# COMP90042 Web Search & Text Analysis

## Workshop Week 5

Zenan Zhai

April 2, 2019

University of Melbourne

# Outline

- Text Classification
  - Definition
  - Tasks
  - Methods
- N-gram Language Model
  - Definition
  - Smoothing
    - Laplacian
    - Back-off & Interpolation
    - Kneser-Ney
  - Evaluation
- Excercise - Notebook

# Text Classification - Definition

## Supervised Learning Task

Input :

- Document: $d$
- Labels: $C = \{c_1, c_2, ...c_k\}$

Output :

- Prediction : $\hat{C}$

# Features

Discussion:

- How to convert document $d$ to a feature vector?

# Labels

- Topic classification
- Sentiment analysis
- Authorship attribution
- Native language identification
- Fact checking

Select features suitable for each task.

# Outline

- Text Classification
    - Definition
    - Tasks
    - Methods
- N-gram Language Model
    - Definition
    - Smoothing
        - Laplacian
        - Back-off & Interpolation
        - Kneser-Ney
    - Evaluation
- Excercise - Notebook

# Definition

Modeling $P(w-1, w_2, ..., w_i)$ in a Language $L$

Intuitively:

- $P(W) = P(w_1)P(w_2|w_1)P(w_3|w_2, w_1)...P(w_n|w_{n-1}...w_1)$

Markov Assumption:

- $P(w_i)$ only depends on previous $n$ words
- $n = 2, P(S) = \prod_{w_i \in S} P(w_i|w_{i-1}, w_{i-2})$

# Training

Maximum likelihood estimation (MLE).

Coin toss of $n$ times:

$$P(face) = \frac{c(face)}{n}$$

2-gram LM:

$$P(w_i|w_{i-1}) = \frac{c(w_i, w_{i-1})}{c(w_{i-1})}$$

How to model sequence not exists in the training material?

# Smoothing - Laplacian

Simply add value $k$:

$$P_{addk}(w_i|w_{i-1}) = \frac{c(w_i, w_{i-1}) + k}{c(w_{i-1}) + k|V|}$$

Problem:

- $k$ needs to be tuned
- $|V|$ grows fast when $n$ increase.

# Smoothing - Back-off & interpolation

Both back-off and interpolation uses information from lower-order models.

Back-off:

- Use $n-1$-gram probability iff $n$-gram count is zero.

$$P_{backoff}(w_i|w_{i-1}) = \begin{cases} \frac{c(w_i,w_{i-1})}{c(w_{i-1})}, & \text{if } c(w_i,w_{i-1}) > 0 \\ \alpha(w_{i-1})c(w_i), & \text{otherwise} \end{cases}$$

Interpolation:

- Incorporate lower-order information by factor $\lambda$ s.
- $P_{interpolation}(w_i|w_{i-1}) = \lambda(w_i, w_{i-1})P(w_i|w_{i-1}) + (1-\lambda(w_{i-1}))P(w_i)$

Question: Disadvantage of this approach?

# Smoothing - Kneser-Ney

$$P_{KN}(w_i|w_{i-1}) = \frac{max(0, c(w_i, w_{i-1}) - d)}{c(w_{i-1})} + \lambda(w_{i-1})P_{continuation}(w_i)$$

- Why do we discount $d$ ?
- What does continuation means ?

# Absolute discounting

$$P_{absDiscount}(w_i|w_{i-1}) = \frac{c(w_i, w_{i-1}) - d}{c(w_{i-1})} + \lambda(w_{i-1})P(w_i)$$

$$\lambda(w_{i-1}) = \frac{d}{c(w_{i-1})}$$

Intuition:

- Interpolation by using $\frac{d}{c(w_{i-1})}$ from lower-order model.
- Lower impact on *n*-gram with higher count.

| Bigram count in training set | Bigram count in heldout set |
|---|---|
| 0 | 0.0000270 |
| 1 | 0.448 |
| 2 | 1.25 |
| 3 | 2.24 |
| 4 | 3.23 |
| 5 | 4.21 |
| 6 | 5.23 |
| 7 | 6.21 |
| 8 | 7.21 |
| 9 | 8.26 |

*(Church and Gale, 1991)*

- Record all bi-gram with count in $[0, 9]$ in training set.
- Calculate Avg. count of these bi-grams in the held-out set.
- Difference for bi-grams with count in $[2, 9]$ are roughly the same.

# Continuation counts

$$P_{continuation} = \frac{|\{w_{i-1} : c(w_i, w_{i-1})\}|}{\sum_{w_i'} |\{w_{i-1} : c(w_i', w_{i-1})|}$$

Intuition:

- Interpolation incorporates prob. from lower-order models.
- Lower-order probs. without context can be unreliable.

Examples: If 1-gram has high count, but only appears as bi-grams.

- San Francisco
- New Zealand

Solution:

- Use count of bi-gram where words appear in the context instead.
- Normalise by counts of all possible contexts.

# Continuation counts

Example I:

- $w = food$, valid context: Asian food, Indian food, Mexican food.
- Calculate continuation counts for *food* in *Asian food*

$$P_{continuation}(food) = \frac{c(Asian, food)}{c(Asian, food) + c(Indian, food) + c(Mexican, food)}$$

Example II:

- Now consider continuation counts for *Zealand* in *New Zealand*.
- Why $P_{continuation}(Zealand) = 1$ ?

# Smoothing - Kneser-Ney

$$P_{KN}(w_i|w_{i-1}) = \frac{max(0, c(w_i, w_{i-1}) - d)}{c(w_{i-1})} + \lambda(w_{i-1})P_{continuation}(w_i)$$

- Use absolute discounting as interpolation.
- Use continuation counting for lower-order probs.

Question:

- Why not use continuation count for the highest order prob. ?

# Evaluation

Recall the objective of language model:

- Modeling probability for an arbitrary sequence of $m$ words.

Evaluate based on probability of all sequences in test set

$$PP(w_1, w_2, w_3, ..., w_m) = \sqrt[m]{\frac{1}{P(w_1, w_2, w_3, ..., w_m)}}$$

- Inverted prob. : lower perplexity $\rightarrow$ better model
- Normalization : take $m^{th}$ root of sequence prob. , $m = length(S)$

# Outline

- Text Classification
    - Definition
    - Tasks
    - Methods
- N-gram Language Model
    - Definition
    - Smoothing
        - Laplacian
        - Back-off & Interpolation
        - Kneser-Ney
    - Evaluation
- Excercise - Notebook