Natuurlijke Taalmodellen & Interfaces
Tejaswini Deoskar
Part A (Step 3)

Lucas van Berkel, 10747958
Ruben Gerritse, 10760326
Gideon Mooijen, 10686290

Report

During the third step of part A, we extend our program to implement a technique called smoothing.

Introduction

For natural language purposes - *regarding calculating probabilities* - smoothing is used to make a more realistic view of the occurrences of sentences in a language. Without smoothing, the values of the calculated probabilities will result in a skewed view on possible sentences. N-grams that are assigned a value 0 - *because they do not occur in the corpus* - will be regarded as an impossible combinations of words, rather than an unlikely combination of words. Due to the distinct computational ceiling it is impossible to calculate all exact probabilities of n-grams. To solve this problem, we apply smoothing to be able to make more realistic assumptions about these probabilities. This results in better generalizations and thus we obtain a better view of the language and the occurring n-grams, even though the corpus we used is finite.

General

The fundament of smoothing methods is commonly defined as *'steal from the rich and give to the poor'*. The probabilities of high-frequency sentences will fall out lower, and this gap will be distributed among the sentences with low frequencies.

Methods

1. **Add-1 smoothing (Laplace smoothing).** This is the most trivial type of smoothing. Every occurrences count is increased by one. Then, all frequency counts are normalized. This is necessary because the resulting probabilities must be of the *'normal distribution'*-type.

2. **Good-Turing smoothing.** A different, more sophisticated method to calculate probabilities of unseen events. r* is considered the new assigned probability and is calculated through the following equation.

$$r^* = \frac{(r+1)\frac{n_{r+1}}{n_r} - r\frac{(k+1)n_{k+1}}{n_1}}{1 - \frac{(k+1)n_{k+1}}{n_1}}$$

   We only applied this smoothing for bigrams that appeared less than 5 times in the corpus.

Results

The percentages of sentences that are assigned a probability of 0 are:

No smoothing	0.45%
Add-1 smoothing	0.0%
Good-Turing smoothing	0.0%

The first five sentences that are assigned a probability of 0 are:

	No smoothing	*Add-1 smoothing*	*Good-Turing smoothing*
1	blessed her	the crow's	Mary in
2	It cannot	the Marys	consulted Lady
3	at We	the limited	resentful I
4	o'clock He	the House	overspread Anne's
5	your whole	the wreck	pleasure One

The probabilities in the table above were not exactly 0. For Add-1 smoothing we printed 5 arbitrary sentences with the lowest available probability (value: $2.824 \cdot 10^{-5}$). For Good-Turing smoothing, the same applies. There are no sentences that are assigned a probability of 0. So we executed the same, five arbitrary sentences with the lowest present probability. All these sentences have the value of: $3.511 \cdot 10^{-6}$. We can conclude that Add-1 smoothing assigns a higher probability to unseen sequences than Good-Turing's method.

It makes perfect sense that after smoothing there are no sentences that are assigned a probability of 0. This would prevent the appearance of new sentences, which would be an unrealistic model. This proves why smoothing is a helpful tool in natural language applications.

Running

The code can be executed by running the following code.

```
python BerkelGerritseMooijen3.py -train-corpus [path] -test-corpus [path] -n [value]
-smoothing [no|add1|gt]
```