Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institut für
Technische Informatik und
Kommunikationsnetze

Bericht zur Semesterarbeit SA-2009-07

Automatische Bestimmung von unbekannten Wörtern

Jürg Schellenberg

Betreuer: Tobias Kaufmann

Juni 2009

Erklärung

Ich erkläre hiermit, das Merkblatt Plagiat[1] zur Kenntnis genommen, die vorliegende Arbeit selbständig verfasst und die im betroffenen Fachgebiet üblichen Zitiervorschriften eingehalten zu haben.

Zürich, 9. Juni 2009

Jürg Schellenberg

[1] Merkblatt des Rektorats der ETH Zürich http://www.ethz.ch/students/semester/plagiarism_s_de.pdf

Inhaltsverzeichnis

1 Einleitung

Bei der automatischen Analyse von natürlichsprachigen Sätzen greift ein Parser üblicherweise auf ein umfangreiches Lexikon zurück. Im Falle von lexikalisierten Präzisionsgrammatiken wie zum Beispiel Head-driven Phrase Structure Grammars (HPSG, siehe [SWBS99]) enthält das Lexikon eine detaillierte syntaktische Beschreibung für jedes erfasste Wort. Das sogenannte Zipf'sche Gesetz besagt, dass in der natürlichen Sprache einige wenige Wörter sehr häufig vorkommen, während sehr viele Wörter nur selten auftreten. Dies hat zur Folge, dass man in einem gegebenen Text viele Wörter antrifft, die nicht im (notwendigerweise begrenzten) Lexikon enthalten sind. Um uneingeschränkte Texte automatisch analysieren zu können, muss deshalb das Problem der "unbekannten" (d.h. nicht im Lexikon aufgeführten) Wörter gelöst werden. Im Rahmen dieser Semesterarbeit soll dazu ein statistisches Modell verwendet werden, welches die syntaktischen Eigenschaften eines unbekannten Wortes vorhersagt. Dabei sollen Informationen aus dem Kontext des unbekannten Wortes als Grundlage zur Schätzung von dessen syntaktischen Eigenschaften dienen. Weil die daraus resultierende Detailliertheit der Spezifikationen über jene der Wortarten (engl. part-of-speech tags) hinaus geht, wird dieser Prozess auch als 'Supertagging' bezeichnet.

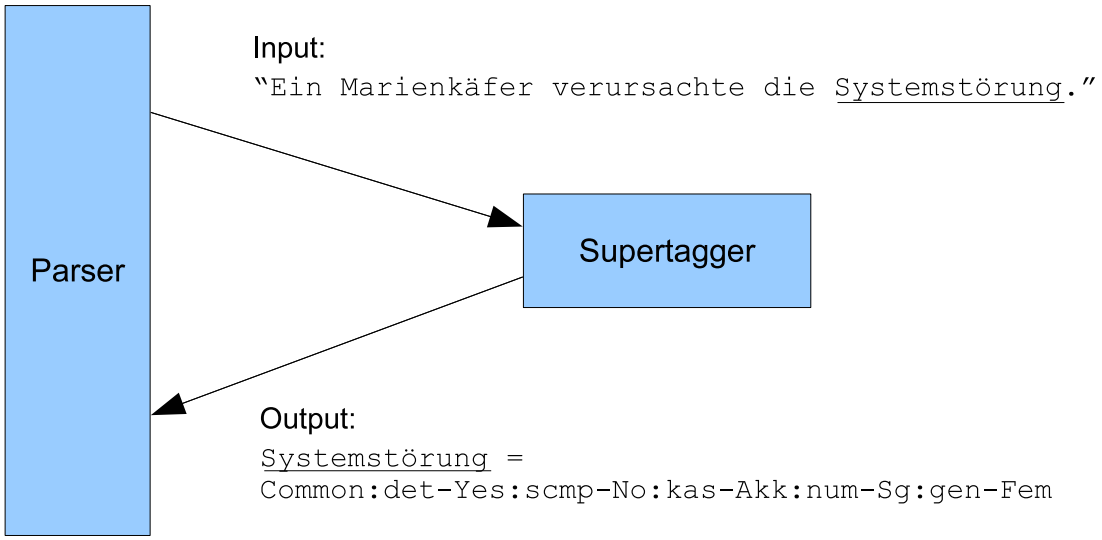
1.1 Aufgabenstellung

- Spezifikation der für den Parser relevanten syntaktischen Eigenschaften im Falle von unbekannten Wörtern
- Planung und Implementation eines Programmes (Supertagger) zur Schätzung der syntaktischen Eigenschaften von Substantiven in Java
- Experimentelle Evaluation des Supertaggers
  - Analyse und Optimierung der Genauigkeit des Supertagger-Ansatzes
  - Einfluss des Supertaggers auf die Qualität der syntaktischen Analyse und die Performance des Parsers
- Vergleich mit bestehenden Arbeiten

2 Ansatz

2.1 Supertagger

Supertagger ist der Name des im Rahmen dieser Semesterarbeit zu erstellenden Programms zur automatischen Bestimmung der syntaktischen Eigenschaften für im Parser-Lexikon nicht aufgeführte Substantive. In der Praxis wird der Parser also für jedes ihm unbekannte Substantiv u den Supertagger aufrufen, um Auskunft über u's syntaktische Eigenschaften zu erhalten. Die alsdann vom Supertagger erfolgende Schätzung von u's syntaktischen Eigenschaften beruht auf Informationen aus dessen Kontext, welcher durch den u enthaltenden Satz abgesteckt werden soll. Figur 1 veranschaulicht die Schnittstelle anhand eines Beispiels.



Figur 1: *Schnittstelle zwischen Parser und Supertagger*

2.2 Supertags

Als Rückgabewert auf eine Anfrage übergibt der Supertagger dem Parser einen oder mehrere Supertags. Dabei stellt ein Supertag die Kapselung mehrerer syntaktischer Eigenschaften eines Substantivs dar. Jede syntaktische Eigenschaft nimmt wiederum genau einen Wert aus einer geschlossenen Wertemenge, den sogenannten Klassen, an. Die folgende Aufstellung nennt die für eine Satzanalyse des Parsers als relevant erachteten syntaktischen Eigenschaften und deren Klassen:

Kasus: Klassen {*Nominativ, Genitiv, Dativ, Akkusativ*}

Numerus: Klassen {*Singular, Plural*}

Genus: Klassen {*Maskulin, Feminin, Neutrum*}

Artikel: Klassen {*Ja, Nein*}. Beispiele: *"das Haus"*, *"diese Sache"*, *"unser Auto"*, *"welche Farbe"*. Untersucht die Zählbarkeit des Substantivs.

Titel: Klassen {*Ja, Nein*}. Beispiele: *"Stadtpräsidentin Mauch"*, aber nicht: *"die Stadtpräsidentin"*. Gesucht ist die grammatisch aussergewöhnliche Konstruktion ohne Artikel.

Massangabe/-einheit: Klassen {*Ja, Nein*}. Beispiele: *"fünf Liter Wasser"*, *"ein Dutzend Krüge warme Milch"*, *"zwei Bücher Unsinn"*. Ein Grossteil dieser Konstrukte könnte mit Hilfe einer Wörterliste erkannt werden, es kommen jedoch auch atypische Masseinheiten vor (siehe letztes Beispiel).

Adverbiale Zeitangabe: Klassen {*Ja, Nein*}. Beispiele: *"Er hat den ganzen Tag geschlafen."*, *"Sie hat diese Woche keine Zeit."*. Diese Konstruktionen können vom Parser nur schwerlich von ganz normalen Akkusativobjektiven unterschieden werden.

Eigenname: Klassen {*Ja, Nein*}

Satzkomplement: Klassen {*Komplementsatz, Interrogativsatz, Hauptsatz, Infinitivverbalphrase, Kein*}. Um grössere Flexibilität zu erzielen, wurden die Untertypen der Satzkomplemente zusätzlich mittels separaten statischen Modellen geschätzt.

Komplementsatz: Klassen {*Ja, Nein*}. Beispiele: *"Die Tatsache, dass es so ist."*

Interrogativsatz: Klassen {*Ja, Nein*}. Beispiele: *"Die Frage, ob es so ist."*

Hauptsatz: Klassen {*Ja, Nein*}. Beispiele: *"Die Behauptung, es sei so."*

Infinitivverbalphrase: Klassen {*Ja, Nein*}. Beispiele: *"Die Lust, so zu sein."*

Nicht alle Kombinationen syntaktischer Eigenschaften sind sinnvoll. So wird beispielsweise kaum je ein Substantiv gleichzeitig die Eigenschaften *Titel* und *adverbiale Zeitangabe* tragen. Um die Supertags schlank zu halten und um dem Parser unnötigen Aufwand zu ersparen, erwies es sich als lohnend, die syntaktischen Eigenschaften hierarchisch zu strukturieren. Es wurden dazu fünf Arten von Supertags, sogenannte Haupttypen, festgelegt, wobei gewisse syntaktische Eigenschaften implizit an den Haupttyp gebunden sind, während andere explizit als Parameter zum Haupttyp mit angegeben werden müssen. Tabelle 1 listet die fünf Haupttypen mit ihren Parametern auf, Tabelle 2 enthält die möglichen Parameterwerte der Haupttypen und ihre Bedeutungen.

Supertag-Haupttyp	Parameter				
stag-np-common	<*det*>	<*scmp*>	[*kas*]	[*num*]	[*gen*]
stag-np-measure	<*det*>		[*kas*]	[*num*]	[*gen*]
stag-np-temporal			[*kas*]	[*num*]	[*gen*]
stag-np-proper			[*kas*]		
stag-np-title					

Tabelle 1: *Die fünf Haupttypen der Supertags und ihre Parameter*

Bemerkungen zu Tabelle 1:

- Auch der Haupttyp wird analog zu den syntaktischen Eigenschaften mittels eines statistischen Modells geschätzt.
- Sowohl der Haupttyp von der Parameterliste, als auch die Parameter untereinander werden durch Doppelpunkte getrennt.
- Die Parameter in spitzen Klammern sind zwingend anzugeben, jene in eckigen Klammern sind optional.

- Je nach Konfiguration ist es dem Supertagger erlaubt, dem Parser auch mehrere Supertags zurück zu geben. Diese werden dann durch Kommas getrennt.
- Sollte es bei Rückgabe mehrerer Supertags vorkommen, dass bei ansonsten identischen Supertags für einen optionalen Parameter alle möglichen Werte in der Resultatemenge erscheinen, so wird dieser Parameter weggelassen.

