Linguistische und semantische Annotation eines Zeitungskorpus

Clematide, S ; Volk, Martin

Abstract: Dieser Artikel beschreibt das Vorgehen beim automatischen inkrementellen Aufbereiten eines rohen Textkorpus mit linguistischer und semantischer Information. Es wird gezeigt, wie das Erkennen von Eigennamen hilft, die Wortartenkategorisierung und partielle syntaktische Analysen zu verbessern. Eine Evaluation über ca. 1000 Sätze zeigt die Stärken und Schwachpunkte der verschiedenen Erkenner auf.

Linguistische und semantische Annotation eines Zeitungskorpus

Simon Clematide, Martin Volk

Universität Zürich
Institut für Informatik
Forschungsgruppe Computerlinguistik
Winterthurerstr. 190
CH-8057 Zürich

{siclemat,volk}@ifi.unizh.ch

Zusammenfassung. Dieser Artikel beschreibt das Vorgehen beim automatischen inkrementellen Aufbereiten eines rohen Textkorpus mit linguistischer und semantischer Information. Es wird gezeigt, wie das Erkennen von Eigennamen hilft, die Wortartenkategorisierung und partielle syntaktische Analysen zu verbessern. Eine Evaluation über ca. 1'000 Sätze zeigt die Stärken und Schwachpunkte der verschiedenen Erkenner auf.

1 Einleitung

Um ein reines Textkorpus linguistisch (Erkennen von Sätzen, Wortarten, einfachen Nominal- und Präpositionalphrasen) und semantisch (Erkennen von Personen-, Firmen-, geographischen Bezeichnungen, temporalen und lokalen Phrasen) zu annotieren, haben wir verschiedene Werkzeuge und Wissensquellen selbst entwickelt und mit fremden integriert. Das inkrementell annotierte Korpus soll zu einer Ressource werden, die statistisches Material liefert, um Präferenzen für die Analysen eines tiefen Parsers zu finden; insbesondere für die Auflösung von Anbindungsmehrdeutigkeiten von Präpositionalphrasen. Für die semantische Annotation verwenden wir Ansätze der Informationsextraktion, die zusammen mit seichten linguistischen Analysen gemessen am Aufwand zufriedenstellende Resultate ergeben.

2 Annotationsschritte

Jeder einzelne Annotationsschritt ist programmiertechnisch als Filter realisiert, der eine textuelle Repräsentation des Korpus einliest und sie mit den entsprechend veränderten und erweiterten Informationen wieder ausgibt. Alle vorgestellten Programme sind in der Programmiersprache Perl implementiert, was eine hohe Flexibilität im Einbinden von Fremdprogrammen für morphologische Analyse und Wortartenbestimmung ermöglicht. Obwohl sich für die Integration von semantischer, Wortarten- und syntaktischer Information in das Korpus ein XML-basiertes Format empfehlen würde, haben wir aus praktischen Gründen zeilenorientierte Formate verwendet. Es

ist jedoch ein vergleichsweise kleiner Aufwand, das Schlussresultat der Annotation in ein XML-Format zu bringen.

2.1 Schritt I: Dokumentstrukturierung

Die deutschsprachige ComputerZeitung (CZ) hat die Jahrgänge 1993-1997 mit etwa 7,5 Mio. Token (50MB) auf CDROM publiziert; den Jahrgang 1996 haben wir für Testzwecke ausgespart. Obwohl nur reines Textformat vorhanden ist, konnten Überschriften und Paragraphenabschnitte über Zeilenlänge, -ende u.ä. zuverlässig erkannt und als dokumentstrukturierendes Markup eingefügt werden. Bei der Satzendeerkennung wurde eine Liste von 1200 Abkürzungen zur Punktklassifikation verwendet.

