu beschreiten. Eine solche Alternative skizziert die folgende Sektion unter dem Stichwort der *Social Software*.

4.1.2 Exploration literarischer Daten im großen Maßstab: Social Software für die literaturwissenschaftliche Fachinformatik

Eine Bezugsgröße für die texttechnologische Vorverarbeitung literarischer Daten besteht in der Nutzbarmachung von *Social Software* (Bächle 2006) im Rahmen des so genannten *crowdsourcing* von Annotationsleistungen. Es geht dabei um die computergestützte Delegation von Annotationsaufgaben an das *Human Computation* (von Ahn 2008), an Gruppen von Annotatoren also, die auf der Basis eines *game with a purpose* (von Ahn & Dabbish, 2008; von Ahn, Liu, e.a., 2006) eine Aufgabe lösen, deren Lösung kaum oder gar nicht automatisierbar ist. Im vorliegenden Fall handelt es

sich dabei um die Annotation von Interpretationen literarischer Daten – im Hinblick auf die Annotation einzelner oder Gruppen semiotischer Aggregate, und zwar bezogen auf deren syntaktische, semantische oder pragmatische interne oder externe Strukturierung. Für solche Ansätze existieren mittlerweile erfolgreiche webbasierte Plattformen wie *mechanical turk* (http://www.mturk.com), auf denen Freiwillige – gegen symbolisches Honorar – als *human intelligence task* (HITs) bezeichnete Aufgaben wie Bildannotation, Textproduktion oder Befragungen – erledigen.

Unter dem Blickwinkel inhaltsbasierter Intertextualität und Intermedialität ist es die externe Strukturierung semiotischer Aggregate die im Fokus der anvisierten LiMeS-Instanz steht. Hier steht man unter anderem vor der Aufgabe der Annotation intermedialer Relationen, von Relationen zwischen Bildzeichen einerseits und Textzeichen andererseits. Es geht beispielsweise um die Frage, in welchem Segment eines Matrixtexts welches Bild beschrieben oder auch nur indirekt thematisiert wird. Offenbar handelt es sich bei der Beantwortung solcher Fragen um Aufgaben, die noch lange einer Automatisierung harren werden. Damit steht die literaturwissenschaftliche Fachinformatik jedoch vor dem Problem, einerseits große Datenmengen (etwa in Form von Trainingsdaten) für die Exploration intermedialer Relationen zu benötigen, diese aber nur über den aufwendigen Weg der intellektuellen Annotation beschaffen zu können. So werden beispielsweise im großen Maßstab textuelle Annotationen von Bildern benötigt, um deren (teil-)automatische Relationierung mit anderen Bild- oder Textmedien zu betreiben. Unter der Voraussetzung, dass sämtliche der zu verarbeitenden Bildmedien textuell annotiert sind, kann diese Aufgabe im Prinzip mittels etablierter Verfahren des *Text Mining* angegangen werden, so z.B. mit einer latenten semantischen Analyse (*latent semantic analysis*) (Berry, Drmač, e.a. 1999), die neben Texten auch Bilder, und zwar mittels ihrer textuellen Repräsentationen verarbeitet. *Wie aber gelangt man zu solchen großmaßstäblichen und zugleich qualitativ hochwertigen Annotationen?* Genau diese Frage führt unter dem vorliegenden Szenario auf das Konzept des *game with a purpose* (von Ahn & Dabbish 2008). In dieser Sektion skizzieren wir kurz die Kernbestandteile einer solchen Art von "Spiel mit literaturwissenschaftlichem Zweck", das zu einem wesentlichen Baustein jeder Art von LiMeS zählen darf, und zwar zur Gewährleistung von Annotationen literarischer Daten, die qualitative Tiefe mit quantitativer Breite verbinden und derzeit nicht anders für die literaturwissenschaftliche Fachinformatik zu beschaffen sind.