Parameter	Werte	Syntaktische Eigenschaft	Klasse
det	stag-det-yes	Artikel	Ja
	stag-det-no		Nein
scmp	stag-scmp-none	Satzkomplement	Kein
	stag-scmp-main		Hauptsatz
	stag-scmp-inf		Infinitivverbalphrase
	stag-scmp-int		Interrogativsatz
	stag-scmp-cmp		Komplementsatz
kas	stag-kas-nom	Kasus	Nominativ
	stag-kas-gen		Genitiv
	stag-kas-dat		Dativ
	stag-kas-akk		Akkusativ
num	stag-num-s	Numerus	Singular
	stag-num-p		Plural
gen	stag-gen-m	Genus	Maskulin
	stag-gen-f		Feminin
	stag-gen-n		Neutrum

Tabelle 2: *Die Parameter der Supertag-Haupttypen und ihre Herleitung aus den syntaktischen Eigenschaften*

In Beispiel (1) zeigt Zeile (b) das korrekte Supertag für das Wort *Systemstörung*, wie es im Satz in Zeile (a) vorkommt:

(1) (a) *Ein kleiner Marienkäfer verursachte die Systemstörung.*

(b) stag-np-common:stag-det-yes:stag-scmp-none:stag-kas-akk:stag-num-s:stag-gen-f

2.3 Statistisches Modell

Um von den in der Umgebung eines Wortes gefundenen Informationen auf dessen syntaktischen Eigenschaften schliessen zu können, werden statistische Modelle eingesetzt. Weil sie den Anforderungen des Supertaggings gut entsprechen, fiel die Wahl dabei auf Maximum-Entropy Modelle (siehe zum Beispiel [MS99, p. 589]). Diese sollen im Folgenden kurz beschrieben werden.
Maximum-Entropy-Modelle repräsentieren in erster Linie die im Training erworbenen Daten. Da die Modellparameter hiermit noch nicht eindeutig bestimmt sind, wird als weiteres Kriterium das Modell mit der grössten Entropie ausgewählt. Die Entropie (siehe [MS99, p. 61]) ist ein

Mass für die Unsicherheit einer Wahrscheinlichkeitsverteilung und ist definiert als:

$$E(x) = -\sum_{x} p(x) \log_2 x \tag{2}$$

Je höher die Entropie (die Unsicherheit) einer Verteilung ausfällt, desto uniformer (gleichverteilter) ist sie. Ein Maximum-Entropy-Modell wird also die aus dem Training bekannten Beobachtungen widerspiegeln, während es für aus dem Training nicht bekannte Beobachtungen keine weiteren Annahmen trifft. Beobachtungen werden dem Modell in Form von Features zur Verfügung gestellt. Seien eine Klasse c und eine Beobachtung o gegeben, dann kodieren Features Elemente aus o, welche für die Schätzung von c nützlich sein könnten. Gleichung (3) zeigt die Repräsentation einer diskreten Featurefunktion $f_i(o, c)$:

$$f_i(o, c) = \begin{cases} 1 & \text{falls } extract\_value_i(o) = value_i \text{ und } c = class_i \\ 0 & \text{sonst} \end{cases} \tag{3}$$

Die Klassenwahrscheinlichkeiten können nun mittels Gleichung (4) berechnet werden.

$$p(class|o) = \frac{1}{Z(o)} \exp \sum_{i=1}^{n} \lambda_i f_i(o, class) \tag{4}$$

λ_i: Featuregewichte. Ein grosser Wert steht für ein informatives Feature.
Die Featuregewichte werden beim Training bestimmt.
$Z(o)$: Normalisierungskonstante

Die Randbedingungen des Modells werden mittels Erwartungswerten für die einzelnen Features formuliert. Dabei wird gefordert, dass der Erwartungswert eines jeden Features f_i seiner durchschnittlichen Häufigkeit in den Trainingsdaten entspricht:

$$\underbrace{\sum_{o,c} p(o, c) f_i(o, c)}_{\text{Erwartungswert von } f_i} = \underbrace{\frac{1}{N} \sum_{j=1}^{N} f_i(o_j, c_j)}_{\substack{\text{durchschnittliche Anzahl Vorkommnisse} \\ \text{von } f_i \text{ im Trainingskorpus}}} \tag{5}$$

In der Anwendung auf das Supertagging entsprechen die Klassen einer syntaktischen Eigenschaft den Klassen eines Klassifikationsproblems, während das zu schätzende Wort mit dem es umgebenden Satz die Beobachtung o ausmacht. Anhand des Beispielsatzes (6) soll die Idee der Features für die syntaktische Eigenschaft *Genus* des Substantivs "*Systemstörung*" veranschaulicht werden. Aussagekräftige Features sind hier einerseits das erste Wort zur Linken ("*die*") und andererseits die Endung des Wortes selber ("*ung*"). Beides sind starke Hinweise für das Vorliegen eines femininen Substantivs. Die Featurefunktionen wandeln nun diese Informationen in diskrete Werte um, wodurch eine mathematische Handhabung ermöglicht wird. In einem ausgearbeiteten Fall erreicht die Anzahl Features ohne Weiteres Werte im vierstelligen Bereich. Maximum-Entropy-Modelle können jedoch auch mit sehr grossen Featuremengen umgehen.

(6) *Ein Marienkäfer verursachte die Systemstörung.*

Smoothing

Im Trainingskorpus selten vorkommende Features sind tendenziell unzuverlässig und lassen keine Aussage über die Allgemeinheit zu. Oftmals erhalten sie grosse Featuregewichte und beschreiben damit zu stark die im Trainingskorpus vorgefundenen Verhältnisse. Dem entgegenwirkend könnten seltene Features einfach ignoriert werden, wodurch aber auch wertvolle Information verloren ginge. Im Wissen, dass zu hohe Featuregewichte in der Praxis gar nicht auftreten, kann darauf eine Glättungsfunktion (*Smoothing*) angewendet werden, welche die Gewichte hoch bewerteter Features abschwächt. Das Ausmass dieser Abschwächung kann durch einen sogenannten Regularisierungsparameter bestimmt werden. Das nachträgliche Verändern der Modellparameter durch Smoothing kann auch als Verunschärfung der Modell-Randbedingungen aufgefasst werden.

2.4 Features

Im Abschnitt 2.3 wurde der Begriff der *Features* eingeführt und anhand eines einfachen Beispiels erklärt. Vorliegender Abschnitt zeigt sowohl die verschiedenen Arten von Features, als auch die verschiedenen Arten ihrer Erstellung auf. Eine Auflistung aller Features für die einzelnen syntaktischen Eigenschaften befindet sich in Anhang A.

2.4.1 Generische Features

In der Praxis fallen für jede syntaktische Eigenschaft Tausende von Features an. Die manuelle Bestimmung und Programmierung all dieser Features wäre mit einem immensen Aufwand verbunden. Mit dem Ansatz von *Feature Templates* in Kombination mit Mutual Information (Informationstheorie) lässt sich dieser Vorgang unter Zuhilfenahme eines annotierten Korpus jedoch weitestgehend automatisieren. Allein das Finden geeigneter Feature Templates obliegt dann noch dem Menschen. Ein Feature Template extrahiert gemäss einer fixen Regel einen Kontext aus der Umgebung eines Substantivs w, welcher für die Bestimmung von dessen syntaktischen Eigenschaften relevant sein könnte. Um beispielsweise w's Geschlecht zu eruieren, könnte unter Anderem das Wort links von w oder w's Endung als Feature Template von Nutzen sein. Doch nicht alle der aus den Feature Templates generierten Kontexte bergen gleich starke Hinweise in sich, in vielen Fällen bleiben die Hinweise sogar aus. Die Kontexte mit der stärksten Aussagekraft lassen sich jedoch empirisch bestimmen, indem für eine grosse Menge von Substantiven der Zusammenhang zwischen ihrem Kontext und ihren syntaktischen Eigenschaften untersucht wird. Die Mutual Information $I(X;Y)$ (siehe [MS99, p. 66]), eine Grösse aus der Informationstheorie, stellt ein Mass für genau diesen Zusammenhang dar. Hiernach können pro Feature Template und syntaktische Eigenschaft die stärksten Features empirisch berechnet werden:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \tag{7}$$

Der Supertagger beinhaltet rund 35 Feature Templates, welche Kontexte aus Elementen der folgenden Ebenen erstellen:

Wortebene: Die einzelnen Wörter des Satzes.

Morphologieebene: Zum Beispiel Suffixe verschiedener Länge des zu bestimmenden Wortes.

Wortartebene: Auch Part-Of-Speech-Tag- oder POS-Tag-Ebene. Wortarten der Wörter im Satz, zum Beispiel Adjektiv oder Artikel.

Viele Features lassen sich nur auf POS-Tag-Ebene zufriedenstellend beschreiben. Doch die Wortarten der einzelnen Wörter sind a priori unbekannt und müssen deshalb zuerst in einem Zwischenschritt geschätzt werden. Wir verwendeten dazu den statistischen POS-Tagger *TnT* (siehe [Bra00]). Das Schätzen der Wortart gelingt für Wörter mit nur einer Bedeutung relativ zuverlässig. Als schwieriger erweist sich diese Aufgabe für Wörter mit mehreren Bedeutungen, wie zum Beispiel das Wort "*der*", das sowohl als Artikel (POS-Tag "ART"), als auch als Relativpronomen (POS-Tag "PRELS") auftreten kann.[2] Das Repertoire der POS-Tags ist in Anhang C aufgeführt.

2.4.2 Komplexe Features

Auf Grund der flexiblen Strukturen natürlicher Sprachen ist es oftmals nicht einfach, die auf die syntaktischen Eigenschaften hinweisenden Elemente eines Wortes in seiner jeweiligen Umgebung zu finden. Ganz abgesehen von der Tatsache, dass die meisten Hinweise optional sind, ist häufig auch deren Stellung im Satz äusserst variabel. So muss in Beispiel (8a) nicht weit gesucht werden, um die Eigenschaft *Artikel* für das Wort "*Systemstörung*" bestimmen zu können[3], während dafür in Beispiel (8b) nebst diversen Adjektiven auch noch ein Komma, ein Substantiv und sogar ein zusätzlicher "fremder" Artikel übersprungen werden muss.

(8) (a) *Die Systemstörung.*

(b) *Die den ganzen Tag währende, landesweite Systemstörung.*

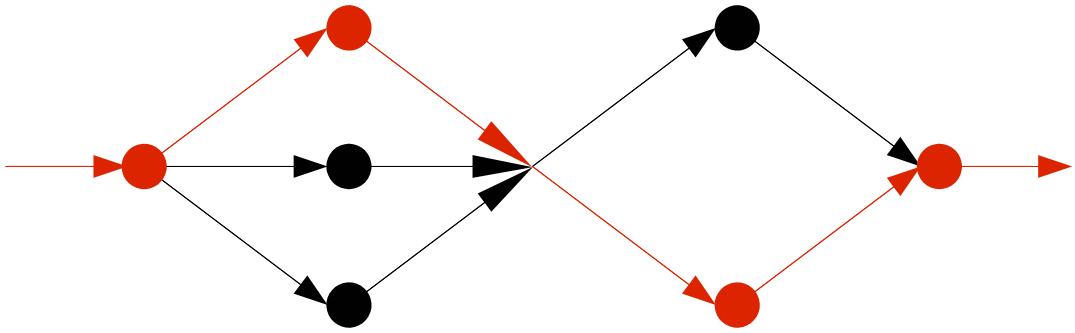
(c) *Die Schüssel, in der Zucker ist.*

Diesem Umstand wird mit *Pattern Features*, einem flexiblen Ansatz, versucht Rechnung zu tragen. Ziel ist es hierbei, ein strukturiertes Regelwerk derart zu durchschreiten, dass alle dabei angetroffenen Regeln erfüllt sind. Figur 2 zeigt einen schematischen Aufbau. Kreise entsprechen den Regeln, ein möglicher kompletter Pfad (Pattern) ist rot markiert. Pro Pattern wird ein Feature generiert. Eine Regel wird immer für eine Stelle im Satz aufgerufen. Falls sie dort erfüllt ist, ermittelt sie die möglichen Startpositionen für die Nachfolgeregel, falls nicht, wird der Pfad aufgegeben. Gestartet wird links bei der ersten Regel mit dem ersten Wort des Satzes. Regeln können äusserst flexibel formuliert werden. Einige für Satzstelle x aufgerufene Beispielregeln sind in Tabelle 3 aufgeführt.

Pattern Features wurden mit einem rekursiven Ansatz implementiert. Bei steigender Regelzahl wird die Performance des Supertaggers rasch durch die Pattern Features dominiert und empfindlich verschlechtert.

[2] Zuweilen erscheint "*der*" auch als Demonstrativpronomen, zum Beispiel in "*Der da!*"

[3] Dass aber auch mit Beispiel (8a) vergleichbare Fälle nicht immer trivial sind, zeigt Beispiel (8c). Hier steht "*Zucker*" ohne Artikel, trotz des vorhergehenden "*der*" (beim vermeintlichen Artikel handelt es sich nämlich um ein Relativpronomen).



