CIS 833 – Information Retrieval and Text Mining

Lecture 14

Language Models

October 20, 2015

Credits for slides: Hofmann, Mihalcea, Mobasher, Mooney, Schutze.

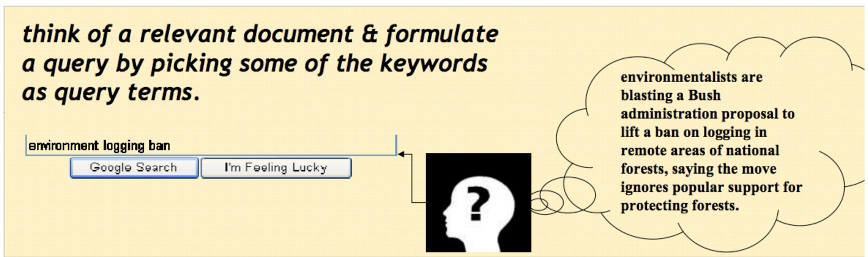
Required Reading

Information Retrieval textbook

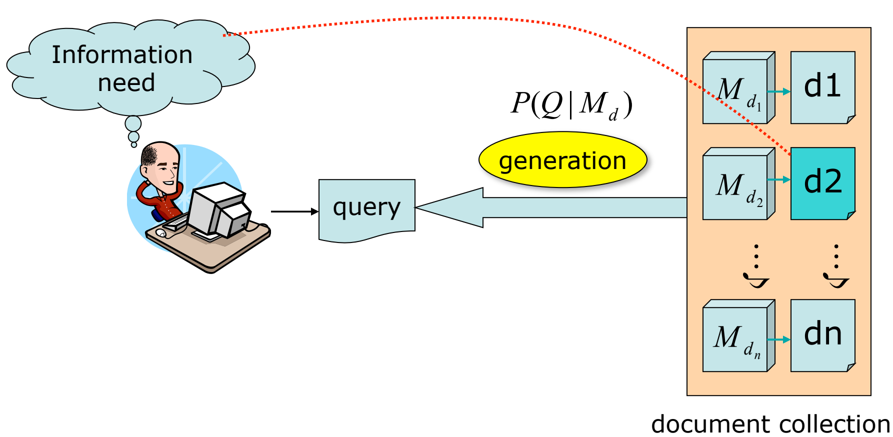
- Chapter 12: Language models for IR

Query Generation Model

- How might a query look like that would ask for a specific document?
  - A common search heuristic is to use words that you expect to find in matching document(s) as your query.
  - The LM approach directly exploits that idea
    - A document is a good match to a query if the document model is likely to generate the query, which will in turn happen if the document contains the query words often.

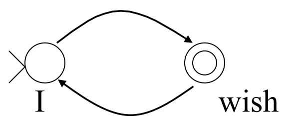


IR based on Language Model (LM)



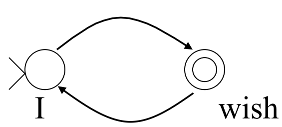
Formal Language (Model)

- Traditional generative model: generates strings
  - Finite state machines or regular grammars, etc.
- Example:



Formal Language (Model)

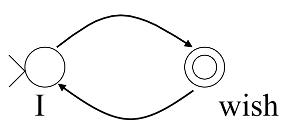
- Traditional generative model: generates strings
  - Finite state machines or regular grammars, etc.
- Example:



I wish
I wish I wish
I wish I wish I wish
I wish I wish I wish I wish
…

Formal Language (Model)

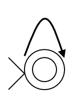
- Traditional generative model: generates strings
  - Finite state machines or regular grammars, etc.
- Example:



I wish
I wish I wish
I wish I wish I wish
I wish I wish I wish I wish
…

wish I wish ???

Stochastic Language Model



Models *probability* of generating strings in the language (commonly all strings over alphabet ∑)

Model M

0.2	the
0.1	a
0.01	man
0.01	woman
0.03	said
0.02	likes
…	

the	man	likes	the	woman
0.2	0.01	0.02	0.2	0.01

multiply

P(s | M) = 0.00000008

Stochastic Language Models

Model M1

0.2	the
0.01	class
0.0001	sayst
0.0001	pleaseth
0.0001	yon
0.0005	maiden
0.01	woman

Model M2

0.2	the
0.0001	class
0.03	sayst
0.02	pleaseth
0.1	yon
0.01	maiden
0.0001	woman

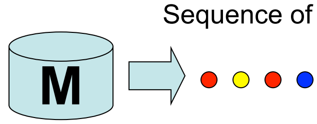
the	class	pleaseth	yon	maiden
0.2	0.01	0.0001	0.0001	0.0005
0.2	0.0001	0.02	0.1	0.01

$$P(s|M2) > P(s|M1)$$

Stochastic Language Models

- A statistical model for generating text
  - Probability distribution over strings (words) in a given language
  - How do we calculate probabilities over sequences of strings?

