Predictive Typing System
Text Analysis and Retrieval

Matija Šantl
Mihael Šafarić
Faculty of Electrical Engineering and Computing

Abstract

I. Introduction

II. Methods

In this section, we survey two smoothing algorithms for n-gram models, Witten-Bell smoothing and Kneser-Ney smoothing.

The first smoohting algorithm we're going to describe is the Witten-Bell smoothing algorithm. Witten-Bell smoothing algorithm is a very simple technique that performs rather poorly [**?**]. Next, we describe the Kneser-Ney smoothing algorithm. Kneser-Ney smoothing works very well and it outperforms Witten-Bell smoothing algorithm.

1. Witten-Bell smoothing

2. Kneser-Ney smoothing

Kneser-Ney smoothing algorithm was introduced in 1995. as an extenstion of absolute discounting where the lower-order distributions that one combines with a higher-order distribution is built in a novel manner [1].

Next we'll present the mathematicall background of the Kneser-Ney smoothing algorithm.

Considering bigram models, we would like to select a smoothed distribution p_{KN} that satisfies the following constraint on unigram marginals for all w_i:

$$\sum_{w_{i-1}} p_{KN}(w_{i-1}w_i) = \frac{c(w_i)}{\sum_{wi} c(w_i)} \quad (1)$$

where the funcion $c(w_i)$ denotes the count of the word w_i in the given corpus. The left hand-side of this equation is the unigram marginal for w_i of the smoothed bigram distribution p_{KN}, and the right-hand side is the unigram frequency of w_i found in the given corpus.

As in absolute discounting where $0 \leq D \leq 1$, we assume that the model has the following form:

$$p_{KN}(w_i|w_{i-n+1}^{i-1}) = \frac{max(c(w_{i-n+1}^{i}) - D, 0)}{\sum_{w_i} c(w_{i-n+1}^{i})} + \frac{D}{\sum_{w_i} c(w_{i-n+1}^{i})} N_{1+}(w_{i-n+1}^{i-1}\cdot) p_{KN}(w_i|w_{i-n+2}^{i-1}) \quad (2)$$

where

$$N_{1+}(\cdot w_i) = |w_i : c(w_{i-1}w_i) > 0| \quad (3)$$

is the number of different words w_{i-1} that precede w_i in the given corpus.

We used this formulation, because as stated in [1], it leads to a cleaner derication of essentially the same formula; no approximations are required, unlike in the original derivation.

By applying the law of total probability, we can write equations given above as following:

$$p_{KN}(w_{i-1}w_i) = \sum_{w_{i-1}} p_{KN}(w_i|w_{i-1}) p(w_{i-1}) \quad (4)$$

which leads to:

$$\frac{c(w_i)}{\sum_{wi} c(w_i)} = \sum_{w_{i-1}} p_{KN}(w_i|w_{i-1}) p(w_{i-1}) \quad (5)$$

Taking into account that $p(w_{i-1}) = \frac{c(w_{i-1})}{\sum_{wi-1} c(w_{i-1})}$, we have

$$c(w_i) = \sum_{w_{i-1}} c(w_{i-1}) p_{KN}(w_i|w_{i-1}) \quad (6)$$

which, after substituting and simplifying leads to the following form:

$$c(w_i) = c(w_i) - N_{1+}(\cdot w_i) + D p_{KN}(w_i) \sum_{w_i} N_{1+}(\cdot w_i) \quad (7)$$

Generalizing to higher-order models, we have the final form for the word probability:

$$p_{KN}(w_i|w_{i-n+2}^{i-1}) = \frac{N_{1+}(\cdot w_{i-n+2}^{i})}{\sum_{w_i} N_{1+}(\cdot w_{i-n+2}^{i})} \quad (8)$$

III. Results

IV. Discussion

References

[1] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–393, 1999.