Expression language modeling approaches as region model queries

Document Space 2006, 23 – 27 January 2006

Djoerd Hiemstra
University of Twente

Today's talk

- Some language modelling stuff we did in the past
  - Ad hoc search
  - Cross-language search
  - Web search
- About the system we build today
  - TIJAH: our XML-IR system
  - TIJAH, for all your IR problems?

Statistical language models

- Noisy channel paradigm

I (input) $\longrightarrow$ $\boxed{\text{noisy channel}}$ $\longrightarrow$ O (output)

- hypothesise all possible input texts I and take the one with the highest probability, symbolically:

$$\hat{I} = \underset{I}{\operatorname{argmax}}\, P(I \mid O)$$
$$= \underset{I}{\operatorname{argmax}}\, P(I) \cdot P(O \mid I)$$

3/52

Statistical language models

- Noisy channel paradigm

D (document) $\longrightarrow$ $\boxed{\text{noisy channel}}$ $\longrightarrow$ $T_1, T_2, \ldots$(query)

- hypothesise all possible documents D and take the one with the highest probability, symbolically:

$$\hat{D} = \underset{D}{\operatorname{argmax}}\, P(D \mid T_1, T_2, \cdots)$$
$$= \underset{D}{\operatorname{argmax}}\, P(D) \cdot P(T_1, T_2, \cdots \mid D)$$

4/52

Statistical language models

- Let's assume we point blindly, one at a time, at 3 words in a document.
- What is the probability that I, by accident, pointed at the words "Document", "Space" and "2006"?
- Compute the probability, and use it to rank the documents.

Statistical language models

- Given a query $T_1, T_2, \ldots, T_n$, rank the documents according to the following probability measure:

$$P(T_1, T_2, \ldots, T_n \mid D) = \prod_{i=1}^{n} ((1-\lambda)P(T_i) + \lambda P(T_i \mid D))$$

- Linear combination of document model and background model

λ : probability of document model

$1-\lambda$: probability of background model

$P(T_i \mid D)$: document model

$P(T_i)$: background model

Statistical language models

- Definition of probability measures:

$$P(T_i = t_i \mid D = d) = \frac{tf(t_i, d)}{\sum_t tf(t, d)}$$

$$P(T_i = t_i) = \frac{cf(t_i)}{\sum_t cf(t)} \quad \text{or}$$

$$= \frac{\sum_d tf(t_i, d)}{\sum_t \sum_d tf(t, d)}$$

7/52

Statistical language models

- How to estimate value of λ?
  - Use EM algorithm on some training queries + relevant documents (or just pick any number: $\lambda = 0.5$)
  - Note that for extreme values:

    $\lambda = 0$: random retrieval

    $\lambda = 1$: only retrieve documents containing all terms

    $\lim \lambda \rightarrow 1$: docs containing n query terms are ranked above docs containing $n-1$ terms

8/52

Statistical language models

- Implementation

$$P(T_1, T_2, \cdots, T_n \mid D) = \prod_{i=1}^{n} ((1-\lambda)P(T_i) + \lambda P(T_i \mid D))$$

$$\vdots$$

$$P(T_1, T_2, \cdots, T_n \mid D) \propto \sum_{i=1}^{n} \log(1 + \frac{\lambda P(T_i \mid D)}{(1-\lambda)P(T_i)})$$

9/52

Statistical language models

- Implementation as vector product:

$$score(q,d) \quad = \sum_{k \in \text{matching terms}} q_k \cdot d_k$$

$$q_k = tf(k,q)$$

$$d_k = \log(1 + \frac{tf(k,d) \sum_t cf(t)}{cf(k) \sum_t tf(t,d)} \frac{\lambda}{1-\lambda})$$

10/52

Experimental results

run	precision at document: 10	30	100	average precision
tfc.tfc	0.240	0.187	0.122	0.126
probabilistic	0.248	0.187	0.153	0.165
Lnu.ltu	0.450	0.345	0.214	0.229
BM25	0.484	0.366	0.234	0.261
LM	0.494	0.385	0.235	0.277

Results of ad hoc queries

11/52

Cross-language IR

cross-language information retrieval
zoeken in anderstalige informatie
recherche d'informations multilingues

Cross-language IR

- Cross-language information retrieval (CLIR):
  - Enter query in one language (language of choice) and retrieve documents in one or more other languages.
  - The system takes care of automatic translation

