Acknowledgments: Some slides in this lecture were adapted from Chris Manning (Stanford) and Jin Kim (UMass’12)
Summary - BIM

• Boils down to

\[ RSV_{BIM}^{BIM} = c_i^{BIM} \quad \text{where} \quad x_i = q_i = 1 \]

\[ c_i^{BIM} = \log \frac{p_i(1 - r_i)}{(1 - p_i)r_i} \]

Log odds ratio

where

<table>
<thead>
<tr>
<th></th>
<th>document</th>
<th>relevant (R=1)</th>
<th>not relevant (R=0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>term present</td>
<td>x_i = 1</td>
<td>p_i</td>
<td>r_i</td>
</tr>
<tr>
<td>term absent</td>
<td>x_i = 0</td>
<td>(1 – p_i)</td>
<td>(1 – r_i)</td>
</tr>
</tbody>
</table>

• Simplifies to (with constant \( p_i = 0.5 \))

\[ RSV = \log \frac{N}{n_i} \quad \text{with} \quad x_i = q_i = 1 \]
Graphical model for BIM

Binary variables

\[ x_i = (tf_i, 0) \]
Okapi BM25

\[ RSV^{BM25} = \log \frac{N}{df_i} \times \frac{(k_1 + 1)tf_i}{k_1((1-b) + b \frac{dl}{avdl}) + tf_i} \]

- \( k_1 \) controls term frequency scaling
  - \( k_1 = 0 \) is binary model; \( k_1 \) = large is raw term frequency
- \( b \) controls document length normalization
  - \( b = 0 \) is no length normalization; \( b = 1 \) is relative frequency (fully scale by document length)
- Typically, \( k_1 \) is set around 1.2–2 and \( b \) around 0.75
- \( MRS \) ch. 11.4.3 discusses incorporating query term weighting and (pseudo) relevance feedback
What about document structure?

- Documents have multiple fields
  - Emails, products (entities), and so on.

- Retrieval models exploit the structure
  - Field weighting is common

\[ \sum_{i=1}^{n} w_i f_i \left( q_1 q_2 \ldots q_m \right) \]
Relevance for Structured Document Retrieval

• Term-level Relevance
  – Which term is important for user’s information need?

• Field-level Relevance
  – Which field is important for user’s information need?

Term-level relevance

\[ P(w|R) \]

\[ V = (w_1, w_2, \ldots, w_m) \]

Field-level relevance

\[ P(F|R) \]

\[ F = (F_1, F_2, \ldots, F_n) \]
Field Relevance: $P(F|w,R)$

$P(F|w,R)$: The \textit{distribution} of \textit{per-term} relevance over document fields

Query:
- $m$ words
- $Q = (q_1, q_2, \ldots, q_m)$

Collection:
- $n$ fields for each document
- $F = (F_1, F_2, \ldots, F_n)$

$p(F|q_1,R)$ $p(F|q_i,R)$ $p(F|q_m,R)$
Why $P(F|w,R)$ instead of $P(F|R)$?

- Different **fields** relevant for different query-terms:

  Query: ‘james registration’

  - ‘james’ is relevant when it occurs in <to>
  - ‘registration’ is relevant when it occurs in <subject>

Example email:

```
From: Young Ah Do <youngah@mit.edu>
Subject: Re: Two Dollar Dinner Registration Confirmation
Date: March 17, 2010 8:02:56 PM EDT
To: James Hong <ttoppo84@mit.edu>

Yes, I did. I got only one email.

Thanks.

Best,
Youngah

On Wed, Mar 17, 2010 at 7:58 PM, James Hong <ttoppo84@mit.edu> wrote:
Hi Youngah,

Let me check it first and get back to you ASAP. Did you receive the confirmation email only once?

James
```
Ranking with zones: Operationalizing

• How to make this practical?
  – Assume term importance is shared across zones
  – ... but the relationship between importance and term frequencies are zone-dependent
    • e.g., denser use of important topic words in title

• Method:
  1. combine evidence across zones for each term (tf)
  2. combine evidence across terms (RSV)
BM25F with zones (1)

- Calculate a weighted variant of total term frequency
- ... and a weighted variant of document length

\[
\tilde{t}_{fi} = \frac{1}{Z} \sum_{z=1}^{Z} v_z t_{zi} \\
\tilde{d}l = \frac{1}{Z} \sum_{z=1}^{Z} v_z len_z \\
\text{avd}l = \text{Average } d\tilde{l} \text{ across all documents}
\]

where

- \( v_z \) is zone weight
- \( t_{zi} \) is term frequency in zone \( z \)
- \( \text{len}_z \) is length of zone \( z \)
- \( Z \) is the number of zones
BM25F (2)

\[ RSV_{SimpleBM25F} = \log \frac{N}{df_i} \times \frac{(k_1 + 1)\tilde{t}_f_i}{k_1((1 - b) + b \frac{d\tilde{l}}{avd\tilde{l}}) + \tilde{t}_f_i} \]

\[ \tilde{t}_f_i = \sum_{z=1}^{Z} v_z t_{f_{zi}} \quad d\tilde{l} = \sum_{z=1}^{Z} v_z len_z \quad avd\tilde{l} = \text{Average across all documents} \]

