Universität Potsdam
Institut für Informatik
Lehrstuhl Maschinelles Lernen

NLP-Pipeline

Tobias Scheffer
Thomas Vanck

NLP-Pipeline

- Folge von Verarbeitungsschritten für
  - Informationsextraktion,
  - Übersetzung,
  - Zusammenfassung.

NLP- (Natural Language Processing-) Pipeline

1. Tokenisierung,
2. Wortarterkennung (Part-of-Speech-Tagging),
3. Eigennamenerkennung (Named Entity Recognition),
4. Shallow Parsing,
5. Chunking,
6. Informationsextraktion

NLP- (Natural Language Processing-) Pipeline

- Sequentielle Verarbeitung von Sprache.
- Viele Varianten der Pipeline, je nach Problemstellung.
- Jede Stufe baut auf Ausgabe der vorangegangenen Stufe auf.
- Manche Verfahren werden für mehrere der Stufen eingesetzt.

NLP- (Natural Language Processing-) Pipeline

- Problem: Fehler summieren sich über die Verarbeitungsschritte auf.
- Trend: Gesamte Verarbeitung als ein einziges Optimierungs-/Lernproblem formulieren.
- Einzelne Stufen können sich so besser beeinflussen.
  - Manchmal ergibt sich erst aus der semantischen Analyse, dass eine bestimmte Wortart für einen Term unplausibel ist.
- Problem: Als einziges Optimierungsproblem extrem aufwändig.
  - Stand der Forschung: Zwei Verarbeitungsschritte gleichzeitig.

POS-Tagging

- Zuordnung der Wortart von Termen.
- Meist Eingabe für weitere Verarbeitungsschritte (Parser, Extraktionsregeln, Übersetzung).
- „Tagset“: Definition der Wortarten, die unterschieden werden sollen.
- Im Prinzip: Nomen, Verben, Pronomen, Präposition, Adverb, Konjunktion, Partizip, Artikel, …
  - John/NN saw/V her/PN duck/V oder
  - John/NN saw/V her/PN duck/N
- 95-97% korrekt gesetzte Tags üblich.

Gebräuchliche Tags für das Englische

AT	Artikel
BEZ	„is“
IN	Preposition
JJ	Adjektiv
NN	Nomen
NNP	„Proper Noun“
NNS	Nomen, plural
PERIOD	„.“, „?“, „!“
PN	Personalpronomen
RB	Adverb
TO	„to“
VB	Verb
VBD	Verb, Vergangenheit
…	

Wortarterkennung: Einflussfaktoren

- Manche POS-Folgen sind gebräuchlich
  - AT, JJ, NN.
- Andere hingegen weniger wahrscheinlich
  - AT, PN VB.
- Die meisten Wörter treten nur in wenigen Wortarten auf, z.B.
  - „Lauf“/VB oder „Lauf“/NN, aber nicht „Lauf“/JJ.
- Vorhersage der Wortform auf Grundlage der Nachbartags: ca. 77% Genauigkeit.
- Vorhersage der wahrscheinlichsten Wortform jedes Wortes ohne Kontext: ca. 90% Genauigkeit.

Wortartenerkennung: Methoden

- Historisch: Transformationsbasierte (regelbasierte) Tagger.
- N-Gramm-Modelle
- HMMs
- Diskriminative Sequenzmodelle (Conditional Random Fields, Hidden Markov Support Vector Machines).

Wortarterkennung mit Markov-Modellen

- Ein Zustand pro POS-Tag.
- Trainingsdaten: „getaggtes“ Korpus. Zustände sind sichtbar.
- Anwendung des Taggers auf neuen Satz: Zustände sind noch nicht sichtbar.

Training des POS-Taggers

- Startwahrscheinlichkeiten: Erste Tags eines Satzes abzählen.
- Übergangswahrscheinlichkeiten: Häufigkeiten der Tagübergänge abzählen.
- Beobachtungswahrscheinlichkeiten: Häufigkeiten aller Wörter in allen Zuständen abzählen.

Anwenden des POS-Taggers

- Folge der Wahrscheinlichsten Einzelzustände gegeben Wortfolge.
  - Durch Forward-Backward-Algorithmus.
- Wahrscheinlichste Gesamtfolge von Zuständen gegeben Wortfolge.
  - Durch Viterbi-Algorithmus bestimmt.

Trigramm-Tagger

- HMM = Bigramm-Tagger.
- Trigramm-Tagger: Jedes Paar von Tags bildet einen Zustand.
- Zustandsübergänge entsprechen Tag-Trigrammen.

