

Part of Speech Tagging

VON CAND. INF. MARKUS BESTEHORN
Universität Karlsruhe (TH)
Kontakt: Markus.Bestehorn@ira.uka.de

Vier Reiche haben 8 Arme. Acht Arme haben 16 Beine. Also haben Arme mehr Beine als Reiche Arme.

Beim Durchlesen des obigen Satzes müssen die meisten Menschen sich zumindest zweimal überlegen, an welcher Stelle im Satz es sich beim Wort "Arme" um ein Körperteil handelt und an welcher Stelle damit eine Person bzw. Personengruppe gemeint ist.

Die Problematik beim Verstehen des Satzes liegt darin, dass das Wort "Arme" des Satzes nur dann verstanden werden kann, wenn der Kontext, in dem es verwendet wird, beim Verstehen des einzelnen Wortes miteinbezogen werden kann.

Von denjenigen, die sich während der Zeit am Gymnasium mit einer humanistisch geprägten Bildung "herumplagen" mussten, dürften viele die täglichen Übungen aus dem Lateinunterricht kennen, in denen beim Übersetzen eines Satzes zunächst für jedes Wort bestimmt werden musste, zu welchem Satzteil das Wort gehört. Für die Übersetzung des Textes war diese Übung unerlässlich, und ein Fehler in der Übersetzung ließ sich häufig auf einen Fehler in der Satzanalyse, also bei der Zuordnung der Wörter zu Satzteilen, zurückführen.

Vor einem ähnlichen Problem, nämlich einem Übersetzungsproblem, steht die maschinelle Spracherkennung, denn es muss ein menschlicher Text in eine für Maschinen verständliche Form übersetzen. Einer der ersten Schritte, die für ein solches maschinelles Textverständnis notwendig sind, ist wie im Lateinunterricht, die korrekte strukturelle Analyse des Satzes.

Im folgenden wird es nun darum gehen, die verschiedenen Verfahren, die zur Markierung von Satzteilen entwickelt wurden, vorzustellen und deren Arbeitsweise zu verstehen.

Was ist Part-of-Speech Tagging?

Bei der Entwicklung von Computer-Programmen wird ebenfalls ein Quelltext, also der Programm-Code, der z.B. in C++ oder Java verfasst wurde, in eine für den Com-

puter verwendbare Form umgesetzt, die so genannte Maschinensprache. Nachdem ein Programm in Maschinensprache vorliegt kann, der Computer die Anweisungen, die der Programmierer dem Computer gegeben hat, umsetzen und entsprechende Ergebnisse berechnen.

Warum geht das nicht mit menschlicher Sprache?

Computer-Sprachen haben eine Eigenschaft, die menschliche Sprachen im Allgemeinen nicht haben: Sie sind kontextfrei. Das heisst (umgangssprachlich formuliert), dass sie keine Kontextabhängigkeiten oder Doppeldeutigkeiten enthalten. Im Eingangsbeispiel wird gezeigt, dass solche Doppeldeutigkeiten durchaus zu Verwirrung führen können, und im nächsten Beispiel wird deutlich, dass es sogar gänzlich unmöglich werden kann, einen Text zu verstehen, wenn Doppeldeutigkeiten und Kontextabhängigkeiten auftreten:

Er schlug den Mann mit dem Stock.

Wird hier jemand mit einem Stock geschlagen oder wird hier jemand geschlagen, der einen Stock hat?

Da wegen der Irregularität menschlicher Sprache eine Verarbeitung, wie sie bei Computer-Sprachen verwendet wird, auf Grundlage des derzeitigen Forschungsstandes nicht möglich ist, versucht man derzeit, Teilprobleme auf dem Weg zur Lösung des Problems der Sprachverarbeitung zu lösen. Eines dieser Teilprobleme ist die Gewinnung von Strukturinformationen aus dem Text, ohne dass Informationen über den Inhalt des Textes vorhanden sein müssen. Einer der ersten Schritte, analog zur Bestimmung von Satzteilen bei der Übersetzung aus dem Lateinischen, ist das so genannte *Part of Speech Tagging*:

Part of Speech Tagging, im folgenden auch kurz Tagging genannt, ist der Vor-

Tag	Beschreibung	Beispiele
ADJA	Adverb	schon, bald, doch
APPR	Präposition; Zirkumposition links	in [der Stadt], [ihm] zufolge
ARTDEF	Artikel	der, die, das, ein, eine
CARD	Kardinalzahl	zwei [Männer], [im Jahre] 1994
NN	normales Nomen	Frau, Haus, München, Ostern, [das] Gehen
PPER	irreflexives Personalpronomen	ich, er, ihm, mich, dir, sie
VVFIN	finites Verb, voll	[du] gehst, [wir] holen
VVIMP	Imperativ, voll	komm [!], Buchen [Sie]

gang, bei dem jedem Wort in einem Satz ein so genanntes *Part-of-Speech Tag*, also eine (grammatikalische) Markierung, zugeordnet wird.

Ziel des Part of Speech Taggings ist es beispielsweise, aus dem Satz "Buchen Sie den Flug" folgende Markierung der Satzteile zu erhalten (Die Bedeutung der Markierungen wird im Abschnitt Tagsets noch genauer erklärt):

Beispiel: Buchen/VVIMP Sie/PPER den/ARTDEF Flug/NN .S

Die Tags werden im Allgemeinen immer durch einen Slash (/) vom Wort getrennt und direkt an das dazugehörige Wort angefügt. Die hier dargestellten Tags stammen aus dem STTS dem Stuttgart-Tübingen Tagset [STTS95]; eine genauere Einführung bezüglich Tagsets findet sich im folgenden Abschnitt. Neben der oben genannten richtigen Variante könnte man den Satz auch wie folgt taggen:

Beispiel: Buchen/NN Sie/PPER den/ARTDEF Flug/NN.S

In der ersten Variante würde das Wort "Buchen" als ein Verb im Imperativ interpretiert und entsprechend mit einem Tag versehen. Die zweite Möglichkeit markiert das Wort "Buchen" als Plural des Nomens "Buche", was offensichtlich falsch ist, aber verdeutlicht, dass es auch bei solch einfachen, kurzen Sätzen durchaus nicht trivial ist, maschinell die richtige Variante herauszufinden.

Tagsets

Um die Strukturinformationen, die aus dem Part of Speech Tagging gewonnen werden, auch nutzen zu können bzw. die Bedeutung der zugewiesenen Tags festzulegen, werden sogenannte Tagsets benutzt:

Ein Tagset ist eine Menge von Markie-

Tabelle 2: Ausschnitt aus dem Penn Treebank Tagset

POS Tag	Beschreibung	Beispiele
CC	coordinating Conjunction	and
CD	cardinal number	1, third
JJ	adjective	green, strong, bad
NN	Noun, singular or mass	house, snow, salt
NNS	Noun, plural	houses
POS	Possessive ending	friend's
VB	Verb, base form	take, make, draw
VBD	Verb, past tense	took, made

rem, die für verschiedene Satzteile während des Part of Speech Taggings benutzt werden.