Language models & translation

- Noisy channel paradigm

D (doc.) $\longrightarrow$ [noisy channel] $\longrightarrow$ $T_1, T_2, \ldots$ (query) $\longrightarrow$ [noisy channel] $\longrightarrow$ $S_1, S_2, \ldots$ (request)

- hypothesise all possible documents D and take the one with the highest probability:

$$\hat{D} = \operatorname*{argmax}_D P(D \mid S_1, S_2, \cdots)$$
$$= \operatorname*{argmax}_D P(D) \cdot \sum_{T_1, T_2, \cdots} P(T_1, T_2, \cdots; S_1, S_2, \cdots \mid D)$$

Language models & translation

- Cross-language information retrieval :
  - Assume that the translation of a word/term does not depend on the document in which it occurs.
  - if: $S_1, S_2, \ldots, S_n$ is a Dutch query of length n
  - and $t_{i1}, t_{i2}, \ldots, t_{im}$ are m English translations of the Dutch query term S_i

$$P(S_1, S_2, ..., S_n | D) =$$
$$\prod_{i=1}^{n} \sum_{j=1}^{m_i} P(S_i | T_i = t_{ij})((1-\lambda)P(T_i = t_{ij}) + \lambda P(T_i = t_{ij} | D))$$

16/52

Language models & translation

- How does it work in practice?
  - Find for each Dutch query term N_i the possible translations $t_{i1}, t_{i2}, \ldots, t_{im}$ and translation probabilities
  - Combine them in a structured query
  - Process structured query

17/52

Language models & translation

- Example:
  1. Dutch query: *gebroken hart*
  2. Translations of *gebroken* :
     broken (1.0) or *fractured* (0.2)
  3. Translations of *hart* :
     heart (0.5) or *ticker* (0.1)
  4. Structured query:
     ((*1.0 broken* $\cup$ *0.5 fractured*) ,
     (*0.5 heart* $\cup$ *0.1 ticker*))

Experimental results

method	average precision	relative
best automatic disambiguation	0.258	69 %
manual disambiguation	0.292	78 %
simple model (unstructured query)	0.285	77 %
translation model (structured query)	0.309	83 %

Web Search

Navigational queries: find the home page ("entry page") of …

Web Search: about priors and static ranking

- Noisy channel paradigm

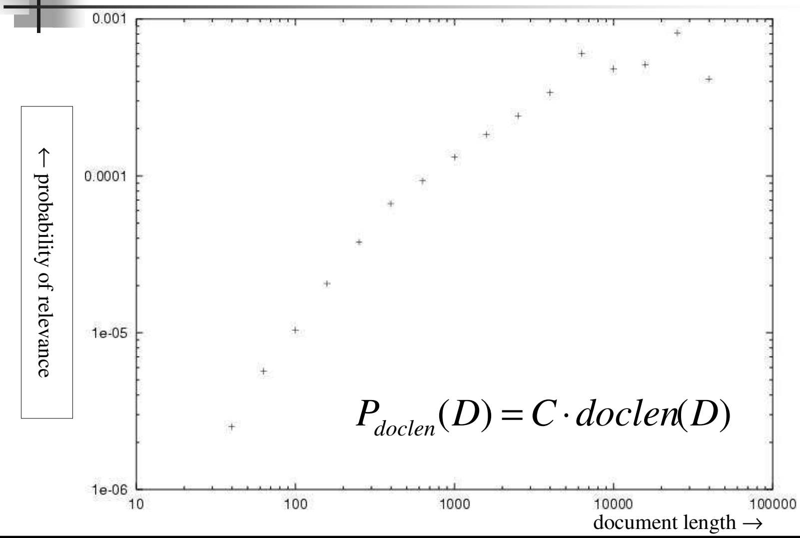
D (document) $\longrightarrow$ noisy channel $\longrightarrow$ $T_1, T_2, \ldots$(query)

- hypothesise all possible documents D and take the one with the highest probability, symbolically:

$$\hat{D} = \underset{D}{\operatorname{argmax}}\, P(D \mid T_1, T_2, \cdots)$$
$$= \underset{D}{\operatorname{argmax}}\, P(D) \cdot P(T_1, T_2, \cdots \mid D)$$

