ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

TIK Institut für Technische Informatik und Kommunikationsnetze

Wortbedeutung und Satzstruktur

Semesterarbeit FS 08

Raphaël Jecker and Benjamin Knecht

Gruppe für Sprachverarbeitung
Betreuer: Tobias Kaufmann

11. Juli 2008

Inhaltsverzeichnis

1 Einleitung und Aufgabenstellung

In dieser Arbeit soll untersucht werden, in welchem Ausmass aus rein statistischer Information über das einzelne und gemeinsame Auftreten von Wörtern in deutschen Texten Aussagen über deren syntaktische Beziehung untereinander gemacht werden können. Dazu wird ein Mass benötigt, welches die Zusammengehörigkeit zweier Wörter in der deutschen Sprache angibt. Beispiele, die hier offensichtlich hohe Werte erhalten sollten, sind *'zum'* und *'Beispiel'* oder *'am'* und *'Freitag'*. Jedoch sollten auch Worte wie *'Wein'* und *'Bier'* höhere Werte erzielen als beispielsweise *'Wein'* und *'Elefant'*, da erstere deutlich öfter im gleichen Kontext zu erwarten sind als letztere. Basierend auf einem möglichst umfangreichen Text sollen diese sogenannten Signifikanzen für alle Wortpaare bestimmt und notiert werden, wobei verschiedene Arten von 'gemeinsamem Auftreten' wie beispielsweise 'innerhalb eines Satzes vorkommend' oder 'mit gewissem Maximalabstand' getrennt betrachtet werden. Mit Hilfe dieses sogenannten Signifikanzmasses soll dann untersucht werden, ob und wie sich syntaktische Beziehungen in diesem Signifikanzmass wiederfinden lassen und auf welche Weise sich möglicherweise die Zuverlässigkeit der auf diesem rein statistischen Mass beruhenden Aussagen erhöht werden kann. Hier kann zum Beispiel versucht werden, die Worte nach Wortarten zu unterscheiden oder sogar ihre semantische Klasse zu berücksichtigen. Weiterhin könnten beispielsweise Filter angewandt werden, welche die relativen Positionen der Worte zueinander vorgeben. Als Referenz zur Verifizierung der Resultate wird hier ein syntaktisch annotierter Korpus, der sogenannte TIGER-Korpus [1], verwendet.

Ein Beispiel zur Veranschaulichung bietet der Satz *„Ihm droht die Auslieferung nach Den Haag“* (Abbildung 1), bei dem nach rein syntaktischem Wissen nicht entschieden werden kann, ob die Präpositionalphrase *'nach Den Haag'* zum Verb *'droht'* oder zum Nomen *'Auslieferung'* zuzuordnen ist. Erst die Hinzunahme von semantischem Wissen in Form einer Signifikanzstatistik liefert hier Entscheidungshilfe.

Eine weitere direkte Anwendung der erstellten Signifikanzstatistik liegt in deren Rolle als Nachschlagewerk bei dem Problem, einzelne Worte in einem über einen verlustbehaftenen Kanal gesendeten Text richtig zu erkennen. Ein Beispiel hierfür stellt die computerseitige Spracherkennung dar, bei der ein unverständlich gesprochenes Wort nun durch Hinzunahme des semantischen Wissens der Signifikanzstatistik zuverlässiger rekonstruiert werden kann. In Abbildung 2 ist ein Beispielsatz dargestellt, bei dem fälschlicherweise das Wort *'erwärmen'* erkannt wurde. Die Konsultation der Signifikanzstatistik hätte in diesem Fall darauf hingedeutet, dass die Wörter *'Waffen'* und *'erwerben'* viel eher gemeinsam zu erwarten sind, als *'Waffen'* und *'erwärmen'*.

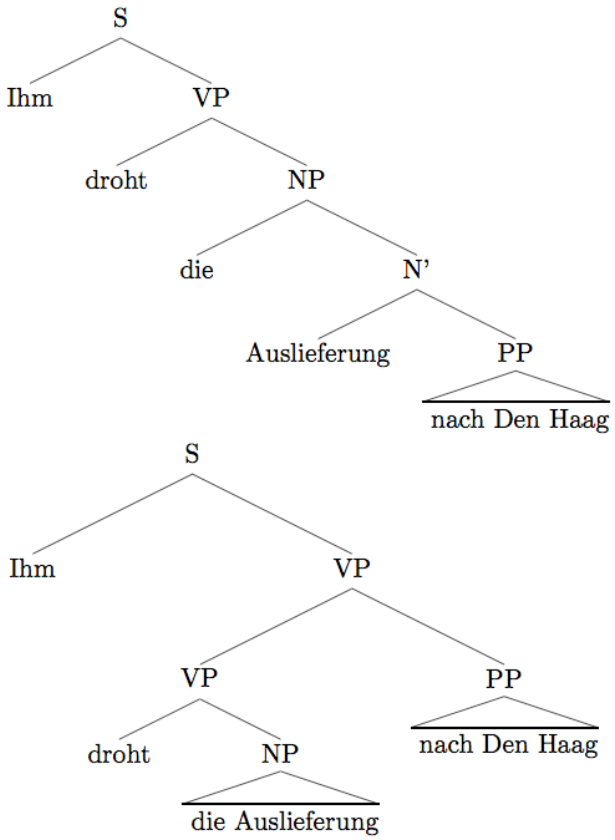

Abbildung 1: Beispielsatz 1
S=Satz, VP=Verbalphrase, NP=Nominalphrase, PP=Präpositionalphrase

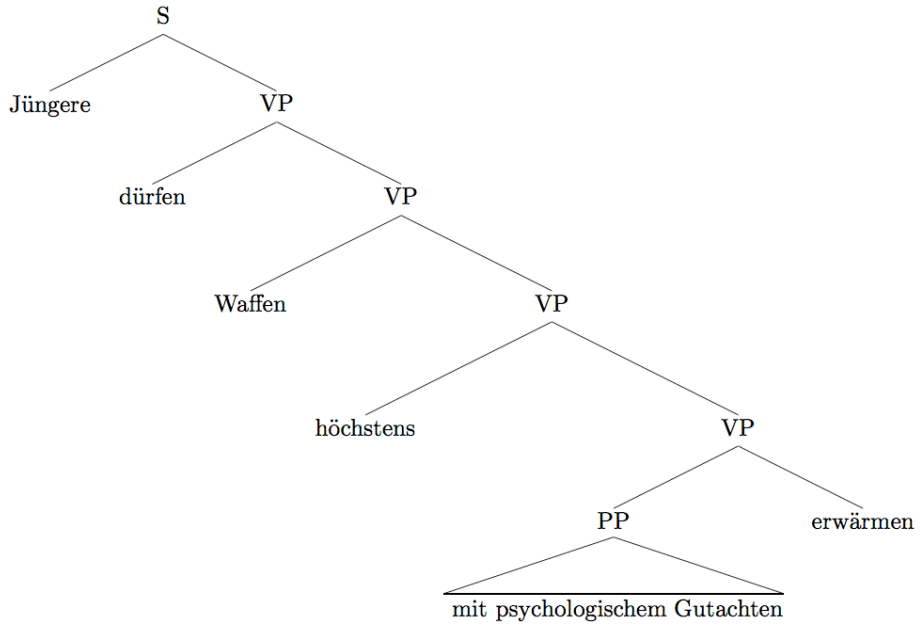

Abbildung 2: Beispielsatz 2
S=Satz, VP=Verbalphrase, NP=Nominalphrase, PP=Präpositionalphrase

2 Statistiken

2.1 Erstellung der Kookurrenzstatistiken

Kookurrenzstatistiken beschreiben die semantische Nähe zweier Wörter in der deutschen Sprache anhand eines für deren gemeinsames Auftreten charakteristischen Wertes, der sogenannten *Signifikanz*. Um ein Mass dafür zu erhalten, wie stark sich zwei Wörter gegenseitig bedingen, muss zum einen gezählt werden, wie oft die beiden Worte zusammen in einer gegebenen Grundstruktur vorkommen, beispielsweise einem Satz oder einem Wortfenster fester Länge, zum anderen muss jedoch auch berücksichtigt werden, wie oft die einzelnen Worte überhaupt selbst vorkommen, um mit deren Verhältnis eine Aussage über die relative Häufigkeit des gemeinsamen Vorkommens in Vergleich zum einzelnen Vorkommen machen zu können. Als konkretes Signifikanz-Mass kommt in dieser Arbeit das sogenannte Poisson-Kollokationsmass zur Anwendung [3]. Man vergleicht hierbei die Zahl tatsächlicher gemeinsamer Auftritte zweier Worte mit der Anzahl erwarteter Auftritte unter der Annahme der statistischen Unabhängigkeit. Sind zwei Worte also selten, so ergibt sich unter dieser Annahme die Erwartung, gemeinsame Auftritte seien noch viel seltener. Treten diese zwei Worte nun beim Zählen in richtigen Texten jedoch viel häufiger auf als erwartet, ergibt dies eine hohe Signifikanz für diese beiden Worte.

