University of Twente

Information Retrieval Modeling

Russian Summer School in Information Retrieval

Djoerd Hiemstra
http://www.cs.utwente.nl/~hiemstra

University of Twente
The Netherlands

4 volunteers needed to submit papers

- Each person: Choose a “personal language model” consisting of 3 words
  - For instance: {RuSSIR, Summer, School}
- Write two English papers, both with:
  - Two authors: *not both papers the same author, put the author names on the paper*,
  - Four words max.: *two words from one author, two from the other, don't show who wrote what*

Overview

1. EM training
2. Index compression and query optimization

Course Material

- Djoerd Hiemstra, “Language Models, Smoothing, and N-grams'', In M. Tamer Özsu and Ling Liu (eds.) *Encyclopedia of Database Systems, Springer*, 2009

What about relevance feedback?

- We assume that a (one) relevant document has generated the query
- So, once we find that document, we might as well stop.
- What we need is a model of "relevance", or language models of sets of relevant documents

Lavrenko's relevance model

- "Construct a relevance model $P(T|R)$ by assuming that once we pick a relevant document *D*, the probability of observing a word is independent from the set of relevant documents"

$$P(T|R) = \sum_{D \in R} P(T|D)P(D|R)$$

- we only have information about *R* through a query

$$P(T|q_1, \ldots) = \sum_{D \in R} P(T|D)P(D|q_1, \ldots)$$

Lavrenko's relevance model 1

- Is really a blind feedback method:
  - do an initial run and assign $P(D|q_1,...)$
  - for every retrieved document, get the most frequent terms *T*, and assign those $P(T|D)$
  - multiply both probabilities, and sum them for each document retrieved

Balog's expert finder

- As in Lavrenko's method, use query to retrieve some initial documents.
- Instead of query (term) expansion, do person name expansion
  - for every retrieved document, get the candidates *ca*, and assign those $P(ca \mid D)$
  - multiply both probabilities, and sum them for each document retrieved

(Balog et al. 2006)

Balog's expert finder

- "Construct a *candidate* model $P(ca|R)$ by assuming that once we pick a relevant document *D*, the probability of observing a *candidate expert* is independent from the set of relevant documents"

$$P(ca|R) = \sum_{D \in R} P(ca|D) P(D|R)$$

- we only have information about *R* through a query

$$P(ca|q_1, \ldots) = \sum_{D \in R} P(ca|D) P(D|q_1, \ldots)$$

$$\sum_{D \in R} P(ca|D) \prod_{i=1}^{n} ((1-\lambda) P(q_i) + \lambda P(q_i|D))$$

University of Twente
The Netherlands

Balog's expert finder

	#rel	MAP	R-prec	MRR	P10
Model 1 (candidate model):					
BASE	**511**	**0.1253**	**0.1914**	0.2759	**0.236**
Model 2 (document model):					
BASE	**580**	**0.1880**	0.2332	0.5149	**0.316**

Figure 2, Candidate model vs. document model

The relevance model in action

Q = "environmental protection laws" 环境保护法

P(word|Q)	word	meaning
0.061	,	[punctuation]
0.036	的	[possessive suffix]
0.027	。	[punctuation]
0.017	和	and
0.016	、	[punctuation]
0.009	环境	environment
0.009	了	[end of sentence]
0.008	海洋	sea
0.008	法	law
0.008	资源	resource
0.007	全国	whole country
0.007	在	in
0.006	保护	protect
0.006	污染	pollution
0.006	胶	rubber
0.006	发泡	defects in plastic
0.005	与	and
0.005	中国	china
0.005	产品	product
0.005	法律	law

University of Twente
The Netherlands

The relevance model in action

Q = “amazon rain forest”

word	*probability*
the	0.0776
of	0.0386
and	0.0251
to	0.0244
in	0.0203
amazon	0.0114
for	0.0109
:	
assistence	0.0009
macminn	0.0008

These are common words: should be explained by general (background) model

interesting word!