- **interpretation**: zone \( z \) is “replicated” \( v_z \) times
- But: want zone-specific parameters \( (k_1, b, \text{IDF}) \)
Empirically, zone-specific length normalization (i.e., zone-specific $b$) has been found to be useful

$$\tilde{tf_i} = \sum_{z=1}^{z} v_z \frac{tf_{zi}}{B_z}$$

$$B_z = (1 - b_z) + b_z \frac{len_z}{avlen_z}$$

$$0 \leq b_z \leq 1$$

$$RSV_{BM25F} = \log \frac{N}{df_i} \times \frac{(k_1 + 1)\tilde{tf_i}}{k_1 + tf_i}$$

See Robertson and Zaragoza (2009: 364)
BM25F: Example Experiment

[Metric: Mean Reciprocal Rank]

<table>
<thead>
<tr>
<th>Test Collection</th>
<th>#Documents</th>
<th>#Queries</th>
<th>#RelDocs/Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>TREC (emails)</td>
<td>198,394</td>
<td>125</td>
<td>1</td>
</tr>
<tr>
<td>IMDB (movies)</td>
<td>437,281</td>
<td>50</td>
<td>2</td>
</tr>
<tr>
<td>Monster.com (resumes)</td>
<td>1,034,795</td>
<td>60</td>
<td>15</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Collection</th>
<th>BM25</th>
<th>BM25F</th>
</tr>
</thead>
<tbody>
<tr>
<td>TREC</td>
<td>54.2%</td>
<td>59.7%</td>
</tr>
<tr>
<td>IMDB</td>
<td>40.8%</td>
<td>52.4%</td>
</tr>
<tr>
<td>Monster.com</td>
<td>42.9%</td>
<td>27.9%</td>
</tr>
</tbody>
</table>

[Jin Kim et al., ECIR 2012]
Summary: Standard Probabilistic IR

Information need

query

\[ P(R|Q,d) \]

matching

document collection

\( d_1 \)

\( d_2 \)

\( \ldots \)

\( d_n \)
Words are drawn independently from the vocabulary using a multinomial distribution.

... the **draft** is that each **team** is given a position in the **draft** ...
Eliteness ("aboutness")

- Model term frequencies using *eliteness*
- What is eliteness?
  - Hidden variable for each document-term pair, denoted as $E_i$ for term $i$
  - Represents *aboutness*: a term is elite in a document if, in some sense, the document is about the concept denoted by the term
  - Eliteness is binary
  - Term occurrences depend only on eliteness...
  - ... but eliteness depends on relevance
The National Football League Draft is an annual event in which the National Football League (NFL) teams select eligible college football players. It serves as the league’s most common source of player recruitment. The basic design of the draft is that each team is given a position in the draft order in reverse order relative to its record …
Graphical model with eliteness

\[ R \]

\[ E_i \]

\[ t_f_i \]

\[ i \quad q \]

Binary variables

Frequencies (not binary)
A common search heuristic is to use words that you expect to find in matching documents as your query – why, I saw Sergey Brin advocating that strategy on late night TV one night in my hotel room, so it must be good!

The LM approach directly exploits that idea!
Goal of Language Modeling

• Want to build models which assign scores to sentences.
  – $P(\text{l saw a van}) \gg P(\text{eyes awe of an})$

• One option: empirical distribution over sentences?
  – Problem: doesn’t generalize (at all)

• Two ways of generalizing
  – **Decomposition**: sentences generated in small steps which can be recombined in other ways
  – **Smoothing**: allow for the possibility of unseen events
Language models for IR: Assumptions

- **Simplifying assumption:** *Queries and documents are objects of same type.* Not true!
  - There are other LMs for IR that do not make this assumption.
  - The vector space model makes the same assumption.

- **Simplifying assumption:** *Terms are conditionally independent.*
  - Again, vector space model (and Naive Bayes) makes the same assumption.

- **Cleaner statement of assumptions than vector space**

- **Thus, better theoretical foundation than vector space**
  - ... but “pure” LMs perform much worse than “tuned” LMs.
Formal Language (Model)

• Traditional generative model: generates strings
  – Finite state machines or regular grammars, etc.