Allgemein gesprochen kann ein zweckorientiertes Annotationsspiel durch Implementierung zweier Arten von Informationsteilsystemen umgesetzt werden. Dies betrifft zum einen das Kernspiel selbst, das zwei oder mehr Spielpartner in einem Annotationspiel vereinigt und deren Spielresultate in Form von Annotationen an einer Serie von Zielobjekten (etwa Bilder, Texte oder Bild-Textbeziehungen) verwaltet. Zum anderen betrifft dies ein übergeordnetes Verwaltungsprogramm, dass Spieler und Zielobjekte auswählt, um deren Spielresultate schließlich je Zielobjekt über viele Spielabläufe hinweg in eine Art *Medianannotation* zu überführen, die als repräsentative Annotation dieses Objekts gilt. Diese beiden Informationsteilsysteme werden im Folgenden kurz beschrieben:

- *Annotationsspiele mit literaturwissenschaftlichem Zweck:* Den Ausgangspunkt bildet die Gestaltung von Spielen mit dem Zweck der Annotation literarischer Daten. Ein solches Spiel bezeichnen wir enger als literarisches Annotationsspiel bzw. als *LitErary Data Annotation* (LEDA) *game*. Anders als im Falle der von von Ahn (2006) und von Ahn, Liu, e.a. (2006) betrachteten Spiele zielt LEDA nicht auf die bloße Annotation von Aggregaten als Ganzes oder auf deren Segmentierung (wie im Falle der Bildsegmentierung). Vielmehr geht es darum, das *Human*

Computation für die gleichzeitige Segmentierung *und* Relationierung nicht explizit verknüpfter Aggregate nutzbar zu machen. Dabei wird der Annotationsprozess unter mindestens zwei Spielpartnern aufgeteilt, und zwar so, dass ein Spieler in der Rolle des Spielleiters bzw. Überwachers sein Gegenüber in der Rolle des Annotators so anleitet bzw. führt, dass dieser möglichst gute Annotationsleistungen erbringt. Im Falle eines literarischen Annotationsspiels steht das Spieldesign vor der besonderen Aufgabe, zugleich ikonographische und symbolische, textuelle Informationen den Spielpartnern darzubieten. Als Seiteneffekt solcher Annotationsspiele wird die korrekte Segmentierung und Relationierung einer Vielzahl von Text- und Bildmedien erwartet. Wegen des grundlegenden Spielcharakters des gesamten Annotationsprozesses wird weiterhin eine besondere Motivation und damit Qualitätssteigerung der Annotationen erwartet. Von Ahn, Liu, e.a. (2006) zeigen, dass *games with a purpose* es erlauben, hochkomplexe Annotations- und Suchaufgaben an Gruppen von Annotatoren zu delegieren, insbesondere in solchen Bereichen, in denen nur geringe Mengen annotierter Daten zur Verfügung stehen.

- *Medianannotationen:* Ein zweites Informationsteilsystem des Social Taggings als Bestandteil des anvisierten LiMeS thematisiert die Verwaltung des *community building* unter den LEDA-Spielern und ihrer Spielergebnisse. Bei einer gegebenen Zahl von Spielern besteht eine Vielzahl von Möglichkeiten der Auswahl von Paarungen, die – graphentheoretisch gesprochen – unterschiedliche Graphen aufspannen, von denen je nach Aufgabe, Gruppenzusammensetzung und tatsächlicher Spielerverfügbarkeit jene Paarungen auszuwählen sind, die einen effizienten Spielablauf garantieren. Es geht also um die Implementierung eines *Annotation Game Management System* (AGMS), das ferner die Speicherung, Verwaltung und das Retrieval von Spielergebnissen ermöglicht. Das AGMS hat unter anderem die Aufgabe, je Spielrunde Spielpartner und Zielobjekte zu selektieren, deren Annotationen zu verwalten und hierauf basierend zielobjektbezogene Medianannotationen zu berechnen. Es geht ferner um die Aufgabe der Nachverarbeitung von Annotationen, um die Kontrolle und Bewertung von Spielen gegebenenfalls mit dem Ziel, defizient annotierte Zielobjekte zum Gegenstand weiterer Spielrunden unter Wahl neuer Paarungen zu machen. Aus informatorischer Sicht bildet die Berechnung und Verwaltung von Medianannotationen die größte Herausforderung im Zuge der Implementierung eines AGMS. Zur Berechnung von Medianannotationen steht im Prinzip das Instrumentarium der Graphähnlichkeitsmessung zur Verfügung. Das mag zwar naheliegen – da Annotationen Graphen aufspannen –, jedoch auch paradox erscheinen, wenn man bedenkt, dass die Berechnung von Graphisomorphien NP-hart ist (Dehmer 2005). Es steht jedoch nunmehr eine Reihe von Graphähnlichkeitsalgorithmen zur Verfügung, welche in geschickter Weise an bestimmte Graphklassen angepasst sind und Näherungen für Ähnlichkeitswerte von Graphen zu berechnen erlauben, deren Komplexität weitaus geringer ist (Bunke & Günter 2001; Dehmer & Mehler 2007; Mehler, Geibel e.a. 2007).

