POS Tagging II

Katja Markert

Institut für Computerlinguistik
Uni Heidelberg
markert@cl.uni-heidelberg.de

December 6, 2019

Bisher und heute

1. Bisher: Linguistischer Hintergrund
2. Bisher: HMM Tagging
3. Jetzt: Effiziente Dekodierung des HMMs
4. Jetzt: POS Evaluation

Übersicht

Übersicht

Lösung: Viterbialgorithmus: Beispiel an der Tafel

	Trump (w_1)	deals (w_2)	are (w_3)	legal (w_4)	Ende
t_F $</s>$	–	–	–	–	
t_6 JJ					
t_5 VBZ					
t_4 NNS					
t_3 NN					
t_2 NNP					
t_1 VBE					
t_0 $<s>$					

Wir merken uns bei jedem Wort, nur die “beste” Vergangenheit per Tag. Sprich wir maximieren per Zelle!

HMMs: Formalisierung/Generalisierung

Ein HMM ist spezifiziert durch:

- $T = \{t_1, t_2, \ldots t_N\}$: N Zustände
- $A = t_{11}, t_{12}, \ldots t_{N1} \ldots t_{NN}$: $N \times N$ Matrix von Übergangswahrscheinlichkeiten; In unserem Fall $t_{lm} = p(t_m|t_l)$
- $W = w_1 w_2 \ldots w_n$ eine Sequenz von n Beobachtungen, wobei jedes w_j aus einem bestimmten Vokabular V stammt
- B eine Matrix von Emissionswahrscheinlichkeiten $p(w_j|t_i)$. Wahrscheinlichkeit, dass w_j von Zustand t_i generiert wird.
- t_0, t_F spezielle Start und Endzustände. In unserem Fall $t_0 =< s >$ und $t_F = </s >$. Haben keine Beobachtungen, aber Übergangswahrscheinlichkeiten $t_{01}, \ldots t_{0N}$ sowie $t_{1F}, t_{2F}, \ldots t_{NF}$

Constraints:

- $\sum_{m=1}^{N} t_{lm} = \sum_{m=1}^{N} p(t_m|t_l) = 1$ für alle l.
- $\sum_{j=1}^{|V|} p(w_j|t_i) = 1$ für alle i

Übergangswahrscheinlichkeiten im POS tagging

$t_{lm} : p(t_m | t_l)$

	NNP	NN	NNS	VBZ	VBE	JJ	$< /s >$
$< s >$	0.067	0.037	0.022	0.0008	0.0015	0.0017	0
NNP	0.211	0.068	0.016	0.014	0.0025	0.00004	0.037
NN	0.009	0.080	0.011	0.012	0.0027	0.00002	0.197
NNS	0.0035	0.012	0.005	0.0027	0.03	0.00004	0.292
VBZ	0.0215	0.039	0.021	0.0005	0.00006	0.124	0.029
VBE	0.004	0.0079	0.022	0.00003	0.0004	0.024	0.0054
JJ	0.0036	0.56	0.39	0.00002	0.000003	0.0005	0.0024

Constraint: Zeilenweise Addition

Emissionswahrscheinlichkeiten im POS tagging

$p(w_j|t_i)$

	Trump	deals	are	legal
NNP	0.000002	0	0	0
NN	$7.6 \cdot 10^{-7}$	0	0	0
NNS	0	0.00015	0	0
VBZ	0	0.01	0	0
VBE	0	0	0.71	0
JJ	0	0	0	0.00008

Constraint: zeilenweise Addition

Wichtige Aufgabe bei HMMs

Dekodierung

Gegeben eine Beobachtungssequenz $W = w_1, w_2 \dots w_n$ und einen HMM $\lambda = (A, B)$, bestimme die wahrscheinlichste Sequenz von Zuständen $t_{1,n} = t_1, t_2 \dots t_n$, die die Beobachtungssequenz generiert hat

1. Vollständige Berechnung ist ineffizient
2. Greedymethode funktioniert nicht und kann falsche Ergebnisse liefern
3. Lösung: Der Viterbialgorithmus berechnet die Dekodierung effizient mit dynamischer Programmierung und Rekursion

Viterbi: Idee

	Trump (w_1)	deals (w_2)	are (w_3)	legal (w_4)	Ende
$t_F = </s>$	–	–	–	–	
t_6 JJ					
t_5 VBZ					
t_4 NNS					
t_3 NN					
t_2 NNP					
t_1 VBE					
$t_0 = <s>$					