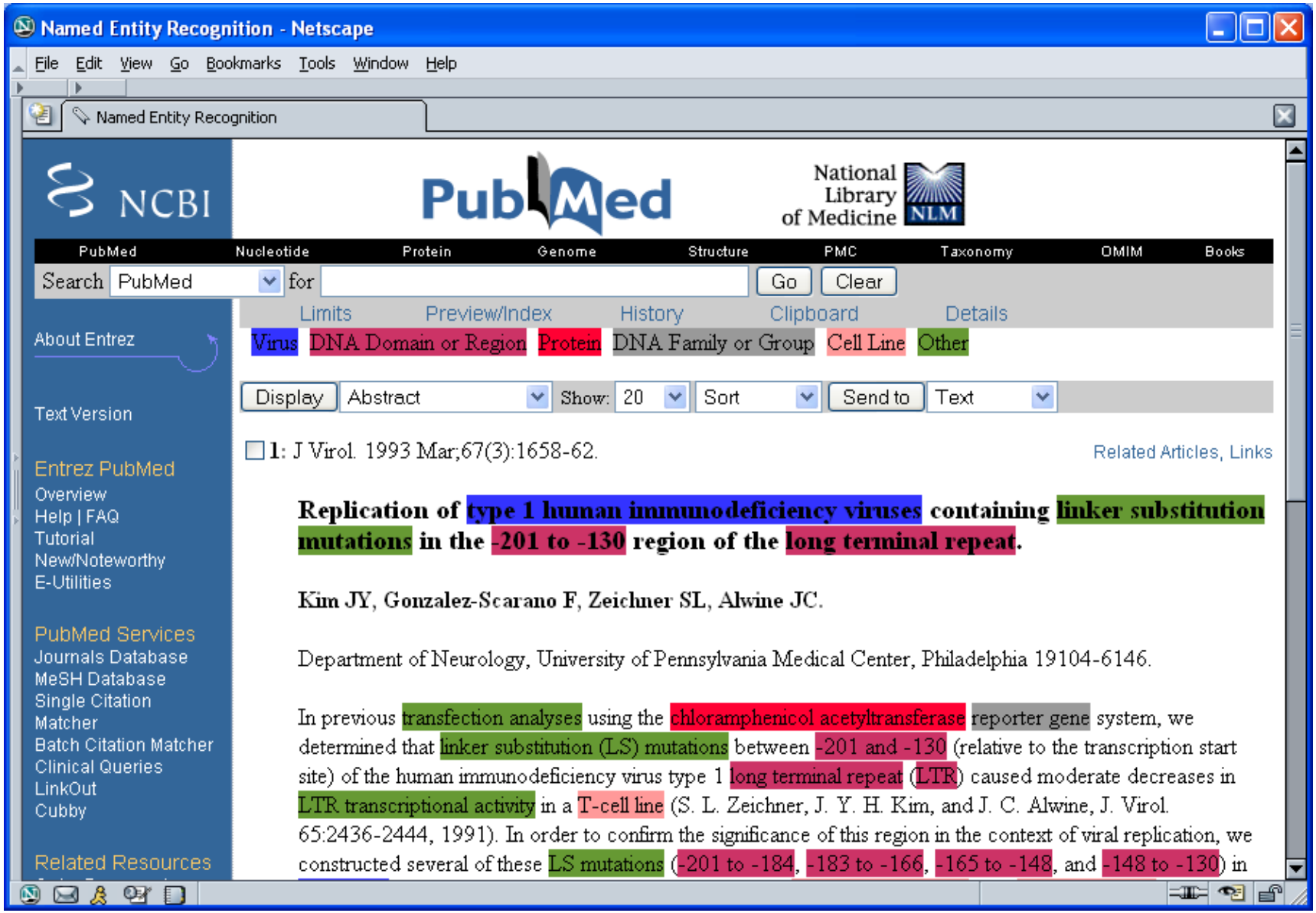
Transformationsbasierte Tagger

- Z.B. Brill Tagger
- Ausgangspunkt: jedes Wort mit dem dafür häufigsten Tag taggen.
- Anwendung von Transformationsregeln: „Wenn bestimmte Bedingungen im Kontext von X_t erfüllt sind, dann ersetze Tag von X_t durch X_t=x“.
- Konstruktion der Regeln durch Handarbeit, unterstützt von Lernen.
- Weniger Robust, historische Technik.