2.2 Schritt II: Partielle Namenerkennung und -klassifizierung

Die Berichte und Meldungen aus dem IT-Business, aus denen die CZ vor allem besteht, enthalten viele Personen-, Firmen- sowie geographische Namen. Das Problem des Identifizierens und Klassifizierens solcher Ausdrücke wird in der Informationsextraktion [1] unter der Bezeichnung "Named Entity Recognition" schon einige Zeit erforscht. Da diese Verfahren immer fehlerbehaftet sind, lohnt es sich nur, häufig vorkommende Klassen zu bestimmen. Wir haben uns deshalb auf die drei oben erwähnten Typen beschränkt. Das Erkennen der einzelnen Typen läuft hintereinander geschaltet ab durch jeweils separate Programme, die öffnende und schliessende Tags als Markup für die jeweilige Kategorie einfügen:

- Personennamen: <PERS> De Benedetti </PERS>
- Firmennamen: <FA> Olivetti Personal Computers </FA>
- Geographische Namen: <GEO> New York </GEO>

Eine Tokenfolge bildet einen zu klassifizierenden Ausdruck, wenn sie beim Nachschlagen im entsprechenden Namensverzeichnis vorkommt. Diese sogenannte interne Evidenz muss z. T. noch verfeinert werden, indem das Verzeichnis während des Erkennens dynamisch wächst oder schrumpft, oder kontextuelle Faktoren interne Evidenz aufheben. Das Hauptproblem ist deshalb das Erstellen der Namensverzeichnisse. Unser Ziel ist es, möglichst viele Elemente dieser Listen automatisch aus dem Korpus selbst zu gewinnen.

Personennamen. Mit Hilfe eines Verzeichnisses von ca. 16'000 Vornamen und anderen Schlüsselwörtern (*Dr.*) werden Nachnamen inklusive Genitivformen identifiziert und in die Liste aufgenommen. Das blinde Anwenden dieser Namensliste ist zu ungenau. Viele Token, die Namen bilden, stehen noch für anderes (z. B. *Prof. Bier* vs. *Bier* als Getränk, *Hagen* als Vor-/Nachname vs. Ortsbezeichnung). Deshalb wird die Namensliste jeweils auf einen eingeschränkten Kontext zugeschnitten.

Geographische Namen. Artikelkennzeichnungen der CZ mit Städtenamen haben geholfen, bestehende Listen mit Städten (950), Ländern sowie Bundesländern (250) zu vervollständigen. Dank morphologischer Analyse durch GERTWOL [3] und der

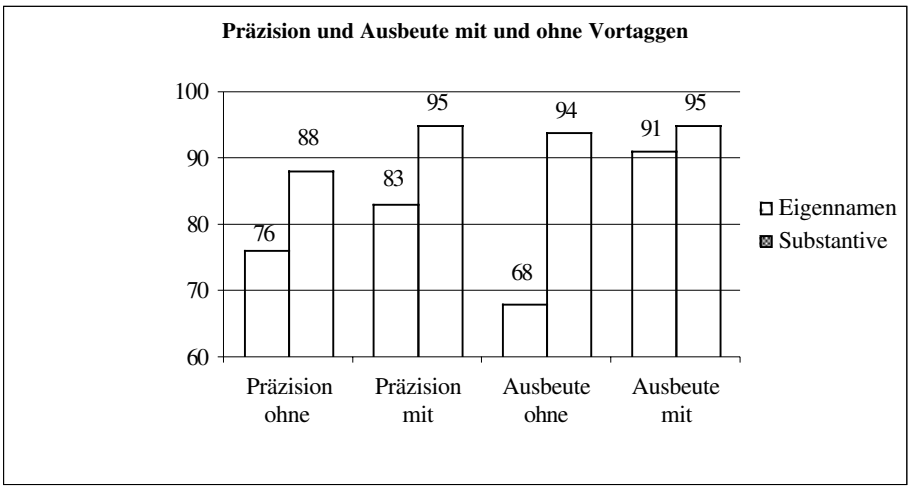
darin vermerkten Unflektierbarkeit konnten 250 grossgeschriebene geographische Adjektive (*Basler*, *Nürnberger*) aus dem Korpus extrahiert werden; sie sind bei der Bestimmung der Vordergrenzen von Firmennamen nützlich, da die Grossschreibung hier kein hinreichender Indikator ist.

