

Und es wäre cool wenn wir im Zuge dessen auch folgende Frage behandeln könnten
(Übungsaufgaben_alle Themen Folie 40): *Nennen Sie 3 Beispielanwendungen von Sprachmodellen in der Computerlinguistik.*

Language modeling is used in speech recognition, machine translation, part-of-speech tagging, parsing, Optical Character Recognition, handwriting recognition, information retrieval and other applications.

https://en.wikipedia.org/wiki/Language_model

10.01:



Bitte aus dem Foliensatz Übungsaufgaben_alle_Themen.pdf slide 58 (Aufgabe 4) besprechen

Aufgabe 4) Eine geglättete Wahrscheinlichkeitsverteilung sei wie folgt definiert:

$$p(w|w') = p^*(w|w') + \alpha(w')p(w) \quad \text{mit} \quad p^*(w|w') = \frac{f(w', w) - \delta}{\sum_{w'} f(w', w')}$$

Leiten Sie die Formel für die Berechnung des Backoff-Faktors $\alpha(w')$ her. (3 Punkte)

Aufgabe 4) Eine geglättete Wahrscheinlichkeitsverteilung sei wie folgt definiert:

$$p(w|w') = p^*(w|w') + \alpha(w')p(w) \quad \text{mit } p^*(w|w') = \frac{f(w', w) - \delta}{\sum_{w'} f(w', w')}$$

Leiten Sie die Formel für die Berechnung des Backoff-Faktors $\alpha(w')$ her. (3 Punkte)

Backoff-Faktor für interpolierte Backoff-Glättung

Der Backoff-Faktor $\alpha(C)$ stellt sicher, dass die Wahrsch. zu 1 summieren:

$$\sum_w p(w|C) = \sum_w \frac{\max(0, f(C, w) - \delta)}{f(C)} + \alpha(C)p(w|C') = 1$$

Wenn $C = w_1^{n-1}$ dann $C' = w_2^{n-1}$

Durch Umformen erhalten wir:

$$\sum_w \alpha(C)p(w|C') = 1 - \sum_w \frac{\max(0, f(C, w) - \delta)}{f(C)}$$

Ausklammern von $\alpha(C)$ liefert:

$$\underbrace{\alpha(C) \sum_w p(w|C')}_{=1} = 1 - \sum_w \frac{\max(0, f(C, w) - \delta)}{f(C)}$$

Dies ist äquivalent zu:

$$\alpha(C) = 1 - \sum_{w: f(C, w) > 0} \frac{f(C, w) - \delta}{f(C)}$$

Übung: diese Aufgabe lösen
(verwende die Notation aus
der Aufgabestellung)

Aufgabe 4) Eine geglättete Wahrscheinlichkeitsverteilung sei wie folgt definiert:

$$p(w|w') = p^*(w|w') + \alpha(w')p(w) \quad \text{mit} \quad p^*(w|w') = \frac{f(w', w) - \delta}{\sum_{w'} f(w', w')}$$

Leiten Sie die Formel für die Berechnung des Backoff-Faktors $\alpha(w')$ her. (3 Punkte)

$$\sum_w p(w|w') = \sum_w (p^*(w|w') + \alpha(w')p(w)) = 1$$

$$\sum_w p^*(w|w') + \sum_w \alpha(w')p(w) = 1$$

$$\cancel{\sum_w p^*(w|w')} + \sum_w \alpha(w')p(w) - \cancel{\sum_w p^*(w|w')} = 1 - \sum_w p^*(w|w')$$

$$\alpha(w') \underbrace{\sum_w p(w)}_1 = 1 - \sum_w p^*(w|w')$$

$$\alpha(w') \cdot 1 = 1 - \sum_w p^*(w|w')$$

$$\alpha(w') = 1 - \sum_w p^*(w|w')$$

Die Summe von aller $p(w)$ ist 1 ,
weil p eine
Wahrscheinlichkeitsverteilung ist.