Eigennamenerkennung – Named Entity Recognition

- Erkennung von:
  - Personen-, Firmen-, Ortsnamen (NAMEX)
  - Daten, Uhrzeiten (TIMEX)
  - Prozentangaben, Geldbeträgen (NUMEX)
  - Genen, Proteinen, Loci, Viren, Krankheiten.
- Normierung von Eigennamen (Named Entity Resolution):
  - Identifizierung von Varianten:
    DCAG = Daimler = DaimlerChrysler AG,
  - Abbildung in Taxonomie: Food/Fruit/Apple ≠ Company/ComputerHardware/Apple.

NER - Beispiel



Schwierigkeiten

- Anzahl möglicher Eigennamen im Prinzip unendlich groß.
- Viele Eigennamen sieht ein Erkenner zum ersten Mal (z.B. 3,3'-diaminobenzidine tetrahydrochlorine).
- Ständig werden neue Eigennamen erfunden (z.B. für neue chemische Verbindungen, neue Unternehmen).
- Nicht einheitliche Varianten des Namens einer Entität (HU, Humboldt-Uni, HU Berlin Humboldt Universität, Humboldt-Universität zu Berlin).

Named Entity Recognition: Methoden

- Generierung von Merkmalen für Token:
  - Meist werden sehr viele Merkmale erzeugt.
  - Wort selbst, POS-Tag, orthographische Merkmale, …
  - Gazetteer-Features.
- Erkennung von Eigennamen auf Grundlage der Merkmale:
  - „Sliding Window“,
  - Hidden-Markov-Modelle,
  - Conditional Random Fields.

Named Entity Recognition

- Kernstück: „Gazetteers“
  - Listen bekannter Namen, auf Anwendungsbereich abgestimmt.
- Gazetteer aber keineswegs ausreichend.
  - Mit „Apple“ kann auch Obst gemeint sein.
  - Neu erfundene Namen tauchen nicht auf.
  - Mit „Ca“, „fat“ können Gene gemeint sein, müssen aber nicht.