21/52

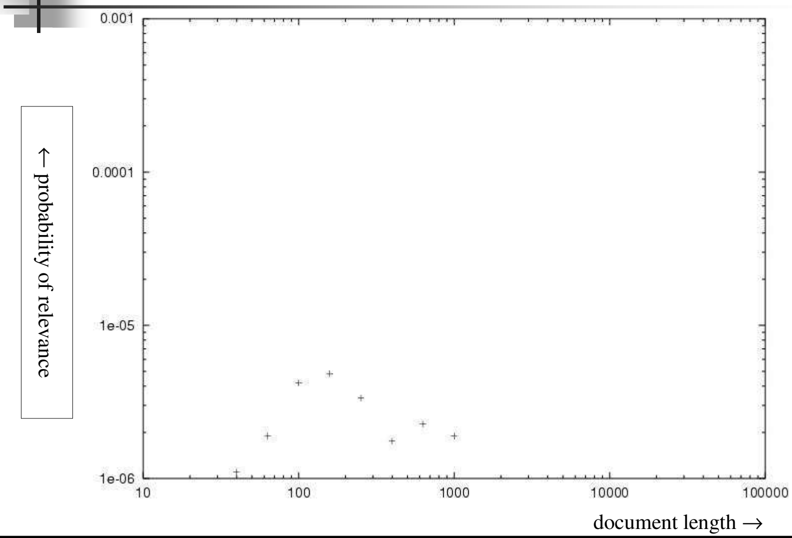
Prior probability of relevance on ad-hoc search task



Priors in Entry Page Search

- Sources of Information
  - Document length
  - Number of links pointing to a document
  - The depth of the URL
  - Occurrence of cue words (‘welcome’,’home’)
  - number of links in a document
  - page traffic

Priors in Entry Page Search



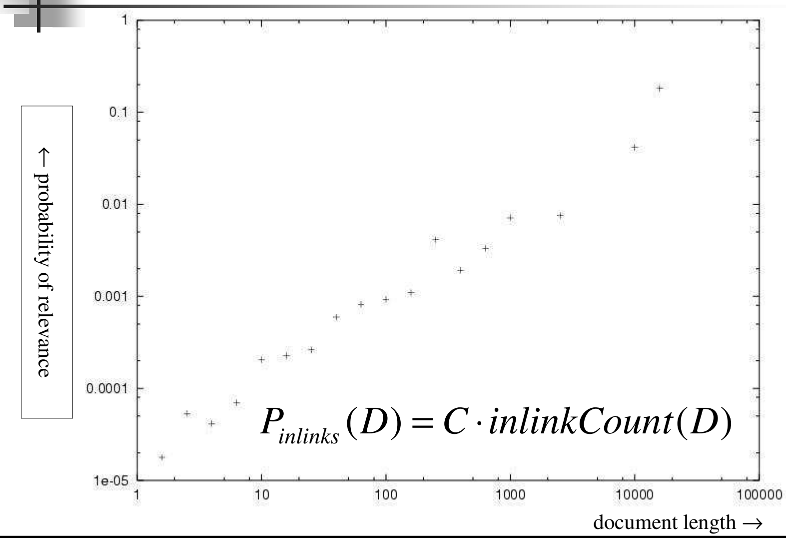
24/52

Priors in Entry Page Search

- Assumption
  - Entry pages referenced more often
- Different types of inlinks
  - From other hosts (recommendation)
  - From same host (navigational)
- Both types point often to entry pages

25/52

Priors in Entry Page Search



$$P_{inlinks}(D) = C \cdot inlinkCount(D)$$

26/52

Priors in Entry Page Search URL depth

- Top level documents are often entry pages
- Four types of URLs
  - root: `www.ipam.ucla.edu`
  - subroot: `www.ipam.ucla.edu/programs/`
  - path: `www.ipam.ucla.edu/programs/ds2006/`
  - file: `www.ipam.ucla.edu/programs/ds2006/reg.html`

27/52

Priors in Entry Page Search results

method	Content	Anchors
$P(Q\|D)$	0.3375	0.4188
$P(Q\|D)P_{doclen}(D)$	0.2634	0.5600
$P(Q\|D)P_{inlink}(D)$	0.4974	0.5365
$P(Q\|D)P_{URL}(D)$	0.7705	0.6301

28/52

Lessons learned

- Probability functions are no queries (yet)!
  - We built systems for cross-language search, web search, video search, etc., using standard software
  - To built these systems we had to re-design APIs, introduce new indexing and storage structures, new query languages, etc.
  - Often, new APIs and indexing structures logically impaired one or more older applications