• Example:
Stochastic Language Models

- Model *probability* of generating strings in the language (commonly all strings over alphabet $\Sigma$)

<table>
<thead>
<tr>
<th>Model M</th>
<th>the</th>
<th>a</th>
<th>man</th>
<th>likes</th>
<th>the</th>
<th>woman</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.03</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.02</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

... multiply

$$P(s \mid M) = 0.00000008$$
Stochastic Language Models

- Model *probability* of generating any string

<table>
<thead>
<tr>
<th>Model M1</th>
<th></th>
<th>Model M2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td><strong>the</strong></td>
<td>0.2</td>
</tr>
<tr>
<td>0.01</td>
<td><strong>class</strong></td>
<td>0.0001</td>
</tr>
<tr>
<td>0.0001</td>
<td><strong>sayst</strong></td>
<td>0.03</td>
</tr>
<tr>
<td>0.0001</td>
<td><strong>pleaseth</strong></td>
<td>0.02</td>
</tr>
<tr>
<td>0.0001</td>
<td><strong>yon</strong></td>
<td>0.1</td>
</tr>
<tr>
<td>0.0005</td>
<td><strong>maiden</strong></td>
<td>0.01</td>
</tr>
<tr>
<td>0.01</td>
<td><strong>woman</strong></td>
<td>0.0001</td>
</tr>
</tbody>
</table>

\[
P(s|M2) > P(s|M1)
\]
Stochastic Language Models

• A statistical model for generating text
  – Probability distribution over strings in a given language

\[
P(\cdot \cdot \cdot | M) = P(\cdot | M)
\]

\[
P(\cdot | M, \cdot)
P(\cdot | M, \cdot \cdot)
P(\cdot | M, \cdot \cdot \cdot)
P(\cdot | M, \cdot \cdot \cdot \cdot)
\]
Generalizability

• No loss of generality to break sentence probability down with the chain rule

\[ P(w_1w_2 \ldots w_n) = \prod_i P(w_i \mid w_1w_2 \ldots w_{i-1}) \]

• Too many histories!

• N-gram solution: assume each word depends only on a short linear history

\[ P(w_1w_2 \ldots w_n) = \prod_i P(w_i \mid w_{i-k} \ldots w_{i-1}) \]
N-gram model

- Each word is predicted according to a conditional distribution based on a limited prior context
- Conditional Probability Table (CPT): $P(X|both)$
  - $P(of|both) = 0.066$
  - $P(to|both) = 0.041$
  - $P(in|both) = 0.038$
- From 1940s onward (or even 1910s – Markov 1913)
- a.k.a. Markov (chain) models
Unigram Models

- Simplest case: unigrams

\[ P(w_1w_2 \ldots w_n) = \prod_{i} P(w_i) \]

- Generative process: pick a word, pick a word, ...
- As a graphical model:

  ![Graphical Model](image)

- To make this a proper distribution over sentences, we have to generate a special STOP symbol last. (Why?)

- Examples:
  - [fifth, an, of, futures, the, an, incorporated, a, a, the, inflation, most, dollars, quarter, in, is, mass.]
  - [thrift, did, eighty, said, hard, 'm, july, bullish]
  - [that, or, limited, the]
  - []
  - [after, any, on, consistently, hospital, lake, of, of, other, and, factors, raised, analyst, too, allowed, mexico, never, consider, fall, bungled, davison, that, obtain, price, lines, the, to, sass, the, the, further, board, a, details, machinists, the, companies, which, rivals, an, because, longer, oakes, percent, a, they, three, edward, it, currier, an, within, in, three, wrote, is, you, s., longer, institute, dentistry, pay, however, said, possible, to, rooms, hiding, eggs, approximate, financial, canada, the, so, workers, advancers, half, between, nasdaq]
The words that the speakers have used during the Democratic convention suggest how the party's themes have changed since the last presidential campaign.

Speakers have hammered home Barack Obama's "change" theme, using the word about eight times as often as they did in 2004.

Also, unlike 2004, when the Kerry campaign sought to avoid direct attacks on the president at the convention, the speakers have regularly been mentioning John McCain by name.

Speakers in 2004 practiced "the art of the implicit slam," a veteran Democratic speechwriter said then, indirectly aliasing Mr. Bush while barely using his name.

Also on the upswing: more mentions of the economy, energy, Iran and Iraq.

Words less frequently used: freedom, Sept. 11 and terrorism.
Bigram Models

- Big problem with unigrams: \( P(\text{the the the the}) \gg P(\text{I like ice cream}) \! \)
- Condition on last word:

\[
P(w_1w_2 \ldots w_n) = \prod_i P(w_i | w_{i-1})
\]

- Any better?

