Schriftliche Wiederholungsprüfung zur Übung
Statistische Methoden in der maschinellen Sprachverarbeitung
SS 2020
Dozent: Helmut Schmid

Sie haben 60 Minuten Zeit plus 5 Minuten zum Absenden.

Allgemeiner Hinweis: Wenn Sie einen Fehler oder ein anderes Problem in einer der Aufgaben entdecken sollten, dann schicken Sie mir bitte eine Nachricht an `schmid@cis.lmu.de`.

Sie sollen den **Forward-Backward-Algorithmus** für einen **Trigramm**-Tagger implementieren. Die Verarbeitung erfolgt wie immer satzweise.

Aufgabe 1) Forward-Algorithmus

Definition der Forward-Wahrscheinlichkeiten $\alpha_{t',t}(k)$

$$\begin{aligned}\alpha_{t',t}(0) &= \begin{cases} 1 & \text{falls } t = t' = \langle s \rangle \\ 0 & \text{sonst} \end{cases} \\ \alpha_{t',t}(k) &= \sum_{t'' \in T} \alpha_{t'',t'}(k-1)\; p(t|t'',t')\; p(w_k|t) \qquad \text{für } 1 \le k \le \mathbf{n+2}\end{aligned}$$

t, t' und t" sind hier Tags, k ist eine Wortposition, $\langle s \rangle$ ist das Satzgrenzen-Tag. $w_1, ..., w_n$ sind die Wörter des Satzes. w_{-1}, w_0, w_{n+1} und w_{n+2} sind Satzgrenzentokens (bspw. der leere String). Die Wahrscheinlichkeit des Satzgrenzentokens gegeben das Satzgrenzentag ist 1 und sonst 0.

Achtung: Sie iterieren hier bis n+2 statt nur bis n+1. Auch die forward-Tabelle ist entsprechend größer.

Sie können den Forward-Algorithmus ähnlich wie den Viterbi-Algorithmus implementieren. Die Maximierung des Viterbi-Algorithmus muss aber durch eine Summe ersetzt werden, und es werden keine Rückwärtszeiger (best_prev) benötigt.

Schreiben Sie Code zur Berechnung der Forward-Wahrscheinlichkeiten. Folgende Variablen und Funktionen sind gegeben und müssen nicht implementiert werden:

- *words*: eine Liste mit den Wörtern des aktuellen Satzes
- *tagset*: eine Liste mit allen vorhandenen Tags
- *lexprob(w,t)*: eine Funktion, welche $p(w|t)$ zurückliefert.
- *contextprob(t",t',t)*: eine Funktion, die $p(t|t'',t')$ zurückliefert

(10 Punkte)

Aufgabe 2) Backward-Algorithmus

Definition der Backward-Wahrscheinlichkeiten $\beta_{t'',t'}(k)$:

$$\begin{aligned}\beta_{t'',t'}(n+2) &= \begin{cases} 1 & \text{falls } t'' = t' = \langle s \rangle \\ 0 & \text{sonst} \end{cases} \\ \beta_{t'',t'}(k) &= \sum_{t \in T} p(t|t'',t')\; p(w_{k+1}|t)\; \beta_{t',t}(k+1) \qquad \text{für } 0 \le k \le n+1\end{aligned}$$

Schreiben Sie Code zur Berechnung der Backward-Wahrscheinlichkeiten. Die Berechnung der Backward-Wahrscheinlichkeiten geht ähnlich wie die Berechnung der Forward-Wahrscheinlichkeiten, aber Sie iterieren in umgekehrter Reihenfolge n+1,...,0 und initialisieren die letzte Spalte der Tabelle. (10 Punkte)

Aufgabe 3) Berechnung der geschätzten Häufigkeiten

Nun berechnen Sie noch die geschätzten Häufigkeiten von Tag-Trigrammen:

$$\gamma_{t'',t',t}(k) \quad = \quad \frac{\alpha_{t'',t'}(k-1)\ p(t|t'',t')\ p(w_k|t)\ \beta_{t',t}(k)}{\alpha_{\langle s\rangle,\langle s\rangle}(n+2)} \qquad \text{für } 1 \leq k \leq n+2$$

Sie iterieren erneut über alle Elemente der Forward-Matrix und berechnen die γ-Werte für die Tag-Tripel. Summieren Sie diese Werte für die einzelnen Tag-Trigramme in einem (bereits vorhandenen) Dictionary *freq*, welches Tag-Tripel auf ihre erwartete Häufigkeit abbildet. (10 Punkte)

(30 Punkte insgesamt)

Viel Erfolg!