Einführung in die Computerlinguistik

HMM POS-Tagging

Laura Kallmeyer

Heinrich-Heine-Universität Düsseldorf

Summer 2016

POS Tags (1)

Jurafsky and Martin (2009)
POS = part-of-speech Tags sind morphosyntaktische Kategorien von Wortformen.
Bsp. (Brown Corpus):

```
(The AT) (Fulton NP-TL) (County NN-TL)
(Grand JJ-TL) (Jury NN-TL) (said VBD)
(Friday NR) (an AT) (investigation NN) (of
IN) (Atlanta's NP) (recent JJ) (primary NN)
(election NN) (produced VBD) (“ “) (no AT)
(evidence NN) (” ”) (that CS) (any DTI)
(irregularities NNS) (took VBD) (place NN)
(. .)
```

POS Tags (2)

Bsp. (Penn Treebank):

```
(Pierre NNP) (Vinken NNP) (, ,) (61 CD)
(years NNS) (old JJ) (, ,) (will MD) (join,
VB) (the DT) (board NN) (as IN) (a DT)
(nonexecutive JJ) (director NN) (Nov. NNP)
(29 CD) (. .)
```

POS Tags (3)

Es gibt kein fest vorgegebenes Inventar von möglichen POS Tags. Die Wahl der POS Tags hängt von der jeweiligen Sprache und von der Anwendung ab, die mit dem getaggten Text vorgenommen werden soll. Morphologisch reichere Sprachen haben in der Regel größere Tagsets.

- Penn Treebank Tagset (PTB, American English): 45 POS-Tags
- Brown Corpus (American English): 87 POS-Tags
- British National Corpus (BNC, British English) basic tagset: 61 POS-Tags
- Stuttgart-Tübingen Tagset (STTS) für das Deutsche: 54 POS-Tags.
- Prague Dependency Treebank (PDT, Tschechisch): 4288 POS-Tags.

HMM POS Tagging (1)

Problem: Gegeben eine Folge w_1^n von n Wörtern, wollen wir die wahrscheinlichste Folge $\hat{t}_1^n$ aller möglichen Folgen t_1^n von n POS Tags für diese Wortfolge ermitteln.

$$\hat{t}_1^n = \arg\max_{t_1^n} P(t_1^n | w_1^n)$$

$\arg\max_x f(x)$ bedeutet "das x, für das $f(x)$ maximal groß wird".

Mit Bayes gilt $$\hat{t}_1^n = \arg\max_{t_1^n} \frac{P(w_1^n | t_1^n) P(t_1^n)}{P(w_1^n)}$$

Daraus folgt $$\hat{t}_1^n = \arg\max_{t_1^n} P(w_1^n | t_1^n) P(t_1^n)$$

HMM POS Tagging (2)

$\hat{t}_1^n = \arg\max_{t_1^n} P(w_1^n|t_1^n)P(t_1^n)$ ist immer noch zu aufwendig zu berechnen.

Annäherungen:

$$P(w_1^n|t_1^n) \approx \prod_{i=1}^{n} P(w_i|t_i)$$

$$P(t_1^n) \approx \prod_{i=1}^{n} P(t_i|t_{i-1})$$

Wenn wir Trainingsdaten in Form eines getaggten Korpus zur Verfügung haben, können wir die benötigten Wahrscheinlichkeiten durch Abzählen ermitteln. $C(x)$ sei die Anzahl von Auftreten von x im Korpus.

HMM POS Tagging (3)

Maximum Likelihood Estimates (MLE)

- Tagübergangswahrscheinlichkeit:

$$P(t_i|t_{i-1}) = \frac{C(t_{i-1}, t_i)}{C(t_{i-1})}$$

Bsp. Brown Corpus: $P(NN|DT) = \frac{C(DT,NN)}{C(DT)} = \frac{56509}{116454} = 0.49$