  - \([\text{texaco, rose, one, in, this, issue, is, pursuing, growth, in, a, boiler, house, said, mr., gurria, mexico, 's, motion, control, proposal, without, permission, from, five, hundred, fifty, five, yen}]\)
  - \([\text{outside, new, car, parking, lot, of, the, agreement, reached}]\)
  - \([\text{although, common, shares, rose, forty, six, point, four, hundred, dollars, from, thirty, seconds, at, the, greatest, play, disingenuous, to, be, reset, annually, the, buy, out, of, american, brands, vying, for, mr., womack, currently, sharedata, incorporated, believe, chemical, prices, undoubtedly, will, be, as, much, is, scheduled, to, conscientious, teaching}]\)
  - \([\text{this, would, be, a, record, november}]\)
• serve as the incoming 92
• serve as the incubator 99
• serve as the independent 794
• serve as the index 223
• serve as the indication 72
• serve as the indicator 120
• serve as the indicators 45
• serve as the indispensable 111
• serve as the indispensible 40
• serve as the individual 234
Sparsity

- Problems with n-gram models:
  - New words appear all the time:
    - Synaptitude
    - 132,701.03
    - fuzzificational
  - New bigrams: even more often
  - Trigrams or more – still worse!

- Zipf’s Law
  - Types (words) vs. tokens (word occurrences)
  - Broadly: most word types are rare
  - Specifically:
    - Rank word types by token frequency
    - Frequency inversely proportional to rank
  - Not special to language: randomly generated character strings have this property
Unigram and higher-order models

$P(\bullet \bullet \bullet \bullet)$

$= P(\bullet) P(\bullet | \bullet) P(\bullet | \bullet \bullet \bullet) P(\bullet | \bullet \bullet \bullet \bullet)$

• Unigram Language Models
  $P(\bullet) P(\bullet) P(\bullet) P(\bullet)$

• Bigram (generally, $n$-gram) Language Models
  $P(\bullet) P(\bullet | \bullet) P(\bullet | \bullet) P(\bullet | \bullet)$

• Other Language Models
  – Grammar-based models (PCFGs), etc.

  • Probably not the first thing to try in IR

Easy. Effective!
A common search heuristic is to use words that you expect to find in matching documents as your query – why, I saw Sergey Brin advocating that strategy on late night TV one night in my hotel room, so it must be good!

The LM approach directly exploits that idea!
Using Language Models in IR (2)

• Treat each document as the basis for a model (e.g., unigram sufficient statistics)
• Rank document \(d\) based on \(P(d \mid q)\)
• \(P(d \mid q) = P(q \mid d) \times P(d) / P(q)\)
  – \(P(q)\) is the same for all documents, so ignore
  – \(P(d)\) [the prior] is often treated as the same for all \(d\)
    • But we could use criteria like authority, length, genre
  – \(P(q \mid d)\) is the probability of \(q\) given \(d\)’s model
• Very general formal approach
Using Language Models in IR (1)

• 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.

• Collection statistics …
  – Are integral parts of the language model.
  – Are not used heuristically as in many other approaches.
    • In theory. In practice, there’s usually some wiggle room for empirically set parameters
The Basic LM Approach
[Ponte & Croft 98]

Document

Language Model

Text mining paper

... text ? mining ? association ? clustering ?...
... food ?...

Query = “data mining algorithms”

Food nutrition paper

... food ? nutrition ? healthy ? diet ?...

Which model would most likely have generated this query?
Ranking Docs by Query Likelihood

\[ \begin{align*}
\text{Doc LM} & \quad \theta_{d_1} \quad \text{Query likelihood} \\
\quad d_1 & \quad \rightarrow \quad \theta_{d_1} & \quad p(q \mid \theta_{d_1}) & \quad \rightarrow & \quad q \\
\quad d_2 & \quad \rightarrow \quad \theta_{d_2} & \quad p(q \mid \theta_{d_2}) \\
\quad d_N & \quad \rightarrow \quad \theta_{d_N} & \quad p(q \mid \theta_{d_N})
\end{align*} \]
The fundamental problem of LMs

• Usually we don’t know the model \( M \)
  – But have a sample of text representative of that model

\[
P( \cdot \cdot \cdot \cdot \cdot | M( \cdot \cdot \cdot \cdot \cdot \cdot \cdot \cdot \cdot \cdot ) )
\]

• Estimate a language model from a sample
• Then compute the observation probability
Query generation probability (first cut)

- Ranking formula
  \[ p(Q, d) = p(d) p(Q | d) \]
  \[ \approx p(d) p(Q | M_d) \]

- The probability of producing the query given the language model of document \(d\) using MLE is:
  \[ \hat{p}(Q | M_d) = \prod_{t \in Q} \hat{p}_{ml}(t | M_d) \]
  \[ = \prod_{t \in Q} \frac{tf_{(t,d)}}{dl_d} \]

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

- \(M_d\): language model of document \(d\)
- \(tf_{(t,d)}\): raw tf of term \(t\) in document \(d\)
- \(dl_d\): total number of tokens in document \(d\)
Insufficient data

- Zero probability $p(t \mid M_d) = 0$
  - May not wish to assign a probability of zero to a document that is missing one or more of the query terms [gives conjunction semantics]