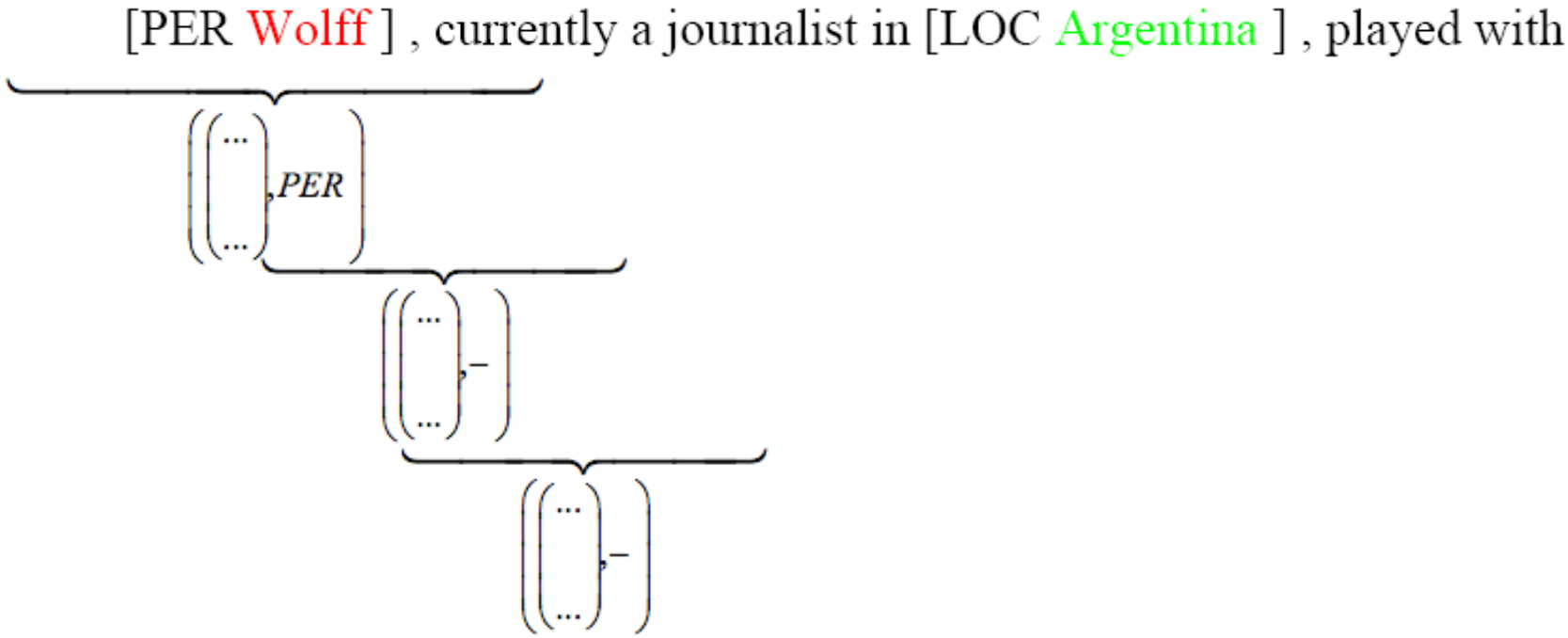
Generierung von Merkmalen

- Bag-of-Word-Features:
  - Wort ist „Aal“, …, Wort ist „zutzeln“.
- Buchstaben-N-Gramm-Merkmale,
  - Wort enthält „aaa“, …, Wort enthält „zzz“).
- Orthographische Merkmale:
  - „Beginnt mit Großbuchstaben“, …,
  - „enthält Sonderzeichen“, „enthält Ziffer“, …
- Gazetteer-Merkmale:
  - Wort taucht in Eigennamenliste 1 auf.

Methoden: Sliding Window

- Merkmale werden in einem Fenster des Satzes generiert, der um ein zu klassifizierendes Token herumgelegt wird.
- Klassifikator entscheidet mit Hilfe dieser Merkmale, ob zentrales Token ein Eigenname ist.
- Klassifikator: Abbildung von Merkmalsvektor auf NER-Klassen.
- Fenster wird über Text geschoben.
- Trainieren: Klassifikator wird trainiert aus Paaren (Vektor, NER-Klasse). Beispiele werden aus getaggten Beispieltexten gewonnen.

Methoden: Sliding Window



- Ein Merkmalsvektor pro Fensterposition.
- Merkmale: Ein Merkmal pro Wort im Dictionary, POS, orthographische Merkmale, Element des Gazetteers?,…

Methoden: HMMs und CRFs

- $b_i(O_t)$: Verteilung über Vektor der zuvor berechneten Merkmale.
- Trainieren durch abzählen aus getaggten Beispieltexten (kein Baum-Welch erforderlich).
- Siehe letzte Vorlesung.

Parsing: PCFGs

- Probabilistic context-free grammar: Statistisches Sprachmodell.
- Hidden-Markov-Modell:
  - Probabilistischer endlicher Automat ≅ probabilistische reguläre Sprache
- Hidden-Markov-Modelle sind spezielle PCFG.
  - Zugrundeliegender endlicher Automat ersetzt durch kontextfreie Grammatik.

PCFG: Definition

- Terminalsymbole $\{w^k\}$, k=1...V.
- Nichtterminale $\{N^i\}$, i=1...n.
- Startsymbol N^1.
- Regeln $\{N^i \rightarrow \xi^j\}$, wobei ξ^j eine Folge aus Terminalen und Nichtterminalen ist.
- Wahrscheinlichkeiten, so dass $\forall i : \sum_j P(N^i \rightarrow \xi^j) = 1$
- $P(N^i \rightarrow \xi^j)$ steht für $P(\xi^j \mid N^i)$.
- Zu parsierender Satz: $w_1, \ldots, w_m$. w_{ab} steht für $w_a, \ldots, w_b$.
- $N^i \Rightarrow^* w_a \ldots w_b$, wenn Satz $w_a \ldots w_b$ aus Nichtterminal N^i abgeleitet werden kann.
- N^i_{pq}: w_{pq} wird von Nichtterminal N^i abgeleitet.

PCFG: Wahrscheinlichkeit eines Satzes

- Über alle Pars-Bäume t: $P(w_{1m}) = \sum_t P(w_{1m}, t)$
- Beispiel:
  - P(S→bla) = 1/3.
  - P(S→S S) = 2/3.

  - P(„bla bla bla“)=?

PCFG: Elementare Fragen

- PCFGs beantworten (wie HMMs) drei Fragen:
  1. Was ist die Wahrscheinlichkeit eines Satzes w_{1m} gegeben eine Grammatik?
  2. Was ist der wahrscheinlichste Pars-Baum gegeben Satz und Grammatik?
  3. Welche Wahrscheinlichkeiten für die Grammatikregeln können wir aus einem gegebenen Trainingskorpus lernen?