29/52

What I need from document space

1. The single most important thing is the document similarity function
   - it might not be symmetric…
   - it might combine more information… (… than fits in a document/term matrix)
2. Documents have internal structure (title, author, part-of-speech, named entities, any other markup)

30/52

What I need from document space

3. Well-defined, well-understood behaviour of systems
4. Applicability to, and good performance for, many search problems
5. Complexity of system has to be encapsulated in a module that responds to simple command strings (similar to SQL or relational algebra)

31/52

PART TWO

region models

Overview

- The TIJAH XML-IR system:
  - What are region models?
- Expressing language modeling approa-ches as 'logical' region expressions
  - simple ad-hoc retrieval
  - video search
  - web search
  - cross-language search

Region models

- A text database is a finite sequence of words, each word has a unique position in the database;
- A region is defined by a start position and an end position (and a score);
- Some regions are pre-defined (for instance documents, titles, author names, etc.).
- Any region is a valid retrieval unit)
- Retrieval is done using a small set of operaters: **`AND`**, **`OR`**, **`CONTAINING`**, **`CONTAINED_BY`**, and **`SCALE`**

34/52

Tags and terms are numbered

```
1 <section>
    2 <title> 3 Information 4 Retrieval 5 Using 6 RDBMS 7 </title>
    8 <section>
        9 <title> 10 Beyond 11 Simple 12 Translation 13 </title>
        14 <section>
            15 <title> 16 Extension 17 of 18 IR 19 Features 20 </title>
        21 </section>
    22 </section>
23 </section>
```

Regions stored as relations:

<section> →	(1,23) (8,22) (14,21) … …
<title> →	(2,7) (9,13) (15,20) … …
“information” →	(3,3) … …
“retrieval” →	(4,4) … …

35/52

Region queries by example

- **`banana`**
  - produces a set of regions, each region defining the position where the word "banana" occurs in the database, *each score=1*
- **`<recipe>`**
  - produces the set of regions defing word sequences tagged as a "recipe" *with score=1*
- **`0.5 SCALE <recipe>`**
  - *produces the set of regions defing word sequences tagged as a "recipe" with score=0.5*

36/52

Region queries by example

- **`<recipe> CONTAINING banana`**
  - produces a set of regions tagged as "recipe" that contain the word "banana", *each scored by the number of occurrences of "banana" divided by the length of the region*.
- **`<ingredient> CONTAINED_BY <recipe>`**
  - produces a set of regions tagged as "ingredient" that are contained by at least one recipe. *If an ingredient is contained by more than one (nested) recipe, then those recipe scores are added in the final score*

37/52

Region queries by example

- **(<doc> CONTAINING db) AND (<doc> CONTAINING ir)**
  - the AND operator is the traditional intersection. The query produces the set of "doc" regions containing both "db" and "ir". *Region scores are multiplied*
- **(<doc> CONTAINING db) OR (<doc> CONTAINING ir)**
  - the OR operator is the traditional union. The query produces the set of "doc" regions containing either "db" or "ir" or both. *Region scores are added if the region contains both*

38/52

Region queries by example

- Nexi:

  //article[about(.//atl|.//kwd, book review)]//sec[about(., databases)]
- Region query:

```
(<sec> CONTAINING databases) CONTAINED_BY
   (<article> CONTAINING
   (((<atl> OR <kwd>) CONTAINING book)
   CONTAINING review))
```

39/52

Region queries by example

<person> ADJ (0.8 SCALE killed) OR (0.2 SCALE murdered)) ADJ (<person> CONTAINING (Abraham ADJ Lincoln)))

- Pronounce as:
  - Find me all <person> regions, *adjacent to* either "killed" (with confidence 0.8) *or* "murdered" (with confidence 0.2) *adjacent to* a <person> *containing* "Abraham" *adjacent to* "Lincoln".