Das AGMS mit seinem integrierten Annotationsspiel soll zur Segmentierung und Relationierung von Zeichnungen, Karikaturen, Bildern, Photographien, literarischen und Gebrauchstexten eingesetzt werden. Unter dem Aspekt der *Social Software* geht es also um die Entwicklung eines Bausteins der literaturwissenschaftlichen Fachinformatik, der gerade nicht auf die ausschließlich maschinelle Exploration intermedialer oder bloß intertextueller Relationen zielt – etwa unter der Entwicklung und Anwendung bekannter, im vorliegenden Anwendungsbereich jedoch hochwahrscheinlich hochgradig fehleranfälligen Algorithmus des maschinellen Lernens.

Vielmehr geht es um die Implementierung eines Algorithmus des *human computation* für das *crowdsourcing* von semantischen Annotationen, welche die Interpretationsleistungen von Literaturwissenschaftlern bzw. Textrezipienten abbilden. Dieser Weg erscheint uns unumgänglich zur Exploration der algorithmisch noch immer als uneinholbar geltenden menschlichen Kompetenz in Bezug auf das Verstehen, Explorieren und Verknüpfen semiotischer Aggregate. Die Implementierung einer solchen Software verspricht, einen Bereich der Intermedialität von Zeichen in einem sehr viel größeren Umfang zu erschließen als es bisher gelungen ist, einen Bereich, der seiner Komplexität wegen allen bisherigen Projekten im Dunstkreis literaturwissenschaftlichen Fachinformatik verschlossen geblieben ist.

4.2 Softwarearchitektur

Das *Literary Memory Information System* ist modular konzipiert. Seine Bausteine sollen als *notwendige* Einheiten die Aufdeckung, Verwaltung, Suchbarmachung und literaturwissenschaftliche Erforschung dieser Art von intertextuellen Beziehungen zwischen literarischen Matrixtexten und genreübergreifenden Echotexten ermöglichen. Dabei gehen wir auf den Umstand ein, dass das Instrumentarium der automatischen Textanalyse insbesondere im Bereich der Merkmalsselektion am Beispiel literarischer Texte auf Probleme stößt, für die es in der bisherigen Forschung zum *Text Mining* kaum Lösungsangebote gibt. Zur Bewältigung dieser Aufgabenlast beschreiben wir LiMeS als ein Informationssystem, das dem heute gebräuchlichen Mehrebenenmodell der Softwarearchitektur (Wolff 2004; Reusner & Hasselbring 2006) folgt und dabei auch Elemente serviceorientierter Architekturen realisiert. Abbildung 1 oben zeigt bereits die wesentlichen Schichten:

1. Die *Persistenzebene* der Datenverwaltung, die in LiMeS die Aufgaben der Corpus- und Medienverwaltung übernimmt. Neben den in LiMeS selbst verwalteten Texten und Medien müssen auch externe Ressourcen (Textarchive, Medienbibliotheken etc.) dynamisch eingebunden werden können.
2. Die in den Abschnitten 4.1.1 und 4.1.2 beschriebenen Komponenten der automatischen, teilautomatischen und intellektuellen Analyse, Relationierung und Annotation machen einerseits wesentliche Teile der *Applikationslogik* von LiMeS aus und bilden gleichzeitig die Brücke in die Interaktionsschicht (Benutzerschnittstelle), da intellektuelle Annotation Systeminteraktion erforderlich macht.
3. In der *Interaktionsschicht* müssen geeignete Sichten für die für LiMeS vorgesehenen vielfältigen Benutzerrollen geschaffen werden: Die Spanne reicht hier von der Text- und Medienverwaltung.

Da eine Nutzung der mit LiMeS erreichten Ergebnisse nicht nur über die unmittelbare Interaktion mit dem System, sondern auch auf dem Weg der Diensteintegration in andere Systeme denkbar erscheint, kommt mittelfristig auch die Erweiterung um Dienste im Sinne *der Service Oriented Architecture* (SOA) in Betracht (Wolff 2003).