Wird das Auftreten zweier Worte in einer Grundeinheit als Zufallsexpe-

riment mit Poisson-Verteilung angesehen, so ergibt sich für die Wahrscheinlichkeit von k gemeinsamen Auftritten

$$p_k = \frac{1}{k!} \cdot \lambda^k \cdot e^{-\lambda}$$

mit $\lambda = n \cdot p_a \cdot p_b$ und p_a, p_b den Einzelwahrscheinlichkeiten der Worte A und B. Für die Wahrscheinlichkeit von mindestens k gemeinsamen Auftritten ergibt sich dann

$$Pr(X > k) = \sum_{l=k}^{\infty} \frac{1}{l!} \cdot \lambda^l \cdot e^{-\lambda}$$

.

Sind nun k die Anzahl tatsächlich gezählter Auftritte, erhält man mit der obigen Formel ein Mass dafür, wie wahrscheinlich nach Annahme der Wort-Unabhängigkeit der Fall der gezählten oder mehr Auftritte ist. Wortpaare, die also sehr häufig auftreten, obwohl sie unwahrscheinlich sind, erhalten sehr kleine Werte, Wortpaare, die sehr selten auftreten, obwohl sie häufig sein sollten, erhalten grosse Werte. Da dies zum einen sehr grosse Werte ergeben kann, zum anderen die intuitive Vortstellung der Signifikanz dazu genau reziprok ist, wendet man auf die obige Formel den inversen Logorithmus an. Um zudem von der Grösse des Textes unabhängig zu werden, normiert man zusätzlich mit dem Logarithmus der Anzahl Grundeinheiten im betrachteten Text. Dies ergibt die folgende Formel zur Berechnung der Signifikanz:

$$sig(A, B) = \frac{-\log \sum_{l=k}^{\infty} \frac{1}{l!} \cdot \lambda^l \cdot e^{-\lambda}}{\log n}$$

bei welcher für kleine λ der Zähler auf den ersten Summanden reduziert und die Fakultät von k für $k \geq 10$ durch die Stirling-Formel vereinfacht werden kann. Daraus folgt schliesslich

$$sig(A, B) = \frac{k \cdot (\log k - \log \lambda - 1)}{\log n}$$

Für die Aussagekraft der Statistik ist es dabei wichtig, mit sehr grossen, sich einer konsistenten Sprache bedienenden Texten zu arbeiten. Für diese Arbeit verwendeten wir stets einen aus Zeitungsartikeln bestehenden Textkorpus mit rund 250 Millionen Wörtern.

2.1.1 Erweiterte Korpora

Nebst der Rohversion des Korpus verwendeten wir zwei weitere, vom Originaltext abgeleitete und bearbeitete Versionen. Als Erstes war dies eine Fassung des Texts, bei der hinter jedem Wort, getrennt durch einen Punkt, ein Kürzel zur Bezeichnung der Wortart folgt. Da die Wortart und das Wort selbst an einem Stück geschrieben sind (beispielsweise *der.ART Baum.NN*),

kann der modifizierte Text genau gleich wie der Original-Text vom Algorithmus bearbeitet werden, wobei die unterschiedlichen Wortarten, zu denen ein Wort gehören kann, automatisch mitberücksichtigt werden und somit beispielsweise das Nomen *Sucht* vom Verb (er/sie) *sucht* getrennt betrachtet werden kann und auch getrennt in der Statistik aufgelistet wird. Das Erweitern der Wörter um ihre Wortarten geschah automatisch mit Hilfe des TNT-Taggers [5]

Zum Zweiten verwendeten wir eine komplett umgeschriebene Fassung des Textes, bei dem jedes Wort durch die Wortgruppe ersetzt wird, zu der es vom im übernächsten Abschnitt erklärten Clustering Algorithmus zugeteilt wurde. Der so entstandene Text besteht also folglich nur noch aus Bezeichnern der Wortgruppen, die jedoch noch in der gleichen Reihenfolge vorliegen, und kann damit wieder gleich wie der Originaltext behandelt werden. Als Ergebnis kommen dabei dann folglich Signifikanzen der Wortgruppen zueinander anstelle der Worte selbst heraus.

2.1.2 Algorithmus

In einem ersten Schritt wird der Text wortweise durchlaufen, wobei zu jedem Wort gespeichert wird, wie oft es im Text vorkommt. Im Textcorpus beläuft sich die Anzahl unterschiedlicher Wörter auf etwa 2.9 Mio Wörter, wovon jedoch der allergrösste Teil sehr selten auftritt ($98\% < 80$ Mal), was vor allem auf Rechtschreibfehler und seltene Eigennamen zurückzuführen ist. Der Grossteil dieses Wörterbuchs konnte daher ohne grösseren Verlust für die Allgemeinheit der deutschen Sprache ignoriert werden, wie später erläutert wird.

Um den Tücken der Gross- und Kleinschreibung zu entgehen, welche insbesondere mit der Grossschreibung an Satzanfängen für unklare Verhältnisse sorgen könnten, werden beim ersten Durchlauf durch den Text alle Worte komplett in Kleinbuchstaben geschrieben. Den somit entstandenen Mehrdeutigkeiten bestimmter Worte wie *die Sucht* und *er/sie sucht* kann, wie im nächsten Abschnitt erläutert wird, durch an jedes Wort angehängte Wortarten beigekommen werden.

Eine implementationstechnische Herausforderung stellte sich in der Wahl der Datenstruktur zur Speicherung der Zähler für das gemeinsame Auftreten zweier Worte. Der erste Versuch, die Zähler als *unsigned int* zusammen mit einem aus den beiden Worten zusammengesetzten String in einer in der *Standard Template Library (STL)* [2] implementierten *map* zu speichern, lieferte unbefriedigende Ergebnisse, was die Anzahl der im Hauptspeicher zu haltenden Wortpaar-Einträge anbelangte.

Im Weiteren kam deshalb ein zweidimensionales, quadratisches Array zum Einsatz, wobei für jedes betrachtete Wort eine Zeile und eine Spalte reserviert sind. Um nun bei einem Wortpaar den Eintrag zu erhöhen, muss lediglich an der richtigen Stelle, bei der sich die beiden Worte im Array tref-

fen, ein Zähler erhöht werden. Die speicherintensiven Strings müssen daher nur gerade ein einziges Mal in einem eindimensionalen Array abgelegt werden, um am Schluss vom betreffenden Index wieder auf das Wort schliessen zu können.

Da zur Allozierung der 2.9 Mio x 2.9 Mio Einträge jedoch selbst im kleinsten char-Format bereits 8.4 Terabyte nötig würden, ist eine Reduktion der Anzahl betrachteter Wörter unvermeidlich. Bei den Versuchen zu dieser Arbeit stand ein Rechner mit 8 Gb RAM zur Verfügung, weshalb die Anzahl Wörter auf rund 84000 reduziert wurde (ergibt ein Array von rund 7 Gb). Aufgrund der oben erwähnten Seltenheit des Grossteils dieser 2.9 Millionen Wörter stellt die drastische Reduktion zwar einen Informationsverlust dar, der wünschenswerterweise vermieden werden sollte, unter praktischen Gesichtspunkten für die interessierende Statistik jedoch verschmerzbar ist.

Dem Problem, dass im char-Format nur 256 Wortpaare gezählt werden können, konnte wie folgt begegnet werden: Da im Array für ein Wortpaar immer zwei Zellen existieren, nämlich eine in der oberen Dreiecksmatrix und eine in der unteren Dreiecksmatrix, konnte der Zahlenbereich auf $256 * 256 = 65536$ erweitert werden, indem die obere Dreiecksmatrix jeweils das lower- und die untere Dreiecksmatrix jeweils das higher Byte einer 16 Bit Zahl darstellen. Für die seltenen Fälle, wo selbst dies nicht ausreichen sollte, wurde zudem die Möglichkeit geschaffen, auf eine STL-map auszuweichen, in der für ein Wortpaar bei Bedarf noch über 65535 hinaus weitergezählt werden kann.

Als gemeinsames Auftreten zweier Wörter wurden 3 verschiedene Szenarien betrachtet:

- direkt aufeinanderfolgendes Vorkommen
- Gemeinsames Vorkommen im gleichen Satz
- gegebene Maximaldistanz über Satzgrenzen hinweg (Wortfenster)

Nachdem die gemeinsamen Auftritte gezählt sind, kann, zusammen mit den Einzelauftrittszählern der Wörter die Signifikanz für alle betrachteten Wortpaare berechnet werden. Diese wurde mit dem Poisson-Mass ermittelt, wie anfangs des Kapitels ausgeführt.