Ein Beispiel für ein solches Tagset für deutsche Sprache ist das bereits erwähnte Stuttgart-Tübingen-Tagset [STTS95]. Ein kleiner Ausschnitt, aus den insgesamt 54 verschiedenen Tags, ist in Tabelle 1 zu sehen.

In den folgenden Beispielen wird das STTS als Referenz für das Tagging deutscher Texte benutzt. Da auch einige Beispiele in englischer Sprache vorkommen, soll an dieser Stelle das Tagset vorgestellt werden, das für das Tagging der englischen Beispiele benutzt wurde (siehe Tabelle 2).

Insgesamt beinhaltet das Penn Treebank Tagset [PTTS93] 45 Tags, womit dieses Tagset eines der kleineren englischen Tagsets ist. Tagsets wie das C7-Tagset [C7] benutzen bis zu 146 Tags, aus Gründen der Übersichtlichkeit und um die Beispiele kurz und einfach zu halten, wurde aber das kürzere Penn Treebank Tagset verwendet.

Regelbasierte Tagger

Nachdem im vorangegangenen Abschnitt eine kurze Einführung in die Thematik des Part of Speech Taggings gemacht wurde, wird die erste Klasse von Taggern, die regelbasierten Tagger, vorgestellt. Die Eingabe für einen regelbasierten Tagger besteht im Wesentlichen aus zwei Teilen:

- Einem Text, der keine Tags enthält, also eine Aneinanderreihung von Wörtern bzw. Buchstaben.
- Einem Tagset, ähnlich denen, die in Abschnitt Tagsets vorgestellt wurden.

Wie bereits in dem vorangegangenen Beispiel "Buchen Sie den Flug" gezeigt wurde, ist es selbst bei so einfachen Sätzen nicht trivial, ein korrektes Tagging zu finden. Ein trivialer Ansatz wäre, sich zu überlegen,

welches Tag das Wort "Buchen" in den meisten Fällen hat. Diese Methode wurde von Greene und Rubin [GR71] bereits umgesetzt und führt dazu, dass der Tagger nur eine Genauigkeit von 77% aufweist.

Dafür gibt es mehrere Gründe: Das Hauptproblem beim Taggen ist, dass es zahlreiche Wörter gibt, die mehrere mögliche Verwendungen haben und folglich auch mehrere mögliche Tags. Untersuchungen, zum Beispiel von DeRose [DeRose88], zeigten, dass nur ein relativ geringer Prozentsatz der Wörter im Englischen, bzw. den zur Untersuchung verwendeten Texten, mehr als einen möglichen Tag hat. Problematisch aber ist, dass gerade diese Wörter relativ häufig benutzt werden. In den meisten modernen Sprachen werden bereits bestehende Wörter durch Verwendung in einem neuen Zusammenhang einer weiteren Wortklasse hinzugefügt, so dass zum Beispiel Wörter, die vorher nur als Nomen auftreten konnten, plötzlich in einem Text auch als Verb verwendet werden. Ein Beispiel für eine solche "Verwandlung" im Englischen ist der Satz "He will *web* our work tomorrow."; hier wird ein Wort, welches vor der Verbreitung des Internets wohl kaum als Verb aufgetreten ist, als Verb benutzt.

Eine der Möglichkeiten ein eindeutiges Tagging zu finden und damit doppeldeutige Taggings zu vermeiden, bilden die regelbasierten Tagger, die jetzt genauer vorgestellt werden.

Grundlagen des regelbasierten Part of Speech Taggings

Die ersten regelbasierten Tagger wurden in den 1960er Jahren entwickelt (Klein und Simmons [KleinSimmons63]; Greene und Rubin [GR71]) und hatten alle eine sogenannte 2-Phasen Architektur. In der ersten Phase werden aus einem "Wörterbuch" für jedes Wort in dem zu taggenden Satz alle möglichen Tags gesucht. Diese Liste der möglichen Tags wird dann dem Wort zugewiesen. Bei allen Wörtern, denen eine Liste mit mehr als einem Element zugewiesen

wurde, wird dann durch Anwendung eines komplexen, grammatikalischen Regelsystems versucht, die Liste auf ein Element zu reduzieren, damit das Tagging eindeutig ist.

Spätere Tagger, wie der in einem der folgenden Abschnitte vorgestellte ENGTWOL, der 1995 von Voutilainen [ENGTWOL] veröffentlicht wurde, basieren immer noch auf dieser 2-Phasen Architektur, allerdings sind bei diesen regelbasierten Taggern sowohl das Wörterbuch als auch das Regelsystem wesentlich ausgefeilter. In den nächsten zwei Abschnitten werden daher zunächst die beiden Phasen der regelbasierten Tagger genauer vorgestellt.

Phase 1: Das Wörterbuch und die initiale Annotation

Die zentrale Komponente der ersten Phase des Taggens ist das Wörterbuch. An Hand des folgenden, bereits bekannten, Beispiels soll diese Phase nun verdeutlicht werden: "Buchen Sie den Flug". Ein Ausschnitt aus einem fiktiven und relativ simplen Wörterbuch könnte, wie in Tabelle 3 dargestellt, aussehen.

Tabelle 3: Beispiel für ein einfaches Wörterbuch

Wort	POS Tag
Buchen	VVIMP
Buchen	VVINF
Buchen	NN
den	ARTDEF
den	PRELS
den	PDS
Flug	NN
Sie	PPER

Es ist deutlich sichtbar, dass ein Wort in dem Wörterbuch durchaus mehrere Einträge haben kann. Um nun die erste Phase abzuschließen, würde ein Suchalgorithmus für jedes der Worte aus dem Beispiel-Satz eine Liste anlegen und diese mit den für dieses Wort möglichen Tags füllen. Ein mögliches Ergebnis für das regelbasierte Part-of-Speech Tagging des Beispielsatzes steht in Tabelle 4.

Für die beiden Worte "Flug" und "Sie" ist damit das Tagging abgeschlossen, da die Mengen bereits einelementig sind und daher eine Auflösung von Doppeldeutigkeiten durch Anwendung des Regelsystems nicht notwendig ist. Dies vereinfacht den Prozess

Tabelle 5: Regelanwendung am Beispiel "Buchen Sie den Flug"

INPUT: \dem\ , \den\ if ((+1 NN) && (-1PPER)) then removeAll (NON ARTDEF-Tags) else remove (ARTDEF-Tag)	INPUT: \Verb\ if ((-1 Satzgrenze) && (+1PPER)) then removeAll (NON VVIMP-Tags) else remove (VVIMP-Tag)
--	---

Tabelle 4: Ergebnis der ersten Phasen für "Buchen Sie den Flug"

Buchen	{VVIMP, VVIN, NN}
Sie	{PPER}
den	{ARTDEF, PRELS, PDS}
Flug	{NN}

der Regelanwendung im folgenden Abschnitt, denn es kann davon ausgegangen werden, dass diese beiden Tags richtig sind.