Figur 2: *Schematischer Aufbau eines Pattern Features*

Regel	Startposition für Nachfolgeregel
An der Stelle x steht ein Artikel.	$x+1$
Es folgt eine beliebige Anzahl Adjektive, optional durch Kommas getrennt.	$x, x+1, .., x_N$
An der Stelle x steht das zu bestimmende Wort.	$x+1$

Tabelle 3: *Beispiele von Regeln, wie sie in einem Pattern Feature vorkommen.*

2.5 Erstellung der Supertags

In der Standardkonfiguration erstellt der Supertagger genau einen Supertag und setzt darin jede syntaktische Eigenschaft auf jene Klasse, die vom zu Grunde liegenden Modell als die wahrscheinlichste errechnet wurde. Grundsätzlich ist es dem Supertagger jedoch erlaubt, auch mehrere Supertags pro Wort zurück zu geben. Da die statistischen Modelle jede Klasse einer syntaktischen Eigenschaft mit einer Wahrscheinlichkeit bewerten, ist die Grundlage für ein Auswahlverfahren gegeben, das pro syntaktische Eigenschaft auch mehrere Klassen zurückgeben kann. Der im Supertagger verwendete Ansatz selektiert für eine syntaktische Eigenschaft alle Klassen i mit Wahrscheinlichkeit:

$$p_i >= k_s \cdot p_{max}, \qquad 0 \leq k_s \leq 1 \tag{9}$$

p_{max}: Maximalwert aller Klassenwahrscheinlichkeiten der syntaktischen Eigenschaft
k_s: Klassenselektionsschwellwert

Mit der Wahl des Klassenselektionsschwellwerts k_s wird das Verhalten des Supertaggers gesteuert. Für $k_s = 1$ beispielsweise wählt der Ansatz nur die wahrscheinlichste Klasse jeder syntaktischen Eigenschaft, während für $k_s = 0.5$ alle Klassen, die wenigstens halb so wahrscheinlich wie die wahrscheinlichste Klasse ihrer syntaktischen Eigenschaft sind, ausgewählt werden. Mit $k_s = 0$ werden alle Klassen selektiert. Die Menge der Supertags ergibt sich schliesslich aus allen möglichen Kombinationen der selektierten Klassen jeder syntaktischen Eigenschaft.

3 Korpus-Experimente

Um die Funktionalität des Supertaggers zu verifizieren und zu optimieren, wurden in einem ersten Teil etliche Experimente am Supertagger selbst durchgeführt. In dieser Reihe wurden die Resultate mit Gold-Standard Daten aus einem annotierten Korpus verglichen. In einem zweiten Teil wurden dann Qualität und Effizienz des Parsers in Zusammenarbeit mit dem Supertagger untersucht.

3.1 Training

Die Trainings- und Testdaten wurden dem TIGER Korpus (siehe [BDH+02]) entnommen. Dieser enthält 50'000 deutsche Sätze annotiert mit linguistischen Daten wie Syntaxbäumen, Morphologie, Wortart (Part-of-Speech-Tag od. POS-Tag) und syntaktischer Funktion der einzelnen Wörter. Für die Erstellung der Trainingsdaten und Auswertung von Supertagger-Resultaten bedarf es korrekter Angaben zu sämtlichen vom Supertagger behandelten syntaktischen Eigenschaften (Gold-Standard Daten). Während einige dieser Eigenschaften direkt aus dem TIGER Korpus ausgelesen werden konnten (zum Beispiel Kasus, Numerus und Genus), mussten andere zuerst aus den darin vorhandenen Daten aufbereitet werden. Dies gelang meist automatisch durch die Formulierung von Algorithmen und in zwei Fällen manuell nach automatischer Vorselektion. Die Gold-Standard Daten wurden mit wenigen Ausnahmen (unten erläutert) für jedes Substantiv[4] im Korpus bestimmt. Folgende Auflistung dokumentiert die Extraktion der einzelnen syntaktischen Eigenschaften aus dem Korpus:

Kasus, Numerus, Genus

Diese Eigenschaften konnten direkt aus dem Korpus ausgelesen werden. Falls mindestens einer der Werte fehlte, wurde das betreffende Wort vom Training und von sämtlichen Experimenten ausgeschlossen.

Artikel

Diese Eigenschaft wurde per Algorithmus gefunden. Folgende Kriterien müssen erfüllt sein:

- w ist Kopf der w enthaltenden Phrase p.
- In p steht links von w ein Wort mit POS-Tag ART, APPRART, PDAT, PIDAT, PPOSAT, PRELAT oder PWAT.

Titel

Diese Eigenschaft wurde per Algorithmus gefunden. Folgende Kriterien müssen erfüllt sein:

- w ist ein NN und steht im Singular.

[4] Es wurden grundsätzlich alle Wörter mit POS-Tag "NN" oder "NE" betrachtet

- Das Wort rechts von w ist ein NE, es ist kein Genitivattribut.
- In w's Phrase steht links von w kein ADJA, ART, APPRART, PDAT, PIDAT, PPOSAT, PRELAT, PIAT oder PWAT.
- In w's Phrase steht links von w kein Genitivattribut.

Massangabe/-einheit

Diese Eigenschaft wurde per Algorithmus (hier vereinfach dargestellt) gefunden. Folgende Kriterien müssen erfüllt sein:

- w ist ein NN und Kopf der w enthaltenden Phrase p.
- w ist nicht Teil einer Namensphrase (label PN).
- w's Funktion in p ist NK (Element der Kern-NP).
- In w's Phrase steht links von w kein Genitivattribut.
- Die Funktion von w's linkem Nachbarn in p ist NK.
- Die Funktion von w's rechtem Nachbarn in p ist NK.
- w's rechter Nachbar ist entweder ein NN oder eine (konjugierte) Namenphrase und deren Kopf ist ein NN.

Adverbiale Zeitangabe

Manuelle Auswahl nach vorselektierendem Algorithmus.

Eigenname

Diese Eigenschaft wurde per Algorithmus gefunden. Folgende Kriterien müssen erfüllt sein:

- w ist ein NE und w's syntaktische Eigenschaft *Artikel* trifft nicht zu.

Satzkomplement

Die vier unten folgenden Satzkomplementtypen müssen zusätzlich diesen Kriterien genügen:

- w ist ein NN.
- w's syntaktische Funktion ist NK.
- In der w enthaltenden Phrase gibt es rechts von w eine Phrase $\hat{p}$ mit syntaktischer Funktion OC (klausales Objekt).

Komplementsatz

Diese Eigenschaft wurde per Algorithmus gefunden. Folgende Kriterien müssen erfüllt sein:

- Bedingungen von Eigenschaft *Satzkomplement*.
- Erstes Wort in $\hat{p}$ ist "*dass*" oder "*daß*".

Interrogativsatz

Diese Eigenschaft wurde per Algorithmus (hier vereinfacht formuliert) gefunden. Folgende Kriterien müssen erfüllt sein:

- Bedingungen von Eigenschaft *Satzkomplement*.
- Erstes Wort in $\hat{p}$ ist "*ob*" oder ein Interrogativpronomen.

Infinitivverbalphrase

Diese Eigenschaft wurde per Algorithmus gefunden. Folgende Kriterien müssen erfüllt sein:

- Bedingungen von Eigenschaft *Satzkomplement*.
- $\hat{p}$ ist (konjugierte) Verbalphrase oder (konjugierter) Verbzusatz.

Hauptsatz

Diese Eigenschaft wurde per Algorithmus gefunden. Folgende Kriterien müssen erfüllt sein:

- Bedingungen von Eigenschaft *Satzkomplement*.
- Aus denjenigen Satzkomplementen, die sich keinem der drei anderen Typen zuteilen liessen, wurden die Hauptsätze manuell ausgelesen.

Die Aufbereitung der Gold-Standard Daten war nicht trivial und nahm einen beträchtlichen Teil der zur Verfügung stehenden Zeit in Anspruch. Dennoch ist die Qualität der gewonnenen Daten nicht immer über alle Zweifel erhaben. Oftmals stellen die Algorithmen nur Näherungen dar, wodurch auch falsch positive und falsch negative Ergebnisse unter die richtigen gelangten. Fehler im Korpus und Inkonsistenzen in der Notation sorgten für zusätzliche Verunreinigungen.

Die POS-Tags aus dem TIGER Korpus wurden ausschliesslich für die Gewinnung der Gold-Standard Daten verwendet. Für die Erstellung von Trainingsdaten und für Evaluationen wurden in Anlehnung an die Praxis stattdessen die durch TnT geschätzten POS-Tags verwendet. TnT wurde dazu per 10-fach-Kreuzvalidierung[5]auf den TIGER Daten selber trainiert.

[5]Bei der 10-fach-Kreuzvalidierung wird der Korpus in 10 Teile geteilt und jeder Zehntel einzeln anhand eines mit den neun anderen Zehnteln trainierten Modells geschätzt. Danach werden die 10 geschätzten Zehntel wieder zu einem Ganzen zusammengefügt.

Vom Training und von sämtlichen Experimenten ausgeschlossen wurden Substantive, die mindestens eines der folgenden Kriterien erfüllten:

- Das Wort ist in der Wörter-Stopliste enthalten. Gewisse Wörter (zum Beispiel "*Mark*") wurden explizit ausgeschlossen, wenn sie in gewissen Algorithmen störend wirkten und wenn sie ihrer Häufigkeit wegen ohnehin im Parser-Lexikon enthalten sind.
- Das Wort ist Teil eines Satzes auf der Sätze-Stopliste. Ein-Wort-Sätze sowie stark falsch notierte Sätze wurden explizit ausgeschlossen.
- Kasus, Numerus oder Genus des Wortes sind nicht vollständig deklariert.
- Das Wort ist Teil eines mehrere Substantive enthaltenden Eigennamens und bei mindestens einem dieser Substantive fehlen Angaben zu Kasus oder Numerus.

Die verbleibenden gültigen Daten (knapp 230'000 Tokens) wurden in ein Trainings-Set (70%, gut 160'000 Tokens), ein Development-Set (10%, ca 22'500 Tokens) und ein Test-Set (20%, knapp 46'000 Tokens) aufgeteilt. Für das Training der Maximum-Entropy Modelle wurde *Maxent-Optimizer* (implementiert von Thomas Ewender, basierend auf der Discriminative-Reranking-Software von [CJ05]) mit Smoothing-Konstante $c = 1$ beigezogen[6]. Während der Entwicklungszeit wurden die Modelle am Development-Set ausgewertet und durch Optimierung der Features und der Gold-Standard Daten verbessert. Erst die abschliessenden Experimente wurden auf dem Test-Set durchgeführt, welches soweit ungenutzt geblieben war. Mit dieser Massnahme wurde verhindert, dass die Optimierungen der Modelle nur auf beste Ergebnisse mit den Testdaten hinzielten.

3.2 Evaluation

Die Treffgenauigkeit des Supertaggers wurde auf vielschichtige Art und Weise untersucht. Folgende Auswertungen wurden durchgeführt:

- Treffgenauigkeit für die einzelnen syntaktischen Eigenschaften sowie für alle Eigenschaften zusammen
- Treffgenauigkeit bei Verwendung verschiedener Feature Templates Gruppen
- Treffgenauigkeit bei Verwendung einzelner Feature Templates
- Treffgenauigkeit bei Variation der Anzahl Features pro Feature Template
- Treffgenauigkeit für auf seltene Wörter beschränkte Training- und/oder Test-Sets
- Treffgenauigkeit für verschiedene Klassenselektionsschwellwerte k_s

Der Rechenaufwand des Supertaggers ist im Vergleich zum Rechenaufwand des Parsers vernachlässigbar klein und wurde darum nicht genauer betrachtet.

[6] Bei höheren Werten, d.h. stärkerem Smoothing, wurden schlechtere Resultate erzielt.

Die Bedeutung der Feature Template Bezeichnungen ist in Anhang A beschrieben. Die Zugehörigkeit der einzelnen Templates zu den Feature Templates Gruppen ist in Anhang B definiert. Wenn nicht anders erwähnt, wurde die Anzahl Features pro Feature Template auf 100 festgelegt.

Im Folgenden werden die Auswertungen sortiert nach syntaktischer Eigenschaft aufgeführt. Weiter unten im Abschnitt folgen Untersuchungen, welche die Gesamtheit aller syntaktischen Eigenschaften betreffen.