Die Beschreibung der Formate der dabei erzeugten Dateien befindet sich im Anhang.

2.1.3 Der Clustering Algorithmus

Ziel des Clustering Algorithmus ist es, eine gegebene Anzahl Wörter in eine Anzahl semantisch möglichst homogener Wortgruppen oder Wortklassen, den sogenannten *Clusters*, einzuteilen. Abhängig von der Zahl der angestrebten Gruppen lässt sich damit eine in Theorie fast beliebig genaue Zusammennahme sinn-ähnlicher Wörter erreichen, wie bei [4] eindrücklich demonstriert

wird. Funktioniert der Algorithmus gut und wählt man eine grosse Anzahl unterschiedlicher Klassen, so erwartet man Thesaurus-ähnliche Resultate, also Wortklassen die als Gruppierung zusammengehörender Synonyme auftreten wie beispielsweise *'trinken, saufen, schlürfen, betrinken, leeren'* oder aber sinnverwandte Wörter wie *'Bier, Wein, Wasser, Cola, Whisky'*. Verwendet man für die Kollokationsstatistik dann einen Korpus, bei dem die Worte durch ihre Wortklasse ersetzt wurden, so tritt bei den gewonnenen Daten der Sinn eines Worts in den Vordergrund vor dem Wort selbst. Hat man unter Verwendung des normalen Texts beispielsweise gute Ergebnisse für *'Wein'* und *'trinken'*, so hat man mit Berücksichtigung der Wortklassen im günstigen Fall gerade auch eine Aussage über *'Bier'* und *'saufen'*, falls nämlich *'Wein'* und *'Bier'* sowie *'trinken'* und *'saufen'* im gleichen Cluster liegen.

Ausgehend vom dortigen Vorschlag, erweiterten wir den Algorithmus derart, dass statt der alleinigen Berücksichtigung der unmittelbar umgebenden Worte ein frei wählbares Wortfenster um das betreffende Wort mitberücksichtigt wird. Von dieser Zusatzinformation erhofften wir uns zusätzliche Verbesserung der Gruppierungsfähigkeit des Algorithmus aufgrund der Einsicht, dass semantisch zusammenhängende Worte oft nicht unmittelbar aufeinander folgen, sondern andere Wörter dazwischen liegen, welche weniger interessant sind (beispielsweise Pronomen oder Präpositionen). Die Grundlage des Algorithmus besteht in der Annahme, dass Wortumgebungen eine hohe Aussagekraft über die in ihnen auftretenden Worte haben. Ähnlich wie bei der Signifikanzberechnung geht man davon aus, dass gewisse Worte häufiger zusammen auftreten als andere. Neu ist allerdings, dass nun nicht das *gemeinsame* Auftreten zweier Worte interessiert, sondern vielmehr das *stellvertretende* Auftreten eines Worts für ein anderes in vergleichbarer Umgebung. Eine Schlüsselrolle spielt dabei das Konzept der sogenannten Mutual Information, einem Begriff aus der Informationstheorie, mit dessen Hilfe der Grad der gegenseitigen Information zweier Wortklassen in eine Zahl gefasst werden kann. Demnach haben Klassen bei denen das Vorhandensein der Einen mit guter Zuverlässigkeit auf das Vorhandensein der Zweiten schliessen lässt, eine hohe Mutual Information; solche, die wenig über gemeinsames Auftreten aussagen, eine geringe. Um eine Anzahl gegebener Worte in Klassen einzuteilen, würde man idealerweise vorerst jedes Wort als Cluster interpretieren und danach wiederholt aus allen möglichen Zusammenfügungen jene wählen, bei welcher die resultierende aufsummierte Mutual Information aller Cluster maximal wäre, bis nur noch die gewünschte Anzahl Cluster übrigbleibt.

Da bei diesen Berechnungen stets für jede mögliche Cluster-Kombination die Mutual Information aller Cluster unter sich berechnet werden müsste, steigt der Rechenaufwand für grosse Anzahl Worte schnell ins Astronomische. Aus diesem Grund kommt in dieser Arbeit ein Greedy Algorithmus zum Einsatz, wie er in [4] vorgeschlagen wird. Hierbei geht man

von der letztlich angestrebten Anzahl Cluster aus und berechnet stets unter Hinzunahme eines neuen, als zusätzlichem Cluster interpretierten Worts, welche Zusammennahme in der höchsten Mutual Information resultiert. Steht das beste Paar fest, werden die Worte der beiden Cluster zusammengeführt.Anschliessend führt man dasselbe mit dem nächsten Wort durch, bis alle Wörter in die festgelegte Anzahl Cluster eingeteilt sind.

Die Vorgehensweise ist dabei die folgende:

In einem ersten Schritt wird die gewünschte Anzahl Cluster erzeugt. Jeder Cluster wird dabei mit einem Wort gefüllt, wobei mit dem am häufigsten vorkommenden Wort begonnen wird und danach durch die nach Häufigkeit sortierte Wortliste gegangen wird. Sind die Cluster mit je einem Wort initialisiert, wird das Durchlaufen durch die sortierte Wortliste fortgesetzt, wobei das neu betrachtete Wort einen eigenen Cluster darstellt. Da nun ein Cluster mehr existiert als erwünscht, wird in der Folge ermittelt, welche zwei der momentan existierenden Cluster zusammengeführt werden sollen. Dazu berechnet man für jede mögliche Zusammenführung die daraus resultierende aufsummierte Mutual Information aller Cluster zueinander, wie in der folgenden Formel dargestellt:

$$I = \sum_{c_1 c_2} Pr(c_1 c_2) \log \frac{Pr(c_2|c_1)}{Pr(c_2)}$$

Summiert wird über alle vorhandenen Cluster, wobei mit $Pr(c_1 c_2)$ die Wahrscheinlichkeit bezeichnet wird, dass die Cluster c_1 und c_2 im Wortfenster gemeinsam auftreten und $Pr(c_2|c_1)$ deren bedingte Wahrscheinlichkeit darstellt.

Jenes Paar, deren Zusammenführung den kleinsten Verlust der Mutual Information im Vergleich zum unberührten Zustand bringt, wird danach zusammengeführt. Hierbei werden die Häufigkeits-Zähler sowohl der Cluster selbst als auch der Cluster mit jedem übrigen Wort zusammengezählt und als Zähler des neu entstandenen Clusters gespeichert. Diese Methode wird solange fortgesetzt, bis alle betrachteten Wörter einem Cluster zugeteilt sind.

Die für die Berechnung der Wahrscheinlichkeiten benötigten Informationen über die Anzahl Vorkommen der Worte einzeln und gemeinsam konnten direkt aus der Implementierung der Signifikanzberechnung übernommen werden, also dem zweidimensionalen Array mit den Wortpaarzählern. Da jedoch zur Erstellung des Arrays der zur Verfügung stehende Hauptspeicher bereits bis an die Grenzen ausgenützt wurde, standen für die nun durch Addition der Zähler zusammengefügter Klassen schnell über 256^2 steigenden Werte kein freier Speicher mehr zur Verfügung. Einziger Ausweg stellte das Neuzuordnen des durch Zusammenführung freigewordenen Speicherplatzes dar. Wurde also beispielsweise ein Cluster c_{21} mit einem bereits existierenden Cluster c_4 zusammengeführt, so konnten die Array-Zellen aller Wortpaar-

Zähler des Clusters c_{21} mit allen anderen Wörtern für den neuen $c_{4,21}$-Cluster verwendet werden. Um eine konsistente Verwaltung der Zellen zu ermöglichen, wurde stets auf der Stelle des Clusters mit dem tieferen Index zusammengeführt. Da pro Wortzähler zwei char-Arrayzellen zur Verfügung standen, ergab sich somit für die Zähler zusammengeführter Cluster ein Zahlenbereich bis $256^4 - 1$, was selbst bei wiederholtem Zusammenführen sehr häufiger Wortpaare für deren addierte Zähler gut ausreichte.

Die Komplexität der Berechnungen der Mutal Informations bei jedem neu hinzuzufügenden Worts, sind von der Ordnung $O(n^2)$ in der Anzahl Cluster und sind mit unserer Implementation bis etwa zu einer Anzahl von ungefähr 100 Clustern praktisch sinnvoll durchführbar.