Firmennamen. Diese werden über Muster identifiziert: Grossgeschriebene Token nach *Firma* oder vor *GmbH*, Erstglieder von bestimmten Bindestrichkomposita wie in *Siemens-Tochter*. Wegen der Grossschreibung gewöhnlicher Substantive ist das Problem der Begrenzung von Firmennamen fürs Deutsche schwieriger zu behandeln als etwa im Englischen. Deshalb werden alle Kandidaten von GERTWOL morphologisch analysiert, wobei alles herausfällt, was nicht unbekannte Wörter (*Acotec*), Abkürzungen (*AT&T*) oder bekannte Eigennamen (*Apple*) sind. Für einige verbreitete Abkürzungen (*EDV*) mussten wir zudem manuell eine Stoppwortliste erstellen. Gefunden wurden 3'588 einteilige Firmennamen, Genitivformen eingerechnet. Bei den 3'421 mehrteiligen Firmennamen (*Axel Springer AG*) werden oft Akronyme eingeführt, die als Alias im Textverlauf wieder erscheinen, oder es werden elliptische Wiedererwähnungen gemacht. Insgesamt sind 9'423 Einträge in der Firmennamenliste. Darin enthaltene Personennamen sind unproblematisch, da diese in einer vorhergehenden Verarbeitungsstufe klassifiziert werden.

2.3 Schritt III: Wortartenerkennung

Wir verwenden den statistischen TreeTagger [8] fürs Deutsche, der an einem umfangreichen STTS-getaggten Korpus trainiert wurde. Er kann SGML-Tags ignorieren, die wir für semantisches Markup und Dokumentstrukturierung einsetzen, und vorgegebene Wortartentags berücksichtigen. In Schritt II erkannte Namen bekommen das korrekte Tag NE für Eigennamen mitgegeben, was wiederum die Erkennung von Substantiven (NN) positiv beeinflusst (vgl. Tabelle 1).

Tabelle 1. Einfluss des Vortaggens von Eigennamen



Volk und Schneider [11] haben gezeigt, dass die Unterscheidung zwischen Eigennamen und Substantiven eine notorische Fehlerquelle beim Taggen deutscher Texte ist. Das Vortaggen lindert dieses Problem und senkt die ursprüngliche Fehlerrate des Taggers von 6% auf 4%. Mit weiteren im Verarbeitungsverlauf gemachten Korrekturen kann die Fehlerrate auf 3% gedrückt werden.

2.4 Schritt IV: Erkennung von Nominal- (NP) und Präpositionalphrasen (PP)

Die Annotation mit syntaktischen Teilstrukturen setzt ein Grammatikmodell voraus. Wir halten uns möglichst nah an das Grammatikmodell, welches für das NEGRA-Korpus [10] entwickelt wurde, und theorieunabhängige sowie datenorientierte Beschreibungen anstrebt. Es werden keine leeren Kategorien eingefügt und die Phrasenstrukturen sind flach gehalten. Dafür erhalten Knotenverbindungen konsequent funktionale Bestimmungen. Auf der von uns analysierten Ebene fällt allerdings nur unspezifisch gehaltene funktionale Information an, da z. B. auf eine explizite Bestimmung des Kopfes einer NP bewusst verzichtet wird.

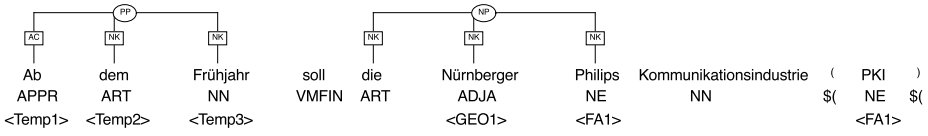


Abb. 1. Ausschnitt eines partiell annotierten Satzes in der graphischen Darstellung des annotate-Tools[1]

In Abb. 1 wurde automatisch eine PP und eine NP erkannt (ovale Knoten). Das Beispiel zeigt auch, dass die Erkennung zum Teil unvollständig ist. In den eckigen Kästchen stehen die funktionalen Bestimmungen (NK=noun kernel, AC=adpositional case marker).