Syntaktische Eigenschaft ”Kasus”

Trainings-Set			Test-Set		
Klasse	Abs. Häufigkeit	Rel. Häufigkeit	Klasse	Abs. Häufigkeit	Rel. Häufigkeit
Nominativ	58121	36.2%	Nominativ	16508	35.9%
Dativ	47255	29.4%	Dativ	13645	29.7%
Akkusativ	34332	21.4%	Akkusativ	10002	21.8%
Genitiv	18333	11.4%	Genitiv	4465	9.7%
n/a	2681	1.7%	n/a	1302	2.8%
Korrekte Vorhersagen				38617	86.5%
Mehrheitsklasse (Baseline)				16508	37.0%

Feature Template Gruppe	Treffgenauigkeit
Alle	86.5%
Kombinierte	85.5%
POS-Tagkontext-basierte	71.9%
Wortkontext-basierte	68.1%
Morphologie-basierte	48.0%

Feature Template	Treffgenauigkeit
Tm1Em13	63.8%
Tm1Em12	63.5%
Tm2Tm1	63.5%
Em13	62.0%
Wm1	60.4%
Em12	58.0%
Tm1T1	50.9%
Wm2Em12	50.8%
Tm2	50.0%
Tm1	49.9%

Syntaktische Eigenschaft ”Numerus”

Trainings-Set			Test-Set		
Klasse	Abs. Häufigkeit	Rel. Häufigkeit	Klasse	Abs. Häufigkeit	Rel. Häufigkeit
Singular	120191	74.8%	Singular	33010	71.9%
Plural	38358	23.9%	Plural	11719	25.5%
n/a	2173	1.4%	n/a	1193	2.6%
Korrekte Vorhersagen				42710	95.5%
Mehrheitsklasse (Baseline)				33010	73.8%

Feature Template Gruppe	Treffgenauigkeit
Alle	95.5%
Kombinierte	95.4%
Morphologie-basierte	89.5%
Wortkontext-basierte	82.5%
POS-Tagkontext-basierte	79.2%

Feature Template	Treffgenauigkeit
E03	88.3%
Tm1E02	88.0%
E04	87.3%
E02	87.2%
Tm1E03	85.6%
E05	84.9%
Wm1E02	82.5%
Em13E03	80.3%
W0	79.3%
Wm1E03	78.9%

Syntaktische Eigenschaft ”Genus”

Trainings-Set			Test-Set		
Klasse	Abs. Häufigkeit	Rel. Häufigkeit	Klasse	Abs. Häufigkeit	Rel. Häufigkeit
Maskulin	58911	36.7%	Maskulin	16191	35.3%
Feminin	57885	36.0%	Feminin	16120	35.1%
Neutrum	35170	21.9%	Neutrum	10735	23.4%
n/a	8756	5.5%	n/a	2876	6.3%
Korrekte Vorhersagen				35885	83.4%
Mehrheitsklasse (Baseline)				16191	37.6%

Feature Template Gruppe	Treffgenauigkeit
Alle	83.4%
Kombinierte	83.3%
Morphologie-basierte	72.4%
Wortkontext-basierte	64.2%
POS-Tagkontext-basierte	52.2%

Feature Template	Treffgenauigkeit
E03	63.8%
E02	63.4%
E04	62.2%
E05	58.4%
Tm1E02	56.9%
Tm1Em13	56.5%
Tm1E03	56.4%
Tm1Em12	56.3%
Em13	56.0%
Em12	54.8%

Bemerkungen:

- Die Unterscheidung zwischen Substantiven des Geschlechts maskulin und neutrum ist in der deutschen Sprache teilweise schwierig.
- Substantive im Plural bieten ebenfalls nur schwache geschlechtsspezifische Merkmale. Da jedoch für den Parser das Geschlecht von Substantiven im Plural gar nicht relevant ist, wurde untersucht, ob eine Verschmelzung der beiden syntaktischen Eigenschaften *Numerus* und *Genus* deren Treffgenauigkeit verbessern würde. Eine neue syntaktische Eigenschaft *GenusNumerus* mit den Klassen *Maskulin*, *Feminin*, *Neutrum* und *Plural* wurde erstellt. Wörter im Plural wurden ungeachtet ihres Geschlechts der Klasse *Plural* zugeteilt, während Wörter im Singular eine der drei Geschlechterklassen einnahmen. In den Auswertungen schnitt die Eigenschaft *GenusNumerus* dann jedoch leicht schlechter ab als die beiden Eigenschaften separat.

Syntaktische Eigenschaft "Artikel"

Trainings-Set			Test-Set		
Klasse	Abs. Häufigkeit	Rel. Häufigkeit	Klasse	Abs. Häufigkeit	Rel. Häufigkeit
Nein	75524	47.0%	Nein	21953	47.8%
Ja	85198	53.0%	Ja	23969	52.2%
Korrekte Vorhersagen				44595	97.1%
Mehrheitsklasse (Baseline)				23969	52.2%

Feature Template Gruppe	Treffgenauigkeit
Alle	97.1%
Kombinierte	96.9%
POS-Tagkontext-basierte	96.3%
Wortkontext-basierte	84.8%
Morphologie-basierte	63.9%

Feature Template	Treffgenauigkeit
Tm2Tm1	91.7%
Tm1	90.1%
Tm1Em12	89.1%
Tm1Em13	87.8%
Em13	84.4%
Wm1	83.1%
Tm1T1	82.0%
Em12	81.3%
Tm1E02	74.2%
T0	66.1%

Syntaktische Eigenschaft ”Titel”

Trainings-Set			Test-Set		
Klasse	Abs. Häufigkeit	Rel. Häufigkeit	Klasse	Abs. Häufigkeit	Rel. Häufigkeit
Nein	159266	99.1%	Nein	45563	99.2%
Ja	1456	0.9%	Ja	359	0.8%
Korrekte Vorhersagen				45802	99.9%
Mehrheitsklasse (Baseline)				45563	99.2%

Feature Template Gruppe	Treffgenauigkeit
Alle	99.9%
Kombinierte	99.9%
POS-Tagkontext-basierte	99.8%
Wortkontext-basierte	99.4%
Morphologie-basierte	99.2%

Feature Template	Treffgenauigkeit
W1	99.3%
W2	99.3%
Tm1E02	99.3%
W1W2	99.3%
Wm1E04	99.3%
Wm1E02	99.3%
Wm1E03	99.3%
Em13E03	99.3%
Tm1E03	99.3%
T1T2	99.2%

Syntaktische Eigenschaft ”Massangabe/-einheit”

Trainings-Set			Test-Set		
Klasse	Abs. Häufigkeit	Rel. Häufigkeit	Klasse	Abs. Häufigkeit	Rel. Häufigkeit
Nein	160214	99.7%	Nein	45778	99.7%
Ja	508	0.3%	Ja	144	0.3%
Korrekte Vorhersagen				45825	99.8%
Mehrheitsklasse (Baseline)				45778	99.7%

Feature Template Gruppe	Treffgenauigkeit
Alle	99.8%
Kombinierte	99.8%
POS-Tagkontext-basierte	99.7%
Wortkontext-basierte	99.7%
Morphologie-basierte	99.7%

Feature Template	Treffgenauigkeit
T1T2	99.7%
W1	99.7%
W2	99.7%
W0	99.7%
Tm1Em12	99.7%
Tm1Em13	99.7%
E02	99.7%
E05	99.7%
Tm2Tm1	99.7%
E04	99.7%

Bemerkungen:

- Auf Grund des seltenen Auftretens von Masseinheiten im Trainings-Set war kein solides Training möglich.
- Ohne semantische Informationen ist die Erkennung von Masseinheiten schwierig.
- Die Qualität der Gold-Standard Daten ist unzureichend.
- Mit dem Einsatz komplexer Features könnte das Ergebnis womöglich verbessert werden.

Syntaktische Eigenschaft ”Adverbiale Zeitangabe”

Trainings-Set			Test-Set		
Klasse	Abs. Häufigkeit	Rel. Häufigkeit	Klasse	Abs. Häufigkeit	Rel. Häufigkeit
Nein	160525	99.9%	Nein	45856	99.9%
Ja	197	0.1%	Ja	66	0.1%
Korrekte Vorhersagen				45871	99.1%
Mehrheitsklasse (Baseline)				45856	99.9%

Feature Template Gruppe	Treffgenauigkeit
Alle	99.9%
Kombinierte	99.9%
Wortkontext-basierte	99.9%
POS-Tagkontext-basierte	99.9%
Morphologie-basierte	99.9%

Feature Template	Treffgenauigkeit
T1T2	99.9%
W1	99.9%
W2	99.9%
W0	99.9%
Tm1Em12	99.9%
Tm1Em13	99.9%
E02	99.9%
E05	99.9%
Tm2Tm1	99.9%
E04	99.9%

Bemerkungen:

- Auf Grund des seltenen Auftretens adverbialer Zeitangaben im Trainings-Set war kein solides Training möglich.
- Adverbiale Zeitangaben bieten dem Supertagger kaum Erkennungsmöglichkeiten.
- Mit dem Einsatz komplexer Features könnte das Ergebnis womöglich verbessert werden.

Syntaktische Eigenschaft ”Eigenname”

Trainings-Set			Test-Set		
Klasse	Abs. Häufigkeit	Rel. Häufigkeit	Klasse	Abs. Häufigkeit	Rel. Häufigkeit
Nein	132043	82.2%	Nein	37998	82.7%
Ja	28679	17.8%	Ja	7924	17.3%
Korrekte Vorhersagen				44854	97.7%
Mehrheitsklasse (Baseline)				37998	82.7%

Feature Template Gruppe	Treffgenauigkeit
Alle	97.7%
Kombinierte	97.7%
POS-Tagkontext-basierte	97.5%
Wortkontext-basierte	88.8%
Morphologie-basierte	88.5%

Feature Template	Treffgenauigkeit
T0	95.7%
Tm2Tm1	88.7%
Tm1Em12	88.0%
Tm1T1	87.9%
Tm1Em13	87.3%
Tm1E02	86.8%
Em13	86.7%
Wm1	86.3%
E04	86.2%
W0	86.1%

Syntaktische Eigenschaft "Satzkomplement"

	Trainings-Set		Test-Set	
Klasse	Abs. Häufigkeit	Rel. Häufigkeit	Abs. Häufigkeit	Rel. Häufigkeit
Kein	159564	99.3%	45666	99.4%
Infinitivverbalphrase	640	0.4%	140	0.3%
Komplementsatz	263	0.2%	56	0.1%
Hauptsatz	144	0.1%	37	0.1%
Interrogativsatz	111	0.1%	23	0.1%
Korrekte Vorhersagen			45626	99.4%
Mehrheitsklasse (Baseline)			45666	99.4%

Feature Template Gruppe	Treffgenauigkeit
Wortkontext-basierte	99.5%
Morphologie-basierte	99.4%
Alle	99.4%
Kombinierte	99.4%
POS-Tagkontext-basierte	99.3%

Feature Template	Treffgenauigkeit
T1T2	99.4%
W1	99.4%
W2	99.4%
W0	99.4%
Tm1Em12	99.4%
Tm1Em13	99.4%
E02	99.4%
E05	99.4%
Tm2Tm1	99.4%
E04	99.4%

Bemerkungen:

- Auf Grund des seltenen Auftretens von Satzkomplementen im Trainings-Set war kein solides Training möglich.
- Ohne semantische Informationen ist die Erkennung von Satzkomplementen schwierig. So entscheidet in Beispiel (10) nur die Bedeutung des zu untersuchenden Wortes, ob der Komplementsatz am Substantiv "*Tatsache*" oder am Verb "*schreibt*" hängt. Ansonsten sind die Sätze identisch.
- Die Qualität der Gold-Standard Daten ist nicht immer ausreichend hoch.
- Mit dem Einsatz komplexer Features könnte das Ergebnis womöglich verbessert werden.

(10) *Er schreibt in seinem Buch über die Republikaner, dass der Umgang mit den Rechtsextremen einen Gradmesser für die Reife einer Gesellschaft abgibt.*

Er schreibt in seinem Buch über die Tatsache, dass der Umgang mit den Rechtsextremen einen Gradmesser für die Reife einer Gesellschaft abgibt.