3 Auswahl von Wortpaaren

3.1 Einführung

Für die Wahl der Wortpaare waren mehrere Faktoren wichtig. Als erstes mussten wir uns für die Wortarten entscheiden, welche von besonderer semmantischer Bedeutung sind. Die Entscheidung fiel auf Nomen, Verben, Präpositionen, Adjektiven und Adverben. Präpositionen sollten vor allem die Verwendung von Nomen im Satz verdeutlichen, während bei Adjektiven und Adverben deren Verwendung im Zusammenhang mit Nomen und Verben interessant ist. In einem zweiten Schritt mussten Wortpaare aus Wörtern dieser Wortarten definiert werden, um diese dann in einem Satz finden zu können. Dabei war natürlich die syntaktische Beziehung zwischen den beiden Wörtern entscheidend. Als Startpunkt haben wir Nominalphrasen, welche im nächsten Abschnitt noch genauer erklärt werden, verwendet, um Wortpaare zu bilden. Später sind wir dann zu Präpositional- und Verbphrasen übergegangen.

3.1.1 Phrasen und deren 'Köpfe'

Eine Phrase ist eine Gruppe von Wörtern, welche wie ein einzelnes Element in der Syntax eines Satzes wirkt. Die Phrase kann wiederum mehrere Phrasen enthalten, womit sich eine Art Baum aufspannt. Meistens zeichnen sich Phrasen durch ein besonders zentrales Wort aus, ohne das die Phrase ihre Bedeutung und Funktion verliert. Dieses Wort wird *Kopf* der Phrase genannt.

Nominalphrase (NP) Eine Nominalphrase fungiert als Nomen im Satz. Somit sind Subjekte und Objekte im Satz Nominalphrasen. Der Kopf der NP ist das Nomen. Eine NP besteht aus einem Kern, der in den meisten Fällen aus einem Nomen, dazugehörigen Artikel und Adjektiv besteht. Ein Beispiel für eine NP wäre: *„Die schwarze Katze“*. *„Katze“* ist also der Kopf

der NP. Es gibt auch kopflose NPs, welche aber bei der Wahl der Wortpaare nicht beachtet wurden.

Verbphrase (S, VP) Der Kopf einer Verbphrase ist das Verb. Dies bedeutet weiter, dass mit jedem Verb im Satz auch eine zusätzliche Verbphrase gegeben ist. Ein ganzer Satz ist eine Verbphrase mit finitem Verb und kann deshalb wie eine VP betrachtet werden. Beispiel für eine VP: *„Die schwarze Katze schläft am Nachmittag“*. Ein Satz als Verbphrase, worin *„schläft“* das finite Verb darstellt. Das Subjekt ist die Nominalphrase *„Die schwarze Katze“*. *„am Nachmittag“* ist eine Präpositionalphrase als Modifikator der Verbphrase.

Präpositionalphrase (PP) Der Kopf einer Präpositionalphrase ist die Präposition. Neben der Präposition enthält die PP eine Nominalphrase. Eine Präpositionalphrase kann sowohl in einer Verbphrase (s.o.) als auch in einer Nominalphrase existieren. Beispiel: *„Die schwarze Katze auf dem Baum auf dieser saftigen Wiese schläft am Nachmittag.“* Das Subjekt ist nun länger geworden: *„auf dem Baum“* ist eine Präpositionalphrase und postnominaler Modifikator der Nominalphrase, die das Subjekt bildet, *„auf dieser saftigen Wiese“* ist ebenfalls eine Präpositionalphrase und postnominaler Modifikator der vorherigen Präpositionalphrase *„auf dem Baum“*.

3.2 Wortpaare aus Nominalphrasen

Für den Versuch wurden 7 verschiedene Arten von Wortpaaren aus Nominal- und Präpositionalphrasen auf ihren semantischen Zusammenhang untersucht.

NP-Kopf mit Adjektiv Das Wortpaar bildet sich aus dem Kopf einer NP (oder PP) und dem dazugehörigen attributiven Adjektiv.

Beispiel: *„Die schwarze Katze“*

- **NP-Kopf:** Katze
- **Adjektiv:** schwarze

NP-Kopf mit postnominalem Genitiv Das Wortpaar bildet sich aus dem Kopf einer NP (oder PP) und dessen Genitivattributes Kopf. Das Genitivattribut ist in der Regel eine NP und steht immer nach dem Kopf der ersten NP. Das Wortpaar besteht also aus zwei Köpfen einer NP, also aus zwei Nomen.

Beispiel: *„Die Katze meiner Tante“*

- **NP-Kopf:** Katze
- **NP-Kopf des Genitivs:** Tante

NP mit Genitivparaphrase oder Präpositionalobjekt Besitzt eine Nominalphrase eine Präpositionalphrase als Genitivparaphrase oder als Präpositionalobjekt, so lassen sich im Ganzen drei Wortpaare bilden:

1. Die beiden Köpfe der beiden Phrasen. Beispiel: *„Der **Wettkampf** um den **Pokal**“.*
2. Der Kopf der NP und die Präposition der PP. Beispiel: *„Der **Wettkampf um** den Pokal“.*
3. Die Präposition mit dem Kopf der dazugehörigen PP. Beispiel: *„Der Wettkampf **um** den **Pokal**“.*

NP mit postnominalem Modifikator Analog zum vorherigen Paragraphen.

1. Die beiden Köpfe der beiden Phrasen. Beispiel: *„Die **Katze** auf dem **Baum**“.*
2. Der Kopf der NP und die Präposition der PP. Beispiel: *„Die **Katze auf** die Baum“.*
3. Die Präposition mit dem Kopf der dazugehörigen PP. Beispiel: *„Die Katze **auf** dem **Baum**“.*

Im Gegensatz zu den ersten beiden Wortpaaren wird bei der dritten Art von Präposition und PP-Kopf nicht zwischen postnominalem Modifikator und Genitivparaphrase oder Präpositionalobjekt unterschieden. Da beide Wörter des Wortpaares aus derselben PP sind, ist deren Funktion nicht relevant für das Wortpaar.

3.3 Wortpaare aus Verbphrasen

Viele Verben lassen sich auftrennen und werden dann zweiteilig: *„Ich fange mit der Übung an.“* Verb: anfangen. In anderen Fällen kommen im Satz Hilfsverben zum Einsatz: *„Ich sollte mit der Übung anfangen.“* oder *„Ich habe die Übung angefangen.“*

Es scheint ersichtlich, dass *an* sowie auch *habe* oder *sollte* wenig Bedeutung ohne das Vollverb *anfangen* haben. Auch lässt sich argumentieren, dass der Kopf des Objekts *„die Übung“* mit dem Vollverb eine stärkere semantische Beziehung hat und sich diese beiden Wörter besser als Wortpaar eignen, als Wortpaare mit Hilfsverben. Da sich das Vollverb in untergeordneten Phrasen befinden kann, wird der Satz, den man sich als Baum vorstellen kann, von oben nach unten verarbeitet. Auf dem Weg zum Vollverb werden alle Komponenten für die Wortpaare zwischengespeichert, bis das Vollverb gefunden wurde. Dann werden folgende Wortpaare erstellt:

VP-Kopf mit Adjektiv Dieses Wortpaar bildet sich aus dem Vollverb und den zugehörigen adverbialen oder prädikativen Adjektiven in der Verphrase und deren klausalen Objekten. Der Dateiname lautet *V_ADJ*

Beispiel: *„Die schwarze Katze schläft tief.“*

- **VP-Kopf:** schläft
- **Adjektiv:** tief

VP-Kopf mit Adverb Dieses Wortpaar besteht aus dem Vollverb der Verbphrase und deren Adverbien. Dateiname: *V_ADV*.

Beispiel: *„Die Katze schläft immer auf dem Baum.“*

- **VP-Kopf:** schläft
- **Adverb:** immer

VP-Kopf mit Subjekt oder Objekt Eine Verbphrase besitzt in der Regel ein Subjekt, welches selber meistens eine NP ist. Dazu kommen manchmal auch Akkusativobjekte, Genitivobjekte und Dativobjekte, die ihrerseits ebenfalls Nominalphrasen sind. Das Wortpaar bildet sich aus dem Vollverb und den Nomen des Subjekts und den verschiedenen Objekten der VP und wird in die Datei *V_N_npcomp* geschrieben.

Beispiel: *„Die schwarze Katze schläft auf dem Baum.“*

- **VP-Kopf:** schläft
- **Nomen (Subjekt):** Katze

VP-Kopf mit Genitivparaphrase, Präpositionalobjekt oder postnominalem Modifikator Präpositionalphrasen können auch Verbphrasen untergeordnet sein, ähnlich wie bei den Nominalphrasen. Eine solche Konstruktion ergibt zwei Arten von interessanten Wortpaaren.