Phase 2: Die Anwendung von Regeln zur Auflösung von Doppeldeutigkeiten

Die zweite Phase zur Eliminierung von Doppeldeutigkeiten innerhalb des Taggings wird nur dann benutzt, wenn nicht jedes Wort bereits in der ersten Phase ein eindeutiges Tagging erhalten hat, also die Liste der Tags aus dem Wörterbuch mehrdeutig ist. In dem Beispiel aus dem vorherigen Abschnitt Phase 1 gibt es noch zwei Wörter, denen eine Liste mit mehr als einem Element zugeordnet wurde. In Systemen, wie sie heute verwendet werden, kommen an dieser Stelle die Regelsysteme mit bis zu 1100 verschiedenen, teilweise recht komplexen Regeln zum Einsatz. Für das Beispiel seien zwei einfache Regeln definiert, die "zufälligerweise" recht gut passen (vgl. Tabelle 5).¹

Die linke der beiden Regeln bewirkt, dass bei einem Wort "den" geprüft wird, ob dem Wort danach ein Tag NN zugeordnet wurde und sich davor ein Wort befindet, dass mit PPER markiert wurde. Wenn beide Bedingungen wahr sind, dann wird jeder Tag aus der Tag-Liste entfernt, außer dem ARTDEF Tag, das heisst diese Regel findet heraus, ob es sich um einen Artikel handelt oder eben nicht. Sollte eine der Bedingungen nicht wahr sein, dann kann der ARTDEF Tag entfernt werden, weil es sich dann sicher um keinen Artikel handelt. Diese Art der Regelanwendung nennt man *negative Regelanwendung*.

Ähnlich wie bei der linken Regel verhält

¹ Es geht hier um die Anwendung der Regeln, nicht um deren grammatikalische Richtigkeit. Die beiden Regeln sind nur für dieses Beispiel sinnvoll und für die allgemeine Anwendung wahrscheinlich nicht geeignet.

es sich auch bei der rechten: Der Unterschied ist, dass hier Verben im allgemeinen als Eingabe verwendet werden dürfen. Die Regel besagt, dass Worte, in deren Tag-Menge ein Verb-Tag vorkommt, die am Anfang eines Satzes stehen und die von einem Personalpronomen (PPER) gefolgt werden, Verben im Imperativ sind und daher mit dem Tag VVIMP markiert werden müssen.

Der ENGTWOL Tagger

Wie bereits in den vorhergehenden Abschnitten erwähnt, sind echte regelbasierte Systeme wesentlich komplexer, als das bisher vorgestellt wurde. Um einen kurzen Einblick in ein solches System zu geben, wird nun an Hand des ENGTWOL Taggers [ENGTWOL] ein weiteres Beispiel mit Part-of-Speech Tags versehen:

Beispiel: It is expected that Mr. Montoya will race tomorrow.

Eine wesentliche Erweiterung des ENGTWOL Taggers gegenüber den älteren Systemen sind die Struktur und der Inhalt des Wörterbuches. Neben Stammformen für die meisten Verben werden auch Wortstämme von Substantiven sowie morphologische und syntaktische Informationen in so genannten "Additional Part of Speech Features" gespeichert. Ein Ausschnitt aus dem Wörterbuch des ENGTWOL Taggers könnte daher so aussehen wie in Tabelle 6.

Es ist deutlich sichtbar, dass versucht wird, neben dem POS-Tag zu jedem Wort eine Reihe anderer Informationen, wie zum Beispiel die Stammform, zu bestimmen und für das Tagging zur Verfügung zu stellen. Außerdem werden Markierer verwendet um anzuzeigen,

Tabelle 6: Ausschnitt aus dem Wörterbuch des ENGTWOL Taggers

Word	POS	Additional POS features
it	PRON	"it" NonMod ACC SG3
it	PRON	"it" NonMod ACC SG3 SUBJ @SUBJ
is	V	"be" SV SVC/N SVC/A PRES SG3 VFIN
expected	V	"expect" Vcog SVO P/of PAST VFIN @+FMAINV
expected	PCP2	"expect" Vcog SVO P/of
that	ADV	"that" AD-A @AD-A
that	PRON	"that" NonMod **CLB Rel SG/PL
that	PRON	"that" PRON DEM SG
that	DET	"that" CENTRAL DEM SG @DN
that	CS	"that" **CLB @CS
to	INFMARK	"to" @INFMARK
to	PREP	"to"
race	N	"race" NOM SG
race	V	"race" SV SVO INF

dass Wörter in bestimmten Umgebungen besonders oft einen bestimmten Tag haben, zum Beispiel zeigt das POS-Feature Vcog bei "expected", dass dieses Wort besonders häufig in einem Satz vor "that" steht. POS-Features wie Vcog können dann beim Tagging benutzt werden, dass heißt, wenn man auf ein Wort mit Vcog stößt und danach ein "that" auftaucht, dann kann man alle Tags aus der Liste entfernen, die das POS-Feature Vcog nicht enthalten. Da auch der ENGTWOL Tagger auf der 2-Phasen Architektur basiert, ist es nicht verwunderlich, dass das Ergebnis der ersten Phase sich analog zu dem des vorhergehenden Beispiels verhält, wie man in Tabelle 7 sehen kann.

Durch die Erweiterung des Wörterbuches um die Additional POS features sind nun wesentlich feinere Regeln möglich, als das bei den früheren Systemen der Fall war. Nach der 1. Phase wird nun analog zu dem vorhergehenden Beispiel durch Anwendung von Regeln ein eindeutiges Tagging erzeugt.

Die Qualität der Ergebnisse des ENGTWOL Taggers sind hauptsächlich aus diesem Grund besser als bei den älteren Systemen. Zusätzlich zu der Erweiterung des Wörterbuches werden zur Auflösung von Doppeldeutigkeiten auch probabilistische Annahmen und andere syntaktische Informationen verwendet, die aber hier nicht weiter vertieft werden.

Evaluierung: Regelbasierte Tagger

Im Gegensatz zu den im Folgenden vorgestellten stochastischen Taggern benötigen

Tabelle 7: Ergebnis der Phase 1 bei ENGTWOL

Word	POS-Tags
It	{PRON, "it" NonMod ACC SG3}, {V, "be" SV SVC/N SVC/A PRES SG3 VFIN}
is	{V, "be" SV SVC/N SVC/A PRES SG3 VFIN}
expected	{V, "expect" Vcog SVO P/of PAST VFIN @+FMAINV}, {PCP2, "expect" Vcog SVO P/of}
that	{DET, "that" CENTRAL DEM SG @DN}, {CS, "that" **CLB @CS}, {ADV, "that" AD-A @AD-A}, {PRON, "that" PRON DEM SG}, {PRON, "that" NonMod **CLB Rel SG/PL}
Mr.	{ABBR, "mr" Title NOM SG}
Montoya	{N, "Montoya" proper NOM SG}
will	{V, "will" AUXMOD VFIN @+FAUXV}, {N, "will" NOM SG}
race	{N, "race" NOM SG}, {V, "race" SV SVO INF}
tomorrow	{N, "tomorrow" NOM SG}, {ADV, "tomorrow" ADVL @ADVL}

regelbasierte Tagger keinen manuell annotierten Trainingskorpus². Ein Nachteil ist jedoch, dass es einen enormen Aufwand bedeutet, ein so komplexes Regelsystem aufzustellen, wie es z.B. der ENGTWOL Tagger benutzt. Um qualitativ hochwertige Ergebnisse zu erzielen, benötigt man zudem Sprachwissenschaftler und andere Spezialisten, die dann die Regeln aufstellen und testen. Ein weiterer Nachteil ist, dass die Benutzung von existierenden Regeln einer Sprache (z.B. Englisch) für das Tagging einer anderen Sprache (z.B. Deutsch) so gut wie unmöglich ist, so dass für jede Sprache ein komplett neues Regelsystem erzeugt werden muss.