These are too specific: might be explained by a single document model

What we need is *parsimony*

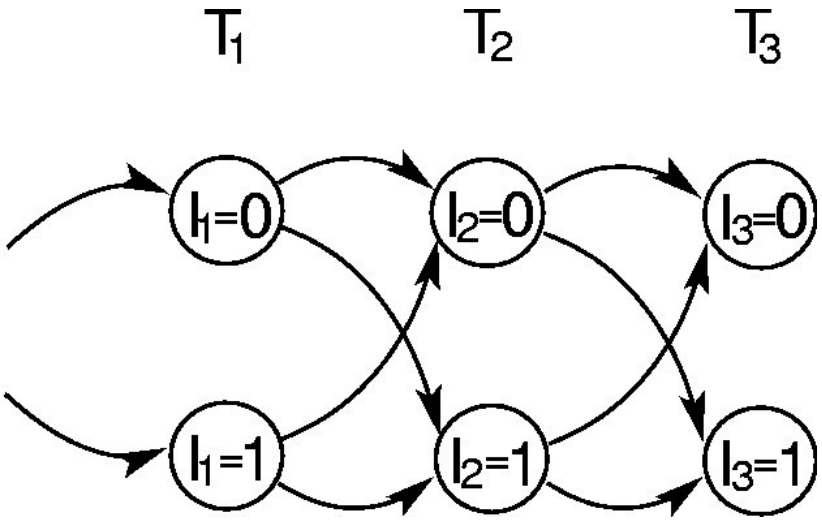
- Optimize the probability to predict language use
- Minimize the total number of parameters needed for that
- Expectation Maximization Training

(Hiemstra, Robertson & Zaragoza 2004).

Statistical language models

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

- Presentation as hidden Markov model
  - finite state machine: probabilities governing transitions
  - sequence of state transitions cannot be determined from sequence of output symbols (i.e. are hidden)



Fundamental questions for HMMs

1. Given a model, how do we efficiently compute the probability $P(O)$ of the observation sequence O ?
2. Given the observation sequence O and a model how do we choose a state sequence that best explains the observations?
3. Given an observation sequence O how do we find the model that maximises the probability $P(O)$ of the observation sequence O ?

Fundamental answers

1. Forward procedure or backward procedure
2. Viterbi algorithm
3. Baum Welch algorithm / forward-backward algorithm (special case of the expectation maximisation-algorithm, or "EM-algorithm")

Statistical language models

- Re-estimate the value of λ_i from relevant documents (relevance feedback)
  - Expectation Maximisation algorithm
  - Estimate different value of λ_i for each term (i.e. different importance of each term.)

Parsimonious models

- Define background models, document models and relevance models in a layered fashion
  1. First define background model
  2. Higher order model(s) should not model language that is well explained by the background model already
  3. Use EM training (we'll see how later on)

How does it work?

- Remember this equation?

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

- In the old days:

$$P(T_i) = \frac{\text{nr. of occurrences in collection}}{\text{size of collection}}$$

$$P(T_i|D) = \frac{\text{nr. of occurrences in document}}{\text{size of document}}$$

How does it work?

- Parsimonious model estimation

$$P(T_i) = \frac{\text{nr. of occurrences in collection}}{\text{size of collection}}$$

$$P(T_i|D) = \text{some random initialisation}$$

Repeat E-step and M-step until $P(T|D)$ does not change significantly anymore

E-step $$e(T) = tf(T,D)\frac{\lambda P_{old}(T|D)}{(1-\lambda)P(T)+\lambda P_{old}(T|D)}$$

M-step $$P_{new}(T|D) = \frac{e(T)}{\sum_T e(T)}$$

How does it work?

- A two-layered model for documents at index time
  1. general model
  2. document model

Train document model

$$P_{index}(T|D) = (1-\lambda)P(T) + \lambda P(T|D)$$

Fix parameter λ

Fix background

How does it work?

- A two-layered model for queries at search time
  1. general model
  2. relevance model

Train relevance model

$$P_{search}(T|R) = (1-\lambda)P(T) + \lambda P(T|R)$$

Fix parameter λ

Fix background

How does it work?

