

language models for document retrieval

Question Answering



Searching Documents: Assumptions

- Set of documents
- Retrieve relevant documents
 - Search engine expects a query (e.g. keywords)
- Documents, Query = Text
- Order documents by relevance

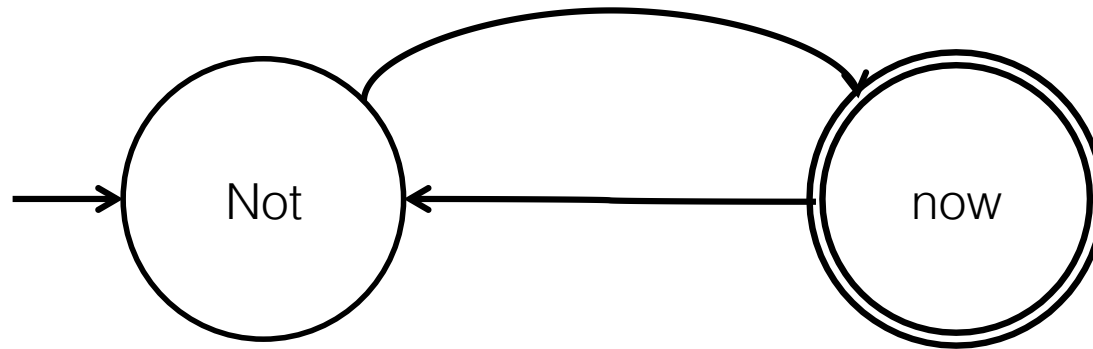
Searching Documents: Assumptions

- User has a certain document in mind
- Document is good match if
 - Query terms appear frequently in the document
 - document (language) model is likely to *generate* the query
- Documents relevance is dependent on likeliness

Using LM for DR

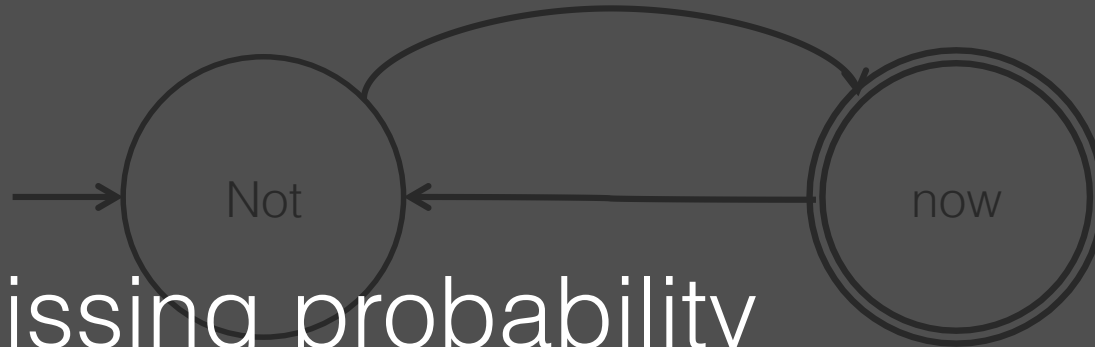
1. Infer a language model for each document
2. Compute **probability** of **generating** the query
 - Query Likelihood Model specific instance of LM approach

Generative Models



- Not now
- Not now Not now
- Not now Not now Not now
- ...

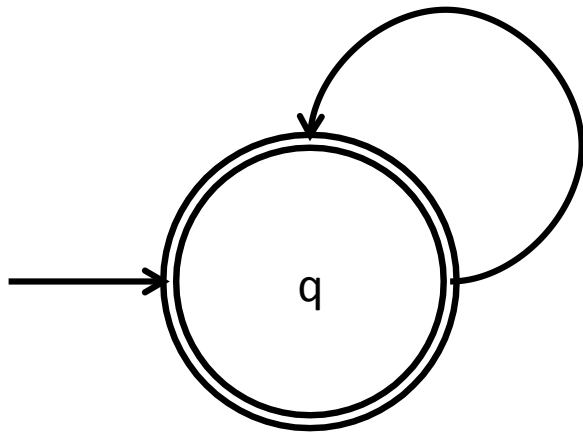
Generative Models



Missing probability
No arbitrary ordering of terms

- Not now
- Not now Not now
- Not now Not now Not now
- ...