Syntaktische Eigenschaft "Komplementsatz"

Trainings-Set			Test-Set		
Klasse	Abs. Häufigkeit	Rel. Häufigkeit	Klasse	Abs. Häufigkeit	Rel. Häufigkeit
Nein	160459	99.8%	Nein	45866	99.9%
Ja	263	0.2%	Ja	56	0.1%
Korrekte Vorhersagen				45891	99.9%
Mehrheitsklasse (Baseline)				45866	99.9%

Feature Template Gruppe	Treffgenauigkeit
Alle	99.9%
Kombinierte	99.9%
POS-Tagkontext-basierte	99.9%
Wortkontext-basierte	99.9%
Morphologie-basierte	99.9%

Feature Template	Treffgenauigkeit
T1T2	99.9%
W1	99.9%
W2	99.9%
W0	99.9%
Tm1Em12	99.9%
Tm1Em13	99.9%
E02	99.9%
E05	99.9%
Tm2Tm1	99.9%
E04	99.9%

Bemerkungen: Siehe unter syntaktischer Eigenschaft *Satzkomplement*

Syntaktische Eigenschaft ”Infinitivverbalphrase”

Trainings-Set			Test-Set		
Klasse	Abs. Häufigkeit	Rel. Häufigkeit	Klasse	Abs. Häufigkeit	Rel. Häufigkeit
Nein	160082	99.6%	Nein	45782	99.7%
Ja	640	0.4%	Ja	140	0.3%
Korrekte Vorhersagen				45802	99.7%
Mehrheitsklasse (Baseline)				45782	99.7%

Feature Template Gruppe	Treffgenauigkeit
Alle	99.7%
Kombinierte	99.7%
Wortkontext-basierte	99.7%
POS-Tagkontext-basierte	99.7%
Morphologie-basierte	99.7%

Feature Template	Treffgenauigkeit
T1T2	99.7%
W1	99.7%
W2	99.7%
W0	99.7%
Tm1Em12	99.7%
Tm1Em13	99.7%
E02	99.7%
E05	99.7%
Tm2Tm1	99.7%
E04	99.7%

Bemerkungen: Siehe unter syntaktischer Eigenschaft *Satzkomplement*

Syntaktische Eigenschaft ”Hauptsatz”

Trainings-Set			Test-Set		
Klasse	Abs. Häufigkeit	Rel. Häufigkeit	Klasse	Abs. Häufigkeit	Rel. Häufigkeit
Nein	160578	99.9%	Nein	45885	99.9%
Ja	144	0.1%	Ja	37	0.1%
Korrekte Vorhersagen				45887	99.9%
Mehrheitsklasse (Baseline)				45885	99.9%

Feature Template Gruppe	Treffgenauigkeit
Alle	99.9%
Kombinierte	99.9%
POS-Tagkontext-basierte	99.9%
Wortkontext-basierte	99.9%
Morphologie-basierte	99.9%

Feature Template	Treffgenauigkeit
T1T2	99.9%
W1	99.9%
W2	99.9%
W0	99.9%
Tm1Em12	99.9%
Tm1Em13	99.9%
E02	99.9%
E05	99.9%
Tm2Tm1	99.9%
E04	99.9%

Bemerkungen: Siehe unter syntaktischer Eigenschaft *Satzkomplement*

Syntaktische Eigenschaft ”Interrogativsatz”

Trainings-Set			Test-Set		
Klasse	Abs. Häufigkeit	Rel. Häufigkeit	Klasse	Abs. Häufigkeit	Rel. Häufigkeit
Nein	160611	99.9%	Nein	45899	99.9%
Ja	111	0.1%	Ja	23	0.1%
Korrekte Vorhersagen				45905	100.0%
Mehrheitsklasse (Baseline)				45899	99.9%

Feature Template Gruppe	Treffgenauigkeit
Alle	100.0%
Kombinierte	100.0%
POS-Tagkontext-basierte	99.9%
Wortkontext-basierte	99.9%
Morphologie-basierte	99.9%

Feature Template	Treffgenauigkeit
T1T2	99.9%
W1	99.9%
W2	99.9%
W0	99.9%
Tm1Em12	99.9%
Tm1Em13	99.9%
E02	99.9%
E05	99.9%
Tm2Tm1	99.9%
E04	99.9%

Bemerkungen: Siehe unter syntaktischer Eigenschaft *Satzkomplement*

Gesamttreffgenauigkeit

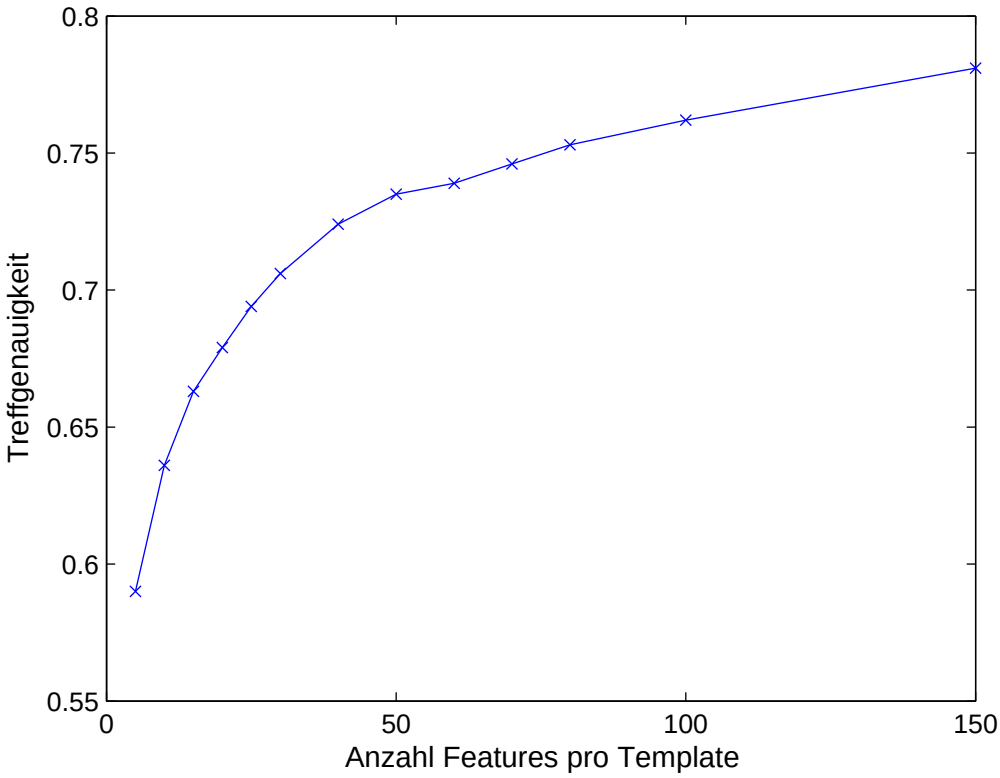
Nur Wörter, für die sämtliche syntaktischen Eigenschaften korrekt geschätzt wurden, wurden als Treffer gewertet.

Feature Template Gruppe	Treffgenauigkeit
Alle	76.2%
Kombinierte	75.2%
Wortkontext-basierte	39.8%
POS-Tagkontext-basierte	37.4%
Morphologie-basierte	23.8%

Feature Template	Treffgenauigkeit
Tm1Em13	30.1%
Tm1Em12	29.7%
Wm1	28.1%
Em13	26.7%
Tm2Tm1	25.2%
Em12	22.6%
Tm1E02	19.9%
E02	19.4%
Tm1T1	19.2%
E03	18.3%

Gesamttreffgenauigkeit bei Variation der Anzahl Features pro Feature Template

Als Standardwert für die meisten Experimente wurden 100 Features pro Template gewählt, obwohl die Treffgenauigkeit für höhere Werte weiter im Steigen begriffen war. Ein Wert > 100 wäre also womöglich angebracht gewesen, auch wenn das statistische Modell mit höheren Featurezahlen tendenziell stärker auf die Trainingsdaten hin trainiert wird.



Gesamttreffgenauigkeit für auf seltene Wörter beschränkte Training- und/oder Test-Sets

In der Annahme, den Supertagger mit möglichst umfangreichen Trainings-Sets auf höchste Genauigkeit trimmen zu können, wurden die statistischen Modelle stets auf der maximal zur Verfügung stehenden Datenmenge trainiert. Da der Supertagger in der Praxis jedoch nur seltene Wörter klassifizieren muss, liegt es eigentlich nahe, ihn auch nur mit seltenen Wörtern zu trainieren. Um diesen Widerspruch zu untersuchen, wurden sowohl Trainings- als auch Test-Set auf 30% ihrer Grösse reduziert, so dass sie nur noch die seltensten Wörter[7] enthielten.

Die folgende Tabelle zeigt die Gesamttreffgenauigkeit des Supertaggers für verschiedene Kombinationen dieser Sets:

[7] Die Worthäufigkeiten wurden einem Index entnommen, welcher auf Artikeln der Frankfurter Rundschau erschienen zwischen dem 1. Januar 1997 und dem 1. Januar 2002 basiert.

Kombination	Gesamttreffgenauigkeit
Trainings- und Test-Set vollständig	76.2%
Trainings- und Test-Set reduziert	72.7%
Nur Test-Set reduziert	69.6%

Demzufolge liefert ein mit nicht reduzierten Daten trainiertes Modell tatsächlich die höchste Genauigkeit, solange es auch auf nicht reduzierte Testdaten angewendet wird. Ist hingegen nur das Test-Set reduziert, fallen die Resulate schlechter aus, als wenn beide Sets reduziert sind. Leider war dieser Umstand zu Beginn der zeitaufwändigen Parsing-Experimente noch nicht bekannt, so dass diese auf Basis des ungünstigen Falls (Trainings-Set vollständig, Test-Set reduziert) durchgeführt wurden.

4 Parsing-Experimente

4.1 Aufbau

In diesem Abschnitt werden Qualität und Effizienz des Parsers in Zusammenarbeit mit dem Supertagger untersucht. Die Experimente wurden auf einem Head-driven Phrase Structure Grammar Parser (siehe [KP07] und [KP08]) durchgeführt. Als Testdaten wurden 1000 Sätze aus Transkriptionen von sechs Ausgaben der Deutschen Tagesschau[8] verwendet (aus Spracherkennungs-Experimenten in [MAD03]).
Der Parser findet pro Satz üblicherweise mehrere mögliche Lösungsbäume. Unter diesen wählt er mit Hilfe eines statistischen Modells einen definitiven Syntaxbaum (Disambiguierung). In den folgenden Experimenten basieren alle Werte, falls nicht anders angegeben, auf den Auswertungen dieser vom statistischen Modell gefundenen Bäume. Wird das Modell hingegen als ideal betrachtet, so dass es unter den möglichen Lösungsbäumen immer den besten zurückliefert, ist von *oracle*-Daten die Rede.
Die Performance des Parsers wird in *unification operations* gemessen. Dieses Mass ist in guter Näherung proportional zum Rechenaufwand. Die Qualität wird üblicherweise durch die Masse Precision, Recall und F-Score beschrieben (siehe auch [AFG+91]). Precision P und Recall R sind Begriffe aus dem Gebiet des Information Retrievals und werden in den Gleichungen (11) und (12) beschrieben. Der F-Score F berechnet sich sodann als harmonisches Mittel von Precision und Recall.

$$P = \frac{K}{N} \tag{11}$$

$$R = \frac{K}{G} \tag{12}$$

$$F = \frac{2RP}{R+P} \tag{13}$$

[8] Es handelt sich dabei um die 20-Uhr-Ausgaben vom 14. April, 21. April, 29. April, 7. Juni, 15. Juni und 23. Juni 2002.

