Übung zur Vorlesung "Computerlinguistik II / Sprachtechnologie"

Sommersemester 2018, Prof. Dr. Udo Hahn, Tobias Kolditz
Übungsblatt 4 vom 28.05.2018
Abgabe bis 04.06.2018 per E-Mail (PDF- & Python-Datei) an
tobias.kolditz@uni-jena.de

Aufgabe 1 : State Emission Probabilities — 3 pt

Schreiben Sie eine Python-Funktion `emission_prob(word, tag)`, welche die *state emission probability* für ein Wort unter der Bedingung eines bestimmten Tags zurückgibt. Die Emissionswahrscheinlichkeiten sollen auf Grundlage eines getaggten Korpus (einer Liste von Wort-Tag-Tupeln) berechnet werden, zum Beispiel:

```
from nltk.corpus import brown

# [('The', 'DET'), ('Fulton', 'NOUN'), ('County', 'NOUN'), ... ]
tagged_corpus = list(brown.tagged_words(tagset='universal')[:100000])
```

Orientieren Sie sich bei der Implementierung am Code der letzten Übung und achten Sie auf eine möglichst geringe Zeitkomplexität: Das Programm soll nur einmal über das gesamte Korpus iterieren – auch wenn die Funktion mehrmals aufgerufen wird! Testen Sie Ihren Code, z.B. mit ambigen Wörtern wie 'show' (NOUN/VERB).

Aufgabe 2 : Theoretisches — 3 pt

a) Zipfsches Gesetz — 1 pt

Erläutern Sie **kurz** (am besten anhand eines Beispiels) das Zipfsche Gesetz in Bezug auf die Linguistik.

b) Markow-Annahme — 2 pt

Stellen Sie (formal und in eigenen Worten) die Annahme dar, die hinter Markow-Modellen steht. Warum approximieren wir frequenzbasierte Wahrscheinlichkeiten mit Markow-Ketten, statt sie mathematisch exakt über die Kettenregel für Wahrscheinlichkeiten zu berechnen?

Aufgabe 3 Chunking — 2 pt

Nehmen Sie sich den von Ihnen in Aufgabe 2, Übungsblatt 1 annotierten Text und gruppieren Sie dessen Tokens in Chunks. Verwenden Sie dafür das IOB-Format (auch: BIO-Format).

Aufgabe 4 : Named-Entity Recognition (NER) — 5pt

Gegeben sei der folgende fachsprachliche Text:

> Als Leucine fasst man zunächst die vier isomeren Aminosäuren Leucin, Isoleucin, tert-Leucin und Norleucin zusammen. Im Vergleich mit den vier Butanolen kann man sie als butylsubstituierte Glycine auffassen; damit sind alle vier Varianten vertreten.
> Leucin und Isoleucin gehören zu den proteinogenen Aminosäuren, d. h. sie sind Bausteine der Proteine von Lebewesen und über den genetischen Code kodiert.
> Berücksichtigt man noch die Stereoisomerie, so sind noch 6 weitere Isomere hinzu zu rechnen: (a) D-Leucin, (b) D-Isoleucin, (c) L-allo-Isoleucin, (d) D-allo-Isoleucin, (e) D-tert-Leucin und (f) D-Norleucin. Sie werden jeweils unter den ihnen zugehörigen Aminosäureartikeln beschrieben.

a) Regulärer Ausdruck — 3 pt

Erstellen Sie **einen** regulären Ausdruck um sämtliche Leucin-Varianten im Text zu finden. Ihr Ausdruck kann auch auf 'Leucine' passen, die nicht im Text erwähnt werden, darf im Beispieltext aber keine Fehler machen. Orientieren Sie

sich an der Folie mit der Überschrift “Regular Expression Basics” des aktuellen Teils der Vorlesung. Sie können [A-Z] benutzen um alle Großbuchstaben abzudecken. Wenn Sie Zeichen mit Spezialbedeutungen wie "." oder "-" abdecken wollen, so müssen Sie diese mit einem vorangestelltem "\" escapen, z.B. "\.".

Hinweis: Sie können Ihren Ausdruck mit Python überprüfen.

b) NER-Ansätze 2 pt

Welche anderen Ansätze zur Entitätenerkennung könnten verwendet werden, um die restlichen biochemischen NEs im Text zu finden?