

Assignment 5 + Smoothing 2

(SNLP Tutorial 6)

Vilém Zouhar, Awantee Deshpande, Julius Steuer

1st, 2nd June 2021

Assignment 5

- Exercise 1: OOV Words
- Exercise 2: Additive smoothing
- Exercise 3: Perplexity, infinite smoothing, interpolation
- Bonus: Other language models

Cross-validation

- K-fold cross-validation: Divide data into k subsets, train on $k-1$ subsets and test on the remaining 1.
- Leave One Out cross-validation: Train on all data points except one. Do this N times.

Questions

- Why is cross-validation beneficial?
- How does shuffling the dataset affect the LOOV score?
- When is k -fold cross-validation beneficial over standard cross-validation?

Smoothing Techniques

Remember the basics!

We perform smoothing to keep a language model from assigning 0 or ~ 0 probabilities to rare/unseen events.

Different ways to do this. . .

Floor Discounting

$$P(w|h) = \frac{N(w, h) + \epsilon}{N(h) + \epsilon \cdot V}$$

Variants: Laplace smoothing, Lidstone smoothing, add- α smoothing. . .

Good-Turing

Data: 

- $N_4 = \{\text{banana}\}$
- $N_3 = \{\text{apple}, \text{eggplant}\}$
- $N_2 = \{\text{cherry}\}$
- $N_1 = \{\text{grape}, \text{branch}\}$
- $N_0 = \{\text{fruit bowl}\}$

$$p_r = \frac{(r+1)N_{r+1}}{N_r} \cdot \frac{1}{N}$$

- Nominator: expected total number of occurrences of words that occur $r+1$ times
- Denominator-left: previous bucket size
- Fraction-left: expected number of occurrences of a single word from that bucket
- Denominator-right: divide by total occurrences

Good-Turing - Questions

- Let k be the maximum occurrence of a word. What's the issue?
- A similar issue related to the one above?
- Do the probabilities sum up to 1?
- How to make it work for anything above unigrams?

Linear Intepolation/Jelinek-Mercer Smoothing

B_1 : (FROZEN YOGHURT)

B_2 : (FROZEN RED)

What will floor discounting do here? Can we interpolate our bigram model with a unigram model?

$$P(w|h) = \lambda_1 P(w|h) + (1 - \lambda_1) P(w)$$

Can be generalised to higher order n-grams.

Questions

- What condition must be fulfilled for higher n-grams?
- How is λ_i determined?
- Can you smooth the above probabilities?

Backing-Off models

What other way can we use the lower-order n-gram distributions? Is a lot of context always a good thing?

Idea behind back-off models: Use information from a lower order n-gram distribution.

A “recursion” strategy...

$$P(w|h) = \begin{cases} \frac{N(w,h)-d}{N(h)} + \alpha(h)\beta(w|h) & \text{for } N(w,h) > 0 \\ \alpha(h)\beta(w|h) & \text{otherwise} \end{cases} \quad (1)$$

Absolute Discounting

Corpus

- Train set:



- Test set:



Distribution

- Vocabulary counts

6 5 3 2 0 0

- Decrease all non-zero counts by some parameter $d = 0.75$

$6-0.75$ $5-0.75$ $3-0.75$ $2-0.75$ 0 0

- Divide by $N = 16$

0.33 0.26 0.14 0.11 0 0

Sum = $0.33+0.26+0.14+0.11 = 0.84 \neq 1$.

Idea: Utilise this probability mass for zero counts.