Wortbedeutungsdesambiguierung



Implementieren Sie einen Wortbedeutungsdesambiguierer auf Basis eines Naive-Bayes-Modelles. Auch hier schreiben Sie eine Funktion desamb(word, tokens), welche das zu desambiguierende Wort und einen Text in Form einer Liste von Tokens als Argumente nimmt. Folgende Funktionen sind gegeben:

- Die Funktion `senses(word)` liefert die Liste der Bedeutungen des in "word" gespeicherten Wortes. (Bspw. könnte für das Wort Bank die Liste {Bank1, Bank2, Bank3} zurückgegeben werden.
- `wordProb(word, sense)` liefert die Wahrscheinlichkeit des Wortes "word" im Kontext der Bedeutung "sense" des zu desambiguierenden Wortes. Der Rückgabewert der Funktion ist immer größer als Null.
- `priorProb(sense)` liefert die Apriori-Wahrscheinlichkeit der Bedeutung "sense". Sie sollen in der Tokenliste nach Vorkommen des zu desambiguierenden Wortes suchen und diese auf Basis der Kontextwörter mit maximalem Abstand 50 desambiguieren. Ausgegeben werden soll jeweils die Position des desambiguierten Wortes und die zugewiesene Bedeutung. Sie können die Ergebnisse entweder auf die Konsole ausgeben oder in einer Liste von Paaren (Position, Bedeutung) zurückgeben.

"ich gehe zur **Bank** um Geld abzuheben und dann setze ich mich auf eine **Bank**"
`desamb("Bank", "ich gehe zur Bank um Geld abzuheben".split())`
`def desamb(word,tokens):`

```
....  
    print( position, bedeutung)  
# (3, Bank1)  
# (14, Bank2)
```

$$\hat{s} = \arg \max_s p(s|C)$$

$$= \arg \max_s \frac{p(s, C)}{p(C)}$$

$$= \arg \max_s p(s, C)$$

Naïve Bayes-Klassifikator

$$\hat{s} = \arg \max_s p(s) \prod_{i=1}^n p(w_i|s)$$

Apriori von Bedeutung s

WK von word w_i gegeben Bedeutung s

```

def desamb(word,tokens):
    word_index_and_contexts = []
    # [(3, ['ich', 'gehe', 'zur', 'und', 'geld', 'aufheben'])],
    max_dist = 3
    # word in tokens suchen und index speichern
    for i in range(len(tokens)):
        if tokens[i] == word:
            contexts = []
            # Kontextwörter extrahieren
            start = max(0, i-max_dist)
            end = min(len(tokens), i+max_dist+1)
            for k in range(start, end):
                if k != i:
                    contexts.append(tokens[k])
            word_index_and_contexts.append((i, contexts))

    print(word_index_and_contexts)
    for word_index, contexts in word_index_and_contexts:
        best_score = None
        best_sense = ""
        for s in senses(word):
            score = priorProb(s)
            for c in contexts:
                score *= wordProb(c,s)
            if best_score is None:
                best_score = score
                best_sense = s
            if score > best_score:
                best_score = score
                best_sense = s
    print(word_index,best_sense)

```

Naïve Bayes-Klassifikator

$$\hat{s} = \arg \max_s p(s) \prod_{i=1}^n p(w_i|s)$$

senses(word) -> liste von Bedeutungen
 # sense("bank") -> [bank1, bank2,bank3]

wordProb(word,sense) -> wk von word gegeben
 sense. Also $p(w_i|s)$

priorProb(sense) -> $p(s)$

desamb("bank","ich gehe zur bank und geld aufheben und sitze auf eine bank im park".split())

Aufgabe 2)

Zeigen Sie, dass die Glättung mit dem Addiere- λ -Verfahren äquivalent zu einer Interpolation (gewichteten Mittelung) mit der uniformen Verteilung ist, d.h. zeigen Sie, dass folgende Gleichung gilt:

$$\frac{f(w_1^n) + \lambda}{N + B\lambda} = \mu \frac{f(w_1^n)}{N} + (1 - \mu) \frac{1}{B} \quad \text{mit } \mu = \frac{N}{N + B\lambda}$$

$$\begin{aligned} \frac{f(w) + \lambda}{N + B\lambda} &= \frac{N}{N + B\lambda} \cdot \frac{f(w)}{N} + \left(1 - \frac{N}{N + B\lambda}\right) \cdot \frac{1}{B} \\ &= \frac{\cancel{N}}{N + B\lambda} \cdot \frac{f(w)}{\cancel{N}} + \left(1 - \frac{\cancel{N}}{N + B\lambda}\right) \cdot \frac{1}{B} \\ &= \frac{f(w)}{N + B\lambda} + \frac{\cancel{N} + B\lambda - \cancel{N}}{N + B\lambda} \cdot \frac{1}{B} \\ &= \frac{f(w)}{N + B\lambda} + \frac{\cancel{B}\lambda}{N + B\lambda} \cdot \frac{1}{\cancel{B}} \\ &= \frac{f(w) + \lambda}{N + B\lambda} \end{aligned}$$