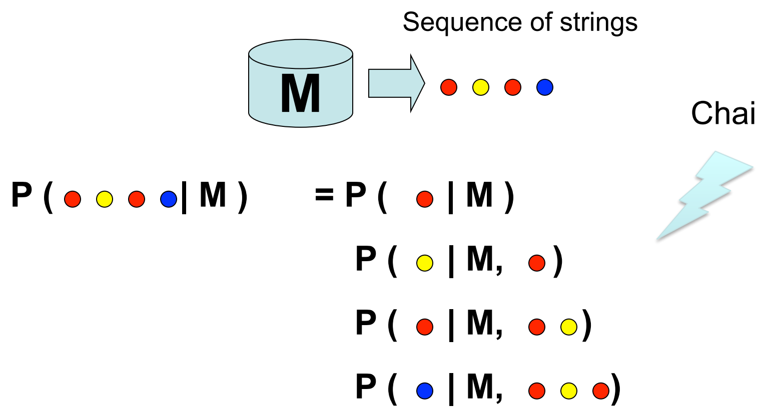
Sequence of strings



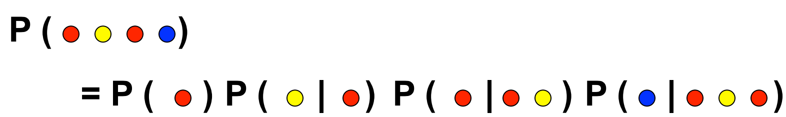
P (● ● ● ● | M) = ??? Chain rule?

Stochastic Language Models

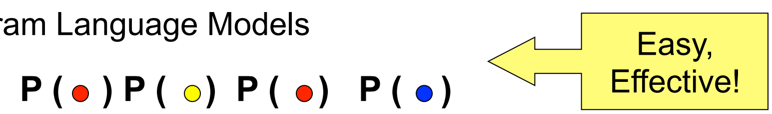
- A statistical model for generating text
  - Probability distribution over strings (words) in a given language



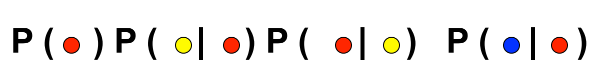
Unigram and higher-order models



- Unigram Language Models



- Bigram (generally, *n*-gram) Language Models



- Other Language Models
  - Grammar-based models (probabilistic context-free grammars or PCFGs), etc.
  - Used for speech recognition, spelling correction, machine translation, etc.

Language Models for IR

- Treat the generation of queries as a random process.
  - Users have a reasonable idea of terms that are likely to occur in documents of interest.
  - They will choose query terms that distinguish these documents from others in the collection.

Query Likelihood Model

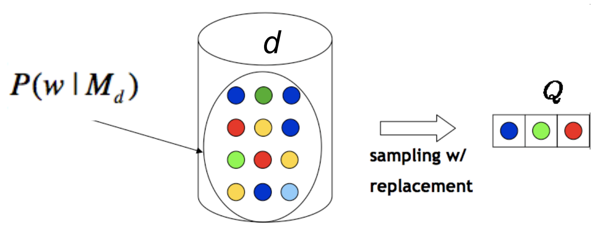
- Treat each document d as the basis for a model M_d (e.g., unigram language model)
  - Infer a language model for each document.
- Estimate the probability of generating the query according to each of these models

  $P(M_d \mid Q) = P(Q \mid M_d) \ \ x \ \ P(M_d) \ / \ P(Q)$
  - $P(Q)$ is the same for all documents, so ignore
  - $P(M_d)$ [the prior] is often treated as the same for all documents
    - But we could use criteria like authority, length, genre
  - $P(Q \mid M_d)$ is the probability of Q given d's model M_d
- Rank documents d based on $P(M_d \mid Q)$

  (or equivalently, $P(Q \mid M_d)$)
- Very general formal approach

Ranking

- Documents are ranked by the probability that a query would be observed as a random sample from the respective document model.



$$P(Q \mid M_d) = \prod_{w \in Q} P(w \mid M_d) = \prod_{\substack{w \text{ distinct} \\ \text{words in } Q}} P(w \mid M_d)^{q_w}$$