Language Model



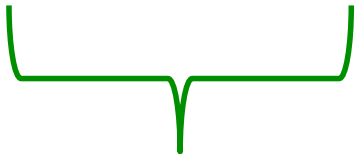
term	$P(\text{term}, q)$
the	0.2
not	0.1
frog	0.01
dog	0.03
now	0.02
<i>STOP</i>	0.2

$P(\text{not now dog not now}) =$

Language Model

- puts a probability measure over terms
- Different types:
 - Unigram: Estimate each term independently
 - $P(t_1 t_2 t_3) = P(t_1) P(t_2) P(t_3)$
 - $P(t_1 t_2 t_3) = P(t_1) P(t_2 | t_1) P(t_3 | t_2)$ (bigram)
 - $P(t_1 t_2 t_3) = P(t_1) P(t_2 | t_1) P(t_3 | t_1 t_2)$

Query Likelihood Model

$$P(d|q) = \frac{P(q|d)}{P(q)} P(d)$$


Relevance of a document d to a query q

Query Likelihood Model

Use Language Model

initial relevance (a priori)
equal over all documents

$$P(d|q) = \frac{P(q|d)}{P(q)} P(d)$$

The diagram shows the equation $P(d|q) = \frac{P(q|d)}{P(q)} P(d)$. The terms are enclosed in green boxes with lines pointing to explanatory text. $P(q|d)$ is in a box with a line pointing to 'Use Language Model'. $P(d)$ is in a box with a line pointing to 'initial relevance (a priori) equal over all documents'. $P(q)$ is in a box with a line pointing to 'Constant value'. The entire fraction $\frac{P(q|d)}{P(q)}$ is enclosed in a larger box with a line pointing to 'Impact of query on initial document relevance'.

Constant value

Impact of query on initial document
relevance

Query Likelihood Model

$$P(d|q) \sim P(q|d)$$

Query Likelihood Model

- Computing $P(q | d)$ using LM
- Unigram assumption:

$$P(q|M_d) = \prod_{t \in q} P(t|M_d)^{tf_{t,d}}$$

Query Likelihood Model

- Formula so far

$$P(d|q) \sim P(q|d) = \prod_{t \in q} \frac{tf_{t,d}}{L_d}$$

Zero Probabilities

Query larry ellison

Document 1 larry ellison is the co-founder of oracle

Document 2 ellison was born in new york city

$$P(\text{Query} \mid \text{Document 1}) = 1/7 \times 1/7 = 1/49$$

$$P(\text{Query} \mid \text{Document 2}) = 0/7 \times 1/7 = 0$$

Zero Probabilities

- Case: $P(t_{\text{missed}} | M_d) = 0$
- Basic Idea: allow non-occurring terms

$$P(t_{\text{missed}} | M_c) = \frac{tf_{t_{\text{missed}}}}{T}$$

Model built from collection

number of terms of collection

Linear Interpolation Smoothing

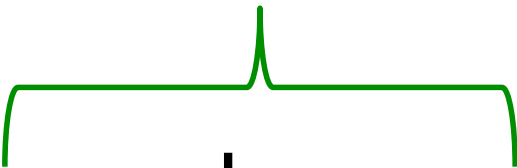
$$P(d|q) \sim \prod_{1 \leq k \leq |q|} (\lambda P(t_k | M_d) + (1 - \lambda) P(t_k | M_c))$$

- Assign λ weight
- M_c Model built from Collection of all documents
- Chose λ according to query length
 - Large value for short queries (unlikely that terms are missing)

Bayesian Smoothing

$$P(t|d) = \frac{tf_{t,d} + \alpha P(t|M_c)}{L_d + \alpha}$$

prior probability



t = (missing) term
L_d = Document size
Alpha = Weight

Extensions

- LMs do not address issues of alternate expression (synonyms)
- Translation Models:
 - Generate missing query terms by translation
 - January 2 2012 by Philipp Langer



end

Resources

- <http://nlp.stanford.edu/IR-book/>
 - Chapter 11, 12
- http://www.uni-stuttgart.de/gi/research/schriftenreihe/quo_vadis/pdf/koch.pdf