Da man versuchen wollte, diesen enorm hohen materiellen und intellektuellen Aufwand zur Erstellung der Regeln zu vermeiden, versuchten andere Forscher einen gänzlich anderen Ansatz zu verfolgen und entwickelten die *Stochastischen Tagger*.

Stochastische Tagger

Die wichtige Idee hinter dem stochastischen Part of Speech Tagging ist, dass es bestimmte Kombinationen von Wörtern und Tags gibt, die besonders häufig sind und wiederum andere Kombinationen eher un-

² Was das genau heisst wird in den folgenden Abschnitten erklärt.

wahrscheinlich sind. Es leuchtet zum Beispiel ein, dass es sehr wahrscheinlich ist, dass ein Artikel vor einem Nomen steht ("das Haus"), und im Gegensatz dazu die Wahrscheinlichkeit, dass ein Artikel vor einem Verb steht, eher gering ist. Analog dazu dürfte es jedem einleuchten, dass die Wahrscheinlichkeit, dass das Wort "Buchen" (vgl. Beispiel) als Verb benutzt wird, wesentlich größer ist als die Wahrscheinlichkeit, dass es als Nomen zur Beschreibung einer Baumart oder Baumgruppe benutzt wird. Die stochastischen Part of Speech Tagger basieren genau auf dieser Eigenschaft und versuchen, für einen Satz die wahrscheinlichste Tagging-Variante zu finden.

Mathematische Grundlagen

Für die Anwendung im Bereich des stochastischen Part of Speech Tagging werden im wesentlichen drei bedingte Wahrscheinlichkeiten immer wieder benutzt:

- $P(t_n|t_1, \dots, t_{n-1})$: Die Wahrscheinlichkeit, dass ein Wort mit dem Tag t_n markiert wird unter der Voraussetzung, dass die vorherigen $n-1$ Worte mit den Tags t_1 bis t_{n-1} markiert wurden.
- $P(Wort|t)$: Die Wahrscheinlichkeit, dass ein Wort mit einem bestimmten Tag t markiert wird.
- $P(t|START)$: Die Wahrscheinlichkeit, dass ein bestimmter Tag t am Anfang eines Satzes steht.

Die meisten stochastischen Part of Speech Tagger benutzen die *Markov Annahme*:

Sei $X = \{X_1, X_2, \dots, X_j\}$ eine Kette von Zufallsvariablen, die Werte aus einem Wertebereich $S = \{s_1, s_2, \dots, s_j\}$ annehmen können. Wenn für die Vorhersage des Wertes der Variable X_{t+1} die Markov Eigenschaften

- Lokalität:
 $P(X_{t+1} = s_k | X_1, \dots, X_t) = P(X_{t+1} = s_k | X_t)$
- Zeitliche Invarianz:
 $P(X_{t+1} = s_k | X_1, \dots, X_t) = P(X_2 = s_k | X_1)$

gelten, dann wird X als *Markov Kette* bezeichnet.

Stochastische Part of Speech Tagger (genauer: Bigramm-Tagger), die von dieser Annahme ausgehen, benutzen also nur den Tag des vorhergehenden Wortes, um den Tag des aktuellen Wortes zu bestimmen. Zunächst mag diese Annahme relativ radikal erscheinen, aber es hat sich gezeigt, dass diese Annahme relativ gute Ergebnisse (~ 90% korrekt markierte Wörter) erzielt. Eine Aufweichung der Annahme dahingehend, dass der Tag des aktuellen Wortes von den beiden vorhergehenden Tags abhängt (Trigramm-Tagger) liefert zwar bessere Ergebnisse, aber der Aufwand, für die Berechnung des Taggings steigt dramatisch an, wie im Folgenden noch deutlich wird.

Training stochastischer Tagger

Zuvor wurde bereits angedeutet, dass es sich beim dem Wort "Buchen" deutlich wahrscheinlicher um ein Verb als um ein Nomen handelt. Die Frage, die sich an dieser Stelle aufdrängt ist, woher die Wahrscheinlichkeiten für das Tagging überhaupt kommen. Woher "weiß" ein Tagger, welches Wort in welchem Zusammenhang wahrscheinlich ein Nomen ist?

Zur Erzeugung einer solchen Wissensbasis werden große, von Hand annotierte Textkorpora benutzt, also Sammlungen von Texten, die von Hand mit den grammatisch richtigen Part of Speech Tags versehen wurden. Es gibt eine ganze Reihe von solchen Textkorpora, von denen hier aber lediglich zwei vorgestellt werden, damit man einen Eindruck vom Aufwand bekommt, der für diesen Schritt des stochastischen Part of Speech Taggings notwendig ist:

- NEGRA: Korpus mit mehr als 20.000 Sätzen (ca. 350.000 Wörter) in deutscher Sprache der Frankfurter Rundschau.[NEGRA]
- Brown Korpus: Mehr als 1 Mio. Wörter aus ca. 500 Texten aus verschiedenen Bereichen (z.B. Zeitungen, wissenschaftliche Arbeiten, Anleitungen...). [PTTS93]

Das Training für einen stochastischen Part of Speech Tagger ist im Prinzip nichts anderes als ein Durchlaufen des Textkorpus und Zählen von Häufigkeiten. Es wird zum Beispiel gezählt, wie oft ein Wort mit einem Tag für Adjektiv vor einem Noen

steht, wie oft ein bestimmter Tag, z.B. für Nomen, ein Wort am Anfang eines Satzes annotiert und wie oft ein bestimmtes Wort (z.B. "Buchen") mit einem bestimmten Tag versehen wird.

Aus der Zählung können dann Wahrscheinlichkeiten berechnet werden, die dann in die Matrizen des Hidden Markov Modells eingetragen werden.

Part of Speech Tagging mit dem Hidden Markov Modell

Nachdem die mathematischen Grundlagen und das "Erlernen" von Wahrscheinlichkeiten vorgestellt wurden, geht es in diesem Abschnitt darum, mit Hilfe des Hidden Markov Modells für einen gegebenen Text ein Part of Speech Tagging zu erhalten.

Der zu taggende Satz ist "Buchen Sie den Flug". Damit ist die Menge der Beobachtungen $O = \{o_1 = \text{Buchen}, o_2 = \text{Sie}, o_3 = \text{den}, o_4 = \text{Flug}\}$. Die Matrizen A, B und Π finden sich in Tabelle 8.