Die NP/PP-Erkennung ignoriert die lexikalische Ebene und setzt auf den Wortartentags auf. Einfache nicht-rekursive[2] kontextfreien Regeln kombinieren die Tags zu Konstituenten, bzw. verbinden Konstituenten zu grösseren Einheiten. Die folgenden zwei Regeln sind für die Erkennung der Phrasen in Abb. 1 verantwortlich:

```
NP --> ART=NK ADJA=NK NE=NK

PP --> APPR=AC ART=NK NN=NK
```

Der Regelrumpf besteht aus Ausdrücken folgender Form:

Tag/Konstituente = Funktionsbestimmung

---

[1] Informationen zu den Tags der verschiedenen Stufen und zum Annotationstool finden sich im WWW unter http://www.coli.uni-sb.de/sfb378/negra-corpus/. Die unterste Zeile, die für Wortkommentare gedacht ist, haben wir für die semantische Annotation zweckentfremdet. Das semantische Markup für phrasale Kategorien ist durch ansteigend durchnummerierte Tags für jedes Token der Phrase markiert.

[2] Rekursive Regeln könnten durchaus verarbeitet werden, es gibt momentan nur keine in unserer Grammatik. Tilgungsregeln hingegen – die das Grammatikmodell 'glücklicherweise verbietet' – wären für die Buttom-Up-Strategie des partiellen Parsers vernichtend.

Der zur NP/PP-Erkennung verwendete Parsingalgorithmus gleicht einem deterministischen Shift-Reduce-Verfahren [5: 64ff.] und erzeugt in der in Perl implementierten Version aus vertikalisiertem getaggten Text direkt das zeilenbasierte NEGRA-Exportformat. Die effizienten und mächtigen Pattern-Matching-Fähigkeiten von Perl erlauben auch bei grossen Textmengen und über 150 Regeln eine zügige Verarbeitung. Neben einfachen und koordinierten NPs und PPs werden dabei auch noch Adjektivphrasen erkannt. Um das Verfahren zu verstehen, eignet sich eine Formulierung in der logischen Programmiersprache PROLOG besser.

Match/Reduce-Algorithmus in PROLOG. Die beiden obigen Regeln können durch folgende 2 Klauseln des Prädikats rule/2 ausgedrückt werden:

```
rule('NP'(['ART'(Art)='NK','ADJA'(Adja)='NK','NE'(Ne)='NK']),
     ['ART'(Art),'ADJA'(Adja),'NN'(Ne)]).

rule('PP'(['APPR'(Appr)='AC','ART'(Art)='NK','NN'(Nn)='NK']),
     ['APPR'(Appr),'ART'(Art),'NN'(Nn)]).
```

Das Hauptprädikat partial_parse/2 nimmt getaggten Text entgegen, sammelt alle Regeln auf und wendet diese auf den getaggten Text an.

```
partial_parse(Tagged, Result) :-
    findall(rule(LHS,RHS), rule(LHS,RHS), Rules),
    apply_rules(Rules, Tagged, Result).
```

Das Prädikat apply_rules/3 stellt sicher, dass jede Regel auf das Zwischenresultat angewendet wird, das aus den vorher benutzten Regeln entstanden ist

```
apply_rules([], Result, Result).

apply_rules([rule(LHS,RHS)|Rules], List, Result) :-
    match_reduce(List, RHS, LHS, Reduced),
    apply_rules(Rules, Reduced, Result).
```

Die eigentliche Arbeit leistet match_reduce/4, das in der Konstituentenliste nach Segmenten sucht, die auf einen Regelrumpf passen (*match*) und diese durch den Regelkopf ersetzt (*reduce*).

```
match_reduce([], _, _, []).

match_reduce(List, RHS, LHS, Result) :-
    append(RHS, Rest, List),
    match_reduce([LHS|Rest], RHS, LHS, Result).

match_reduce([X|Xs], RHS, LHS, [X|Result]) :-
    match_reduce(Xs, RHS, LHS, Result).
```

Im Unterschied zum klassischen Shift-Reduce-Verfahren wird jede Regel nur einmal angefasst und reduziert dann gleich alle passenden Segmente. Damit lässt sich durch die Reihenfolge der Regeln die Struktur der entstehenden Konstituenten steuern.