Query generation probability

- The probability of producing the query given the language model of document *d* (using maximum likelihood estimate) is:

$$P(Q \mid M_d) = \prod_{w \in Q} P(w \mid M_d)$$

$$= \prod_{w \in Q} \frac{tf_{(w,d)}}{dl_d}$$

Unigram assumption:
Given a particular language model, the query terms occur independently

M_d : language model of document d
$tf_{(w,d)}$: raw tf of word w in document d
dl_d : total number of tokens in document d

Insufficient data

- Zero probability $P(w \mid M_d) = 0$
  - May not wish to assign a probability of zero to a document that is missing one or more of the query terms [gives strict conjunctive semantics]
  - Need to smooth probabilities
  - There's a wide space of approaches to smoothing probability distributions to deal with this problem, such as adding 1, ½ or ε to counts, Dirichlet priors, discounting, and interpolation
- General approach here
  - A non-occurring term is possible, but no more likely than would be expected by chance in the collection.
  - If $tf_{(w,d)} = 0$ then $P(w \mid M_d) = \frac{cf_w}{cs}$

    cs : raw count of word w in the collection

    cf_t : raw collection size (total number of tokens in the collection)

Mixture model

- A simple idea that works well in practice is to use a mixture between the document multinomial and the collection multinomial distribution

  $P(w|d) = \lambda P(w|M_d) + (1 - \lambda) P(w|M_c)$

  - linear interpolation language model
- Mixes the probability from the document with the general collection frequency of the word
- Correctly setting λ is very important
- A high value of lambda makes the search "conjunctive-like" – suitable for short queries
- A low value is more suitable for long queries
- Can tune λ to optimize performance
  - Perhaps make it dependent on document size (cf. Dirichlet prior or Witten-Bell smoothing)

Basic mixture model summary

- General formulation of the LM for IR

$$p(Q \mid d) = \prod_{t \in Q} ((1-\lambda) p(t \mid M_c) + \lambda p(t \mid M_d))$$

general language model

individual-document model

  - The user has a document in mind, and generates the query from this document.
  - The equation represents the probability that the document that the user had in mind was in fact this one.

Collection Statistics in IR

- Collection statistics …
  - Are integral parts of the language model.
  - Are not used heuristically as in many other approaches.
    - At least in theory…

Example

- Document collection (2 documents)
  - d_1: Xerox reports a profit but revenue is down
  - d_2: Lucent narrows quarter loss but revenue decreases further
- Model: MLE unigram from documents; $\lambda = ½$
- Query: *revenue down*
  - $P(Q|d_1)$ = ?
  - $P(Q|d_2)$ = ?
- Ranking: ?

$$p(Q \mid d) = \prod_{t \in Q} ((1-\lambda) p(t \mid M_c) + \lambda p(t \mid M_d))$$

Language models: pros & cons

- Novel way of looking at the problem of text retrieval based on probabilistic language modeling
  - Conceptually simple and explanatory
  - Formal mathematical model
  - Natural use of collection statistics, not heuristics (almost…)
- LMs provide effective retrieval and can be improved to the extent that the following conditions can be met
  - Our language models are accurate representations of the data.
  - Users have some sense of term distribution.

LM vs. Prob. Model for IR

- The main difference is whether “Relevance” figures explicitly in the model or not
  - LM approach attempts to do away with modeling relevance
- LM approach asssumes that documents and expressions of information problems are of the same type
- Computationally tractable, intuitively appealing

LM vs. Prob. Model for IR

- Problems of the basic LM approach
  - Assumption of equivalence between document and information problem representation is unrealistic
  - Very simple models of language
  - Relevance feedback is difficult to integrate, as are user preferences, and other general issues of relevance
  - Can’t easily accommodate phrases, passages, Boolean operators
- Current extensions focus on putting relevance back into the model, etc.

Comparison With Vector Space

- There's some relation to traditional TF-IDF models:
  - terms often used as if they were independent
  - (unscaled) term frequency is directly in model
  - the probabilities do length normalization of term frequencies
  - the effect of doing a mixture with overall collection frequencies is a little like idf: terms rare in the general collection but common in some documents will have a greater influence on the ranking

Comparison With Vector Space

- Differences
  - Based on probability rather than similarity
    - Intuitions are probabilistic rather than geometric
  - Details of use of document length and term, document, and collection frequency differ
- Recent work has shown that LM approach performs better than TF-IDF and other retrieval models.