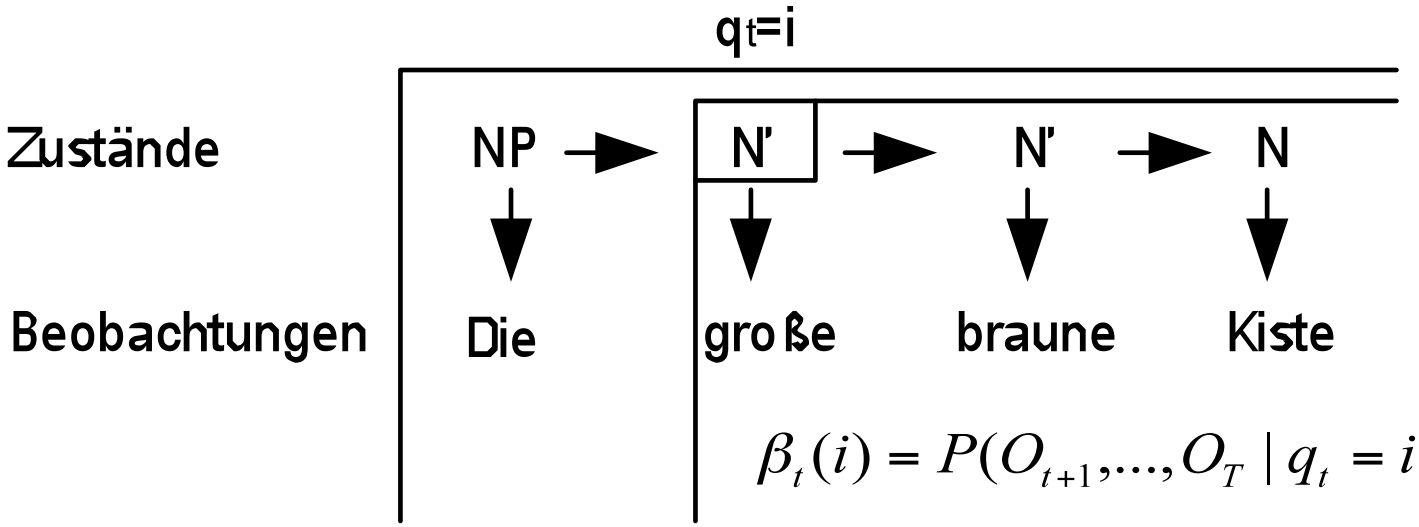
Chomsky-Normalform

- Vereinfachung hier: Regeln haben eine der Formen:
  - $N^i \rightarrow N^j\ N^k$,
  - $N^i \rightarrow w^j$.
- Für jede kontextfreie Grammatik lässt sich eine Grammatik in Chomsky-Normalform finden, die dieselbe Sprache erzeugt.
- (Allerdings entstehen nicht die gleichen Pars-Bäume).

Proabilistische reguläre Grammatiken

- Spezialfall: Jede Regel hat eine der Formen:
  - $N^i \rightarrow w^j N^k$,
  - $N^i \rightarrow w^j$.
- Für jedes HMM existiert eine PCFG (Spezialfall PRG), die dieselbe Wahrscheinlicheitsverteilung modelliert.

