Politeness analysis: POS tagging

Carla Groenland (10208429), Joost van Doorn (10805176)

November 10, 2014

Given a sequence of words $w_1, \dots, w_T$, we calculate

$$\text{argmax}_{t_1,\dots,t_T} \prod_{t=1}^{T} [P(t_i|t_{i-1}, t_{i-2})P(w_i|t_i)]P(t_{T+1}|t_T)$$

for $t_1, \dots, t_T$ in the tag set and $t_0, t_{-1}/t_{T+1}$ begin/end of sentence markers (where sentences end with .!?;). The probabilities can be estimated by

$$\hat{P}(A|B) = \frac{\text{frequency of } A \text{ and } B \text{ together}}{\text{frequency of } B}.$$

Several components can be added:

- Capitalization: double the amount of tags, by keeping track of whether a word is capitalized in the tag.
- Smoothing: apply linear interpolation,

  $$P(t_3|t_1, t_2) = \lambda_1 \hat{P}(t_3) + \lambda_2 \hat{P}(t_3|t_2) + \lambda_3 \hat{P}(t_3|t_1, t_2)$$

  where $\lambda_1 + \lambda_2 + \lambda_3 = 1$ are fixed and need to calculated using *deleted interpolation*. See Figure 1 in the article for pseudo code; the convention $\frac{0}{0} = 0$ is used, f denotes frequencies and N is total number of tags.
- Handling of unknown words: let w be an unknown word of length n, consisting of letters $l_1, \dots, l_n$ (keep into account capitalization!). Select all words that appear at most 10 times and put those (with counts) in the word dictionary W. We are going to estimate $p(w|t)$ based on the last m letters of w, where

  $$m = \min\{10, \text{length of longest (equal) suffix in } W\}.$$

  Beforehand, we have calculated (based on W) the values of weight

  $$\theta = \frac{1}{s-1} \sum_{j=1}^{s} (\hat{P}(t_j) - \text{mean}\hat{P}))^2,$$

  i.e. the standard deviations of the (unsmoothed) maximum likelihood estimates for the tags. We set $P(t) = \hat{P}(t)$, for $i = 0, \dots, m$ we recursively do

  $$P(t|l_{n-i+1}, \dots, l_n) = \frac{\hat{P}(t|l_{n-i+1}, \dots, l_n) + \theta P(t|l_{n-i}, \dots, l_n)}{1 + \theta})$$

for

$$\hat{P}(t|l_{n-i+1}, \ldots, l_n) = \frac{f(t, l_{n-i+1}, \ldots, l_n)}{f(l_{n-i+1}, \ldots, l_n)}$$

as before. By choice of m, $P(l_{n-m+1}, \ldots, l_n) \neq 0$ and we can use Bayes rule to obtain

$$P(w|t) \approx P(l_{n-m+1}, \ldots, l_n|t) = \frac{f(l_{n-m+1}, \ldots, l_n)P(t|l_{n-m+1}, \ldots, l_n)}{f(t)}.$$

Using the components above, we make functions

$$\mathit{emission}(w, t) = P(w|t) \quad \text{and} \quad \mathit{transition}(t_1, t_2, t_3) = P(t_3|t_2, t_1).$$

The implementation of Viterbi can be speed-up using Beam search (Section 2.5).