Assignment 6,7 + Smoothing 3

(SNLP Tutorial 7)

Vilém Zouhar, Awantee Deshpande, Julius Steuer

8th, 10th June 2021

Slides repository

github.com/zouharvi/uds-snlp-tutorial

- Contributions welcome
- “Cheating” allowed

Assignment 6

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

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

Data: 🍌 🍏 🍆 🍌 🍒 🍇 🌿 🍒 🍇 🍌 🍒 🍇 🍏

$$N(\text{🍏}) = 2 \implies P(\text{🍏}) = 2/13$$

$$N(\text{🍇}) = 3 \implies P(\text{🍇}) = 3/13$$

But,

$$N(\bullet\,\text{🍏}) = 2\ (\text{🍌}\,\text{🍏}\ ,\ \text{🍇}\,\text{🍏})$$

$$N(\bullet\,\text{🍇}) = 1\ (\text{🍒}\,\text{🍇})$$

$$\therefore P(\text{🍏}) = 2/12$$

$$P(\text{🍇}) = 1/12$$

Kneser-Ney Smoothing

$$P_{CONTINUATION}(w) = \frac{|\{w' : C(w', w) > 0\}|}{|\{(w_i, w_j) : C(w_i, w_j) > 0\}|}$$

For bigrams,

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

$$\lambda(w_{i-1}) = \frac{d}{c(w_{i-1})} \cdot |\{w : C(w_{i-1}, w) > 0\}|$$

Kneser-Ney Smoothing

General Formula

$$P_{KN}(w_i|w_{i-n+1:i-1}) = \frac{\max\{C_{KN}(w_{i-n+1:i-1}, w_i) - d, 0\}}{\sum_{w'} C_{KN}(w_{i-n+1:i-1} w')} + \lambda(w_{i-n+1:i-1}) \cdot P_{KN}(w_i|w_{i-n+2:i-1})$$

$$\text{where} C_{KN}(\bullet) = \begin{cases} \text{count}(\bullet) & \text{for highest order} \\ \text{continuation\_count}(\bullet) & \text{for lower orders} \end{cases} \tag{1}$$

Kneser-Ney Smoothing Questions

- How are unseen words handled by KN Smoothing?
- For a KN-Smoothed language model,

$$\text{Information content("Vader")} = 15$$

$$\text{Information content("Star")} = 10$$

  Which unigram is assigned a higher probability by KN smoothing? Would the information content be the same for absolute discounting?
- Will the probabilities be affected for frequent higher order n-grams?
- Can Kneser-Ney smoothing be implemented for unigrams?

Pruning

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

We are storing all $V^n + V^{n-1} + ... + V + 1$ distributions!

- Idea: Store the counts which exceed a threshold $c(\bullet) \geq K$. Also called a “cut-off”.

Pruning: Comment on the implementation

```python
assert tree.get("5634") == 1
```

Pruning: How to build a count tree

```
tree.add("ABCE")
tree.add("ABCD")
tree.add("ABCD")
tree.add("QBCD")
tree.add("QQCD")
tree.add("BCDA")
tree.add("1234")
tree.add("1234")
tree.add("1234")
tree.add("1234")
tree.add("1234")
tree.add("5634")
```

Assignment 7

- Exercise 1: Count Trees and Pruning
- Exercise 2: Kneser-Ney Smoothing
- Bonus: Comparison of smoothing techniques

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. Kneser-Ney Smoothing: https://medium.com/@dennyc/a-simple-numerical-example-for-kneser-ney-smoothing-nlp-4600addf38b8
4. Comparison of Smoothing Techniques: https://people.eecs.berkeley.edu/~klein/cs294-5/chen_goodman.pdf