CS657A: Information Retrieval
Language Models

Arnab Bhattacharya
arnabb@cse.iitk.ac.in

Computer Science and Engineering,
Indian Institute of Technology, Kanpur
http://web.cse.iitk.ac.in/~cs657/

2nd semester, 2021-22

Tue 1030-1145, Thu 1200-1315

Language Models

- Probabilistic models of language *generation*
- Inspired by speech processing
  - food born thing, good corn sing, mood morning, good morning
- Each document builds a language model
- A term is "generated" from a model with some probability
- Thus, probability distribution over all possible term sequences
- A "query" can also be generated from the language model
- Documents are ranked by probability of generating the query using the corresponding language model

$$P(q|LM(d_j))$$

Types of Language Models

- Unigram model
- Terms are sampled independently
- Joint probability of terms is separable to product of individual probabilities
- "tiger eats deer" is same as "deer eats tiger"
  - Mostly all right in Indian languages, though
- Higher-order models
- Bi-gram or n-gram models capture phrases better
- Preceding context model
- Parse tree grammar model
- Higher-order models require more expensive parameter estimation
- In IR applications, not much better
  - Can show better results for NLP applications, though

Probability Distribution

- Multinomial model
- Natural model
- Each term has a probability of being generated from the model

$$P(t_1, t_2, \ldots, t_N | LM(d_j)) = \prod_{\forall t_i} P(t_i | LM(d_j))$$

- Multiple Bernoulli model
- In each position, term t_i occurs and other terms do *not* occur

$$P(t_1, t_2, \ldots, t_N | LM(d_j)) = \prod_{\forall t_i} \left[P(t_i | LM(d_j)). \prod_{\forall t_j \neq t_i} (1 - P(t_i | LM(d_j))) \right]$$

Maximum Likelihood Estimator

- Model *LM* is not known
- Its parameters are estimated using maximum likelihood estimator (MLE)
- Probability of a term begin generated from a model is its *relative frequency*

$$P(t_i|LM(d_j)) = \frac{tf_{i,j}}{dl_j}$$

- Suffers from zero frequency problem
  - If some term is missing, probability falls to 0
- Laplace/Lindstone correction: Add pseudo-counts of ϵ, and re-normalize
  - ϵ is typically 0.1 or 0.5
- Pseudo-counts act like priors and model resembles maximum a posteriori (MAP)
- All terms get *same* prior

Smoothing

- An absent term is possible, but its probability should not exceed the *background* probability
- *Prior* of terms should depend on their frequency in the *corpus*
- If frequency of term t_i in the entire corpus is $cf_i = \sum_{\forall d_j} tf_{i,j}$ and the total size of the corpus is $cl = \sum_{\forall d_j} dl_j$ terms

$$P(t_i|C) = \frac{cf_i}{cl}$$

- Jelinek-Mercer smoothing uses a weighted combination

$$P_\lambda(t_i|LM(d_j)) = \lambda.P(t_i|LM(d_j)) + (1-\lambda).P(t_i|C)$$

  - λ is typically 0.9
- Dirichlet smoothing uses Dirichlet priors on the multinomial model

$$P_\mu(t_i|LM(d_j)) = \frac{tf_{i,j} + \mu.P(t_i|C)}{dl_j + \mu}$$

Tf-idf Resemblance

- Jelinek-Mercer smoothing resembles tf-idf

$$
\begin{aligned}
P(q|d_j) &= \prod_{\forall t_i \in q} P(t_i|d_j) \\
&= \prod_{\forall q_i \in d_j} [\lambda.P(q_i|LM(d_j)) + (1-\lambda).P(q_i|C)]. \prod_{\forall q_i \notin d_j} (1-\lambda).P(q_i|C) \\
&= \prod_{\forall q_i \in d_j} [\lambda.P(q_i|LM(d_j)) + (1-\lambda).P(q_i|C)] \\
&\quad . \prod_{\forall q_i} (1-\lambda).P(q_i|C) \Big/ \prod_{\forall q_i \in d_j} (1-\lambda).P(q_i|C) \\
&\sim \prod_{\forall q_i \in d_j} [\lambda.P(q_i|LM(d_j)) + (1-\lambda).P(q_i|C)] \Big/ [(1-\lambda).P(q_i|C)] \\
&= \prod_{\forall q_i \in d_j} \left[1 + \frac{\lambda}{1-\lambda} . \frac{tf_{i,j}}{dl_j} . \frac{cl}{cf_i} \right]
\end{aligned}
$$

Ranking Functions

- Instead of ranking by $P(q|d_j)$, other functions can be used
- Can be ranked by $P(d_j|q)$
- Given the query, how likely is the document

$$P(d_j|q) = P(q|d_j).P(d_j)/P(q)$$

- If document prior is ignored, ranking remains the same

Comparison of Language Models

- Language models are essentially probability distributions
- Language model can be learned from q as well
  - Treats q as another document
- How dissimilar two language models are

$$D_{KL}(LM(q), LM(d_j)) = \sum_{\forall t_i} P(t_i|LM(q)) \log \frac{P(t_i|LM(q))}{P(t_i|LM(d_j))}$$

- Kullback-Leibler divergence measure or relative entropy measures how bad $LM(d_j)$ is in modeling $LM(q)$
  - $LM(q)$ is the "true" distribution while $LM(d_j)$ is "approximation"
  - Asymmetric

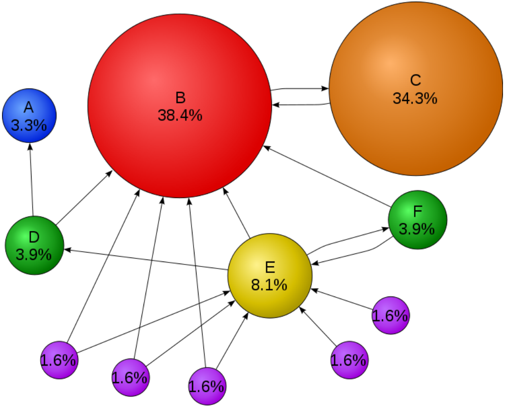
Document Prior

- Document priors can be sometimes very useful
- For example, in web page ranking
- Google's PageRank
- Random surfer assumption to model a person randomly clicking
- Suppose a person continues randomly clicking with probability λ
- With $1-\lambda$ probability, the person decides to stop following a link, and jumps to a random new page
- Assuming a total of N pages, the page rank of a page p is

$$PR(p) = (1-\lambda).\frac{1}{N} + \lambda.\sum_{q \in B_p} \frac{PR(q)}{|F_q|}$$

  where B_p and F_q are back links and forward links respectively
  - λ is assumed to be 0.15
- Can be solved iteratively or analytically

Example



- Pages with no link (e.g., A) are assumed to have links to all pages
- C has a much bigger page rank than E as a highly important page (i.e., B) points to it
- D and F have same page rank as they are pointed to by E only

HITS

- Hyperlink Induced Topic Search (HITS) ranks webpages according to authority (*not* relevance)
- An authority is a page that contains the "best" information
- A hub is a page that contains links to many authorities
- Mutually recursive: an authority is a page that contains backlinks from many hubs
- For a particular query, first the set of relevant documents is retrieved
- For each page in this induced subgraph, two scores, *hub* and *authority*, are computed
- Each page's authority score is updated as (normalized) *sum of hub scores* of its backlinks
- Each page's hub score is updated as (normalized) *sum of authority scores* of its forward links
- All hub and authority scores are normalized
- Iteratively continue till convergence
- Run at query time, but only for relevant pages