Es ist geplant, die Software-Architektur von LiMeS so zu implementieren, dass sie als Aufsatz auf dem eHumanities-Desktop (Mehler, Gleim e.a. 2009) ruht. Das bedeutet insbesondere, dass LiMeS den Desktop als Korpusmanagementsystem wie auch als System zur Verwaltung von Nutzungsrechten verwendet. Auch ist damit die Möglichkeit gegeben, das umfangreiche System an text-

technologischen Methoden der Korpusvorverarbeitung und Korpusanalyse, das der Desktop bietet, unmittelbar zu nutzen. Die modulare und objektorientierte Architektur des rein webbasierten eHumanities Desktop unterstützt gerade solche Erweiterungen, wie sie mit der Implementierung von LiMeS am Beispiel des Werkes von Kafka geplant sind. Diesen Weg zu beschreiten wird in diesem Zusammenhang eine der kommenden texttechnologischen Hauptaufgaben sein.

5 Ausblick

In diesem Beitrag fokussieren wir auf die literatur- und medientheoretische Konzeption von LiMeS und die Möglichkeiten ihrer Umsetzung mit Hilfe von Verfahren aus der Texttechnologie und des sich schnell entwickelnden Feldes sozialer webbasierter Anwendungen. Im Hintergrund steht dabei die Annahme, dass Hypertextrelationierung, Medieneinbindung, automatische Erzeugung und intellektuelle Annotation in transbiblionomen Räumen zu materiellen Ergebnissen führen, die einen erheblichen qualitativen Unterschied zu den Ergebnissen traditioneller Editionsphilologie und deren Fortführung mit digitalen Mitteln aufweisen. Dies führt schließlich auch zu neuen Nutzungsmöglichkeiten für den forschenden und lernenden Anwender. Für die Operationalisierung dieses Modells wird daher die Bezugnahme auf nutzerseitige Anforderungen (vgl. Toms & O'Brien 2008) an e-Humanities-Anwendungen ein wesentlicher Erfolgsfaktor sein.

6 Literatur

Bächle, M. (2006), Social software. In: Informatik Spektrum, 29(2):121-124.

Barrow, J. (1997), A Writing Support Tool with Multiple Views. Computing and the Humanities, 31(1), 13-30.

Baßler, M. (2005), Die kulturpoetische Funktion und das Archiv. Eine literaturwissenschaftliche Text-Kontext-Theorie, Tübingen: Francke.

Benjamin, W. (1981), Franz Kafka. Zur zehnten Wiederkehr seines Todestages, in: Benjamin über Kafka. Texte, Brieferzeugnisse, Aufzeichnungen, ed. Hermann Schweppenhäuser, Frankfurt a.M.: Suhrkamp, 9-38.

Berry, M.W., Z. Drmač, & E. R. Jessup (1999), Matrices, vector spaces, and information retrieval. SIAM Review, 41(2):335-362.

Biemann, C.; Bordag, S., Quasthoff, U., & Wolff, C. (2004, Mai 2004), Web Services for Language Resources and Language Technology Applications. Paper presented at the Proceedings Fourth International Conference on Language Resources and Evaluation [LREC 2004], Lissabon.

Bolter, J. D. (2001), Writing Space. Computers, Hypertext and the Remediation of Print, Mahwah, NJ/London: Erlbaum.

Bunke, H. & Günter, S. (2001), Weighted mean of a pair of graphs. Computing, 67(3), 209-224.

Burnard, L. (2007), New tricks from an old dog: An overview of TEI P5. In L. Burnard, M. Dobreva, N. Fuhr, and A. Lüdeling, editors, Digital Historical Corpora- Architecture, Annotation, and Retrieval, number 06491 in Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany.

Dehmer, M. (2005). Strukturelle Analyse Web-basierter Dokumente. Multimedia und Telekooperation. Berlin: DUV.

Dehmer, M. & A. Mehler, A (2007), A new method of measuring the similarity for a special class of directed graphs. Tatra Mountains Mathematical Publications, 36:39-59.

Deleuze, G.; Guattari, F. (1976), Kafka. Für eine kleine Literatur, Frankfurt a.M.: Suhrkamp.

Derrida, J. (1967), Grammatologie, Frankfurt a.M.: Suhrkamp.