K: Anzahl korrekte Phrasen in Kandidat
N: Anzahl Phrasen in Kandidat
G: Anzahl Phrasen in Gold-Standard Referenz

Trotz dieser einfachen Formeln ist die Bestimmung der Parserqualität nicht unproblematisch. Weil ein Syntaxbaum Informationen aus verschiedenen Ebenen beinhaltet, ist der Begriff seiner Korrektheit nicht ohne Weiteres klar definiert. So müssen beim Vergleich zweier Syntaxbäume nebst den Baumkanten und Konstituenten unter Anderem auch die POS-Tags und syntaktischen Funktionen der Wörter verglichen werden. Dieser Vergleich wird dadurch erschwert, dass Grammatiken verschiedene Notationen eines Baumes beziehungsweise seiner Konstituenten zulassen (”flache”, ”tiefe”, etc). Zusätzlich müssen sowohl der Referenzbaum als auch der Antwortbaum zuerst in ein Zwischenformat konvertiert werden, damit ein Vergleich überhaupt angestellt werden kann. Hierbei entstehen Fehler, die das Ergebnis der Auswertung weiter zu Unrecht verschlechtern. Die beschreibenden Grössen der Parserqualität sind darum als Absolutwerte nicht zwingend aufschlussreich. Deshalb wurden zusätzliche Experimente durchgeführt, aus welchen Referenzwerte resultierten, die qualitative Schlüsse zulassen. Die Setups dieser Experimente seien hier kurz beschrieben:

no-unknown-words: Sämtliche in den Testsätzen vorkommenden Wörter sind im Parser-Lexikon vorhanden. Es werden keine Supertags benötigt. Die diesem Setup entstammenden Werte messen direkt die Eigenschaften des Parsers und enthalten keinerlei Einflüsse des Supertaggers.

default-supertag: Unbekannte Wörter erhalten teilweise unterspezifizierte Einträge. Es wird angenommen, dass ein unbekanntes Wort ein Common Noun ohne Satzkomplement und mit optionalem Artikel ist. Kongruenzmerkmale (Kasus, Numerus und Genus) werden nicht spezifiziert.

no-complete-parse: Unbekannte Wörter können nicht in einen Satz eingebaut werden. Das beste, was der Parser tun kann, ist in den Lücken zwischen den unbekannten Wörtern korrekte Phrasen zu erkennen.

4.2 Evaluation

Die Figuren 3 und 4 zeigen das Verhalten des Parsers, wenn 20% der Substantive[9] in den Testsätzen als unbekannt markiert und durch den Supertagger mit Supertags versehen wurden. Hierin lässt sich Folgendes erkennen:

- Für alle Klassenselektionsschwellwerte[10] ist der Parser mit Supertagger (*f-score*) deutlich präziser als ohne Angabe von Supertags (*no-complete-parse*). Auch im Vergleich mit der Variante *default-supertag* schneidet der Supertagger durchwegs besser ab, wenn auch für hohe k_s nicht mehr so deutlich. Der Rechenaufwand ist bei *default-supertag* jedoch höher.
- Für kleine k_s nähert sich der Supertagger der Variante *no-unknown-words*, bleibt aber stets unterhalb. Dies bedeutet, dass das Parser-Lexikon den Sprachgebrauch soweit gut

[9] Dabei fiel die Wahl auf die am seltensten vorkommenden Substantive. Siehe auch bei den Untersuchungen zu reduzierten Trainings- und Test-Sets in Abschnitt 3.2

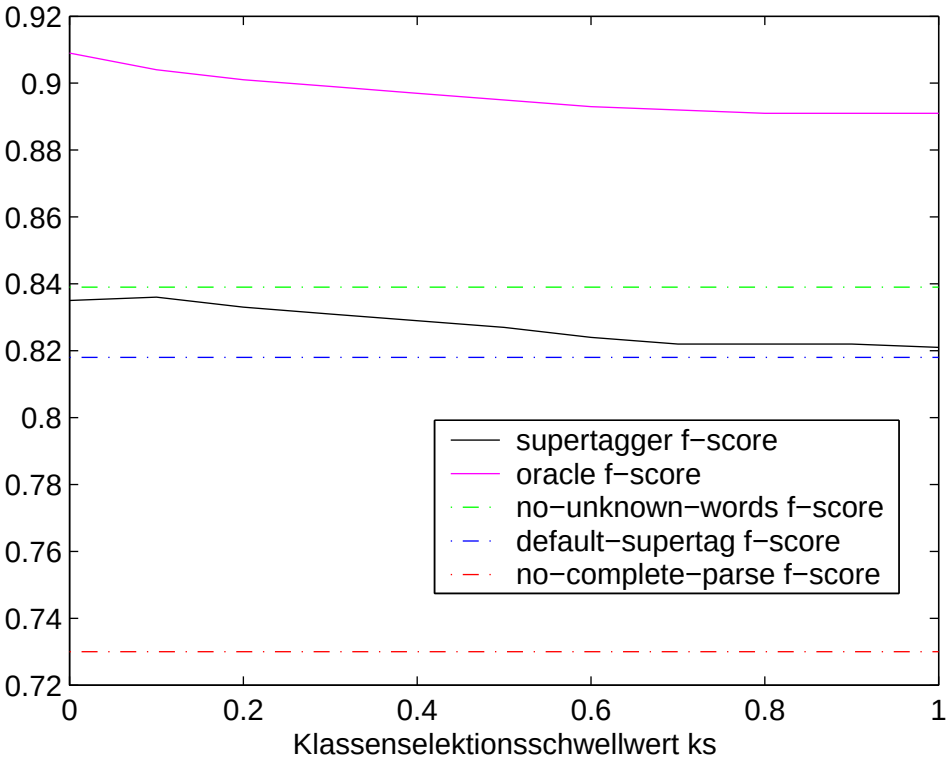
[10] Die Bedeutung des Klassenselektionsschwellwerts k_s ist in Abschnitt 2.5 erklärt.

abdeckt. Dies ist nicht selbstverständlich, denn das Lexikon wird in Handarbeit erstellt. Es kann fehlerhaft sein und wird nicht immer alle möglichen Anwendungsformen eines Wortes enthalten. Es ist deshalb auch durchaus denkbar, dass die Angaben des Supertaggers diejenigen im Lexikon übertreffen, weil der Supertagger den Kontext des Wortes analysiert und so dessen Anwendungsform von Fall zu Fall neu einschätzen kann. Ohne Einsatz des Supertaggers wird der Parser jedoch sämtliche Lexikoneinträge zu einem Wort in die Suche miteinbeziehen müssen, weshalb die Performance von *no-unknown-words* auch grundsätzlich schlechter ausfällt.

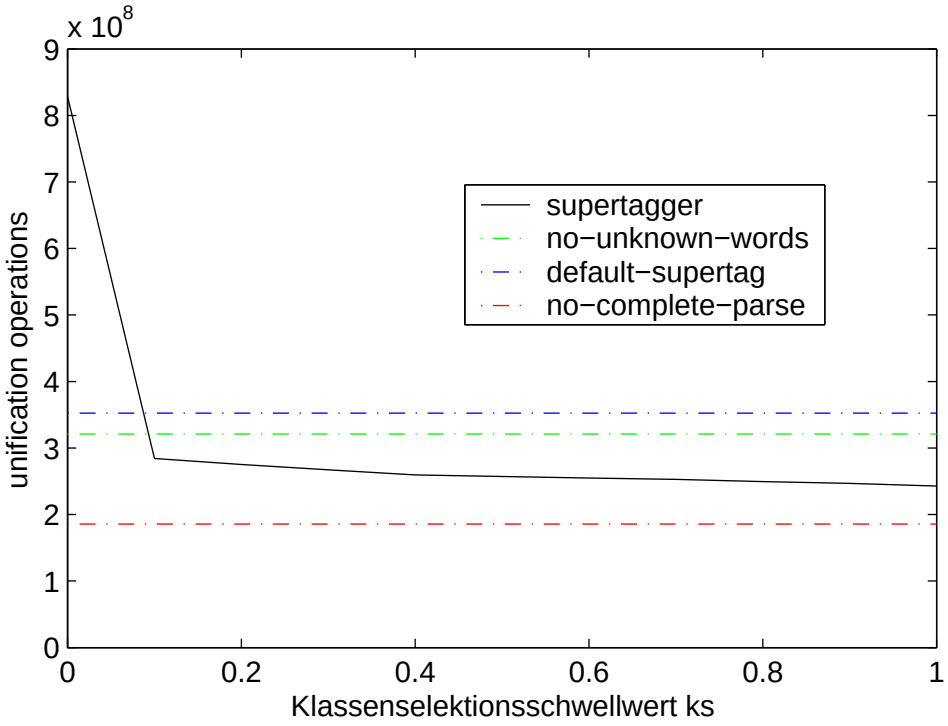
- Während die Präzision und der Rechenaufwand des Parser für $k_s = 1$ bis $k_s = 0.1$ linear ansteigen, verdient der Fall $k_s = 0$ spezielle Beachtung. Hier generiert der Supertagger ungeachtet der Schätzungen seiner statistischen Modelle stets sämtliche möglichen Supertags. Daher steigt gemäss Figur 5 die Anzahl Supertags pro unbekanntem Wort bei sonst moderatem Wachstum beim Übergang von $k_s = 0.1$ nach $k_s = 0$ um beinahe das 30-fache.[11] Die damit verbundene Vergrösserung des Parser-Suchraumes ist erheblich und die Fehlerquote des für die Disambiguierung verantwortlichen statistischen Modells steigt empfindlich. Dies lässt sich an einem leichten Knick im ansonsten gegen $k_s = 0$ hin monoton ansteigenden Supertagger F-Score erkennen, während der *oracle* F-Score weiter ansteigt.

In Figur 6 wurde der F-Score bei verschiedener Anzahl unbekannter Wörter aufgezeichnet. Es ist kaum überraschend, dass die Präzision des Parsers sinkt, je mehr unbekannte Wörter vorhanden sind. Weniger selbstverständlich ist jedoch, dass der F-Score gegen kleine Klassenselektionsschwellwerte k_s hin ansteigt. Zwar generiert der Supertagger für kleinere k_s mehr Supertags, wodurch sich auch die Wahrscheinlichkeit erhöht, dass sich darunter der korrekte Supertag befindet. Doch erhöht sich gleichzeitig auch der Suchraum und damit die Lösungsmenge des Parsers, weshalb die korrekte Lösung tendenziell in der Masse unterzugehen droht. Dass die Präzision des Parsers dennoch steigt, ist deshalb ein Hinweis für das gute Funktionieren von dessen Disambiguierungseinheit. Leider fehlen in Figur 6 die Werte für $k_s = 0$, weil während des Parsens mit mehr als 20% unbekannten Wörtern auf Grund der schnell wachsenden Grösse des Suchraumes mehr und mehr Out-Of-Memory-Fehler auftraten.

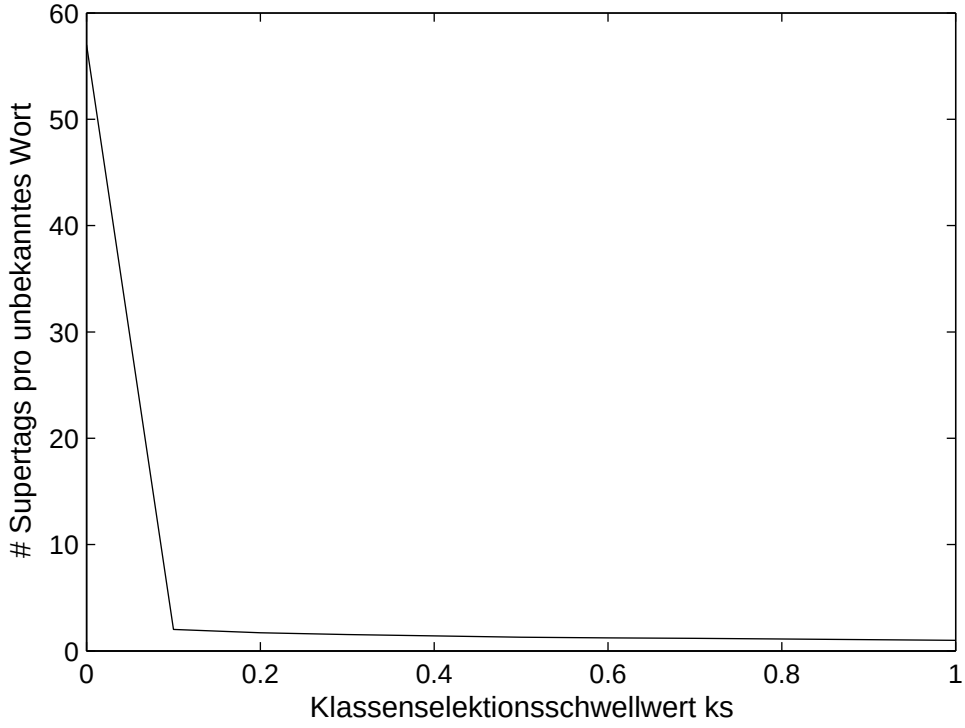
[11] Die Quantisierung von k_s in 10%-Schritten ist etwas unglücklich geraten. Eine Skala mit feinerer Auflösung für kleine k_s wäre aufschlussreicher gewesen.