Mit Hilfe der Matrizen soll nun für die gegebenen Beobachtungen ein Tagging erzeugt werden, welches die größte mögliche Wahrscheinlichkeit hat.

Berechnung der Wahrscheinlichkeiten

Allgemein berechnet sich die Wahrscheinlichkeit eines Taggings nach Formel 1:

$$\frac{\pi_1 \times b_{11} \times a_{11} \times b_{12} \times a_{12} \times b_{13} \times a_{13} \times \dots \times b_{1n} \times a_{1n}}{1. \text{ Wort} \quad 2. \text{ Wort} \quad 3. \text{ Wort} \quad \dots \quad n\text{-tes Wort}}$$

Für die Berechnung des Taggings, welches die größte Wahrscheinlichkeit hat, gibt es nun verschiedene Varianten, die nachfolgend beschrieben werden:

► **Brute-Force:** Es wird versucht, alle Kombinationen von Tags für alle Wörter durchzuprobieren und das wahrscheinlichste auszuwählen. Das Problem hier ist, dass bei diesem Verfahren schon für den Satz "Buchen Sie den Flug" mit 4 Worten und dem "Tagset" (Vgl. Tabelle 8) mit 5 Tags $5^4 = 625$

Tabelle 8: Die Matrizen A, B und P für das Beispiel "Buchen Sie den Flug"

	NN	VVIMP	ARTDEF	PPER	PDS	Buchen	Sie	den	Flug	Π
NN	0,1	0	0,3	0,4	0,2	0,2	0	0	1	0,2
VVIMP	0	0	0,2	0,4	0,4	0,8	0	0	0	0,3
ARTDEF	0,4	0,1	0	0,2	0,3	0	0	0,7	0	0,2
PPER	0,1	0,1	0,1	0,3	0,4	0	1	0	0	0,1
PDS	0,1	0,2	0,2	0,3	0,2	0	0	0,3	T	0,2

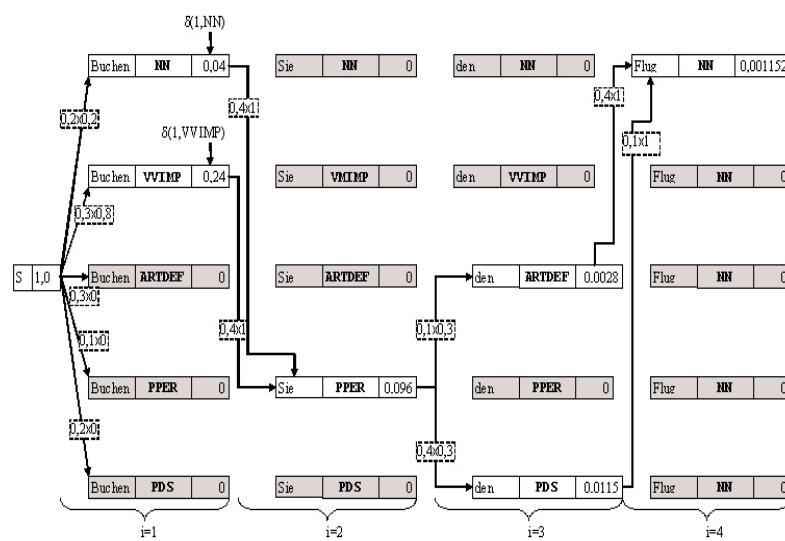


Abbildung 1: Graph am Ende der zweiten Phase des Viterbi-Algorithmus

Möglichkeiten zu prüfen sind. Bei einem normalen Satz mit ca. 20 Worten und einem Tagset, wie dem STTS mit ca. 50 Tags würde das dazu führen, dass $54^{20} (\sim 4 \times 10^{34})$ Möglichkeiten zu überprüfen sind.

► **Dynamische Programmierung:** Die Berechnung des Ergebnisses erfolgt hierbei schrittweise und es werden nur Berechnungen fortgesetzt, die einen Beitrag zum Ergebnis leisten könnten. Wichtig ist hier, dass die Berechnung der Wahrscheinlichkeit für ein Tagging nur Multiplikationen enthält, was dazu führt, dass eine Berechnung automatisch abgebrochen werden kann, wenn eines der Elemente darin gleich 0 ist. Diese Eigenschaft wird durch den Viterbi Algorithmus, der im Folgenden vorgestellt wird, ausgenutzt.

Der Viterbi-Algorithmus

Der Viterbi-Algorithmus besteht im wesentlichen aus drei Phasen und baut einen Graphen auf, auf dessen Grundlage dann das wahrscheinlichste Tagging bestimmt werden kann (vgl. Abbildung 1):

Initialisierung: In dieser Phase wird ein Startknoten S erzeugt.

Aufbauen des Graphen: Mittels Induktion

wird hier ein Graph aufgebaut, aus dem das wahrscheinlichste Tagging abgelesen werden kann. Die Knoten des Graphen enthalten die Wahrscheinlichkeit eines Taggings, welches durch den Pfad vom Startknoten zu diesem Knoten repräsentiert wird, und Verweise auf den jeweiligen Vorgängerknoten.

Auslesen des wahrscheinlichsten Taggings: Unter Benutzung des Verweises auf den Vorgänger-Knoten wird ausgehend vom wahrscheinlichsten Endknoten die Belegung bestimmt, die die höchste Wahrscheinlichkeit hat.

Jeder Knoten des Graphen entspricht einem Wort und einem dazugehörigen Tag, z.B. {"Buchen", NN}. Das Aufbauen des Graphen erfolgt im wesentlichen mit zwei Funktionen: Die Funktion $\delta(i,t)$ berechnet die Wahrscheinlichkeit, dass das i-te Wort mit dem Tag t versehen wird. Die Berechnung erfolgt genau so, wie in Formel 1 beschrieben. Die zweite Funktion $\psi(i,t)$ berechnet für das i-te Wort mit dem Tag t den Vorgängerknoten.

Phase 1: Initialisierung

Bei der Initialisierung wird nur ein Startknoten S erzeugt. Außerdem wird diesem Knoten $\delta(0, \text{START})=1$ zugewiesen.

Phase 2: Aufbau des Graphen

Hier werden für jedes Wort i und jeden Tag im Tagset t die beiden Funktionen $\delta(i,t)$ und $\psi(i,t)$ berechnet. Für das Beispiel "Buchen Sie den Flug" wird damit der Graph wie folgt aufgebaut:

Nach dem durch den Initialisierungsschritt der Knoten S erzeugt wurde und

$\delta(0, \text{START}) = 0$ gesetzt wurde, wird für jeden Tag t ein Knoten {"Buchen", t} erzeugt. Am Beispiel des Knotens {"Buchen", NN} sollen die nachfolgenden Schritte verdeutlicht werden:

1. Es wird $\psi(1, \text{NN}) = S$ gesetzt, d.h. der Vorgängerknoten von {"Buchen", NN} ist S.

2. Um den Wert der Wahrscheinlichkeitsfunktion δ zu berechnen werden die folgenden Werte aus den Matrizen benötigt:

- a) Der Wert $P(\text{Buchen}|\text{NN}) = b_{11} = 0,2$, der die Wahrscheinlichkeit angibt, mit der das Wort "Buchen" mit NN markiert wird.
- b) Der Wert $P(\text{NN}|\text{START}) = \Pi_1 = 0,2$, der die Wahrscheinlichkeit angibt, mit der ein Nomen (NN) am Anfang eines Satzes steht.