Um die Analyse zu starten, muss nur noch der getaggte Text im entsprechenden Format übergeben werden und die entsprechende Struktur für die erste PP aus Abb. 1 wird berechnet.

```
?- partial_parse(['APPR'('Ab'),'ART'(dem),'NN'('Frühjahr')],S).
```

Erweiterung des Regelformats. Der Verzicht auf morphologische und lexikalische Information limitiert die Leistungsfähigkeit der NP/PP-Erkennung klar. Um wenigstens das Wissen zu berücksichtigen, das bei der Namenserkennung eingefügt wurde, haben wir das Regelformat erweitert, damit es semantisches Markup berücksichtigen kann.

Semantisches Tag `%` *Tag/Konstituente = Funktionsbestimmung*

Die semantischen Tags sind optional, und die Regeln, die solche Tags enthalten, werden möglichst früh und in einer Reihenfolge verwendet, die *longest match* erlaubt.

Falls der zweiteilige Firmenname "Philips Kommunikationsindustrie" korrekt erkannt wäre, würde mit Hilfe der beiden folgenden Regeln die gewünschte Struktur wie in Abb. 2 aufgebaut.

```
MPN --> <FA1>%NE=PNC  <FA2>%NE=PNC

NP  --> ART=NK  ADJA=NK  MPN=NK
```

MPN steht dabei für "multi-word proper noun" und PNC für "proper noun component".

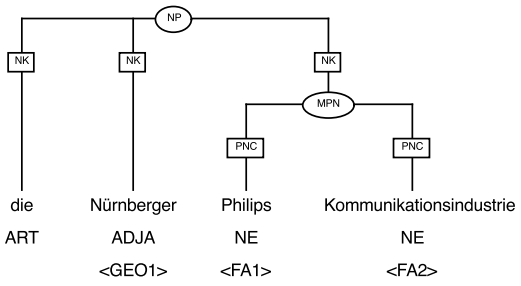


Abb. 2: Erwünschte Struktur durch Zusammenfassen von mehrteiligen Namen

2.5 Schritt V: Erkennung temporaler und lokaler Phrasen

Die semantische Klassifikation von Phrasen erfordert Disambiguierung von Wortbedeutungen, da nur wenige Präpositionen und Substantive eindeutig zuzuordnen sind. Es wurden Listen von temporalen und lokalen Substantiv-Mustern erstellt, die sich sowohl auf Einzelwörter wie Komposita anwenden lassen. Temporale bzw. lokale Präpositionalphrasen werden dann über die Kookkurrenz von potentiellen temporalen bzw. lokalen Präpositionen und von Nominalen kategorisiert, die durch die Substan-

tiv-Muster erkannt wurden. Zu letzteren gehören auch alle vorgängig erkannten geographischen Namen.

3 Evaluation der semantischen Annotation

Ein Testkorpus aus 990 Sätzen (13'987 Token, davon 1'119 dokumentstrukturierendes Markup) wurde für die Evaluation bezüglich Wortarten und semantischer Klassifikation zuerst automatisch annotiert und dann von Hand kontrolliert bzw. korrigiert. Für eine zuverlässige Überprüfung und qualitative Fehlersuche sind graphische Benutzerschnittstellen mit intuitiven Darstellungsformen wichtig. Es gibt sowohl für semantisches Annotieren (TATOE) [7] wie für syntaktisches Annotieren (annotate) solche Werkzeuge, leider haben wir kein Tool gefunden, das beides komfortabel integriert. Immerhin lässt es annotate zu, dass in die Wortkommentare die nötige semantische Information gesteckt werden kann (siehe Abb. 1).

Semantische Klassifikation ist schwierig, wie die Diskussion um SENSEVAL [4] belegt, und die Übereinstimmungswerte zwischen Annotierenden sind viel kleiner als bei Wortarten: Oft ist Weltwissen notwendig, um Firmen von anderen Organisationen (*Meta Group*) zu unterscheiden. Geographische Namen – etwa solche, die Orte mit politische Institutionen bezeichnen (*Bonn*) – werden gerne metonymisch verwendet. Wörtlicher und metaphorische Verwendung von lokalen Ausdrücken fliessen ineinander über; im Hinblick auf die Verwendung des Korpus für die Auflösung von Anbindungsmehrdeutigkeiten stellt sich die Frage, wie Lokalitäten in virtuellen Räumen, d.h. Internetadressen, interpretiert werden.