Absolute Discounting

$$P(w|h) = \frac{c(w, h) - d}{c(h)}$$

Adjust the probability mass $1 - \sum_h \frac{c(w, h) - d}{c(h)}$

e.g. For bigrams,

$$P_{abs}(w_i|w_{i-1}) = \frac{\max\{N(w_{i-1}, w_i) - d, 0\}}{\sum_{w'} N(w_{i-1}, w')} + \lambda(w_{i-1})P_{abs}(w_i)$$

$$P_{abs}(w_i) = \frac{\max\{N(w_i) - d, 0\}}{\sum_{w'} N(w')} + \lambda(.)P_{unif}(w_i)$$

$$\text{where } \lambda(w_{i-1}) = \frac{d}{\sum_{w'} N(w_{i-1}, w')} \cdot N_{1+}(w_{i-1}, \bullet)$$

$$\lambda(.) = \frac{d}{\sum_{w'} N(w')} \cdot N_{1+}$$

Absolute Discounting - Questions

- How does the discounting parameter d affect perplexity?
- What values can d take? Why?
- What problems does Absolute Discounting have?

Kneser-Ney Smoothing

Idea: Can we use the lower order distributions in a better way?

I WENT TO THE GROCERY _____ .

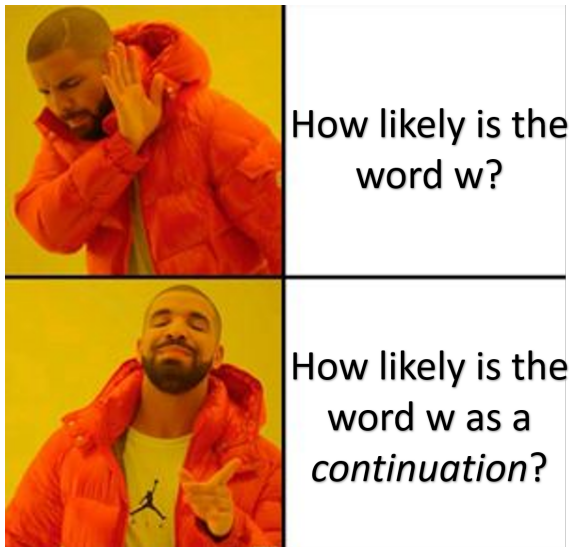
Options:

W_1 : STORE

W_2 : YORK

Use the fact that YORK generally appears as context or *continuation* of the word NEW.

Kneser-Ney Smoothing



Kneser-Ney Smoothing

$$P_{\text{continuation}}(w) \propto |\{w' : C(w', w) > 0\}|$$

****Don't forget to normalise!****

$$P_{KN}(w_i | w_{i-n+1:i-1}) = \frac{\max\{C_{KN}(w_{i-n+1:i-1}, w_i) - d, 0\}}{\sum_{w'} C_{KN}(w_{i-n+1:i-1} w')} + \lambda(w_{i-1}) P_{\text{continuation}}(w_i)$$

$$\text{where } C_{KN}(\bullet) = \begin{cases} \text{count}(\bullet) & \text{for highest order} \\ \text{continuationcount}(\bullet) & \text{for lower orders} \end{cases} \quad (2)$$

Will be covered in detail in the next tutorial...

Pruning

- Back-off models and interpolation save n-grams of all orders.

You are storing all $V^n + V^{n-1} + \dots + V + 1$ distributions!

- Idea: Store the counts which exceed a threshold $c(\bullet) > K$. Also called a “cut-off”.
- Another idea: Use some information-theory based approach to determine the nature of the probabilities, and then prune the lower orders. Known as *Stolcke Pruning*.

Questions

- Does pruning assign 0 probability to the pruned n-grams?
- Can you prune an entire branch/subtree? What does this mean?
- What is a good pruning strategy?

Assignment 6

- Exercise 1: MAP and MLE estimates
- Exercise 2: Good Turing Smoothing
- Exercise 3: Cross-Validation

Resources

- ① UdS SNLP Class: <https://teaching.lsv.uni-saarland.de/snlp/>
- ② n-gram models: <https://web.stanford.edu/~jurafsky/slp3/3.pdf>
- ③ Entropy pruning: <https://arxiv.org/pdf/cs/0006025.pdf>
- ④ Twitter emojis