- General approach
  - $tf_{(t,d)} = 0$ $\quad p(t \mid M_d) = \frac{cf_t}{cs}$
  - A non-occurring term is possible, but no more likely than would be expected by chance in the collection.
  - If
    - $cf_t$ : raw count of term t in the collection
    - $cs$ : raw collection size(total number of tokens in the collection)
Smoothing

- We often want to make estimates from sparse statistics:

  \[ P(w \mid \text{denied the}) \]
  3 allegations
  2 reports
  1 claims
  1 request
  7 total

- Smoothing flattens spiky distributions so they generalize better

  \[ P(w \mid \text{denied the}) \]
  2.5 allegations
  1.5 reports
  0.5 claims
  0.5 request
  2 other
  7 total

- Very important all over NLP, but easy to do badly!
- We’ll illustrate with bigrams today (h = previous word, could be anything).
Language Model Smoothing
(Illustration)

Max. Likelihood Estimate

\[ p_{ML}(w) = \frac{\text{count of } w}{\text{count of all words}} \]

Smoothed LM

\( P(w) \)

Word \( w \)
How to Smooth?

- All smoothing methods try to
  - discount the probability of words seen in a document
  - re-allocate the extra counts so that unseen words will have a non-zero count

- Method 1 Additive smoothing [Chen & Goodman 98]: Add a constant $\delta$ to the counts of each word, e.g., “add 1”

$$p(w \mid d) = \frac{c(w,d) + 1}{|d| + |V|}$$

Counts of $w$ in $d$  

“Add one”, Laplace

Length of $d$ (total counts)  

Vocabulary size
Smoothing

• Estimating multinomials
• We want to know what words follow some history $h$
  – There’s some true distribution $P(w \mid h)$
  – We saw some small sample of $N$ words from $P(w \mid h)$
  – We want to reconstruct a useful approximation of $P(w \mid h)$
  – Counts of events we didn’t see are always too low ($0 < N P(w \mid h)$)
  – Counts of events we did see are in aggregate too high

• Two issues:
  – Discounting: how to reserve mass what we haven’t seen
  – Interpolation: how to allocate that mass amongst unseen events
Laplace smoothing

• Idea: pretend we saw every word once more than we actually did [Laplace]
  – Corresponds to a uniform prior over vocabulary
  – Think of it as taking items with observed count $r > 1$ and treating them as having count $r^* < r$
  – Holds out $V/(N+V)$ for “fake” events
  – $N1+/N$ of which is distributed back to seen words
  – $N0/(N+V)$ actually passed on to unseen words (nearly all!)
  – Actually tells us not only how much to hold out, but where to put it

• Works astonishingly **poorly** in practice

• Quick fix: add some small $\delta$ instead of 1 [Lidstone, Jefferys]

• Slightly better, holds out less mass, still a bad idea
How Much to subtract?

• Remember the key discounting problem:
  – What count should $r^*$ should we use for an event that occurred $r$ times in $N$ samples?
  – $r$ is too big

• Idea: held-out data [Jelinek and Mercer]
  – Get another $N$ samples
  – See what the average count of items occurring $r$ times is (e.g. doubletons on average might occur 1.78 times)
  – Use those averages as $r^*$
  – Much better than add-one, etc.
Smoothing add-one summary

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c$</td>
<td>number of word tokens in training data</td>
</tr>
<tr>
<td>$c(w)$</td>
<td>count of word $w$ in training data</td>
</tr>
<tr>
<td>$c(w, w_{-1})$</td>
<td>count of word $w$ following word $w_{-1}$</td>
</tr>
<tr>
<td>$V$</td>
<td>total vocabulary size</td>
</tr>
<tr>
<td>$N_k$</td>
<td>number of word types with count $k$</td>
</tr>
</tbody>
</table>

- **One class of smoothing functions:**
  - Add-one / delta: assumes a uniform prior
    
    $$
P_{ADD-\delta}(w | w_{-1}) = \frac{c(w, w_{-1}) + \delta (1/V)}{c(w_{-1}) + \delta}
    $$
  - Better to assume a unigram prior
    
    $$
P_{UNI-PRIOR}(w | w_{-1}) = \frac{c(w, w_{-1}) + \delta \hat{P}(w)}{c(w_{-1}) + \delta}
    $$
Linear Interpolation

- One way to ease the sparsity problem for n-grams is to use less-sparse n-1-gram estimates.