1. Vollverb und Kopf der PP. Beispiel: *„Der Richter **urteilt** über den **Angeklagten**.“*
2. Vollverb und Präposition der PP. Beispiel: *„Der Richter **urteilt über** den Angeklagten.“*

VP-Kopf mit Modifikator Ein Modifikator ist unter anderem eine der Verbphrase untergeordnete Präpositionalphrase. Das Wortpaar besteht analog wie das vorherige Paar aus dem Vollverb und dem Nomen oder der Präposition der PP.

1. Vollverb und Kopf der PP. Beispiel: *„Die Katze **schläft** auf dem **Baum**.“*

2. Vollverb und Präposition der PP. Beispiel: *„Die Katze **schläft auf** dem Baum.“*

3.4 Implementation

Als Quelle für die Wortpaare wurde ein Korpus [1] mit über 50'000 Sätzen gewählt, der mit dem TIGER-Annotationsshema [6] annotiert ist. Der Korpus lag im XML-Format vor, welches die Baumstruktur eines jeden Satzes enthält. Mit der TIGER Java API [7] wird der Korpus geladen und die Sätze in Form von Bäumen abrufbar gemacht. Wegen der hohen Speicheranforderungen (ca. 7 GB) musste der Korpus viergeteilt werden.

Nun werden aus jedem Satz die 14 verschiedenen Arten von Wortpaaren extrahiert, indem dessen syntaktische Struktur untersucht wird. Für jede Art von Wortpaar wird eine eigene Datei erstellt. Zusätzlich zu den Wortpaaren mit der vermuteten starken semantischen Beziehung, werden zusätzlich Wortpaare generiert, die zwar den gleichen Wortarten entsprechen, aber syntaktisch keine besondere Beziehung haben. Im Beispiel mit der Präposition und dem Nomen aus der untergeordneten NP, werden also nebst dem gesuchten Wortpaar auch Wortpaare aus allen anderen Präpositionen und Nomen gebildet. Diese zusätzlichen Wortpaare werden negative oder inkorrekte Wortpaare genannt und werden später beim Testen der Kookkurrenzstatistik verwendet. Für diese Wortpaare werden ebenfalls für alle 14 Arten von Wortpaaren Dateien erstellt. Mehr zu den erzeugten Dateien, deren Namen und Aufbau im Anhang.

Sofern keine Fehler bei den annotierten Sätzen im Korpus vorhanden sind, extrahiert das Java-Programm die Wortpaare korrekt und beachtet sogar Koordinationen von zwei oder mehreren Phrasen. Falls erwünscht werden die Wortpaare auch zusammen mit ihren Wortarten ausgegeben. Dabei wird das Kürzel der Wortart, mit einem Punkt “.äls Trennung hinten an die beiden Wörter gehängt.

3.5 Auswertung

Wie schon angetönt, dienen die Wortpaare dazu, die erzeugte Statistik zu bewerten. Die Wortpaare der verschiedenen Sorten dienen damit als Testsets. Die Bewertung der Statistik bezüglich den unterschiedlichen Wortpaararten sollte grafisch sichtbar sein und so entschieden wir uns für die Erstellung von ROC-Kurven, welche im nächsten Abschnitt näher erläutert werden. Dazu brauchen wir korrekte Wortpaare, sogenannte Positives, und inkorrekte Wortpaare, die Negatives. Diese Wortpaare werden nun in der Statistik gesucht und deren Signifikanz mit einem Schwellwert verglichen. Liegt der Wert der Signifikanz über dem Schwellwert und das Wortpaar gehört zu den Positives, so wird ein korrekt erkanntes Wortpaar verzeichnet, ist das Wortpaar ein Negatives, notiert sich das Programm ein inkorrekt erkanntes

Wortpaar. Die ROC-Kurve zeichnet dann das Verhältnis zwischen korrekt erkannten und inkorrekt erkannten Wortpaaren auf. Die ROC-Kurve dient also als grafisches Mittel, um die True Positives Rate und die False Positives Rate zu vergleichen.

3.5.1 ROC-Kurve

Die ROC-Kurve (Receiver Operating Characteristic) ist eine Methode zur Bewertung von Analyseverfahren. Sie zeichnet das Verhältnis von Effizienz und Fehlerrate auf und gibt so Aufschluss über die Qualität unserer Qualifizierung. Die Achsen des Graphes reichen von 0 nach 1. Auf der X-Achse ist die Fehlerrate angegeben, während auf der Y-Achse die True Positives aufgetragen werden. Lässt man nun den zuvor erwähnten Schwellwert zwischen 0 und unendlich variieren, erhält man verschiedene Zuordnungen von korrekt erkannten Wortpaaren und die dabei gemachten Fehler. Diese Zuordnungen werden auf dem ROC-Graph als Punkte repräsentiert. Zum Schluss werden die Punkte zu einer Linie verbunden. Dabei startet die Linie immer im Nullpunkt und endet in der rechten oberen Ecke am Punkt (1,1). Für einen Schwellwert von unendlich gibt es weder Positives noch Negatives, deren Signifikanz über dem Schwellwert liegen. Aus diesem Grund befindet sich der Punkt an der Stelle (0,0) im Graph. Ist der Schwellwert null, geht man davon aus, dass sowohl alle Positives als auch Negatives über dem Schwellwert liegen und somit zwar alle Positives, aber auch die Negatives, als korrekt erkannt werden. Daraus folgt der Punkt an der Stelle (1,1) im Graph.

Da in der Statistik jedoch nicht für jedes Wortpaar ein Eintrag existiert, gibt es immer einige Wortpaare deren Signifikanz nie über dem Schwellwert liegen, weil sie gar keine besitzen. Unsere Kurven endeten also nicht im Punkt (1,1) sondern irgendwo auf der Ebene. Wortpaare, die nicht in der Statistik gefunden wurden, bekamen automatisch den Wert 0 zugeordnet. Aus dieser Änderung folgt nun zwar die erwünschte ROC-Kurve, jedoch ist ein Teil der ROC-Kurve eine Gerade vom Punkt (1,1) zur vorherigen Kurve.

Die Diagonale, welche den Nullpunkt mit dem Punkt (1,1) verbindet, zeigt eine gleichverteilte Klassifizerung, welche man erhält, wenn alle Signifikanzwerte zufällig wären. Das bedeutet, dass Punkte, die unter der Diagonale liegen eine schlechtere Klassifizierung als zufälliges Zuteilen repräsentieren. Je näher die Kurve aber dem Punkt (0,1) kommt, desto besser ist die Bewertung, da dieser Punkt keine falsch erkannten und alle korrekt erkannten Wortpaare repräsentiert. Sehr steile Kurven deuten auf wenig Fehler hin und sind anzustreben.

3.5.2 Distanzfilter

Die Distanz zwischen den beiden Wörtern der Wortpaare ist ebenfalls bekannt und kann genutzt werden, um die Resultate zu verbessern. Dabei

Wortpaar	Genauigkeit	Fenster	Geordnet	Cluster/Wortart
N_P_ppcomp	99,06%	Paar	Nein	-
N_N_ppcomp	98,85%	Paar	Ja	Wortart
V_N_ppcomp	98,62%	Paar	Ja	Wortart
V_P_ppcomp	97,47%	Satz	Nein	-
N_P_ppmod	96,65%	5	Nein	Cluster
N_N_postgen	95,66%	Paar	Ja	Wortart
N_N_ppmod	94,58%	Paar	Ja	Cluster
P_N	92,58%	5	Nein	-
N_ADJ	89,74%	5	Nein	Cluster
V_N_ppmod	87,92%	Paar	Ja	Wortart
V_ADV	83,43%	5	Nein	-
V_ADJ	82,51%	5	Nein	-
V_N_npcomp	82,04%	5	Nein	-
V_P_ppmod	79,44%	5	Nein	Cluster

Tabelle 1: Parameter der gezeigten Kurven

werden alle Wortpaare mit Distanzen, die ausserhalb des vorher definierten Intervalls des Filters liegen, ignoriert. Ist also über eine Wortpaarart zusätzlich die wahrscheinliche Distanz im Satz bekannt, kann die Anwendung eines Distanzfilters Fehler ausmerzen.

4 Resultate

4.1 Überblick

In diesem Kapitel werden die erzeugten ROC-Kurven untersucht und miteinander verglichen. Zuerst werden die besten und die schlechtesten Resultate präsentiert und weiter im Kapitel verschiedene Faktoren, die zu diesen Resultaten geführt haben, genauer betrachtet.