Das Auswerten von Wortartenkategorisierung ist einfach, da jedes Tag einzeln entweder richtig oder falsch erkannt ist. Bei der Kategorisierung von zusammengesetzten Ausdrücken kann zusätzlich teilweise richtig und teilweise falsch erkannt werden. In dieser Schwierigkeit spiegeln sich die beiden Teilprobleme beim Erkennen von zusammengesetzten Ausdrücken:

- Identifikation von Beginn und Ende eines Ausdrucks
- Klassifikation eines Ausdrucks.

Die Module für die Namenserkennung lösen die beiden Probleme gleichzeitig. Die Bestimmung von Beginn und Ende temporaler und lokaler Phrasen hängt von der Zuverlässigkeit der Phrasen-Erkennung ab. Letztere hat insbesondere mit dem Erkennen von Appositionen Schwierigkeiten, so dass ein Mass, das nur vollständig erkannte Phrasen zählt, besonders bezüglich Ausbeute unfair wäre.

Um trotzdem eine einfache automatische Auswertung analog zur Wortartenauswertung zu ermöglichen, haben wir zwei Werte ausgezählt:

E1. Die richtige oder falsche Klassifizierung des *ersten* Bestandteils eines Ausdrucks

E2. Die richtige oder falsche Klassifizierung einzelner Token eines Ausdrucks unabhängig von der korrekten Identifikation des ersten Bestandteils

Aus der Menge der vorhandenen (A) und automatisch entdeckten (B) Klassifizierungen, wurden

$$\text{Präzision } P = \frac{|A \cap B|}{|B|} \quad , \text{ Ausbeute } R = \frac{|A \cap B|}{|A|}$$

sowie das harmonische Mittel *F* dieser Werte ermittelt:

$$F = \frac{2 \times P \times R}{P + R}$$

Geeigneter wäre für Zwecke der Korpusannotation ein kombiniertes Mass, das die Präzision höher gewichtet, da wir möglichst wenig Falschinformation einfügen möchten.

Tabelle 2. Evaluation der semantischen Klassifikation

Kategorie	*Mod.*	*vorhanden A*	*erkannt B*	*Präzision P*	*Ausbeute R*	*Mittel F*
Personen-namen	E1	116	109	94%	89%	92%
	E2	199	192	95%	91%	93%
Geograph. Namen	E1	165	187	81%	92%	86%
	E2	182	199	83%	91%	87%
Firmen-namen	E1	347	372	80%	86%	83%
	E2	435	431	79%	78%	79%
Lokale Phrasen	E1	131	62	82%	39%	53%
	E2	360	159	80%	35%	49%
Temporale Phrasen	E1	263	246	81%	76%	79%
	E2	547	340	91%	57%	70%

Tabelle 2 ist wie folgt zu interpretieren: Im Testkorpus sind 116 Personennamen vorhanden, diese bestehen aus 199 Token. In Zeile E1 wird gezählt, wieviele Anfangstoken von Personennamen korrekt erkannt wurde (109). In Zeile E2 wird dagegen gezählt, wieviele der 199 Token als Personenamentoken erkannt wurden (192).