- General linear interpolation:

  \[ P(w | w_{-1}) = [1 - \lambda(w, w_{-1})] \hat{P}(w | w_{-1}) + [\lambda(w, w_{-1})] P(w) \]

  - Having a single global mixing constant is generally not ideal:

    \[ P(w | w_{-1}) = [1 - \lambda] \hat{P}(w | w_{-1}) + [\lambda] P(w) \]

- Solution: have different constant buckets
  - Buckets by count
  - Buckets by average count (better)
Mixture model

- \( P(w \mid d) = \lambda P_{mle}(w \mid M_d) + (1 - \lambda) P_{mle}(w \mid M_c) \)
- Mixes the probability from the document with the general collection frequency of the word.
- Correctly setting \( \lambda \) 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 \( \lambda \) 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

\[ p(Q, d) = p(d) \prod_{t \in Q} \left((1 - \lambda) p(t) + \lambda p(t \mid M_d)\right) \]

- 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.
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; $\lambda = \frac{1}{2}$
- Query: revenue down
  - $P(Q|d_1) = [(1/8 + 2/16)/2] \times [(1/8 + 1/16)/2]$
    \[= \frac{1}{8} \times 3/32 = 3/256\]
  - $P(Q|d_2) = [(1/8 + 2/16)/2] \times [(0 + 1/16)/2]$
    \[= \frac{1}{8} \times 1/32 = 1/256\]
- A term is missing... can you find it?
- Ranking: $d_1 > d_2$
Ponte and Croft Experiments

• Data
  – TREC topics 202-250 on TREC disks 2 and 3
    • Natural language queries consisting of one sentence each
  – TREC topics 51-100 on TREC disk 3 using the concept fields
    ⇒ Lists of good terms

<num>Number: 054
<dom>Domain: International Economics
<title>Topic: Satellite Launch Contracts
<desc>Description:
... </desc>

<con>Concept(s):
1. Contract, agreement
2. Launch vehicle, rocket, payload, satellite
3. Launch services, ... </con>
Precision/recall results 202-250

<table>
<thead>
<tr>
<th>prec.</th>
<th>tf.idf</th>
<th>LM</th>
<th>%chg</th>
<th>I/D</th>
<th>Sign</th>
<th>Wilc</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.7439</td>
<td>0.7590</td>
<td>+2.0</td>
<td>10/22</td>
<td>0.7383</td>
<td>0.5709</td>
</tr>
<tr>
<td>0.10</td>
<td>0.4521</td>
<td>0.4910</td>
<td>+8.6</td>
<td>24/42</td>
<td>0.2204</td>
<td>0.0761</td>
</tr>
<tr>
<td>0.20</td>
<td>0.3514</td>
<td>0.4045</td>
<td>+15.1</td>
<td>27/44</td>
<td>0.0871</td>
<td>0.0081*</td>
</tr>
<tr>
<td>0.30</td>
<td>0.2761</td>
<td>0.3342</td>
<td>+21.0</td>
<td>28/43</td>
<td>0.0330*</td>
<td>0.0054*</td>
</tr>
<tr>
<td>0.40</td>
<td>0.2093</td>
<td>0.2572</td>
<td>+22.9</td>
<td>25/39</td>
<td>0.0541</td>
<td>0.0158*</td>
</tr>
<tr>
<td>0.50</td>
<td>0.1558</td>
<td>0.2061</td>
<td>+32.3</td>
<td>24/35</td>
<td>0.0205*</td>
<td>0.0018*</td>
</tr>
<tr>
<td>0.60</td>
<td>0.1024</td>
<td>0.1405</td>
<td>+37.1</td>
<td>22/27</td>
<td>0.0008*</td>
<td>0.0027*</td>
</tr>
<tr>
<td>0.70</td>
<td>0.0451</td>
<td>0.0760</td>
<td>+68.7</td>
<td>13/15</td>
<td>0.0037*</td>
<td>0.0062*</td>
</tr>
<tr>
<td>0.80</td>
<td>0.0160</td>
<td>0.0432</td>
<td>+169.6</td>
<td>9/10</td>
<td>0.0107*</td>
<td>0.0035*</td>
</tr>
<tr>
<td>0.90</td>
<td>0.0033</td>
<td>0.0063</td>
<td>+89.3</td>
<td>2/3</td>
<td>0.5000</td>
<td>undef</td>
</tr>
<tr>
<td>1.00</td>
<td>0.0028</td>
<td>0.0050</td>
<td>+76.9</td>
<td>2/3</td>
<td>0.5000</td>
<td>undef</td>
</tr>
<tr>
<td>Avg.</td>
<td>0.1868</td>
<td>0.2233</td>
<td>+19.55</td>
<td>32/49</td>
<td>0.0222*</td>
<td>0.0003*</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>prec.</th>
<th>tf.idf</th>
<th>LM</th>
<th>%chg</th>
<th>I/D</th>
<th>Sign</th>
<th>Wilc</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.4939</td>
<td>0.5020</td>
<td>+1.7</td>
<td>10/21</td>
<td>0.6682</td>
<td>0.4106</td>
</tr>
<tr>
<td>10</td>
<td>0.4449</td>
<td>0.4898</td>
<td>+10.1</td>
<td>22/30</td>
<td>0.0081*</td>
<td>0.0154*</td>
</tr>
<tr>
<td>15</td>
<td>0.3932</td>
<td>0.4435</td>
<td>+12.8</td>
<td>19/26</td>
<td>0.0145*</td>
<td>0.0038*</td>
</tr>
<tr>
<td>20</td>
<td>0.3643</td>
<td>0.4051</td>
<td>+11.2</td>
<td>22/34</td>
<td>0.0607</td>
<td>0.0218*</td>
</tr>
<tr>
<td>30</td>
<td>0.3313</td>
<td>0.3707</td>
<td>+11.9</td>
<td>28/41</td>
<td>0.0138*</td>
<td>0.0070*</td>
</tr>
<tr>
<td>100</td>
<td>0.2157</td>
<td>0.2500</td>
<td>+15.9</td>
<td>32/42</td>
<td>0.0005*</td>
<td>0.0003*</td>
</tr>
<tr>
<td>200</td>
<td>0.1855</td>
<td>0.1903</td>
<td>+15.0</td>
<td>35/44</td>
<td>0.0001*</td>
<td>0.0000*</td>
</tr>
<tr>
<td>500</td>
<td>0.1004</td>
<td>0.1119</td>
<td>+11.4</td>
<td>36/44</td>
<td>0.0000*</td>
<td>0.0000*</td>
</tr>
<tr>
<td>1000</td>
<td>0.0653</td>
<td>0.0887</td>
<td>+5.1</td>
<td>36/43</td>
<td>0.0000*</td>
<td>0.0002*</td>
</tr>
<tr>
<td>RPr</td>
<td>0.2473</td>
<td>0.2876</td>
<td>+16.32</td>
<td>34/43</td>
<td>0.0001*</td>
<td>0.0000*</td>
</tr>
</tbody>
</table>
## Precision/recall results 51-100