- Trellis mit $N(+2)$ Zeilen und $n(+1)$ Spalten. Zeilen haben Tags, Spalten haben Wörter
- Zelleninhalt $vit(i,j)$: Viterbipfadwahrscheinlichkeit zum Zeitpunkt j für Zustand/tag i
- Diese werden rekursiv berechnet und man merkt sich den besten Pfad per Zelle.
- Laufzeit: $O(N^2 \cdot n)$ naiv: $O(N^n)$

Viterbi-Initialisierung

Wie bei anderen Algorithmen der dynamischen Programmierung müssen wir irgendwie beginnen. Wir initialisieren die erste Spalte.

Für $1 \leq i \leq N$ setze:

$$vit(i,1) = vit(t_i, w_1) = p(t_i|t_0) \cdot p(w_1|t_i) = p(t_i| < s >) \cdot p(w_1|t_i)$$

$$backp(i,1) = 0$$

Viterbi-Rekursion

Spaltenweise Rekursion.

Für $2 \leq j \leq n$ berechne:

$$vit(i,j) = \max_{k=1}^{N}[vit(k,j-1)p(t_i|t_k)p(w_j|t_i)] \; fuer \; 1 \leq i \leq N$$

$$backp(i,j) = \operatorname{argmax}_{k=1}^{N}[vit(k,j-1)p(t_i|t_k)p(w_j|t_i)] \; fuer \; 1 \leq i \leq N$$

In die Berechnung der jeweils nächsten Viterbi-WS $vit(i,j)$ gehen ein

- $vit(k,j-1)$: der vorherige Viterbipfad für alle möglichen Zustände k
- $p(t_i|t_k)$: Übergangswahrscheinlichkeiten von möglichen Vorzuständen k
- $p(w_j|t_i)$: Emittierungswahrscheinlichkeit für w_j im Zustand t_i

Die Viterbi-WS für Zustand t_i ist jeweils die maximale WS aus diesen Faktoren für mögliche Vorzustände $k = 1 \ldots N$.

Viterbi-Terminierung

Bester score unter Berücksichtigung des Satzendes:

$$\max_{k=1}^{N} vit(k,n)p(t_F|t_k) = \max_{k=1}^{N} vit(k,n)p(</s>|t_k)$$

Anfang des Backtrace:

$$\operatorname{argmax}_{k=1}^{N} vit(k,n)p(t_F|t_k)$$

Übersicht

Logarithmen

Wir benutzen natürlich im echten Leben wie immer lieber die Logarithmen:

$$\begin{aligned}
\operatorname*{argmax}_{t_{1,n}} p(w_{1,n}|t_{1,n})p(t_{1,n}) &= \operatorname*{argmax}_{t_{1,n}} \prod_{i=1}^{n} [p(w_i|t_i) \cdot p(t_i|t_{i-1})] \\
&= \operatorname*{argmax}_{t_{1,n}} \sum_{i=1}^{n} [\log p(w_i|t_i) + \log p(t_i|t_{i-1})]
\end{aligned}$$

TnT

Brants, T. (2000). A statistical part-of-speech tagger. In: ANLP 2000

`http://www.coli.uni-saarland.de/~thorsten/`

- Trigram Tagger
- Optimiert für Geschwindigkeit
- Sprachunabhängig (?)
- Optimiert Smoothing
- Optimiert Behandlung unbekannter Wörter

Trigram Tagging

- Schaue zwei Tags zurück bei den Übergangswahrscheinlichkeiten
- Bei den Emissionswahrscheinlichkeiten ändert sich nichts.

$$\prod_{i=1}^{n}[p(w_i|t_i) \cdot p(t_i|t_{i-1}, t_{i-2})]p(t_{n+1}|t_n)$$

Smoothing in TnT

Interpolation über Maximum-Likelihood $\hat{p}$

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

mit $\lambda_3 + \lambda_2 + \lambda_1 = 1$.

Schätzung für unbekannte Wörter

Emissionswahrscheinlichkeit $p(w|t)$, wenn wir w nicht kennen?
Ideen?