- Wortwahrscheinlichkeit:

$$P(w_i|t_i) = \frac{C(t_i, w_i)}{C(t_i)}$$

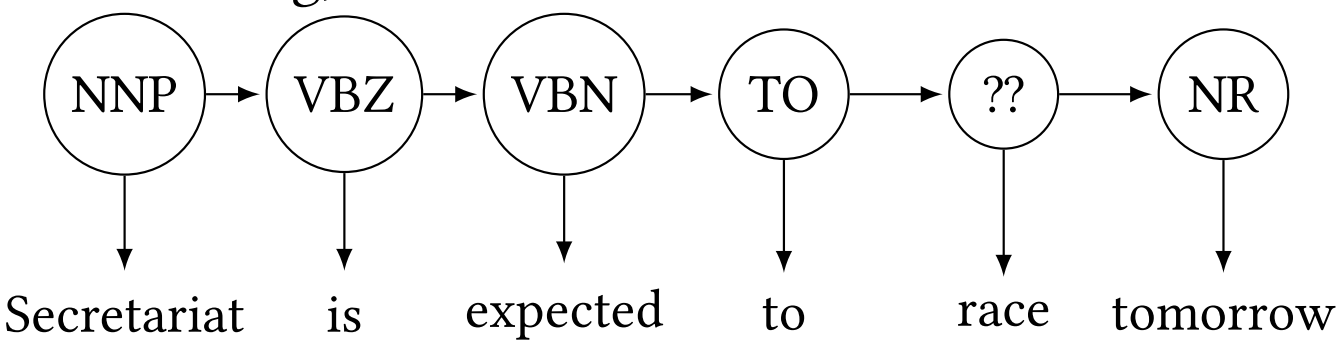
Bsp. Brown Corpus: $P(is|VBZ) = \frac{C(VBZ,is)}{C(VBZ)} = \frac{10073}{21627} = 0.47$

POS Tag Sequences (1)

Bsp.: *race*, Ambiguität zwischen VB und NN.

(1) Secretariat/NNP is/VBZ expected/VBN to/TO race/??
tomorrow/NR (Brown tagset)

Entscheidung, ob VB oder NN:



POS Tag Sequences (2)

Übergangs- und Wortwahrscheinlichkeiten, die unterschiedlich sind:

$P(\text{NN}|\text{TO}) = 0.00047$ $\quad$ $P(\text{VB}|\text{TO}) = 0.83$

$P(\text{race}|\text{NN}) = 0.00057$ $\quad$ $P(\text{race}|\text{VB}) = 0.00012$

$P(\text{NR}|\text{NN}) = 0.0012$ $\quad$ $P(\text{NR}|\text{VB}) = 0.0027$

Je nach POS Tag für *race* unterscheiden sich die Wahrscheinlichkeiten der beiden Folgen in dem Faktor

NN: $\quad P(\text{NN}|\text{TO})P(\text{race}|\text{NN})P(\text{NR}|\text{NN}) = 0.00000000032$

VB: $\quad P(\text{VB}|\text{TO})P(\text{race}|\text{VB})P(\text{NR}|\text{VB}) = 0.00000027$

Bigram-HMMs (1)

Formalisierung als gewichteter endlicher Automat. Der Automat erlaubt uns, über beobachtete Ereignisse (die zu taggenden Wörter) zu sprechen und über versteckte Ereignisse (Folgen von POS Tags). Daher die Bezeichnung Hidden Markov model (HMM).

Ein HMM besteht aus Zuständen (= POS Tags), Übergangswahrscheinlichkeiten $a_{i-1,i}$ (= $P(t_i|t_{i-1})$) und Emissionswahrscheinlichkeiten $b_i(w_i)$ ($P(w_i|t_i)$).

Bigram-HMMs (2)

Ein HMM ist ein Tupel $\langle Q, A, O, B, q_0, q_F \rangle$, wobei

- Q eine endliche Menge von Zuständen ist;
- A eine $|Q| \times |Q|$ Matrix ist, deren Elemente aus $\mathbb{R}$ sind, die Übergangswahrscheinlichkeiten,
- O eine endliche Menge von Beobachtungen ist;
- B eine Folge von $|Q|$ Funktionen $b_i : O \to \mathbb{R}$, Emissionswahrscheinlichkeiten;
- $q_0, q_F \notin Q$ der Startzustand und der Endzustand sind. Diese beiden Zustände haben keine Emissionen, und es sind zusätzlich Übergangswahrscheinlichkeiten a_{0i} und a_{iF} für alle $1 \leq i \leq |Q|$ gegeben.