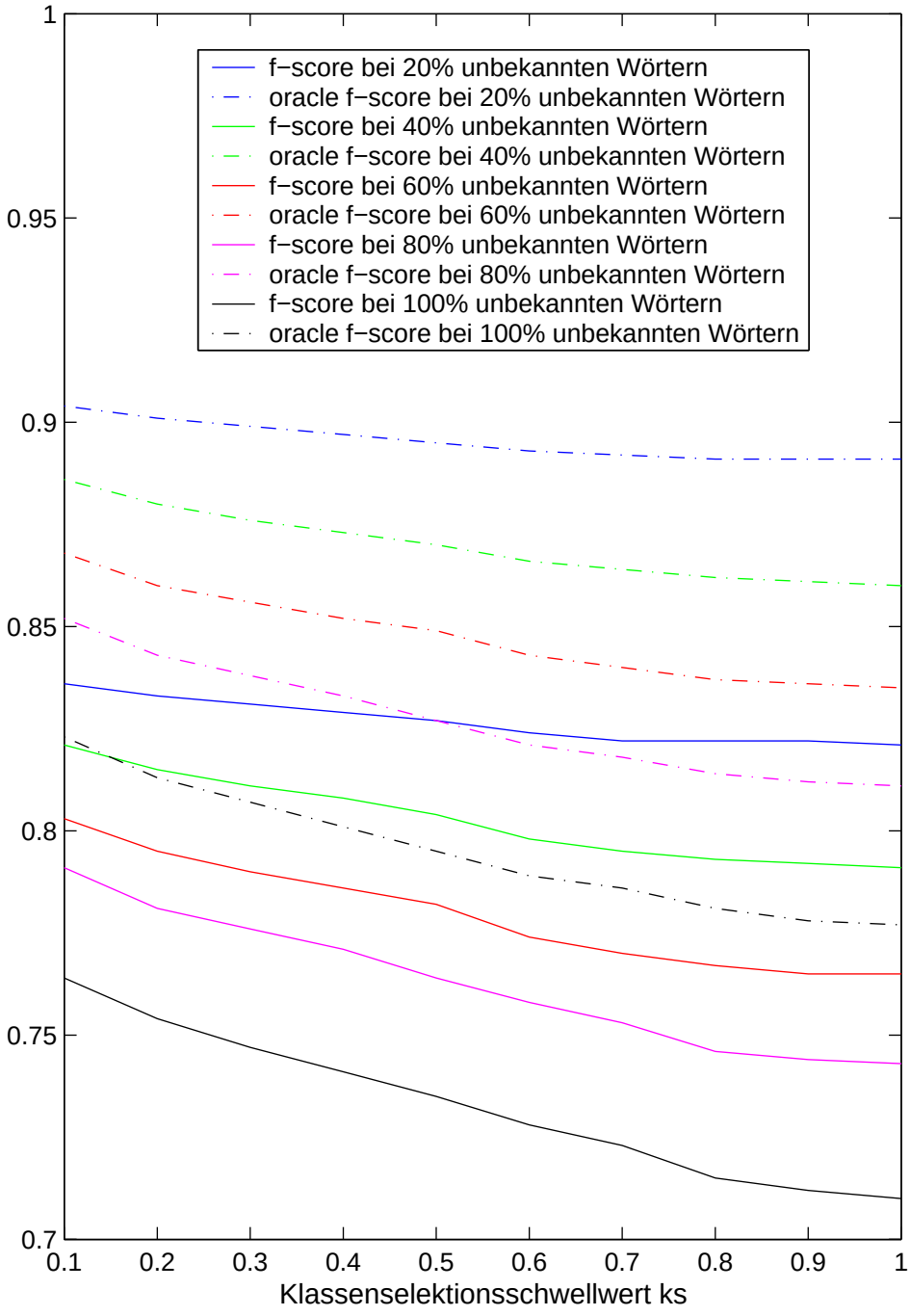
Figur 3: *Parser F-Score bei 20% unbekannten Wörtern*



Figur 4: *Parser Unification Operations bei 20% unbekannten Wörtern*



Figur 5: *Anzahl Supertags pro unbekanntes Wort bei 20% unbekannten Wörtern*



Figur 6: *Parser F-Score und oracle F-Score bei Variation der Anzahl unbekannter Wörter*

5 Schlussfolgerungen

Das Ziel dieser Semesterarbeit war es, den Umgang mit unbekannten Wörtern zu verbessern. Dazu haben wir den Supertagger entwickelt, der im Falle eines unbekannten Wortes die im Parser-Lexikon fehlenden Einträge zu syntaktischen Eigenschaften aus dem Kontext des Wortes schätzen soll. Diese Schätzung basiert einerseits auf den generischen Features, die durch den Ansatz der Feature Templates und Mutual Information erstellt wurden. Andererseits kommen auch komplexe Features zum Einsatz, wie zum Beispiel die Pattern Features, mit denen der Kontext eines Wortes in beinahe uneingeschränkter Weise untersucht werden kann.
In Verwendung mit dem Supertagger lieferte der Parser teils deutlich höherwertige Resultate als im Gebrauch ohne Supertagger. Unter gewissen Bedingungen lag die Präzision sogar nur knapp unter jener, die erreicht wurde, wenn sämtliche Wörter als bekannt galten. Die Vorhersagen der statistischen Modelle können also durchaus als gültiger Ersatz für das Parser-Lexikon verwendet werden. Die verminderte Qualität des Parsers wird dabei durch die Verbesserung seiner Effizienz wieder aufgewogen. Denn aus den Experimenten ging hervor, dass mit dem Einsatz des Supertaggers die Effizienz des Parsers auf Kosten der Genauigkeit verbessert werden kann. Mit der Wahl des Klassenselektionsschwellwertes, ein Wert für die Unsicherheit des Supertaggers, kann das Verhältnis Effizienz – Qualität beeinflusst werden, denn obwohl die Disambiguisierung mit zunehmender lexikalischer Mehrdeutigkeit nicht nur aufwändiger, sondern auch schwieriger wird, steigt dabei die Qualität des Parsers kontinuierlich an. So ist es grundsätzlich auch denkbar, dass eine weiterentwickelte Version des Supertaggers die Qualität der Einträge im Parser-Lexikon sogar übertreffen kann. Nichtsdestotrotz sind gewisse syntaktische Eigenschaften durch Supertagging nur sehr schwer vorherzusagen, weil sie selten auftreten und/oder sich im Wortkontext nicht deutlich genug preisgeben. Hier ist und bleibt das in Handarbeit erstellte Lexikon unabdingbar.

5.1 Ähnliche Arbeiten

In der Arbeit ”Enhancing Performance of Lexicalised Grammars” (siehe [DKN08]) wurden verschiedene Methoden angewendet, um die Präzision und Effizienz eines Parsers zu verbessern. Unter Anderem wurde darin auch Supertagging im Zusammenhang mit unbekannten Wörtern untersucht.
Unser Zusatzaufwand im selben Gebiet motivierte und legitimierte sich dabei insbesondere durch folgende Punkte:

Wir setzten mehr und raffiniertere Features ein. Mit zahlreichen, durch den Ansatz der Mutual Information gefundenen, generischen Features und dem Einsatz von komplexen Features boten wir unseren statistischen Modellen eine solidere Grundlage zur Vorhersage der Supertags.

Wir verwendeten andere Supertags. Durch die Einführung von syntaktischen Eigenschaften brachten wir Strukturen in die Vielzahl an lexikalischen Typen ein, wodurch sich die Vorhersage der Supertags in Teilprobleme aufsplitten liess. Jede syntaktische Eigenschaft wurde bei uns im Einzelnen untersucht und optimiert.

Wir werteten die Ergebnisse anders aus. Die in der anderen Arbeit ermittelte ”Parser Coverage” bezeichnet den Anteil an Sätzen, aus denen beim Parsing mindestens ein kompletter

Baum resultierte.[12] Dieses Mass gibt jedoch keinerlei Aufschluss über die Korrektheit der Ergebnisse (Die Parser-Präzision wurde von den Autoren lediglich in einem einmaligen, nur 100 Sätze umfassenden Test von Hand ermittelt.) und erlaubt es auch nicht, Rückschlüsse auf den Disambiguierungsschritt des Parsers zu ziehen. Der Verlauf der Präzision bei zunehmender lexikalischer Mehrdeutigkeit, d.h. bei zunehmender Anzahl Supertags pro unbekanntem Wort, war für uns aber von zentralem Interesse.

[12] Üblicherweise definiert sich die ”Parser Coverage” jedoch als der Anteil an Sätzen, aus denen der Parser die korrekte Struktur ableiten kann.

A Die Features der syntaktischen Eigenschaften

Standard Feature Templates

Jeder syntaktischen Eigenschaft wurden folgende generische Features zugewiesen:

`Em23Em13E03`	`Tm3`	`Wm2Em12`
`Em12`	`Tm2`	`Wm2Em13`
`Em13`	`Tm2Tm1`	`Wm2Em14`
`Em13E03`	`Tm1`	`Wm2Wm1`
`E02`	`Tm1Em12`	`Wm1`
`E03`	`Tm1Em13`	`Wm1E02`
`E04`	`Tm1E02`	`Wm1E03`
`E05`	`Tm1E03`	`Wm1E04`
	`Tm1T1`	`Wm1W1`
	`T0`	`W0`
	`T1`	`W1`
	`T1T2`	`W1W2`
	`T2`	`W2`

Die Bezeichnungen der Features setzen sich aus den folgenden Elementen zusammen. Die Angaben beziehen sich immer auf das Wort w, für welches das Template aufgerufen wurde.

`Em`$< x >< y >$	Die letzten y Buchstaben des Wortes x Stellen links von w
`E`$< x >< y >$	Die letzten y Buchstaben des Wortes x Stellen rechts von w
`Tm`$< x >$	Der POS-Tag des Wortes x Stellen links von w
`T`$< x >$	Der POS-Tag des Wortes x Stellen rechts von w
`Wm`$< x >$	Das Wort x Stellen links von w
`W`$< x >$	Das Wort x Stellen rechts von w

Falls sich ein Feature aus mehreren Elementen zusammenstellt, sind diese Elemente UND-verknüpft.

Die Features der syntaktischen Eigenschaften

Folgende syntaktischen Eigenschaften besitzen zusätzlich komplexe Features:

Syntaktische Eigenschaft	Featurebeschreibung
Kasus	Ein Pattern Feature, das auch in grösserer Entfernung nach einer eventuell vorhandenen Präposition sucht. Anhand der Präposition kann oft auf den Fall geschlossen werden.
Numerus	Ein Feature, das untersucht, ob w's letzter Vokal ein 'e' und w's zweitletzter Vokal ein Umlaut ist.
Genus	Ein generisches Feature wie `Em12E02`. Aber nur, falls w's linker Nachbar ein Adjektiv ist.
Artikel	Ein Pattern Feature, das auch in grösserer Entfernung nach einem eventuell vorhandenen Artikel sucht.
Hauptsatz	Ein Feature, das auch in grösserer Entfernung nach einem eventuell vorhandenen Hauptsatz sucht.

B Feature Template Gruppen

Gruppe	Feature Templates
Morphologie-basierte	{`E02, E03, E04, E04`}
Wortkontext-basierte	{`Wm2Wm1, Wm1, Wm1W1, W0, W1, W2, W1W2`}
POS-Tagkontext-basierte	{`Tm3, Tm2, Tm2Tm1, Tm1, Tm1T1, T0, T1, T1T2, T2`}
Kombinierte	Morphologie-basierte, Wortkontext-basierte, POS-Tagkontext-basierte und {`Tm1E02, Tm1E03, Tm1Em12, Tm1Em13, Wm2Em12, Wm2Em13, Wm2Em14, Wm1E02, Wm1E03, Wm1E04, Em12, Em13, Em23Em13E03, Em13E03`}
Alle	Kombinierte und allfällige Features der jeweiligen syntaktischen Eigenschaft

C POS-Tag-Set

Das hier verwendete Tagset ist das ”Stuttgart/Tübinger Tagset” (STTS), das von Anne Schiller (ehemals IMS/STR, jetzt RXRC/Grenoble), Christine Thielen (SfS/TÜB), Simone Teufel (ehemals IMS/STR, jetzt Cogsci/Edinburgh) und Christine Stöckert (IMS/STR) entwickelt wurde (Thielen, Schiller, Teufel & Stöckert, 1999). Siehe auch [TSTS].