Region algebra

<person> CONTAINING (Abraham OR Abe)

Set of all <person> regions

Name	Start	End	Score
<person>	2	6	1.0
<person>	8	11	1.0
<person>	43	47	1.0
:			

$\bowtie$

contain-ment join

Set of all Abraham OR Abe regions

Name	Start	End	Score
Abraham	9	9	1.0
Abraham	88	88	1.0
Abe	161	161	1.0
:			

Region algebra

all <person> regions

Name	Start	End	Score
<person>	2	6	1.0
<person>	8	11	1.0
<person>	43	47	1.0
:			

$\bowtie$ contain-ment join

Set of all Abraham OR Abe regions

Name	Start	End	Score
Abraham	9	9	1.0
Abraham	88	88	1.0
Abe	161	161	1.0
:			

Name	Start	End	Score
<person>	8	11	0.5

43/52

Hold on…

What does this have to do with language models?

Simple language model

$$P(T_1 \cdots T_n \mid D) = \prod_{i=1}^{n} P(T_i \mid D)$$

$$P(T_1 = db, T_2 = ir \mid D) = P(T_1 = db \mid D) \cdot P(T_2 = ir \mid D)$$

(<doc> CONTAINING db) AND (<doc> CONTAINING ir)

45/52

Smoothing

$$P(T_1 = db, T_2 = ir \mid D) =$$
$$(0.2 \cdot P(T_1 = db \mid C) + 0.8 \cdot P(T_1 = db \mid D))$$
$$\cdot (0.2 \cdot P(T_2 = ir \mid C) + 0.8 \cdot P(T_2 = ir \mid D))$$

```
(<doc> CONTAINED_BY
   ((0.2 SCALE (<root> CONTAINING db) OR
                  (0.8 SCALE (<doc> CONTAINING db))))
AND (<doc> CONTAINED_BY
   ((0.2 SCALE (<root> CONTAINING ir) OR
                  (0.8 SCALE (<doc> CONTAINING ir))))
```

46/52

Video shot retrieval

$$P(T = ni \mid Shot) =$$
$$(0.18 \cdot P(T = ni \mid C) + 0.02 \cdot P(T = ni \mid Video)$$
$$+ 0.4 \cdot P(T = ni \mid Scene) + 0.4 \cdot P(T = ni \mid Shot))$$

```
(<shot> CONTAINED_BY
    ((0.18 SCALE (<root> CONTAINING ni)) OR
     (0.02 SCALE (<video> CONTAINING ni)) OR
     (0.4 SCALE (<scene> CONTAINING ni)) OR
     (0.4 SCALE (<shot> CONTAINING ni)))
```

47/52

Cross-language retrieval

$$P(S_1 = gebroken, S_2 = hart \mid D) =$$
$$(1.0 \cdot P(T_1 = broken \mid D) + 0.2 \cdot P(T_1 = fractured \mid D))$$
$$\cdot (0.5 \cdot P(T_2 = heart \mid D) + 0.1 \cdot P(T_1 = ticker \mid D))$$

```
  (1.0 SCALE (<doc> CONTAINING broken)) OR
       (0.2 SCALE (<doc> CONTAINING fractured)))
AND (0.5 SCALE (<doc> CONTAINING heart)) OR
       (0.1 SCALE (<doc> CONTAINING ticker)))
```

48/52

Web Retrieval

$$P(T_1 \cdots T_n, D) = P(D) \prod_{i=1}^{n} P(T_i \mid D)$$

$$P(T = google, D) = P(D) \cdot P(T = google \mid D)$$

```
$PageRank AND (<doc> CONTAINING google)
```

49/52

Conclusion

- A major break-though!

  or

- a funny coincidence…

50/52

… but please note

- Probability functions are arithmetic expressions specifying the probability of a single document
- Region queries are algebraic expressions processing sets of document regions. The algebra is fully composable

More info

- http://www.cs.utwente.nl/~cirquid
- http://www.cs.utwente.nl/~hiemstra

- Djoerd Hiemstra and Wessel Kraaij, ``A Language Modeling Approach to TREC'', In Ellen Voorhees and Donna Harman (eds.), *TREC: Experiment and Evaluation in Information Retrieval*, MIT Press, 2005
- Djoerd Hiemstra and Vojkan Mihajlovic, ``A database approach to information retrieval: The remarkable relationship between language models and region models'', *CTIT Technical Report 05-35*, 2005
- Henning Rode, Georgina Ramirez, Thijs Westerveld, Djoerd Hiemstra, and Arjen de Vries, ``The Lowlands' TREC Experiments 2005'', *Proceedings of the 14th Text Retrieval Conference (TREC)*, 2005


Many thanks to:
Henning Rode, Vojkan Mihajlovic,
Arjen de Vries, Georgina Ramirez