Ein Resultat wird als 'gut' bezeichnet, wenn die Genauigkeit einen hohen Wert hat. Die mathematische Formel für die Genauigkeit lautet wie folgt:

$$ACC = \frac{TP + TN}{P + N} = 1 - \frac{FP + P - TP}{P + N}$$

4.1.1 Genauigkeit als Gütefaktor

Es folgen die ROC-Kurven für jede Wortpaarart, mit Angaben über Fensterbreite, Einhaltung der Reihenfolge, sowie ob Cluster oder Wortarten verwendet wurden.

Dabei fällt auf, dass gerade alle Wortpaare, welche den Kopf einer modifizierenden oder komplementären Präpositionalphrase enthalten oder auch

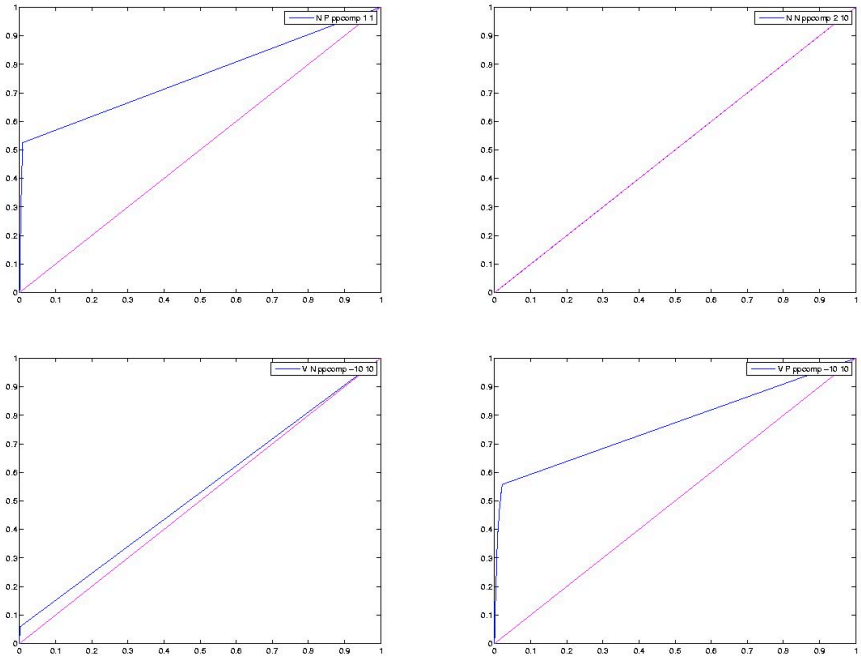

Abbildung 3: ROC-Kurven mit maximaler Genauigkeit

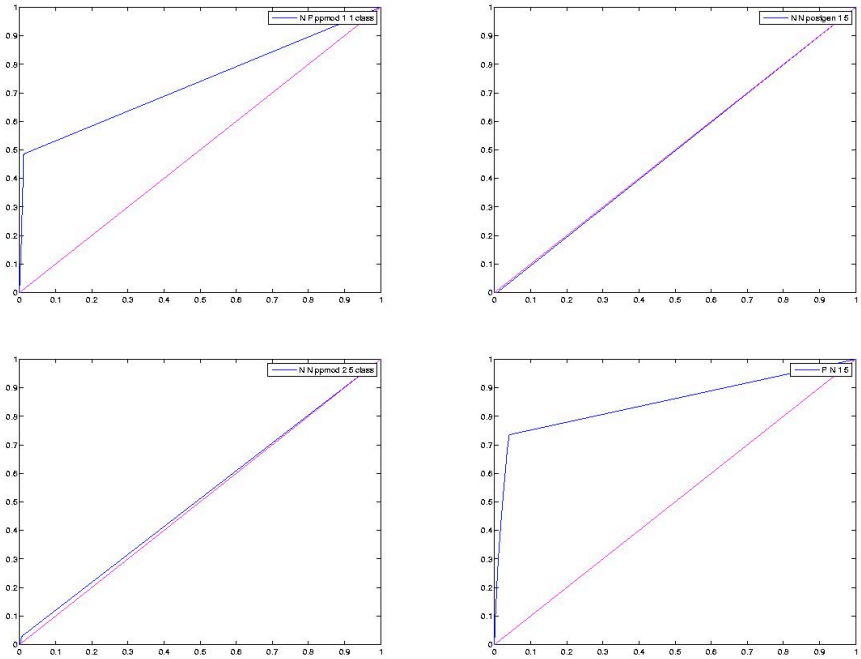

Abbildung 4: ROC-Kurven mit maximaler Genauigkeit

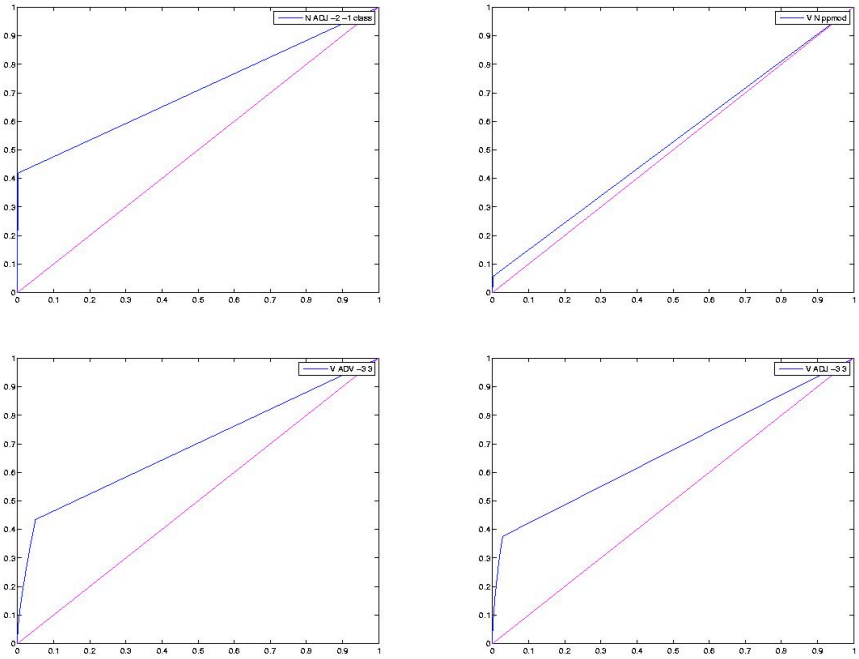

Abbildung 5: ROC-Kurven mit maximaler Genauigkeit

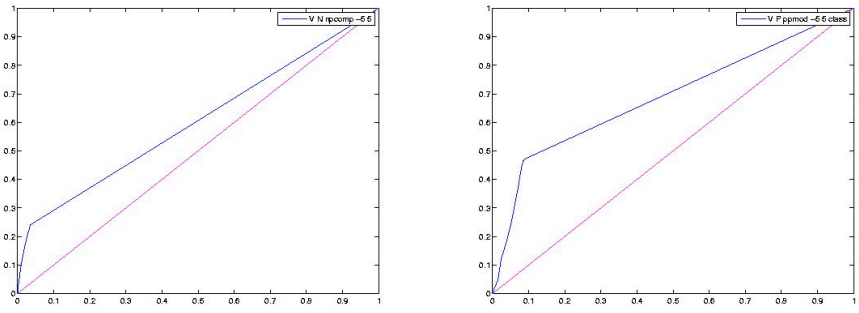

Abbildung 6: ROC-Kurven mit maximaler Genauigkeit

den Kopf eines Postgenitivs, trotz der hohen Genauigkeit sehr wenig korrekt erkannte Paare (True Positives) aufweisen. Betrachtet man die Schwellwerte dieser Wortpaararten, stellt man fest, dass diese im Vergleich zu den anderen ungewöhnlich hoch sind. Das bedeutet, dass die hohe Genauigkeit erst bei einem sehr hohen Schwellwert erreicht wird.

In Anbetracht dessen, haben wir zusätzlich andere Auswertungsmöglichkeiten für die ROC-Kurven gesucht. Wie schon bei der Erläuterung der ROC-Kurve erwähnt, bedeutet der Punkt (0,1) eine optimale Klassifizierung der Wortpaare. Nun lässt sich geometrisch der Abstand zu diesem Punkt berechnen, um dann Kurven mit einem sehr kleinen Abstand zum Optimalpunkt zu suchen. Eine andere Variante wäre die Anzahl der True Positives - also die Anzahl der korrekt erkannten Wortpaare - mit der Genauigkeit zu multiplizieren. So würden sehr genaue Kurven mit wenig True Positives aussortiert und stattdessen weniger genaue Kurven mit vielen True Positives in Betracht gezogen.

Interessanterweise erhält man mit beiden Auswertungsmöglichkeiten beinahe die gleichen Kurven. Wir haben uns für die Erstere entschieden. Es folgt eine Auflistung der neuen Kurven für die problematischen Wortpaararten mit Angaben zu Fensterbreite, Distanzfilter etc.

