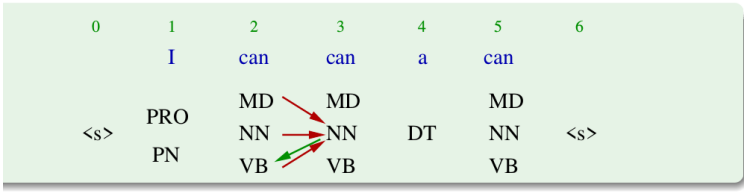
Übung7: Wortart-Tagger (Training)
Übung8: Wortart-Tagger (Anwenden)

P(Wort|Tag)

$$p(w_1^n, t_1^n) \propto \prod_{i=1}^{n+1} p(t_i|t_{i-2}, t_{i-1}) p(t_i|w_i) / p(t_i)$$

Im Training wollen wir die Parametern p(tag| tag_context) und p(word|tag) berechnen und speichern in geeigneten Datenstrukturen.



$$\delta_{NN}(3) = \max \begin{pmatrix} \delta_{MD}(2)\ p(NN|MD)\ p(can|NN), \\ \delta_{NN}(2)\ p(NN|NN)\ p(can|NN), \\ \delta_{VB}(2)\ p(NN|VB)\ p(can|NN) \end{pmatrix}$$

In Taggen-Schritt (Anwendung) wollen wir die Tagfolge eines neuen Textes vorhersagen/bestimmen.

Wir bestimmen die beste Tagfolge, indem wir den Viterbi-Algorithmus anwenden.

Für die Berechnung einer Viterbi-WK brauchen wir HMM-Parameter.

p(tag| tag_context) und p(word|tag) schätzen wir aus Wort-Tag und Tag-Tag Häuftigkeiten.

$$\begin{aligned} p(t|w) &= \frac{f(w,t) - \delta_6}{\sum_{t'} f(w,t')} + \alpha(w) p(t|a_{n-4}^n g(w)) \\ p(t|b_1^k g) &= \frac{f(b_1^k g, t) - \delta_k}{\sum_{t'} f(b_1^k g, t')} + \alpha(b_\square^k g) p(t|b_2^k g) \text{ für } 0 < k < 5 \\ p(t|g) &= \frac{f(g,t)}{\sum_{t'} f(g,t')} \end{aligned}$$

$$p(w_1^n, t_1^n) \propto \prod_{i=1}^{n+1} p(t_i|t_{i-2}, t_{i-1}) p(t_i|w_i) / p(t_i)$$

Apriori-Tagwahrscheinlichkeiten $p(t) = f(t) / \sum_{t'} f(t')$

- Kontextwahrscheinlichkeiten $p(t|t', t'')$

  Diese Wahrscheinlichkeiten sollen mit Kneser-Ney-Glättung geglättet werden. Sie müssen hier nur die relativen Häufigkeiten $p^*(t|t', t'')$, $p^*(t|t'')$ und $p^*(t)$ (vgl. Übung 3) berechnen. $p(t|t', t'')$ wird erst im eigentlichen Taggerprogramm berechnet.

$$
\begin{aligned}
p(t|w) &= \frac{f(w,t)-\delta_6}{\sum_{t'} f(w,t')} + \alpha(w)p(t|a_{n-4}^{n}g(w)) \\
p(t|b_1^k g) &= \frac{f(b_1^k g,t)-\delta_k}{\sum_{t'} f(b_1^k g,t')} + \alpha(b_\diamond^k g)p(t|b_2^k g) \text{ für } 0<k<5 \\
p(t|g) &= \frac{f(g,t)}{\sum_{t'} f(g,t')}
\end{aligned}
$$

Apriori-Tagwahrscheinlichkeiten $p(t) = f(t)/\sum_{t'} f(t')$

- Kontextwahrscheinlichkeiten $p(t|t',t'')$

  Diese Wahrscheinlichkeiten sollen mit Kneser-Ney-Glättung geglättet werden. Sie müssen hier nur die relativen Häufigkeiten $p^*(t|t',t'')$, $p^*(t|t'')$ und $p^*(t)$ (vgl. Übung 3) berechnen. $p(t|t',t'')$ wird erst im eigentlichen Taggerprogramm berechnet.

Corpus:

the dog is hungry
DT NN VB ADJ

dog loved fish
NN VBD NN

cat does not want to eat
NN MD RB VB P MD

Parameter schätzen:
Für p(w|t) brauchen wir:
p*(t|w)
p*(t| suffix(w))
p(t)

Für p(t| tag context) brauchen wir
p(t| t',t'')

*backoff factor wird später berechnet

Extract f(w, t) and f(t', t'', t)

Z.B.
f(the, DT): 1
f(dog, NN): 2

f(DT, NN, VB): 1
f(NN, VB, ADJ): 1

```
# Modell in einer Datei speichern
with open(sys.argv[2],'wb') as file:
    model = (max_suff_len, apriori_tag_prob, word_tag_prob,
             suffix_tag_prob, tag_ngram_prob)
    pickle.dump(model, file)
```

Wie berechnen wir p*(t|w), p*(t| suffix(w)), p(t)?

Dafür brauchen wir Wort-Tag-HF und denDiscount

p*(t|w)

$$p(t|w) = \frac{f(w,t) - \delta_6}{\sum_{t'} f(w,t')} + \alpha(w) p(t|a_{n-4}^{n} g(w))$$

$$p(t|b_1^k g) = \frac{f(b_1^k g, t) - \delta_k}{\sum_{t'} f(b_1^k g, t')} + \alpha(b_\circ^k g) p(t|b_2^k g) \text{ für } 0 < k < 5$$

$$p(t|g) = \frac{f(g,t)}{\sum_{t'} f(g,t')}$$

p*(t| suffix(w))

Dafür brauchen wir Suffix(w)-Tag-HF.

Für p(t), brauchen wir die Tag-Häufigkeiten für alle Tag t

- Apriori-Tagwahrscheinlichkeiten $p(t) = f(t) / \sum_{t'} f(t')$

Beispiel von Tag-Häufigkeiten
f(NN, DT): 5
f(DT,DT): 0

Datenstruktur Beispiel

```
# Wort-Tag-Paare und Tag-Trigramme extrahieren
word_tag_freq  = defaultdict(lambda: defaultdict(int))
tag_ngram_freq = defaultdict(lambda: defaultdict(int))
```

```
tag_freq = defaultdict(int)
```

p*(t|w)
p*(t|t', t'')

prob[context] = {t: p*(t| context), …...}