Bigram-HMMs (3)

Bsp.: HMM-POS Tagger mit $Q = \{Det, N, Adj, V\}$,
$O = \{the, chief, rules, \ldots\}$, folgende Wahrscheinlichkeiten:

- Emissionswahrscheinlichkeiten ($b_i(w) = P(w|q_i)$):
  $P(\text{the}|\text{Det}) = 1$ $P(\text{chief}|\text{N}) = 3 \cdot 10^{-3}$ $P(\text{rules}|\text{N}) = 5 \cdot 10^{-3}$
  $P(\text{chief}|\text{Adj}) = 4 \cdot 10^{-3}$ $P(\text{rules}|\text{V}) = 6 \cdot 10^{-3}$
  Alle anderen Emissionswahrscheinlichkeiten für *chief*, *the* und *rules* seien 0.

- Teil der Übergangswahrscheinlichkeiten ($A_{i,j} = P(q_j|q_i)$):
  $P(\text{N}|\text{Det}) = 5 \cdot 10^{-1}$ $P(\text{N}|\text{N}) = 1 \cdot 10^{-1}$ $P(\text{N}|\text{Adj}) = 5 \cdot 10^{-1}$
  $P(\text{Adj}|\text{Det}) = 3 \cdot 10^{-1}$ $P(\text{V}|\text{N}) = 4 \cdot 10^{-1}$ $P(\text{V}|\text{Adj}) = 1 \cdot 10^{-1}$

  Die Wahrscheinlichkeit, dass ein Det am Satzanfang steht, ist 1, die, dass auf ein N oder V ein Satzende folgt, ist jeweils $0.1 = 1 \cdot 10^{-1}$.

Bigram-HMMs (4)

Ziel: Gegeben ein HMM und eine Folge von Beobachtungen $o_1 \ldots o_n$ soll ermittelt werden, welche Folge von Zuständen (POS Tags) die Beobachtung am wahrscheinlichsten macht.

Viterbi Algorithmus:

- Wir durchlaufen die Beobachtungen von links nach rechts und füllen dabei eine Matrix *viterbi* von Pfadwahrscheinlichkeiten.
- *viterbi* ist eine $(|Q| + 1) \times n$)-Matrix. Das Feld $viterbi[q, i]$ soll angeben, wie wahrscheinlich es ist, dass wir nach Abarbeiten der ersten i Beobachtungen in Zustand q sind.
- Außerdem führen wir Backpointer mit, die uns für $viterbi[q, i]$ sagen, welcher Zustand q voranging.

Bigram-HMMs (5)

Bsp.:

q_F				$360 \cdot 10^{-9}$, V
N		$15 \cdot 10^{-4}$, Det	$300 \cdot 10^{-8}$, Adj	
V			$360 \cdot 10^{-8}$, N	
Adj		$12 \cdot 10^{-4}$, Det		
Det	1, q_0			
	1: the	2: chief	3: rules	

chief, Adj: $1 \cdot 3 \cdot 10^{-1} \cdot 4 \cdot 10^{-3}$ *chief*, N: $1 \cdot 5 \cdot 10^{-1} \cdot 3 \cdot 10^{-3}$
rules, N: $\max\{12 \cdot 10^{-4} \cdot 5 \cdot 10^{-1} \cdot 5 \cdot 10^{-3}$ Vorgänger Adj , $15 \cdot 10^{-4} \cdot 1 \cdot 10^{-1} \cdot 5 \cdot 10^{-3}$ Vorgänger N$\}$
rules, V: $\max\{12 \cdot 10^{-4} \cdot 1 \cdot 10^{-1} \cdot 6 \cdot 10^{-3}$ Vorgänger Adj , $15 \cdot 10^{-4} \cdot 4 \cdot 10^{-1} \cdot 6 \cdot 10^{-3}$ Vorgänger N$\}$

Die beste POS-TAG Folge ist demnach Det N V.

