Smoothing Techniques

Visualization

Sidharth Aggarwal[1] Amar Vashishth[1]

[1]Department of Computer Science
IIT Madras

Natural Language Processing(CS6370)
Prof. Sutanu Chakraborti

Outline

1. Language Modelling
   - N-gram Model
   - Need of Smoothing
2. Smoothing Techniques
   - Add-1 Smoothing
   - Laplace Smoothing
   - Good Turing Smoothing
   - Interpolation Smoothing
   - Other Smoothing
3. Visualization
4. Applications
5. Evaluation

Outline

Language Modelling

- The N-gram language model is given by formula as,

$$P(w_1, w_2, \ldots, w_n) = \Pi P(w_i | w_{i-k}, \ldots, w_{i-1})$$

- Further we can assume a Markov assumption that the current word doesn't depend on all the previous words, so then we can form unigrams, bigrams, etc.

$$P(w_1, w_2, \ldots, w_n) = \Pi P(w_i | w_{i-1})$$

Outline

Language Modelling

Need of Smoothing

- In language model we are calculating probabilities of the sentence.
- The probability includes the count of the n-gram.
- So, if the count of the n-gram turns out to be zero then whole probability of the sentence will be zero.
- So, we need to prevent this.
- And we will prevent the zero probabilities by smoothing them.

Outline

Add-1 Smoothing

Idea

In this we add 1 to the numerator and size of the vocabulary in the denominator.

Formula

$$P(w_i|w_{i-1}^{i-1-n}) = \frac{count(w_{i-1}^{i-1-n}, w_i) + 1}{count(w_{i-1}^{i-1-n}) + |V|}$$

Outline

Laplace Smoothing

Idea

In this we add k to the numerator and product of size of the vocabulary and k in the denominator. And k can be adjusted empirically.

Formula

$$P(w_i|w_{i-1}^{i-1-n}) = \frac{count(w_{i-1}^{i-1-n}, w_i) + k}{count(w_{i-1}^{i-1-n}) + k * |V|}$$

Outline

Good Turing Smoothing

Idea

In this technique it provides the probability to the unigram by seeing the frequency of the word.

Formula For c=0

$$P(entitywithZerofreq.) = \frac{N_1}{N}$$

Formula For c>0

$$c^* = \frac{(c+1)N_{c+1}}{N_c}$$

$$P(entitywithfreq - c) = \frac{c^*}{N}$$

Outline

Interpolation Smoothing

Idea

In this we make use the all the lesser grams also while calculating the probability of the n-gram.

Formula

Trigram

$$P(w_i|w_{i-1}, w_{i-2}) = \lambda_1 P(w_i|w_{i-1}, w_{i-2}) + \lambda_2 (P(w_i|w_{i-1}) + \lambda_3 P(w_i)$$

Outline

Other Smoothing

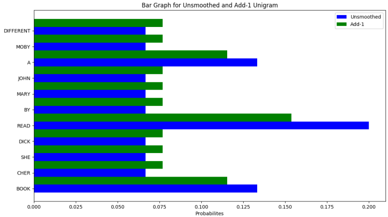
Katz Smoothing

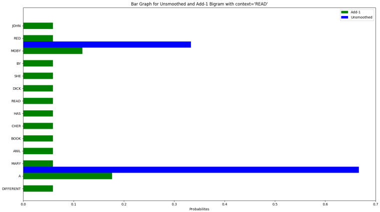
In this we give some discount to the original n-gram count and distribute this discount among the n-gram whose count is zero.

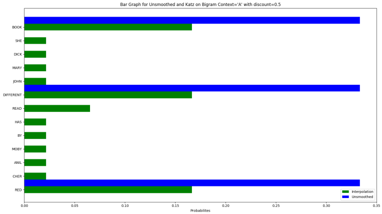
Absolute Discounting Smoothing

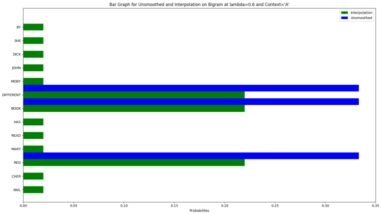
In this it makes use of two techniques which includes the discounting technique and the interpolation.

Visualization









Applications

Machine Translation

P(high winds tonite) $>$ P(large winds tonite)

Speech Recognition

P(I saw a van) $>$ P(eyes awe of an)

Spell Check

P(john read a book) $>$ P(john red a book)

Evaluation

Extrinsic

In this we put the model in a task and then see the results. It is very time consuming.

Intrinsic

In this we use the **Perplexity** which give us the amount of surprise model feel when it see the test instance.

$$PP(w) = \sqrt[N]{\frac{1}{P(w_1, w_2, \dots, w_N})}$$