Outline

- Language modeling
- Language models for IR
- **Smoothing**
- Alternative models
- Comparison with traditional models

Laplace Smoothing

- Count events in observed data
- Add 1 to every count
- Renormalize to obtain probabilities

- If event counts are $(m_1, m_2, \ldots, m_k)$ with $\sum_i^k m_i = N$ then
  - Max likelihood estimates are $(\frac{m_1}{N}, \ldots, \frac{m_K}{N})$
  - Laplace estimates are $(\frac{m_1+1}{N+k}, \ldots, \frac{m_K+1}{N+k})$

Discounting Methods

- Laplace smoothing

- Lindstone correction
  - Add ε to all counts
  - Re-normalize
  - => $\dfrac{m_i + \varepsilon}{N + k\varepsilon}$

- Absolute discounting
  - Subtract ε
  - Re-distribute probability mass

Background Probability

- Key intuition: A nonoccurring term is possible (even though it didn’t occur), . . .
- . . . but no more likely than would be expected by chance in the collection

- Problem with all discounting methods:
  - Discounting treats unseen words equally (add or subtract ε)
  - Some words are more frequent than others

Background Probability

- Idea: use background probabilities
  - Smooth ML estimates with general English expectations

(computed as relative frequency of a word in a large collection)

  - Reflects expected frequency of events by background probability $P(w|M_c)$

$$P(w|M_c) = \frac{CF_w}{|c|}$$

    - M_c : the collection model
    - CF_w: the number of occurrences of w in the collection
    - $|c| = \sum_w CF_w$ the total number of tokens in the collection

Interpolation vs. Back off Smoothing

- Two possible approaches to smoothing

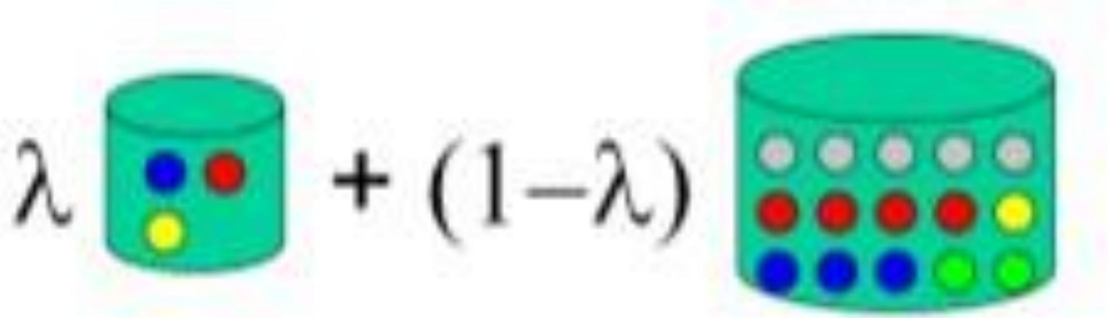
- Interpolation:
  - Adjust probabilities for all events, both seen and unseen
- Back-off:
  - Adjust probabilities only for unseen events
  - Leave non-zero probabilities as they are
  - Rescale everything to sum to one: rescales "seen" probabilities by a constant

- Interpolation tends to work better
  - And has a cleaner probabilistic interpretation

Jelinek-Mercer Smoothing

- Basic interpolation method
- Mixes the probability from the document with the general collection frequency of the word
- Correctly setting λ is very important for good performance
  - High value of λ: "conjunctive-like" search – tends to retrieve documents containing all query words
  - Low value of λ: more disjunctive, suitable for long queries

$$P(w|D) = \lambda \frac{TF_{w,D}}{|D|} + (1-\lambda)\frac{CF_w}{|c|}$$



Smoothing for N-gram Model (Jelinek-Mercer)

- Mixes different n-gram probabilities from the document with the general collection frequency of the word

- Unigram: $$P(w|D) = \lambda \frac{TF_{w,D}}{|D|} + (1-\lambda)\frac{CF_w}{|c|}$$

- Bigram: $$P(w_i|w_{i-1}, D) = \lambda_1 \left[\lambda_2 \frac{TF_{w_{i-1}w_i,D}}{TF_{w_{i-1},D}} + (1-\lambda_2)\frac{TF_{w_i,D}}{|D|}\right] + (1-\lambda_1)\frac{CF_w}{|c|}$$

or $$P(w_i|w_{i-1}, D) = \lambda_1 \frac{TF_{w_{i-1}w_i,D}}{TF_{w_{i-1},D}} + \lambda_2 \frac{TF_{w_i,D}}{|D|} + (1-\lambda_1-\lambda_2)\frac{CF_w}{|c|}$$

Outline

- Language modeling
- Language models for IR
- Smoothing
- Alternative models
- **Comparison with traditional models**

Vector Space vs. BM25 vs. LM

- BM25/LM: based on probability theory
- Vector space: based on similarity
  - A geometric/linear algebra notion

- All models consider term, document, and collection frequency as well as document length but in different ways

Vector Space vs. BM25 vs. LM

- Term frequency
  - It is directly used in all three models
  - LMs: raw term frequency
  - BM25/Vector space: more complex

Vector Space vs. BM25 vs. LM

- Length normalization
  - Vector space: cosine or pivot normalization
  - LMs: probabilities are inherently length normalized
  - BM25: tuning parameters for optimizing length normalization

Vector Space vs. BM25 vs. LM

- Inverse document frequency
  - BM25/Vector space use it directly
  - LMs: mixing term and collection frequencies has an effect similar to IDF
  - Collection frequency (LMs) vs. document frequency (BM25, vector space)

Assumptions in LM

- Simplifying assumption:
  - Terms are conditionally independent

=> Not true! But works in most cases.

- Vector space model make the same assumption
  - Cleaner statement of assumptions than vector space
  - Thus, better theoretical foundation than vector space
  - Moreover, LM has the flexibility of considering term dependency

Questions?