<table>
<thead>
<tr>
<th></th>
<th>tf.idf</th>
<th>LM</th>
<th>%chg</th>
<th>I/D</th>
<th>Sign</th>
<th>Wilc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rec.</td>
<td>10485</td>
<td>10485</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rrel.</td>
<td>5818</td>
<td>6105</td>
<td>+4.93</td>
<td>32/42</td>
<td>0.0005*</td>
<td>0.0003*</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Prec.</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.7274</td>
<td>0.7805</td>
<td>+7.3</td>
<td>10/22</td>
<td>0.7383</td>
<td>0.2961</td>
</tr>
<tr>
<td>0.10</td>
<td>0.4861</td>
<td>0.5002</td>
<td>+2.9</td>
<td>26/44</td>
<td>0.1456</td>
<td>0.1017</td>
</tr>
<tr>
<td>0.20</td>
<td>0.3898</td>
<td>0.4088</td>
<td>+4.9</td>
<td>24/45</td>
<td>0.3830</td>
<td>0.1405</td>
</tr>
<tr>
<td>0.30</td>
<td>0.3352</td>
<td>0.3626</td>
<td>+8.2</td>
<td>28/47</td>
<td>0.1215</td>
<td>0.0277*</td>
</tr>
<tr>
<td>0.40</td>
<td>0.2826</td>
<td>0.3064</td>
<td>+8.4</td>
<td>25/45</td>
<td>0.2757</td>
<td>0.0286*</td>
</tr>
<tr>
<td>0.50</td>
<td>0.2163</td>
<td>0.2512</td>
<td>+16.2</td>
<td>26/40</td>
<td>0.0403*</td>
<td>0.0007*</td>
</tr>
<tr>
<td>0.60</td>
<td>0.1563</td>
<td>0.1798</td>
<td>+15.2</td>
<td>20/30</td>
<td>0.0494*</td>
<td>0.0025*</td>
</tr>
<tr>
<td>0.70</td>
<td>0.0913</td>
<td>0.1109</td>
<td>+21.5</td>
<td>14/22</td>
<td>0.1431</td>
<td>0.0288*</td>
</tr>
<tr>
<td>0.80</td>
<td>0.0540</td>
<td>0.0529</td>
<td>+3.7</td>
<td>8/13</td>
<td>0.2905</td>
<td>0.2108</td>
</tr>
<tr>
<td>0.90</td>
<td>0.0179</td>
<td>0.0152</td>
<td>-14.9</td>
<td>1/4</td>
<td>0.3125</td>
<td>undef*</td>
</tr>
<tr>
<td>1.00</td>
<td>0.0005</td>
<td>0.0004</td>
<td>-11.9</td>
<td>1/2</td>
<td>0.7500</td>
<td>undef*</td>
</tr>
<tr>
<td>Avg.</td>
<td>0.2286</td>
<td>0.2486</td>
<td>+8.74</td>
<td>32/50</td>
<td>0.0325*</td>
<td>0.0015*</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Prec.</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.5320</td>
<td>0.5960</td>
<td>+12.0</td>
<td>15/21</td>
<td>0.0392*</td>
<td>0.0125*</td>
</tr>
<tr>
<td>10</td>
<td>0.5080</td>
<td>0.5260</td>
<td>+3.5</td>
<td>14/30</td>
<td>0.7077</td>
<td>0.1938</td>
</tr>
<tr>
<td>15</td>
<td>0.4933</td>
<td>0.5053</td>
<td>+2.4</td>
<td>14/28</td>
<td>0.5747</td>
<td>0.3002</td>
</tr>
<tr>
<td>20</td>
<td>0.4670</td>
<td>0.4890</td>
<td>+4.7</td>
<td>16/34</td>
<td>0.6962</td>
<td>0.1280</td>
</tr>
<tr>
<td>30</td>
<td>0.4293</td>
<td>0.4593</td>
<td>+7.0</td>
<td>20/32</td>
<td>0.1077</td>
<td>0.0095*</td>
</tr>
<tr>
<td>100</td>
<td>0.3344</td>
<td>0.3562</td>
<td>+6.5</td>
<td>29/45</td>
<td>0.0362*</td>
<td>0.0076*</td>
</tr>
<tr>
<td>200</td>
<td>0.2670</td>
<td>0.2852</td>
<td>+6.8</td>
<td>29/44</td>
<td>0.0244*</td>
<td>0.0009*</td>
</tr>
<tr>
<td>500</td>
<td>0.1797</td>
<td>0.1881</td>
<td>+4.7</td>
<td>30/42</td>
<td>0.0040*</td>
<td>0.0011*</td>
</tr>
<tr>
<td>1000</td>
<td>0.1164</td>
<td>0.1221</td>
<td>+4.9</td>
<td>32/42</td>
<td>0.0005*</td>
<td>0.0003*</td>
</tr>
<tr>
<td>RPr</td>
<td>0.2836</td>
<td>0.3013</td>
<td>+6.24</td>
<td>30/43</td>
<td>0.0069*</td>
<td>0.0052*</td>
</tr>
</tbody>
</table>
Extension: 3-level model