ADJA	attributives Adjektiv	[das] groSSe [Haus]
ADJD	adverbiales oder	[er fährt] schnell
	prädikatives Adjektiv	[er ist] schnell
ADV	Adverb	schon, bald, doch
APPR	Präposition; Zirkumposition	links in [der Stadt], ohne [mich]
APPRART	Präposition mit Artikel	im [Haus], zur [Sache]
APPO	Postposition	[ihm] zufolge, [der Sache] wegen
APZR	Zirkumposition rechts	[von jetzt] an
ART	bestimmter oder	der, die, das,
	unbestimmter Artikel	ein, eine, . . .
CARD	Kardinalzahl	zwei [Männer], [im Jahre] 1994
FM	Fremdsprachliches Material	[Er hat das mit ”A big fish” übersetzt]
ITJ	Interjektion	mhm, ach, tja
KOUI	unterordnende Konjunktion	um [zu leben],
	mit ”zu” und Infinitiv	anstatt [zu fragen]
KOUS	mit Satz	wenn, ob
KON	nebenordnende Konjunktion	und, oder, aber
KOKOM	Vergleichskonjunktion	als, wie
NN	normales Nomen	Tisch, Herr, [das] Reisen
NE	Eigennamen	Hans, Hamburg, HSV
PDS	substituierendes Demonstrativ-	dieser, jener
	pronomen	
PDAT	attribuierendes Demonstrativ-	jener [Mensch]
	pronomen	
PIS	substituierendes Indefinit-	keiner, viele, man, niemand
	pronomen	
PIAT	attribuierendes Indefinit-	kein [Mensch],
	pronomen ohne Determiner	irgendein [Glas]
PIDAT	attribuierendes Indefinit-	[ein] wenig [Wasser],
	pronomen mit Determiner	[die] beiden [Brüder]
PPER	irreflexives Personalpronomen	ich, er, ihm, mich, dir
PPOSS	substituierendes Possessiv-	meins, deiner
	pronomen	
PPOSAT	attribuierendes Possessiv-	mein [Buch], deine [Mutter]
	pronomen	

PRELS	substituierendes Relativ-pronomen	[der Hund ,] der
PRELAT	attribuierendes Relativpronomen	[der Mann ,] dessen [Hund]
PRF	reflexives Personalpronomen	sich, dich, mir
PWS	substituierendes Interrogativpronomen	wer, was
PWAT	attribuierendes Interrogativpronomen	welche [Farbe], wessen [Hut]
PWAV	adverbiales Interrogativ- oder Relativpronomen	warum, wo, wann, worüber, wobei
PAV	Pronominaladverb	dafür, dabei, deswegen, trotzdem
PTKZU	”zu” vor Infinitiv	zu [gehen]
PTKNEG	Negationspartikel	nicht
PTKVZ	abgetrennter Verbzusatz	[er kommt] an, [er fährt] rad
PTKANT	Antwortpartikel	ja, nein, danke, bitte
PTKA	Partikel bei Adjektiv oder Adverb	am [schönsten], zu [schnell]
SGML	SGML Markup	turnid=n022k TS2004
SPELL	Buchstabierfolge	S-C-H-W-E-I-K-L
TRUNC	Kompositions-Erstglied	An- [und Abreise]
VVFIN	finites Verb, voll	[du] gehst, [wir] kommen [an]
VVIMP	Imperativ, voll	komm [!]
VVINF	Infinitiv, voll	gehen, ankommen
VVIZU	Infinitiv mit ”zu”, voll	anzukommen, loszulassen
VVPP	Partizip Perfekt, voll	gegangen, angekommen
VAFIN	finites Verb, aux	[du] bist, [wir] werden
VAIMP	Imperativ, aux	sei [ruhig !]
VAINF	Infinitiv, aux	werden, sein
VAPP	Partizip Perfekt, aux	gewesen
VMFIN	finites Verb, modal	dürfen
VMINF	Infinitiv, modal	wollen
VMPP	Partizip Perfekt, modal	gekonnt, [er hat gehen] können
XY	Nichtwort, Sonderzeichen enthaltend	3:7, H2O, D2XW3
$,	Komma	,
$.	Satzbeendende Interpunktion	. ? ! ; :
$(	sonstige Satzzeichen satzintern	; - [,]()

Literatur

[AFG+91] S. Abney, S. Flickenger, C. Gdaniec, C. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski. Procedure for quantitatively comparing the syntactic coverage of english grammars. In E. Black, editor, *HLT '91: Proceedings of the workshop on Speech and Natural Language*, pages 306–311, Morristown, NJ, USA, 1991. Association for Computational Linguistics.

[BDH+02] S. Brants, S. Dipper, S. Hansen, W. Lezius, and G. Smith. The TIGER treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, pages 24–41, 2002.

[Bra00] T. Brants. TnT–a statistical part-of-speech tagger. In *Proceedings of the sixth conference on Applied natural language processing*, pages 224–231, 2000.

[CJ05] E. Charniak and M. Johnson. Coarse-to-fine n-best parsing and maxent discriminative reranking. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 173–180, 2005.

[DKN08] Rebecca Dridan, Valia Kordoni, and Jeremy Nicholson. Enhancing performance of lexicalised grammars. In *Proceedings of ACL-08: HLT*, pages 613–621, Columbus, USA, June 2008. Association for Computational Linguistics.

[KP07] T. Kaufmann and B. Pfister. Applying licenser rules to a grammar with continuous constituents. In *Proceedings of the 14th International Conference on Head-Driven Phrase Structure Grammar*, pages 150–162, Stanford, USA, 2007.

[KP08] T. Kaufmann and B. Pfister. Applying a grammar-based language model to a broadcast-news transcription task. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, Columbus, USA*, June 2008.

[MAD03] K. McTait and M. Adda-Decker. The 300k LIMSI German broadcast news transcription system. In *Proceedings of Eurospeech*, Geneva, Switzerland, 2003.

[MS99] C.D. Manning and H. Schütze. *Foundations of statistical natural language processing*. MIT Press, 1999.

[SWBS99] I.A. Sag, T. Wasow, E.M. Bender, and I.A. Sag. *Syntactic theory: a formal introduction*. CSLI, 1999.

[TSTS] C. Thielen, A. Schiller, S. Teufel, and C. Stöckert. Guidelines für das Tagging deutscher Textkorpora mit STTS. Technical report, Technical report, IMS and SfS, 1999.

Institut für
Technische Informatik und
Kommunikationsnetze

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich
Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo

Frühlingssemester 2009

SEMESTERARBEIT

für

Jürg Schellenberg

Betreuer: Tobias Kaufmann ETZ D97.7
Stellvertreter: Thomas Ewender ETZ D97.7

Ausgabe: 16. Februar 2008
Abgabe: 1. Juni 2008

Automatische Bestimmung von unbekannten Wörtern

Einleitung

Bei der automatischen Analyse von natürlichsprachigen Sätzen greift ein Parser üblicherweise auf ein umfangreiches Lexikon zurück. Im Falle von lexikalisierten Präzisionsgrammatiken wie zum Beispiel *Head-driven Phrase Structure Grammars* (HPSG, [1]) enthält das Lexikon eine detaillierte syntaktische Beschreibung für jedes erfasste Wort. Das sogenannte Zipf'sche Gesetz besagt, dass in der natürlichen Sprache einige wenige Wörter sehr häufig vorkommen, während sehr viele Wörter nur selten auftreten. Dies hat zur Folge, dass man in einem gegebenen Text viele Wörter antrifft, die nicht im (notwendigerweise begrenzten) Lexikon enthalten sind.

Um uneingeschränkte Texte automatisch analysieren zu können, muss deshalb das Problem der "unbekannten" (d.h. nicht im Lexikon aufgeführten) Wörter gelöst werden. Im Rahmen dieser Semesterarbeit soll dazu ein statistisches Modell verwendet werden, welches die syntaktischen Eigenschaften eines unbekannten Wortes vorhersagt.

Aufgabenstellung

In dieser Semesterarbeit soll ein Programm entwickelt werden, welches die syntaktischen Eigenschaften eines Substantivs bestimmt. Diese Eigenschaften umfassen beispielsweise Merkmale wie Fall, Zahl und Geschlecht, aber auch Zählbarkeit und Valenzinformation. Da im allgemeinen Fall keine eindeutige Bestimmung möglich ist, soll das Programm jeweils alle Möglichkeiten angeben, gewichtet mit den entsprechenden Wahrscheinlichkeiten.

Dabei soll ein Ansatz verwendet werden, der mit dem sogenannten *Supertagging* [2] verwandt ist. Supertagging ist eine Erweiterung des Part-of-speech-Taggings (automatisches bestimmen der Wortart, siehe beispielsweise [3]), bei der an Stelle von Wortarten eine viel detailliertere Kategorisierung verwendet wird. Als zugrundeliegendes statistisches Modell soll ein sogenanntes *Maximum-Entropy-Modell* (für eine Einführung siehe [4], [5]) verwendet werden.

Schliesslich soll der entwickelte Ansatz auf echten Daten ausgewertet werden. Zum einen ist zu untersuchen, mit welcher Genauigkeit der Ansatz die syntaktischen Eigenschaften von beliebigen Wörtern bestimmen kann. Da das eigentliche Ziel in der automatischen syntaktischen Analyse liegt, soll zudem untersucht werden, wie der Ansatz die Qualität der syntaktischen Analyse und die Ausführungszeit beeinflusst. Zudem soll ermittelt werden, welchen Einfluss die Unterspezifikation der syntaktischen Merkmale auf diese beiden Grössen hat. Vergleichbare Experimente wurden etwa von [6] und [7] durchgeführt.

Die durchgeführten Arbeiten und die erhaltenen Resultate sind in einem Bericht zu dokumentieren (siehe dazu [8]), der in gedruckter Form (gebunden) und als PDF abzugeben ist. Zusätzlich sind im Rahmen eines Kolloquiums zwei Präsentationen vorgesehen: Etwa eine Woche nach Beginn soll der Arbeitsplan und am Ende der Arbeit das Resultat vorgestellt werden. Die Termine werden später bekannt gegeben.

Literaturverzeichnis

[1] Ivan A. Sag and Thomas Wasow. *Syntactic Theory: A Formal Introduction.* CSLI Publications, Stanford, 1999.

[2] Srinivas Bangalore and Aravind K. Joshi. Supertagging: An approach to almost parsing. *Computational Linguistics*, 25:237–265, 1999.

[3] A. Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In Eric Brill and Kenneth Church, editors, *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 133–142. Association for Computational Linguistics, Somerset, New Jersey, 1996.

[4] D. Klein and C. Manning. Maxent models, conditional estimation, and optimization. http://www.cs.berkeley.edu/~klein/papers/maxent-tutorial-slides.pdf, 2003.

[5] James Curran. Maximum entropy models for natural language processing. http://www.alta.asn.au/events/altss2004/course_notes/ALTSS-Curran-Maxent.pdf, 2004.

[6] J. Nicholson, V. Kordoni, Y. Zhang, T. Baldwin, and R. Dridan. Evaluating and extending the coverage of hpsg grammars. In *Proceedings of 6th International Conference on Language Resources and Evaluation (LREC2008), Marrakech, Morocco*, 2008.

[7] Rebecca Dridan, Valia Kordoni, and Jeremy Nicholson. Enhancing performance of lexicalised grammars. In *Proceedings of ACL-08: HLT*, pages 613–621, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[8] B. Pfister. *Hinweise für die Präsentation der Semester- oder Diplomarbeit.* Institut TIK, ETH Zürich, März 2004.

[9] S. Brants, S. Dipper, S. Hansen, W. Lezius, and G. Smith. The TIGER treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, Sozopol, Bulgaria, 2002.

Zürich, den 13. Februar 2009