Forward / Backward – Inside / Outside



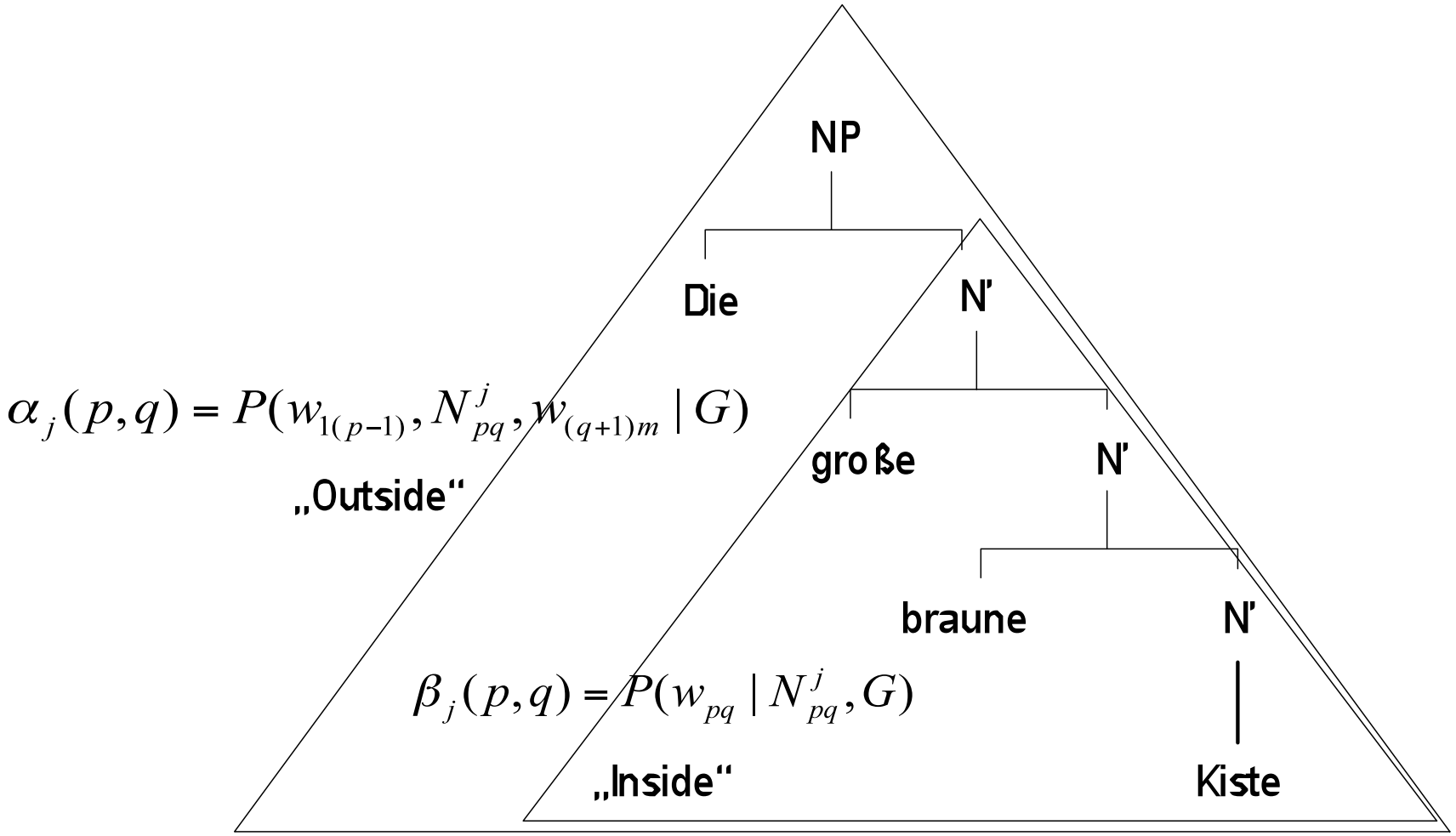
$$\beta_t(i) = P(O_{t+1},\ldots,O_T \mid q_t = i,\lambda)$$

„Backward“

$$\alpha_t(i) = P(O_1,\ldots,O_t, q_t = i \mid \lambda)$$

„Forward“

Forward / Backward – Inside / Outside



$$\alpha_j(p,q) = P(w_{1(p-1)}, N^j_{pq}, w_{(q+1)m} \mid G)$$

„Outside“

$$\beta_j(p,q) = P(w_{pq} \mid N^j_{pq}, G)$$

„Inside“

Inside / Outside

- „Outside-Wahrscheinlichkeit“:

$$\alpha_j(p,q) = P(w_{1(p-1)}, N^j_{pq}, w_{(q+1)m} \mid G)$$

- „Inside-Wahrscheinlichkeit“

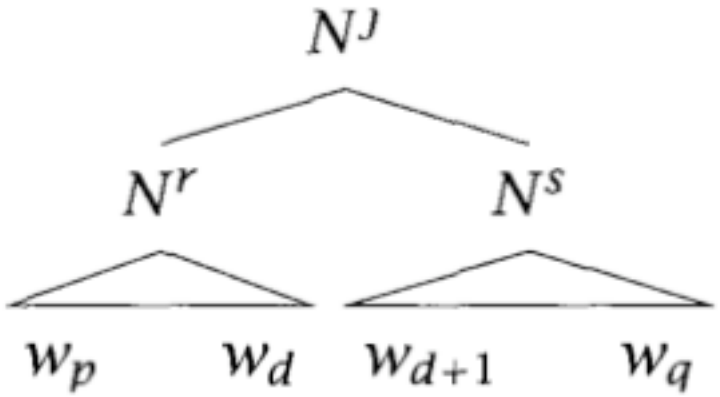
$$\beta_j(p,q) = P(w_{pq} \mid N^j_{pq}, G)$$