Alle Kurven stammen von den Daten aus der Statistik, welche mit einer Fensterbreite von 5 Wörtern, mit Clustering und ohne auf die Reihenfolge der Wörter zu achten, erzeugt wurden. Der Anteil an True Positives hat nun zwar zugenommen, doch die Fehlerrate ebenso. Viele inkorrekte Wortpaare werden also fälschlicherweise als richtig anerkannt.

4.1.2 Effekt von Distanzfilter

Wenn sich korrekte und inkorrekte Wortpaare in ihrer Distanz unterscheiden, so können Distanzfilter effektiv angewendet werden um den Anteil der False Positives zu reduzieren. Dazu folgt ein Beispiel aus der Wortpaarart, welche aus den PPs extrahiert wird.

Bei der Anwendung eines Distanzfilters mit Intervall $[1, 5]$ wird die False-Positives-Rate stark verkleinert, während die True-Positives-Rate keine sichtbare Veränderung erfährt. Die Genauigkeit steigt von $64,76\%$ auf $92,58\%$ an.

4.1.3 Effekt von Clustering

Statistiken, die mit Clustering erzeugt wurden, enthalten keine Signifikanzen für Wortpaare, sondern für Klassenpaare. Dabei wird zuerst die Klasse der zu untersuchenden Wörter des Wortpaars ermittelt, um dann die Signifikanz für dieses Klassenpaar zu verwenden. Um den Effekt von Clustering zu illustrieren, zeigen wir die ROC-Kurven der Wortpaarart NNomen - Adjektiv“.

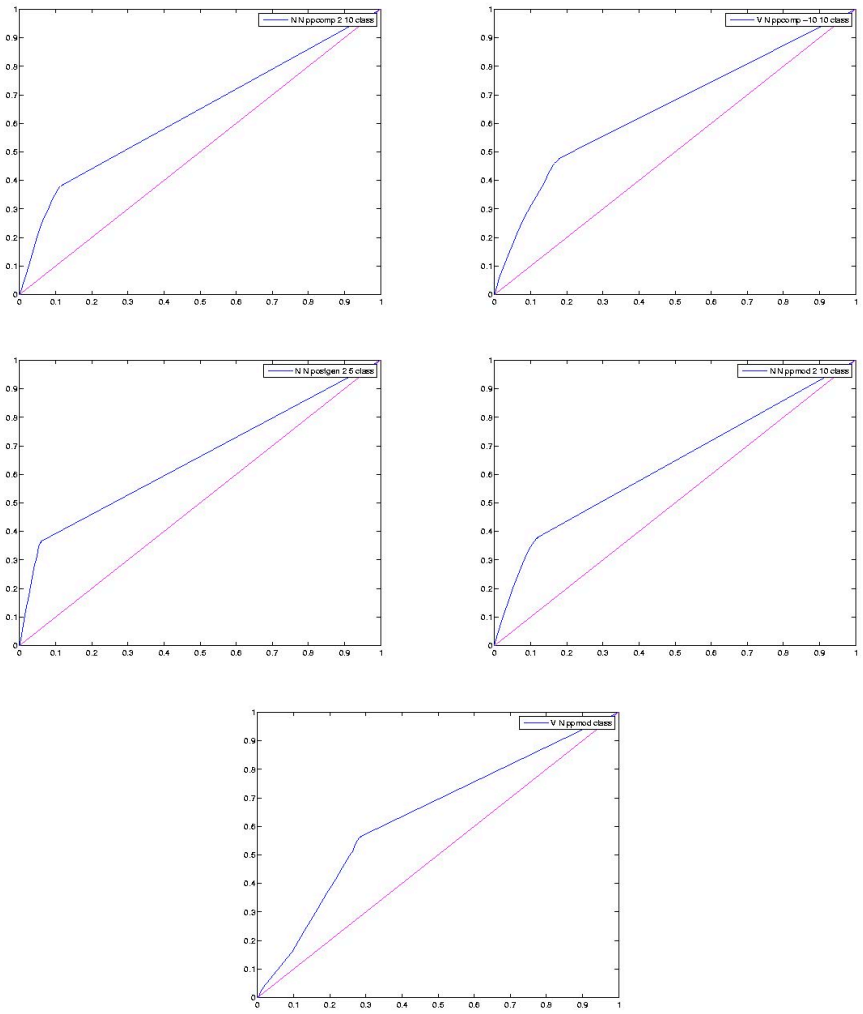

Abbildung 7: ROC-Kurven mit minimalem Abstand zum Optimalpunkt

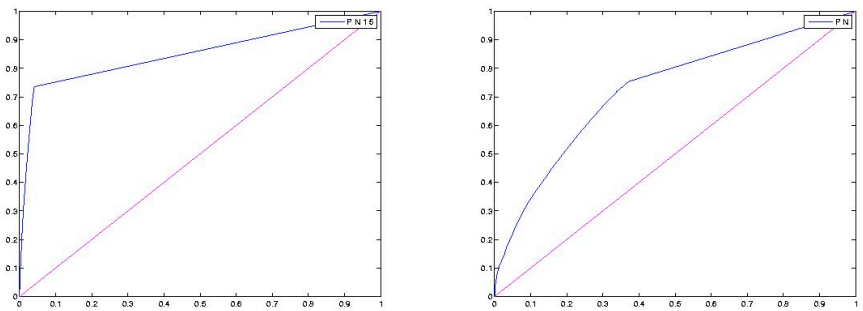

Abbildung 8: Vergleich: mit und ohne Distanzfilter

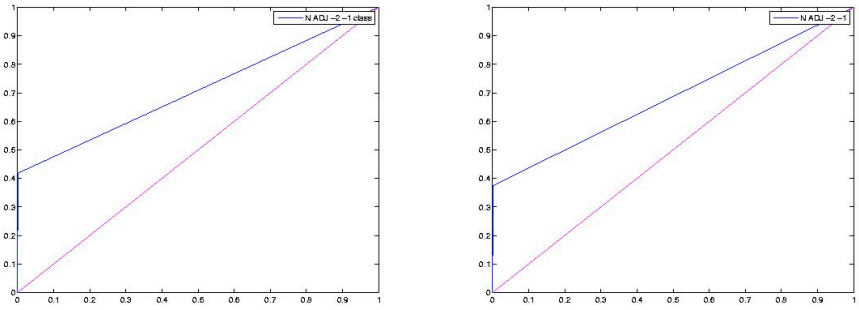

Abbildung 9: Vergleich: mit und ohne Clustering

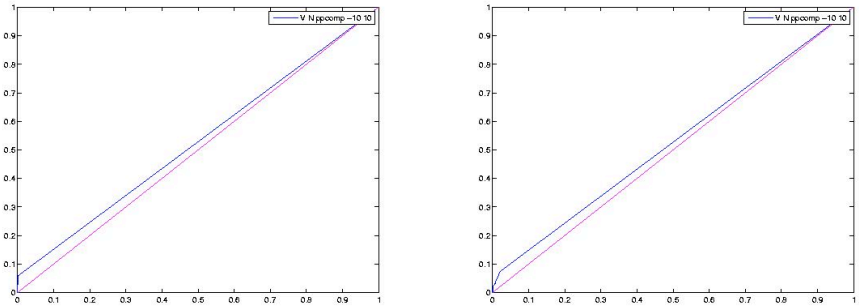

Abbildung 10: Vergleich: mit und ohne Beachtung der Reihenfolge

Der Unterschied ist nicht frappierend, jedoch folgt in diesem Fall aus der Verwendung von Clustering eine leicht höhere True Positives Rate.

4.1.4 Effekt von Beachtung der Reihenfolge

Bei der Erstellung dieser Statistik geht die Information der Reihenfolge der Wörter nicht verloren. Die Signifikanz ist also unterschiedlich, wenn die Wörter eines Wortpaares in umgekehrter Reihenfolge auftreten. Welche Auswirkungen dies auf die ROC-Kurven hat, zeigt das Beispiel mit der Wortpaarart "Verb - Kopf einer komplementären PP":

Die Anzahl True Positives verändert sich nicht, dafür aber jene der False Positives. Offenbar kann die Beachtung der Reihenfolge einige inkorrekte Wortpaare aussortieren, die andernfalls als korrekt erkannt worden wären.