Derrida, J. (1984), Two words for Joyce, in: Attridge, D. & Ferrer, D., eds., Poststructuralist Joyce: Essays from the French, Cambridge: CUP, 145-159.

Dotzler, B. J. (2006), Pervasive Bureaucracy: The Case of Herman Hollerith., in: Austriaca. Cahiers universitaires d'information sur l'Autriche, no. 60, 45-52.

Dotzler, B. J. (2008), Kafka zwischen den Medien, in: J. Paech; J. Schröter, eds., Intermedialität – Analog/Digital, Paderborn: Fink, 181-192.

Embach, M., Rapp, A. (2008), Rekonstruktion und Erschließung mittelalterlicher Bibliotheken Neue Formen der Handschriftenpräsentation. Berlin: Akademie-Verlag [=Beiträge zu den Historischen Kulturwissenschaften, Bd. 1].

Gleim, R., Waltinger, U., Ernst, A. Mehler, A., Esch, D., & Feith, T. (2009), The eHumanities desktop—an online system for corpus management and analysis in support of computing in the humanities. In Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics EACL 2009, 30 March - 3 April, Athens

Hofmannsthal, H. von (1907), Der Dichter und diese Zeit, in: Die Neue Rundschau. XVIIIer Jg. Der freien Bühne, Bd. 1, H.3, 257-276.

Ide, N., & G. Priest-Dorman, G. (1998), Corpus encoding standard. http://www.cs.vassar.edu/CES/.

Jenny, L. (1979), La stratégie de la forme, in: Poétique, no. 27, 257-281.

Kafka, F. (1958), Briefe, hrsg. v. Brod, M., Frankfurt a.M.: S. Fischer.

Kafka, F. (1976), Briefe an Felice, hrsg. v. Born, J., Frankfurt a.M.: S. Fischer.

Lachmann, R. (1984), Ebenen des Intertextualitätsbegriffs, in: Stierle, K.H.; Warning, R., Das Gespräch, München: Fink, 133-138.

Lobin, H. & Lemnitzer, L., eds., (2004). Texttechnologie. Perspektiven und Anwendungen, Tübingen: Stauffenburg.

Mehler, A. (2004a): Textmining. In: Lobin, Henning und Lothar Lemnitzer (eds.): Texttechnologie. Perspektiven und Anwendungen, Seiten 329-352. Stauffenburg, Tübingen.

Mehler, A. (2004b), Textmodellierung: Mehrstufige Modellierung generischer Bausteine der Textähnlichkeitsmessung. In Mehler, A. and Lobin, H., editors, Automatische Textanalyse: Systeme und Methoden zur Annotation und Analyse natürlichsprachlicher Texte, 101-120. Wiesbaden: Verlag für Sozialwissenschaften.

Mehler, A. (2005a), Lexical chaining as a source of text chaining. In Patrick, J. and Matthiessen, C. (eds.), Proceedings of the 1st Computational Systemic Functional Grammar Conference, University of Sydney, Australia, pages 12-21.

Mehler, A. (2005b), Zur textlinguistischen Fundierung der Text- und Korpuskonversion. Sprache und Datenverarbeitung, 1:29-53.

Mehler, A., & Wolff, C. (2005), Einleitung: Perspektiven und Positionen des Text Mining. LDV-Forum, 20(1), Einführung in das Themenheft Text Mining, 1-18.

Mehler, A., Geibel, P., & Pustylnikov, O. (2007), Structural classifiers of text types: Towards a novel model of text representation. LDV Forum – Zeitschrift für Computerlinguistik und Sprachtechnologie, 22(2):51-66.

Mehler, A., Gleim, R., Ernst, A., & Waltinger, U. (2008). WikiDB: Building interoperable wiki-based knowledge resources for semantic databases. Sprache und Datenverarbeitung. International Journal for Language Data Processing, 32(1):47-70.

Mehler, A., Gleim, R., Ernst, A., Waltinger, U, Ernst, A., Esch, D., & T. Feith (2009), eHumanities Desktop — eine webbasierte Arbeitsumgebung für die geisteswissenschaftliche Fachinformatik. In Proceedings of the Symposium "Sprachtechnologie und eHumanities", 26. Und 27. Februar, Duisburg-Essen University.