- Wahrscheinlichkeit einer Wortkette:

$$P(w_{1T} \mid G) = P(N^1 \Rightarrow^* w_{1T}, G)$$
$$= P(w_{1T} \mid N^1_{1T}, G) = \beta_1(1,T)$$

Inside-Berechnung anschaulich

$$\beta_j(p,q) = P(w_{pq} \mid N^j_{pq}, G)$$



Berechnung der Inside-Wahrscheinlichkeit

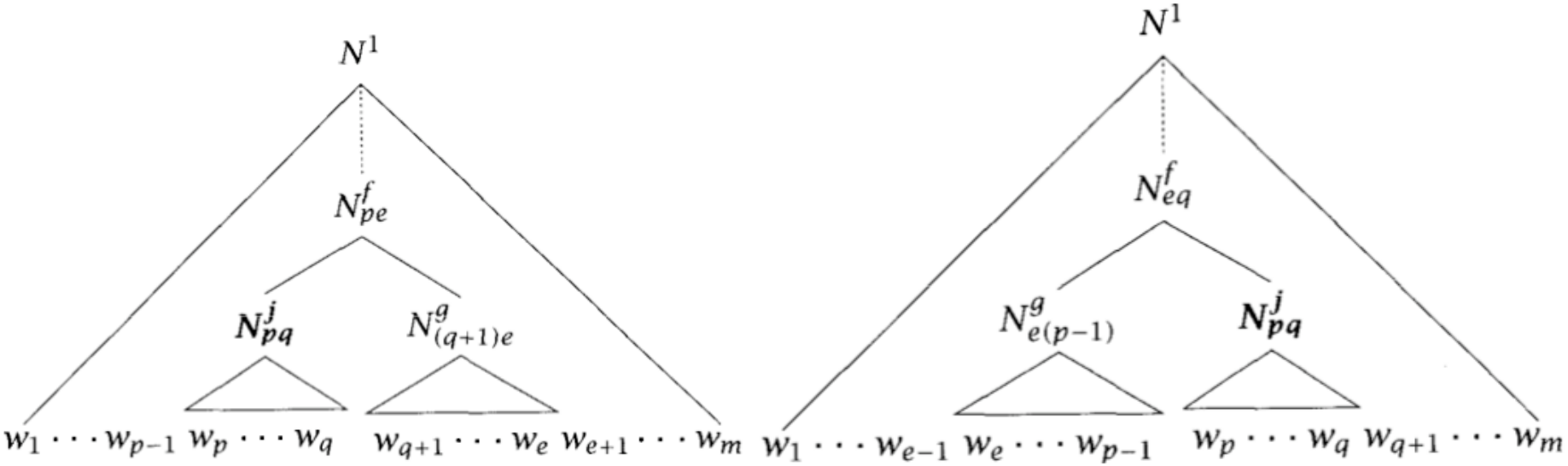
- Verankerung: $\beta_j(k, k) = P(w_k|N^j_{kk}, G)$
  $= P(N^j \rightarrow w_k|G)$

- Induktionsschritt:

$$\begin{aligned}
\beta_j(p,q) &= P(w_{pq}|N^j_{pq}, G) \\
&= \sum_{r,s}\sum_{d=p}^{q-1} P(w_{pd}, N^r_{pd}, w_{(d+1)q}, N^s_{(d+1)q}|N^j_{pq}, G) \\
&= \sum_{r,s}\sum_{d=p}^{q-1} P(N^r_{pd}, N^s_{(d+1)q}|N^j_{pq}, G)P(w_{pd}|N^j_{pq}, N^r_{pd}, N^s_{(d+1)q}, G) \\
&\quad \times P(w_{(d+1)q}|N^j_{pq}, N^r_{pd}, N^s_{(d+1)q}, w_{pd}, G) \\
&= \sum_{r,s}\sum_{d=p}^{q-1} P(N^r_{pd}, N^s_{(d+1)q}|N^j_{pq}, G)P(w_{pd}|N^r_{pd}, G) \\
&\quad \times P(w_{(d+1)q}|N^s_{(d+1)q}, G) \\
&= \sum_{r,s}\sum_{d=p}^{q-1} P(N^j \rightarrow N^r N^s)\beta_r(p,d)\beta_s(d+1,q)
\end{aligned}$$