4.1.5 Effekt der Mitberücksichtigung von Wortarten

Manche Wörter werden gleich geschrieben, haben aber unterschiedliche grammatikalische Funktionen. Dies tritt vor allem auf, wenn Gross- und Kleinschreibung ignoriert werden, was in dieser Arbeit der Fall ist. Als Beispiel

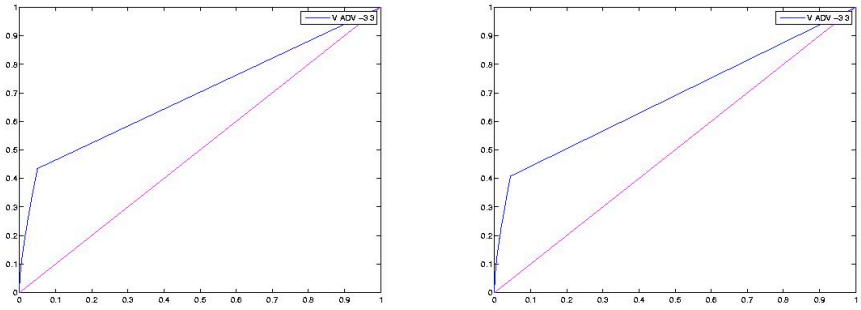

Abbildung 11: Vergleich: mit und ohne Wortarten

verwenden wir die Wortpaarart “Verb - Adverb

Auch hier ist der Effekt nicht gross. Je nach Wortpaarart steigt oder sinkt die True-Positives-Rate. Offenbar gibt es aber nicht allzu viele Wörter, welche auch einer anderen Wortart zugehören können. Die Änderung bei Anwendung dieser Methode ist deshalb relativ gering.

4.2 Fazit

Allgemein hat sich gezeigt, dass bei einer festen Fensterbreite von fünf Wörtern die höchsten True-Positives-Raten zu verzeichnen sind. Dies hat aber bei den Beispielen mit den Köpfen von Präpositionalphrasen auch zur Folge, dass die False Positives Rate schneller ansteigt.

Weiter hat sich bei diesen Beispielen gezeigt, dass hohe Genauigkeit auch einen hohen Schwellwert erfordert und sich dabei auch kleinere True Positives und False Positives Raten ergeben. Ein anderes Mass, welches einen Kompromiss zwischen effektiv korrekt erkannten Wortpaaren und Genauigkeit eingehen würde, wäre zur Auswertung von Vorteil.

Schliesslich hat sich dennoch gezeigt, dass Wortpaararten, welche Präpositionen oder Adjektive als Teil ihrer Wortpaare haben, gut durch die Statistik klassifiziert werden können.

Literatur

[1] 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.

[2] http://www.sgi.com/tech/stl/

[3] Quasthoff, Uwe; Wolff, Christian. The Poisson Collocation Measure and its Applications. In *Proc. International Workshop on Computational Approaches to Collocations*, Vienna, July 2002.

[4] Peter F. Brown , Peter V. deSouza , Robert L. Mercer , Vincent J. Della Pietra , Jenifer C. Lai. Class-based n-gram models of natural language. In *Computational Linguistics*, v.18 n.4, December 1992.

[5] Thorsten Brants. TnT: A Statistical Part-Of-Speech Tagger. In *Proceedings of the sixth conference on Applied natural language processing*, p.224-231, , Seattle, Washington, April-May 2000.

[6] www.ifi.uzh.ch/CL/volk//treebank_course/tiger_annot.pdf

[7] http://www.tigerapi.org/

Dateityp	Grundeinheit	Reihenfolge	Korpus
MEM	sentence	unordered	untagged
STAT	window	ordered	tagged
	pair		clustered

Tabelle 2: Notationsschema der Dateien

5 Appendix A - Dateiformate

5.1 Kookkurrenzstatistiken

Die Bezeichnungen der Dateien folgen der folgenden Regel:

Dabei entstehen die Dateinamen daraus, dass pro Spalte ein Eintrag ausgewählt wird und mit Punkt (nach MEM oder STAT), beziehungsweise underscore getrennt aneinandergereiht wird. Zu den Erklärungen der Einträge in Tabelle 2:

- **MEM/STAT:** kennzeichnet, ob es sich um eine Zählerdatei (MEM) oder eine Signifikanzdatei (STAT) handelt.
- **sentence/window/pair:** kennzeichnet, mit welcher Grundeinheit die Wortpaare gezählt wurden.
- **ordered/unordered:** kennzeichnet, ob die Reihenfolge, in der die Worte auftraten getrennt berücksichtigt wurde.
- **untagged/tagged/clustered:** kennzeichnet ob der unveränderte Korpus (untagged), der um Wortarten ergänzte Korpus (tagged) oder jener Korpus, bei dem die Worte durch ihre Wortklassen ersetzt wurden, verwendet wurde (clustered).

5.1.1 Wortpaar Zähler

Die Dateien mit den Zählern sind wie folgt aufgebaut:

- Anzahl berücksichtigter Wörter
- Anzahl berücksichtigter Grundeinheiten
- Liste der berücksichtigten Worte
- Liste der Einzel-Wort-Zähler
- Liste der Tripel [Index i] [Index j] [Wortpaar-Zähler] (ein Tripel pro Zeile)

5.1.2 Signifikanzstatistiken

Die Dateien mit den Signifikanzstatistiken haben die folgende Form:

WortA | *WortB* -> *Signifikanz*

5.2 Wortpaare aus TIGER-Korpus

Die Dateien, welche die Wortpaare aus dem TIGER-Korpus enthalten, haben folgende Dateinamen:

- N_ADJ: Nomen mit zugehörigem Adjektiv
- N_N_postgen: Nomen mit Kopf-Nomen des postnominalen Genitiv
- N_N_ppcomp: Nomen mit Kopf-Nomen der komplementären Präpositionalphrase (PP)
- N_N_ppmod: Nomen mit Kopf-Nomen der modifizierenden PP
- N_P_ppcomp: Nomen mit Präposition der komplementären PP
- N_P_ppmod: Nomen mit Präposition der modifizierenden PP
- P_N: Präposition mit zugehörigem Nomen
- V_ADJ: Verb mit zugehörigem Adjektiv
- V_ADV: Verb mit zugehörigem Adverb
- V_N_npcomp: Verb mit Kopf-Nomen des Subjekts oder Objekte
- V_N_ppcomp: Verb mit Kopf-Nomen der komplementären PP
- V_N_ppmod: Verb mit Kopf-Nomen der modifizierenden PP
- V_P_ppcomp: Verb mit Präposition der komplementären PP
- V_P_ppmod: Verb mit Präposition der modifizierenden PP

Der erste Teil entspricht den Anfangsbuchstaben der Wortart des ersten Wortes des Wortpaares. Mit Ausnahme von P_N ist das erste Wort gleichzeitig auch der Kopf der untersuchten Phrase. Bei P_N ist das erste Wort die Präposition der PP. Der zweite Teil des Dateinamen entspricht der Wortart des zweiten Wortes des Wortpaars. Der dritte Teil des Dateinamens spezifiziert das zweite Wort des Wortpaares. *ppcomp* heisst in diesem Fall, dass das zweite Wort aus der komplementären Präpositionalphrase stammt.

Dateien, welche die negativen Wortpaare enthalten, haben ein zusätzliches "_negäm Ende des Dateinamens.

In der Datei werden die Wortpaare so präsentiert, dass die Reihenfolge der im Dateinamen angedeuteten entspricht. Der Aufbau ist wie folgt:

- erstes Wort des Wortpaars
- zweites Wort des Wortpaars
- Wortdistanz ($Position(Wort2) - Position(Wort1)$)
- Satz (nur bei positiven Wortpaaren)

Die Elemente stehen auf einer Zeile und sind mit einem Leerzeichen getrennt. Die Leerzeichen im Satz wurden durch einen Unterstrich "_ërsetzt.

5.2.1 Positives- und Negatives-Zähler

Die Dateien, welche Positives und Negatives Zahlen enthalten, sind nach dem gleichen Schema benannt wie die Dateien, welche die Wortpaare enthalten. Zusätzlich stehen zwei Zahlen im Dateinamen oder das Kürzel ïnfüm die Intervallgrenzen des Distanzfilters anzugeben. Fehlen diese Zahlen wurde kein Filter angewendet. Wurden ausserdem Cluster an Stelle von effektiven Wörtern gescannt, steht zusätzlich classïm Dateinamen. Am Schluss wird noch ßtatsängehängt.

Der Aufbau der Datei sieht folgendermassen aus:

- Threshold: Der Grenzwert, der für diese Zeile verwendet wurde.
- True Positives (TP): Anzahl korrekte Wortpaare, die in der Statistik gefunden wurden.
- Postives (P): Anzahl korrekte Wortpaare (in jeder Zeile gleich).
- False Positives (FP): Anzahl inkorrekte Wortpaare, die in der Statistik gefunden wurden.
- Negatives (N): Anzahl inkorrekte Wortpaare (in jeder Zeile gleich).