Personen – und geographische Namen werden am besten – und deshalb auch zuerst erkannt. Das Erkennen von einteiligen Namen ist leicht anspruchsvoller und Fehler wirken sich im Auswertungsmodus E2 etwas weniger aus. Schwieriger sind Firmennamen, die oft aus längeren Tokensequenzen bestehen, oder Funktionswörter enthalten, was die Bestimmung der Endbegrenzung erschwert. Der markant tiefere Ausbeutewert von E2 bei Firmennamen gegenüber E1 zeigt dies deutlich. Die tieferen Werte beim Erkennen von lokalen Phrasen deuten darauf, dass es dort eine grössere Ausdrucksvielfalt gibt als bei temporalen Phrasen. Zudem wirkt sich im tiefen Ausbeutewert unsere Haltung am stärksten aus, unsichere Information nicht zu berücksichtigen. Bei den temporalen Phrasen gibt es zwischen den beiden Auswertungsmodi massive Unterschiede. Die schlechtere Präzision in E1 deutet auf misslungene Segmentierung hin, die Ausbeute in E2 wird durch nicht gefundene lange Konstituenten gedrückt. Eine qualitative Fehleranalyse lässt vermuten, dass eine verbesserte NP/PP-Erkennung eine höhere Gesamtleistung erbringen würde. Es bleibt noch abzuklären, wie gross der Nutzen durch die Verwendung eines statistischen Chunkers [9] wäre.

4 Schluss

Die einzelnen Annotationsschritte bilden eine Kaskade von Programmmodulen, die auch separat eingesetzt werden können. Zu verbessern bleibt die syntaktische Analyse, wobei sicher gewisse lexikalische und morphologische Information eingebunden werden muss. Die Namenerkennung möchten wir noch auf Produktebezeichnungen ausdehnen und einen Teilsatz-Erkenner (*clause recognition*) integrieren.

Unsere Korpusannotation ist vergleichbar mit dem Vorgehen in SPPC [6] für die Zwecke der Informationsextraktion, geht aber bei der Erkennung von lokalen und temporalen Phrasen darüber hinaus. Ein interessanter Punkt von SPPC ist, dass die unterschiedlichen Annotationsinformationen in einer sogenannte "Text-Chart" gespeichert werden, die mehrdeutige Analysen repräsentieren kann und voreilige Disambiguierung verhindern hilft. Für das Englische bietet GATE [2] eine Umgebung, die einen ähnlichen Ansatz erlaubt, wobei die Module komfortabel integriert sind.

Dank. Wir möchten Dominik Merz für die Implementation der NP/PP-Erkennung in Perl und Charlotte Merz für die Korrektur des Evaluationskorpus herzlich danken.

Literaturhinweise

1. Cunningham, H.: Information Extraction – a User Guide. Institute for Language, Speech and Hearing (ILASH): Sheffield (1999)
2. Cunningham, H., Wilks, Y., Gaizauskas, R.: GATE -- a General Architecture for Text Engineering. In: Proceedings of the 16th Conference on Computational Linguistics (COLING-96) (1996)
3. Haapalainen, M., Majorin, A.: GERTWOL: Ein System zur Automatischen Wortformenerkennung deutscher Wörter. (1994)
4. Kilgariff, A.: Gold Standard Datasets for Evaluating Word Sense Disambiguation Programs. Computer Speech and Language, Vol. **12**.4 (1998)
5. Naumann, S., Langer, H.: Parsing: Eine Einführung in die maschinelle Analyse natürlicher Sprache. Leitfäden und Monographien der Informatik. B.G.Teubner, Stuttgart (1994)
6. Neumann, G., Piskorski, J.: An Intelligent Text Extraction and Navigation System. In: Proceedings of 6th International Conference on Computer-Assisted Information Retrieval (RIAO-2000). Paris (2000)
7. Rostek, L., Alexa, M.: Marking up in TATOE and exporting to SGML: Rule development for identifying NITF categories Computer-assisted corpus-based text analysis. Computers and the Humanities, Vol. **31**.4 (1998)
8. Schmid, H.: Improvements In Part-of-Speech Tagging With an Application To German. In: Proceedings of the ACL SIGDAT-Workshop. (1995)
9. Skut, W., Brants, T.: Chunk Tagger – statistical recognition of noun phrases. In: Proceedings of the Sixth Workshop on Very Large Corpora. Montreal (1998).
10. Skut, W., Krenn, B., Brants, T.,Uszkoreit, H.: An Annotations Scheme for Free Word Order Languages. In: Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP). Washington, D.C. (1997)
11. Volk, M.,Schneider, G.: Comparing a statistical and a rule-based tagger for German. In: Proceedings of KONVENS-98. Bonn (1998)