Durch die Multiplikation der beiden Werte $\Pi_1 \leftrightarrow b_{11} = 0,04$ kann nun der endgültige Funktionswert von $\delta(1, \text{NN})$ berechnet werden. Dieser ergibt sich aus dem Produkt des eben berechneten Wertes und dem Wert von $\delta(0, \psi(1, \text{NN})) = 1,0$, also dem δ -Wert des Vorgängerknotens.

Die vorgestellten Schritte werden für jeden Tag t durchgeführt, wobei die Knoten, deren δ -Wert gleich 0 ist, nicht weiter bearbeitet werden müssen (in Abbildung 1 sind das die grauen Knoten). Am Ende gibt es zwei Knoten, deren δ -Wert ungleich 0 ist, nämlich {"Buchen", NN} und {"Buchen", VVIMP}. Diese beiden Knoten sind nun die einzigen beiden Knoten, die weiter bearbeitet werden müssen. Am Beispiel des Knotens {"Sie", PPER} wird nun gezeigt, wie der Graph weiter aufgebaut wird, bis alle Knoten expandiert sind:

1. Es gibt zwei Möglichkeitenzum Endknoten zu gelangen, die eine ist der Pfad über den Knoten {"Buchen", NN}, die andere Variante wäre über den Knoten {"Buchen", VVIMP}. Es müssen also die δ -Werte für beide Varianten errechnet werden:

a) {"Buchen", NN}: Die Wahrscheinlichkeit $P(\text{Sie}|\text{PPER}) = b_{42} = 1$ wird multipliziert mit $P(\text{ARTDEF}|\text{NN}) = a_{13} = 0,4$, was einen Wert von 0,4 ergibt. Dieser Wert wird nun mit $\delta(1, \text{NN}) = 0,04$ multipliziert, was 0,016 ergibt.

b) {"Buchen", VVIMP}: Hier wird analog zu oben verfahren, indem zunächst das Produkt $P(\text{Sie}|\text{PPER}) \leftrightarrow P(\text{PPER}|\text{VVIMP}) = b_{42} \leftrightarrow a_{23} = 0,4$ gebildet wird. Das Ganze wird nun mit dem δ -Wert des Vorgängerknotens 0,24 multipliziert, womit $\delta(2, \text{VVIMP}) = 0,096$ gilt.

2. Die anderen Vorgänger-Knoten müssen nicht miteinbezogen werden, da hier sich eine Multiplikation mit 0 ergeben würde.

3. Aus den beiden Varianten wird nun diejenige ausgewählt, die die größere Wahrscheinlichkeit hat, also letztere. Für den Knoten {"Sie", PPER} gilt damit $\delta(2, \text{PPER}) = 0,096$ und $\psi(2, \text{PPER}) = \text{"Buchen", VVIMP}$.

Wie beschrieben werden die restlichen Knoten für die restlichen Wörter aufgebaut. Wenn der Graph (vgl. Abbildung 1) fertig ist, kann in der dritten Phase des Algorithmus ein Pfad ausgelesen werden, der das Tagging mit der größten Wahrscheinlichkeit repräsentiert.

Phase 3: Auslesen des Part of Speech Taggings

Aus den Knoten des letzten Schrittes beim Graphaufbau, also im Beispiel mit i=4, wird nun derjenige ausgewählt, der die größte Wahrscheinlichkeit hat, also den größten δ -Wert hat. Für das Beispiel wäre das der Knoten {"Flug", NN} mit $\delta(4, \text{NN}) = 0,001152$. Von diesem Knoten aus kann nun solange mit Hilfe der ψ -Funktion immer der Vorgänger-Knoten ermittelt werden, bis der Startknoten S erreicht ist. Für jeden Knoten werden also die folgenden Schritte durchgeführt:

- 1. Berechne den Vorgänger des aktuellen Knotens mit der ψ -Funktion.
- 2. Speichere den berechneten Knoten in einer FIFO-Warteschlange (First in – First out) ab.

Wie zu erwarten, ergibt sich dadurch das gewünschte Tagging des Satzes "Buchen Sie den Flug": Buchen/VVIMP Sie/PPER den/ARTDEF Flug/NN

Evaluierung: Stochastische Tagger

Der Vorteil der stochastischen Tagger gegenüber den regelbasierten Tagger ist, dass das Aufstellen komplexer Regelsysteme zur Elimination von Doppeldeutigkeiten im Tagging wegfällt. Dieser Vorteil wird aber recht schnell relativiert, wenn man bedenkt, dass für die Berechnung der Wahrscheinlichkeiten ein recht großer Trainingskorpus vorhanden sein muss, der von Hand annotiert werden muss. Die Qualität des Taggings hängt daher auch wesentlich mit der des Trainingskorpus zusammen. Ein weiterer, allerdings kleiner Nachteil ergibt sich

aus der Tatsache, dass die für die Repräsentation der Wahrscheinlichkeiten relativ große Matrizen notwendig sind: Bei einem Tagset wie dem STTS ist das, wenn es sich um einen echten HMM-Tagger handelt, alleine für die Matrix A eine $54 \leftrightarrow 54$ Matrix. Vor allem aber die Matrix B kann extrem groß werden, denn ihre Größe hängt von der Anzahl der verschiedenen Wörter ab, die im Trainingskorpus verwendet wurden. Beide Arten von Taggern haben Vor- und vor allem gravierende Nachteile. Deswegen hat Eric Brill einen weiteren Tagger entworfen, der versucht von beiden Seiten die Vorteile zu benutzen und die Nachteile zu umgehen. Das Ergebnis dieser Arbeit ist der sogenannte Brill Tagger und dieser wird im nächsten Abschnitt vorgestellt.

Der Brill Tagger

Grundlage des von Eric Brill entwickelten Brill Taggers ist das sogenannte Transformation Based Learning (TBL) [brill92simple], welches ebenfalls von Eric Brill eingeführt wurde. Transformation Based Tagging hat sowohl regelbasierte Anteile als auch stochastische Anteile.

In Anlehnung an die regelbasierten Tagger hat auch der Brill Tagger Regeln, die zur Korrektur von falsch gesetzten Tags und für das Taggen von unbekanntem Wörtern benutzt werden. Analog zu den stochastischen Taggern müssen Wahrscheinlichkeiten gelernt werden oder anderweitig vor der Anwendung vorhanden sein.

Ein Brill-Tagger hat mehrere verschiedene Komponenten:

- **Lexikon:** Dies ist die Komponente, die analog zu den Wörterbüchern der anderen beiden Tagger-Arten aufgebaut ist.
- **Transformation Rules (TR):** Ähnlich wie bei regelbasierten Taggern werden diese Regeln zur Korrektur von Tagging-Fehlern benutzt. Durch Benutzung des Transformation Based Learning können die Regeln jedoch automatisch "gelernt" werden und müssen nicht manuell definiert werden.
- **Lexical Rules (LR):** Regelwerk, welches vor allem mit Hilfe von morphologischen Regeln versucht, unbekanntem Wörtern den richtigen Tag zuzuweisen.
- **Bigramme:** Zerlegung des zu taggenden Satzes in Wortpaare, die dann im wesentlichen von den Lexical Rules benutzt werden.