Meister, J. C. (2005), Projekt Computerphilologie Über Geschichte, Verfahren und Theorie rechnergestützter Literaturwissenschaft, in: Segeberg, H. & Simone Winko, S., eds., Literarität und Digitalität. Zur Zukunft der Literatur, München: Fink, 315-341.

Pêcheux, M. (1983), Über die Rolle des Gedächtnisses als interdiskursives Material, in: Geier, M., Woetzel, H., eds. (1983), Das Subjekt des Diskurses. Beiträge zur sprachlichen Bildung von Subjektivität, Berlin: Argument-Verlag, 50-58.

Porombka, S. (2001), Hypertext. Zur Kritik eines digitalen Mythos, München: Wilhelm Fink.

Rayward, W. B. (2008), European modernism and the information society: informing the present, understanding the past, Aldershot: Ashgate.

Reussner, R., & Hasselbring, W. (eds.). (2006), Handbuch der Software-Architektur. Heidelberg: dpunkt.

Rolf, E. (1993), Die Funktionen der Gebrauchstextsorten. Berlin/New York: De Gruyter.

Schmidt, T., & Wolff, C. (2004), Dokumentbezogenes Wissensmanagement in dynamischen Arbeitsgruppen. Text Mining, Clustering und Visualisierung. In B. Bekavac, J. Herget & M. Rittberger (Eds.), 9. Internationales Symposium für Informationswissenschaft (ISI 2004) (Vol. Information zwischen Kultur und Marktwirtschaft, pp. 317-336). Chur (CH): UVK.

Sebastiani, F. (2002), Machine learning in automated text categorization. ACM Computing Surveys, 34(1):1-47.

Toms, E. G., & O'Brien, H. L. (2008). Understanding the information and communication technology needs of the e-humanist. Journal of Documentation, 64(1).

Topia, A. (1984), The Matrix and the Echo: Intertextuality in Ulysses, in: Post-Structuralist Joyce: Essays From the French, eds. D. Attridge/D. Ferrer, Cambridge: Cambridge UP, 103-125.

von Ahn, L. (2006), Games with a purpose. IEEE Computer Magazine, 39(6):92-94.

von Ahn, L. (2008), Human computation. In ICDE, pages 1-2. IEEE, 2008.

von Ahn, L. & L. Dabbish (2008), Designing games with a purpose. Commun. ACM, 51(8):58-67.

von Ahn, L., Liu, R, & Blum, M. (2006), Peekaboom: a game for locating objects in images. In CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems, pages 55-64, New York: ACM Press.

Wagner, B. (2006a), Kafkas phantastisches Büro, in: K. Scherpe/E. Wagner, eds., Kontinent Kafka. Mosse Lectures an der Humboldt-Universität zu Berlin, Berlin: Vorwerk 8, 104-118.

Wagner, B. (2006b), Connecting Cultures. Heinrich Rauchberg, Franz Kafka, and the Hollerith Machine, in: Austriaca. Cahiers universitaires d'information sur l'Autriche, no. 60, 53-68.

Wagner, B./Reinhard, T. (2007), Das Virtuelle Kafka-Bureau, in: I. Jonas, ed., Sinn und Nutzen von Datenbanken in den Geisteswissenschaften, Frankfurt: Peter Lang, 95-114.

Wolff, C. (2003), Web Services im e-Learning und e-Publishing. In: K.-P. Fähnrich & H. Herre, eds., Content- und Wissensmanagement. Leipzig: Leipziger Informatik-Verbund / Universität Leipzig, 123-132.

Wolff, C. (2004), Systemarchitekturen. Aufbau texttechnologischer Anwendungen. In L. Lemnitzer & H. Lobin (Eds.), Texttechnologie. Perspektiven und Anwendungen (pp. 165-192). Tübingen: Stauffenburg.

Wolff, C. (2005), Generierung ontologischer Konzepte und Relationen durch Text Mining-Verfahren. Paper presented at the Knowledge eXtended. Die Zusammenarbeit von Wissenschaftlern, Bibliothekaren und IT-Spezialisten, Jülich.

Wolff, C. (2008), Veränderte Arbeits- und Publikationsformen in der Wissenschaft und die Rolle der Bibliotheken. In E. Hutzler, A. Schröder & G. Schweikl (eds.), Strategien zum Aufbau digitaler Bibliotheken (pp. 157-172). Göttingen: Universitätsverlag Göttingen.