Information need

\[ P(Q \mid M_C, M_T) \]
\[ P(Q \mid M_C, M_T, M_d) \]

generation

query

document collection

\[ M_C \]
\[ M_{T_1} \]
\[ M_{T_2} \]
\[ \vdots \]
\[ M_{T_m} \]
\[ M_{d_1} \]
\[ M_{d_2} \]
\[ \vdots \]
\[ M_{d_n} \]

\( d1 \)
\( d2 \)
\( \vdots \)
\( dn \)
LMs vs. vector space model (1)

- LMs have some things in common with vector space models.
- Term frequency is directed in the model.
  - But it is not scaled in LMs.
- Probabilities are inherently “length-normalized”.
  - Cosine normalization does something similar for vector space.
- Mixing document and collection frequencies has an effect similar to idf.
  - Terms rare in the general collection, but common in some documents will have a greater influence on the ranking.
LMs vs. vector space model (2)

- **LMs vs. vector space model: commonalities**
  - Term frequency is directly in the model.
  - Probabilities are inherently “length-normalized”.
  - Mixing document and collection frequencies has an effect similar to idf.

- **LMs vs. vector space model: differences**
  - LMs: based on probability theory
  - Vector space: based on similarity, a geometric/linear algebra notion
  - Collection frequency vs. document frequency
  - Details of term frequency, length normalization etc.
Vector space (tf-idf) vs. LM

<table>
<thead>
<tr>
<th>Rec.</th>
<th>tf-idf</th>
<th>LM</th>
<th>%chg</th>
<th>significant?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.7439</td>
<td>0.7590</td>
<td>+2.0</td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td>0.4521</td>
<td>0.4910</td>
<td>+8.6</td>
<td>*</td>
</tr>
<tr>
<td>0.2</td>
<td>0.3514</td>
<td>0.4045</td>
<td>+15.1</td>
<td>*</td>
</tr>
<tr>
<td>0.4</td>
<td>0.2093</td>
<td>0.2572</td>
<td>+22.9</td>
<td>*</td>
</tr>
<tr>
<td>0.6</td>
<td>0.1024</td>
<td>0.1405</td>
<td>+37.1</td>
<td>*</td>
</tr>
<tr>
<td>0.8</td>
<td>0.0160</td>
<td>0.0432</td>
<td>+169.6</td>
<td>*</td>
</tr>
<tr>
<td>1.0</td>
<td>0.0028</td>
<td>0.0050</td>
<td>+76.9</td>
<td></td>
</tr>
<tr>
<td>11-point average</td>
<td>0.1868</td>
<td>0.2233</td>
<td>+19.6</td>
<td>*</td>
</tr>
</tbody>
</table>

The language modeling approach always does better in these experiments . . . . . . but note that where the approach shows significant gains is at higher levels of recall.
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 assumes that documents and expressions of information problems are of the same type
• Computationally tractable, intuitively appealing
LM vs. Prob. Model for IR

• Problems of 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.