Die einzelnen Komponenten werden jetzt im Detail vorgestellt.

DAS LEXIKON

Wie bereits angemerkt wurde, hat das Lexikon eine ähnliche Struktur, wie das Wörterbuch bei den regelbasierten Taggern: Jedem Wort wird eine Folge möglicher Tags zugeordnet. Der einzige Unterschied ist, dass die Tags zu einem Wort nach Wahrscheinlichkeit absteigend sortiert in dieser Folge vorkommen (siehe Tabelle 10).

Für das Taggen wird dann später zunächst jedem Wort der wahrscheinlichste Tag zu diesem Wort zugeordnet.

TRANSFORMATION RULES

Da für das Taggen zunächst nur der wahrscheinlichste Tag des Wortes benutzt wird, entstehen offensichtlich Fehler:

1. "Buchen/VVIMP Sie/PPER den/ARTDEF Flug/NN."
2. "Die/ARTDEF Buchen/VVIMP sind/VVAMP groß/ADJ."

Im zweiten Satz ist das Wort "Buchen" offensichtlich kein Imperativ und daher ist das Tagging falsch. In Anlehnung an die regelbasierten Tagger werden nun Regeln benutzt, um solche Fehler zu vermeiden. Der wesentliche Unterschied ist, dass die Regeln beim Brill Tagger automatisch gelernt werden können, wie nachfolgend vorgestellt.

Wie oben bereits beispielhaft vorgeführt wurde, wird zunächst ein Trainingskorpus so markiert, dass immer der wahrscheinlichste Tag für das entsprechende Wort benutzt wird. Dadurch entstehen Fehler, die durch den Vergleich mit dem manuell annotierten Trainingskorpus gefunden werden können. Diese Fehler werden dann in eine Tabelle eingetragen, die die Fehler kategorisiert: Die Tabelle enthält Tripel $\{t_{Haben}, t_{Soll}, Anzahl\}$ in denen die Anzahl der Fehler gespeichert werden, bei denen ein Tag t_{Haben} statt eines Tags t_{Soll} verwendet wurde (vgl. Tabelle 9).

Die erste Zeile besagt, dass es an 134 Stellen Fehler gab, bei denen ein Wort, weil

Tabelle 10:

Wort	POS Tags (sortiert nach Wahrscheinlichkeit)
Buchen	{VVIMP, NN}
Sie	{PPER}
den	{ARTDEF, PDS, PRELS}
Flug	{NN}

ches richtig markiert ein Nomen (NN) ist, stattdessen als Verb (VFIN = finites Vollverb) markiert wurde. Der Brill Tagger versucht, aus diesen Fehlern zu lernen und automatisch Regeln zu erzeugen. Problem dabei ist, dass es theoretisch möglich wäre unendlich viele Regeln automatisch zu generieren: "Ändere Tag VFIN in NN wenn das erste Wort davor den Tag x hat, das 2. Wort davor den Tag y,... das n-te Wort davor den Tag... usw."

Um die Anzahl der Regeln überschaubar und dadurch auch die Zeit für die Anwendung dieser Regeln in einem gewissen Rahmen zu halten, werden sog. Templates eingeführt. Diese Templates sorgen dafür, dass automatisch erzeugte Regeln immer eine bestimmte Form haben müssen³:

1. "Ändere Tag a in Tag b, wenn das vorhergehende (nachfolgende) Wort x ist."
2. "Ändere Tag a in Tag b, wenn eines der drei vorhergehenden (nachfolgenden) Wörter x ist."
3. "Ändere Tag a in Tag b, wenn das vorhergehende (nachfolgende) Wort mit dem Tag t markiert wurde und das vorhergehende (nachfolgende) Wort x ist."

In den obigen Regeln sind a,b,t und x jeweils Variablen für beliebige Wörter bzw. Tags. Aus jedem Template und jedem Fehler-Tripel wird nun genau eine Regel erzeugt. Diese Regel wird angewendet und die Fehlerreduktion berechnet:

Korrigierte Fehler: Die Anzahl der Fehler, die durch Anwendung der Regel korrigiert wurden.

Beispiel: Es wäre möglich, dass von den 134 fälschlich markierten Verben nach Anwendung einer Regel nur noch 50 übrig sind. Die Anzahl der korrigierten Fehler wäre demnach 84.

Erzeugte Fehler: Die Anzahl der Fehler, die durch Anwendung der Regel erst gemacht wurden.

Beispiel: Nach der Anwendung der Regel sind 21 neue Fehler hinzugekommen, weil die Regel unter bestimmten Voraussetzungen einen Fehler erzeugt.

Die Fehlerreduktion ergibt sich demnach aus Differenz der Anzahl der korrigierten Fehler und der Anzahl der erzeugten Fehler. In

³ Dies ist nur eine kleine Auswahl, es gibt insgesamt 21 solcher Templates in der entsprechenden Publikation von Eric Brill [brill92simple].

Tabelle 9:

t _{Haben}	t _{Soll}	Anzahl
VFIN	NN	134
VVIMP	NN	1
ADJ	VFIN	54

dem Beispiel oben sind das 63.

Diejenige Regel, die die beste Fehlerreduktion hat, wird dann dem Regelsystem hinzugefügt.

Nachteil dieser Methode zum automatischen Erlernen von Regeln ist, dass immer noch ein manuell annotierter Korpus vorhanden sein muss. Dieses Problem wurde teilweise von Eric Brill [brill95unsupervised] gelöst, wird aber im Folgenden nicht betrachtet.

BIGRAMME UND KONTEXTREGELN

Diese beiden Komponenten werden zusammen vorgestellt, weil sie bei dem Versuch Wörter zu taggen, die nicht im Trainingskorpus vorkamen und daher unbekannt sind, zusammenarbeiten. Grundsätzlich gibt es verschiedene Ansätze, um diese unbekannt Wörter zu taggen:

Die "dümmste" Möglichkeit mit diesen Wörtern umzugehen wäre, ihnen einfach einen konstanten oder zufälligen Tag zuzuweisen, zum Beispiel NN, das heißt alle unbekannt Wörter werden als Nomen markiert. Die "Lösung" ist jedoch sehr problematisch, da es sehr wahrscheinlich ist, dass hier ein Fehler gemacht wird. Das Problem dabei ist nicht der einzelne Fehler, sondern die Tatsache, dass das Auftreten eines falschen Tags bzw. eines Fehlers wahrscheinlich zu einer Reihe von Folgefehlern führt. Dieser Ansatz ist also weniger geeignet.