Bigram-HMMs (6)

Algorithmus:

`for` q `in range(`$|Q|$`):`

$$viterbi(q, 1) = a_{0,q} \cdot b_q(o_1)$$
$$backpointer(q, 1) = 0$$

`for` i `from 2 to` n`:`

`for all` $q \in Q$`:`

$$viterbi(q, i) = \arg\max_{q' \in Q} \left(viterbi(q', i-1) \cdot a_{q',q} \cdot b_q(o_i)\right)$$
$$backpointer(q, i) = \arg\max_{q' \in Q} \left(viterbi(q', i-1) \cdot a_{q',q}\right)$$

$$viterbi(q_F, n) = \arg\max_{q' \in Q} \left(viterbi(q', n) \cdot a_{q',q_F}\right)$$

$$backpointer(q_F, n) = \arg\max_{q' \in Q} \left(viterbi(q', n) \cdot a_{q',q_F}\right)$$

Bigram-HMMs (7)

- Anschließend erhält man die beste Tagsequenz (in umgekehrter Reihenfolge), wenn man, von $backpointer(q_f, n)$ ausgehend den Backpointern folgt.
- Die Wahrscheinlichkeit dieser Tagsequenz steht in $viterbi(q_f, n)$.

Trigram-HMMs (1)

Annahme bisher: Die Übergangswahrscheinlichkeit dafür, dass ein bestimmtes Tag als nächstes folgt, ist nur von dem vorhergehenden Tag abhängig.

Erweiterung: es werden die beiden vorhergehenden Tags berücksichtigt:

$$P(t_1^n) \approx \prod_{i=1}^{n} P(t_i|t_{i-1}, t_{i-2})$$

Zusätzlich wird noch das Satzende berücksichtig, genauer die Wahrscheinlichkeit, dass das letzte Tag am Satzende steht (t_{n+1} ist ein neu eingeführtes Satzendetag):

$$\begin{aligned} \hat{t}_1^n &= \arg\max_{t_1^n} P(w_1^n|t_1^n)P(t_1^n) \\ &= \arg\max_{t_1^n} \left[\prod_{i=1}^{n} P(w_i|t_i)P(t_i|t_{i-1}, t_{i-2})\right] P(t_{n+1}|t_n) \end{aligned}$$

Trigram-HMMs (2)

Problem: Man hat oft nicht genug Daten, um alle Wahrscheinlichkeiten abschätzen zu können:

$$P(t_i|t_{i-1}, t_{i-2}) = \frac{C(t_{i-2}, t_{i-1}, t_i)}{C(t_{i-2}, t_{i-1})}$$

Viele dieser Anzahlen werden 0 sein, da man nicht alle Folgen im Trainingskorpus findet.

Lösung: Kombination von Tri-, Bi- und Unigrams:

$$P(t_i|t_{i-1}, t_{i-2}) = \lambda_3 \hat{P}(t_i|t_{i-1}, t_{i-2}) + \lambda_2 \hat{P}(t_i|t_{i-1}) + \lambda_1 \hat{P}(t_i)$$

Trigram-HMMs (3)

Berechnung der $\lambda_1, \lambda_2, \lambda_3$ z.B. durch deleted interpolation

$\lambda_1 = \lambda_2 = \lambda_3 = 0$
for each trigram t_1, t_2, t_3 with $f(t_1, t_2, t_3) > 0$:
    depending on the maximum of the values
    $x_1 = \frac{C(t_1,t_2,t_3)-1}{C(t_1,t_2)-1}$, $x_2 = \frac{C(t_2,t_3)-1}{C(t_2)-1}$ and $x_3 = \frac{C(t_3)-1}{N-1}$:
        case max $= x_1$: increment λ_3 by $C(t_1, t_2, t_3)$
        case max $= x_2$: increment λ_2 by $C(t_1, t_2, t_3)$
        case max $= x_3$: increment λ_1 by $C(t_1, t_2, t_3)$
normalize $\lambda_1, \lambda_2, \lambda_3$
(N ist die Gesamtanzahl Tokens im Korpus)

Accuracy = (Anzahl korrekter Tags) / (Anzahl Tags) liegt beim POS Tagging deutlich über 95 %.

Jurafsky, D. and Martin, J. H., editors (2009). *Speech and Language Processing. An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.* Prentice Hall Series in Articial Intelligence. Pearson Education International. Second Edition.