1. barabaling
2. Metamind
3. inmaniant

Unbekannte Wörter in TnT

Benutze Suffixe: letzte m Buchstaben eines Wortes w der Länge L

$$p(t|w) = p(t|l_{L-m+1}, \ldots l_L).$$

Beispiel:

$$p(\mathit{VBG}|\mathit{barabaling}) = p(\mathit{VBG}|\mathit{word\ ends\ with\ ing}\ )$$

Aber wir wollten doch $p(w|t)$??

Zusammenfassung

Hidden Markov Modelle für Tagging

- generativ, überwacht
- zahlreiche Unabhängigkeitsannahmen
- Benutzt Emissions- und Übergangswahrscheinlichkeiten
- Übergangswahrscheinlichkeiten: sehr kurze Historie
- Wissen muss nicht enkodiert werden
- Kann an sich nicht vorausschauen $\Longrightarrow$ kann umgangen werden...
- Schwierig, um andere Abhängigkeiten zu erweitern $\Longrightarrow$ CRFs oder MEMMs (siehe Buch 8.4.5)

Übersicht

Wie schwer ist Tagging?

Word Types:

Types	WSJ	Brown
Nicht ambig	44,432 (86%)	45,799 (85%)
Ambig	7,025 (14%)	8,050 (15%)

Word tokens:

Tokens	WSJ	Brown
Nicht ambig	577,421 (45%)	384,349 (33%)
Ambig	711,780 (55%)	786,646 (67%)

Baseline Unigram Tagger (MFC): 92.34% auf WSJ

Upper bound: Menschliches tagging 96-100%

Derzeitige Performanz

Beste statistische Tagger: Um die 97% Akkuratheit auf Penn Treebank (TnT, Stanford Tagger)
Mit neuronalen Netzen (LSTMs): 97.40 (Wang et al, 2015)

Kleine Verbesserungen können auf einen gesamten Text gerechnet einen grossen Unterschied ausmachen

Performanz abhängig von

- Sprache
- Genre: Twitterdaten (siehe speziellen Twittertagger unter `http://www.cs.cmu.edu/~ark/TweetNLP/`)
- Anzahl Training
- Tagset
- Unbekannte Wörter

Tagger fürs Deutsche

TnT: HMMs, Stanford: Bidirektionale MeMMs

Ergebnisse für TIGER:

	TnT	Stanford
Akk. gesamt	96.92	97.63
Akk. bekannt	97.59	?
Akk. unbekannt	89.16	91.66
Prozent unbekannt	7.85	7.52

Tagger fürs Deutsche

TnT: HMMs, Stanford: Bidirektionale MeMMs

Ergebnisse für 10, 1000 Token aus WebKorpus deWAC:

	TnT	Stanford
Akk. gesamt	92.69	92.61
Akk. bekannt	95.90	?
Akk. unbekannt	71.99	75.35
Prozent unbekannt	13.44	13.00

Konfusionsmatrix

Zeigt typische Fehler (hier Ausschnitt von einem HMM tagger). Gold Standard in Spalte, Systemausgabe in Zeile.

	NN	IN	VB	JJ
NN	8844	37	95	235
IN	2	4402	0	0
VB	76	2	2965	5
JJ	58	12	6	1921

Precision, recall, F-Measure können für jede Klasse wie üblich berechnet werden. Micro/Macroaveraging danach möglich.

Demo

Der POS Tagger des Stanford CoreNLP Systems:
`http://corenlp.run/`

Kein HMM; aber auch ein statistisches Modell mit Viterbi zur Dekodierung, aber diskriminativ, nicht generativ.

Aufgaben

- * Jurafsky und Martin, Online 3rd edition, Kapitel 8 (bis aussschließlich 8.4.5)
- Aufgabenblatt 7

Referenzen

- Petrov, Das, McDonald: A Universal Part-of-Speech Tagset. In Lrec 2012
- Voutilainen, A. (1999). Handcrafted rules. In: *Syntactic wordclass tagging*
- Brants, T. (2000). A statistical part-of-speech tagger. In: ANLP 2000
- Wang P, Qian Y, Soong FK, He L, Zhao H. (2015): Part-of-speech tagging with bidirectional long short-term memory recurrent neural network. arXiv preprint arXiv:1510.06168. 2015
- Youtube Videos über HMM im allgemeinen (Danke, Philipp):
  - `https://www.youtube.com/watch?v=6JVqutwtzmo`
  - `https://www.youtube.com/watch?v=kqSzLo9fenk&t=294s`