Eine weitere Möglichkeit ist die Bearbeitung mit Bigrammen: Der zu taggende Text wird in Bigramme zerlegt, also für das englische Beispiel aus dem Abschnitt ENGTWOL, wie folgt:

{(It, is), {is, expected}, {expected, that}, {that, Mr.}, {Mr., Montoya}, {Montoya, will}, {will, race}, {race, tomorrow}}⁴

Es ist wahrscheinlich, dass das Wort "Montoya" nicht im Trainingskorpus vorhanden war und daher das Zuweisen eines initialen Tags nicht möglich war. Mit Hilfe der

⁴ Auf die Betrachtung der Satzzeichen als eigene Tokens wurde hier verzichtet, weil es hier unerheblich ist.

Bigramme wäre es aber möglich, an Hand der umliegenden Wörter den wahrscheinlichsten Tag zu finden.

Ähnlich zu den Transformation Rules ist der Aufbau der Contextual Rules: Hier wird an Hand von Suffixen, Affixen und Präfixen, der Schreibweise oder anderen Wortmerkmalen versucht, den richtigen Tag zu finden. Zum Beispiel sind im Englischen unbekannt Wörter, die groß geschrieben sind, relativ häufig Eigennamen, womit der richtige Tag im obigen Beispiel leicht zu finden ist. Ähnlich könnte es bei einem Wort funktionieren, dass zum Beispiel auf die Endung "-ous" endet, womit ein Tagging des Wortes als Adjektiv wahrscheinlich richtig ist.

Ablauf des Brill Taggings

Der Ablauf des Taggens mit dem Brill-Tagger ist sehr ähnlich zu dem eines regelbasierten Taggers:

1. Zuweisen eines initialen Tags zu allen bekannten Wörtern
2. Zuweisung von Tags bei Wörtern, die nicht im Trainingskorpus vorkamen mit Hilfe von Kontextregeln und Bigrammen.
3. Anwendung der Lexical Rules zur Korrektur von Fehlern, die während der ersten beiden Schritte gemacht wurden.

Evaluierung: Brill Tagger

Im Gegensatz zu den anderen regelbasierten Ansätzen in [KleinSimmons63] und [GR71] ist die Qualität des Ergebnisses des Brill Taggers wesentlich besser. Die Qualität der Ergebnisse der stochastischen Tagger, welche bisher von den regelbasierten Systemen nicht erreicht werden konnten, werden sogar noch geringfügig verbessert. Ein weiterer Vorteil gegenüber den stochastischen Taggern ist die Tatsache, dass wesentlich weniger Information gespeichert werden muss: Wie bereits bei der Evaluation der stochastischen Tagger in Abschnitt "Evaluierung stochastische Tagger" angesprochen wurde, müssen die Kontextinformationen in riesigen Matrizen abgespeichert werden, während der Brill Tagger mit weniger als 100 Regeln auskommt.

Ferner ist es zusätzlich möglich, die Regeln ohne einen manuell annotierten Korpus zu lernen, d.h. es genügt ein nicht annotierter Korpus und ein Wörterbuch, in dem lediglich die möglichen Tags zu einem Wort in beliebiger Reihenfolge (also nicht wie bei dem vorgestellten Modell nach Wahrscheinlichkeit geordnet) enthalten sind, um entsprechende Regeln zu lernen. Dieses Verfahren wird in [brill95unsupervised] vorgestellt.

Durch die Möglichkeit Regeln vollkommen automatisch zu lernen ist der Brill Tagger damit auch sehr portabel und für beliebige Sprachen mit geringem Aufwand einsetzbar.

In [roche95deterministic] wurde desweiteren gezeigt, dass die Regeln des Brill Taggers in endliche Automaten umgewandelt werden können. Eine solche Umwandlung führt dazu, dass der Brill Tagger für das Taggen eines Satzes mit n Wörtern genau n Schritte benötigt. Damit ist der Brill Tagger zehnmal schneller als der schnellste stochastische Part of Speech Tagger. ■

Literatur

- [STTS95] Christine Stöckert, Christine Thielen, Anne Schiller, Simone Teufel. Stuttgart tübinger tagset. <http://www.sfs.nphil.uni-tuebingen.de/Elwis/stts/stts.html>, 1995. Vorläufige Guidelines für das Tagging deutscher Textcorpora mit STTS
- [C7] Paul Rayson. The claws web tagger.
- [GR71] G. Rubin, B. Greene. Automatic grammatical tagging of english. Technical Report, Brown University, Providence, RI, 1971.
- [DeRose88] S. DeRose. Grammatical category disambiguation by statistical optimization. Computational Linguistics, 14: 31-39, 1988.
- [KleinSimmons63] Sheldon Klein and Robert F. Simmons. A computational approach to grammatical coding of english words. J. ACM, 10(3): 334-347, 1963.
- [ENGTWOL] A. Voutilainen. A Syntax-based part of speech analyser, 1995.
- [Brill92simple] Eric Brill. A simple rule-based part-of-speech tagger. In Proceedings of ANLP-92, 3rd Conference on Applied Natural Language Processing, pages 152-155, Trento, IT, 1992.
- [brill95unsupervised] Eric Brill. Unsupervised learning of disambiguation rules for part of speech tagging. In David Yarovsky and Kenneth Church, editors, Proceedings of the Third Workshop on Very Large Corpora, pages 1-13, Somerset, New Jersey, 1995. Association for Computational Linguistics.
- [roche95deterministic] Emmanuel Roche and Yves Schabes. Deterministic part-of-speech tagging with finite-state transducers. Computational Linguistics, 21(2): 227-253, 1995.
- [PTTS93] Mitchell P. Marcus and Beatrice Santorini and Mary Ann Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. Computational Linguistics, 2: pages 313-330, 1994
- [NEGRA] NEGRA Korpus Version 2

Karlsruher Transfer

Chefredaktion Martin Wagener (V.i.S.d.P.)

Layout Christian Bock
Christian Bürger
Matthias Hecking
Bastian Schwark
Martin Wagener

Redaktion Volker Schmitt
Bastian Schwark

Herausgeber Verein Karlsruher
Wirtschaftswissenschaftler e.V.

Druck Idee, Satz & Druck
Scheffelstr. 52
76135 Karlsruhe

Auflage 2500 Exemplare

Bezug Der Karlsruher Transfer erscheint einmal pro Semester. Er kann kostenlos von Interessenten bezogen werden.
ISSN 0937-0803

Anschrift Karlsruher Transfer
Verein Karlsruher
Wirtschaftswissenschaftler e.V.
Waldhornstraße 27
76131 Karlsruhe
Tel.: 0721/608-3078
Fax: 0721/379824
transfer@vkw.org
www.vkw.org/transfer
wap.vkw.org

Namentlich gekennzeichnete Artikel geben nicht unbedingt die Meinung der Redaktion wieder. Die veröffentlichten Beiträge sind urheberrechtlich geschützt. Vervielfältigungen jeglicher Art nur mit Genehmigung der Redaktion und der Autoren.

Richtigstellung: Die im Karlsruher Transfer Nummer 30 auf den Seiten 5-7 abgedruckten Photographien stammen nicht wie angegeben von der IME Graduate School at the University of Karlsruhe. Vielmehr ist der Urheber Dipl. Inf. Peter Baumung.