Outside-Berechnung anschaulich

$$\alpha_j(p,q) = P(w_{1(p-1)}, N^j_{pq}, w_{(q+1)m} \mid G)$$



Berechnung der Outside-Wahrscheinlichkeit

- Verankerung:

$$\alpha_1(1,m) = 1$$

$$\alpha_j(1,m) = 0 \text{ for } j \neq 1$$

- Induktionsschritt:

$$\begin{aligned}
\alpha_j(p,q) &= \Big[\sum_{f,g\neq j}\sum_{e=q+1}^{m} P(w_{1(p-1)}, w_{(q+1)m}, N^f_{pe}, N^j_{pq}, N^g_{(q+1)e})\Big] \\
&\quad + \Big[\sum_{f,g}\sum_{e=1}^{p-1} P(w_{1(p-1)}, w_{(q+1)m}, N^f_{eq}, N^g_{e(p-1)}, N^j_{pq})\Big] \\
&= \Big[\sum_{f,g\neq j}\sum_{e=q+1}^{m} P(w_{1(p-1)}, w_{(e+1)m}, N^f_{pe}) P(N^j_{pq}, N^g_{(q+1)e} | N^f_{pe}) \\
&\quad \times P(w_{(q+1)e} | N^g_{(q+1)e})\Big] + \Big[\sum_{f,g}\sum_{e=1}^{p-1} P(w_{1(e-1)}, w_{(q+1)m}, N^f_{eq}) \\
&\quad \times P(N^g_{e(p-1)}, N^j_{pq} | N^f_{eq}) P(w_{e(p-1)} | N^g_{e(p-1)})\Big] \\
&= \Big[\sum_{f,g\neq j}\sum_{e=q+1}^{m} \alpha_f(p,e) P(N^f \rightarrow N^j\ N^g) \beta_g(q+1,e)\Big] \\
&\quad + \Big[\sum_{f,g}\sum_{e=1}^{p-1} \alpha_f(e,q) P(N^f \rightarrow N^g\ N^j) \beta_g(e,p-1)\Big]
\end{aligned}$$

Wahrscheinlichster Pars-Baum

- Viterbi-Algorithmus für PCFG.
- $O(m^3n^3)$
- $\delta_i(p,q)$ = Höchste Inside-Wahrscheinlichkeit für einen Pars des Teilbaumes N^i_{pq}.
- Initialisierung: $\delta_i(p,p) = P(N^i \rightarrow w_p)$
- Induktion: $\delta_i(p,q) = \max_{\substack{1 \le j,k \le n \\ p \le r < q}} P(N^i \rightarrow N^j\ N^k)\delta_j(p,r)\delta_k(r+1,q)$
- Besten Weg abspeichern:

$$\psi_i(p,q) = \arg\max_{(j,k,r)} P(N^i \rightarrow N^j\ N^k)\delta_j(p,r)\delta_k(r+1,q)$$

- Parse-Baum rekonstruieren: $\psi_i(p,q) = (j,k,r)$ dann sind Unterknoten N^j_{pr} (links) und $N^k_{(r+1)q}$ (rechts) im Baum

PCFG: Lernen aus Beispielkorpora

- Problem: Grammatik (CFG) muss schon gegeben sein, der Lernalgorithmus lernt nur die Wahrscheinlichkeiten $P(N^i \rightarrow N^j N^k)$
- Lernen von CFG: schwierig (Forschungsthema).
- Lernen von CFG+Wahrscheinlichkeiten (komplettes Lernen einer PCFG aus Korpus: Forschungsthema.

1. Wenn Korpus mit Pars-Baum annotiert ist: Abzählen der Regelwahrscheinlichkeiten.
2. Wenn Korpus nicht annotiert ist: EM-Algorithmus.

Penn Treebank

- S simple clause
- SBARQ Wh- question
- SQ Yes/no question
- ADJP adjective phrase
- NP noun phrase
- PP prepositional phrase
- QP quantifier phrase
- VP verb phrase
- HWNP Wh- noun phrase
- -SBJ subject
- -LOC location
- CONJP conjunction phrase
- FRAG fragment
- INTJ interjection
- LST list marker
- NAC not a constituent
- NX nominal constituent inside NP
- PRN parenthitical
- PRT particle
- RRC reduced relative
- …

Penn Treebank

```
( (S (NP-SBJ The move
      (VP followed
          (NP (NP a round)
              (PP of
                  (NP (NP similar increases)
                      (PP by
                          (NP other lenders))
                      (PP against
                          (NP Arizona real estate loans)))))
```

- Insgesamt 4.5 Mio Wörter.
- Baum-Bänke aufzubauen und PCFGs zu lernen ist effektiver als Grammatiken von Hand zu entwickeln.