- A three-layered model for known relevant documents
  1. general model
  2. relevance model
  3. document model

Train relevance model and document model

$$P_{rel}(T|D) = (1-\lambda-\mu)P(T) + \mu P(T|R) + \lambda P(T|D)$$

Fix parameters

Fix background

Only use relevance model

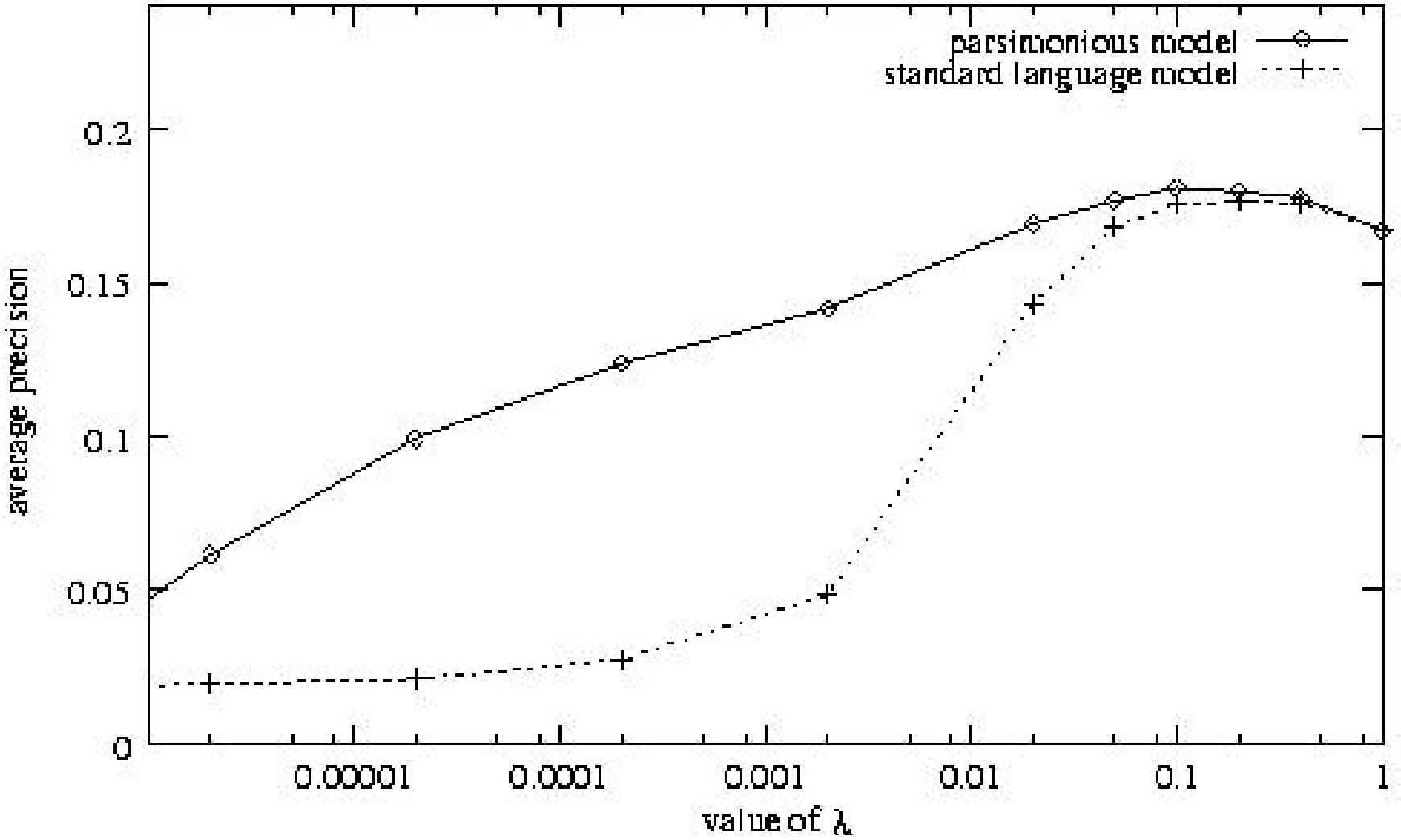
How to use a relevance model?

- Measure cross-entropy between relevance model and document model

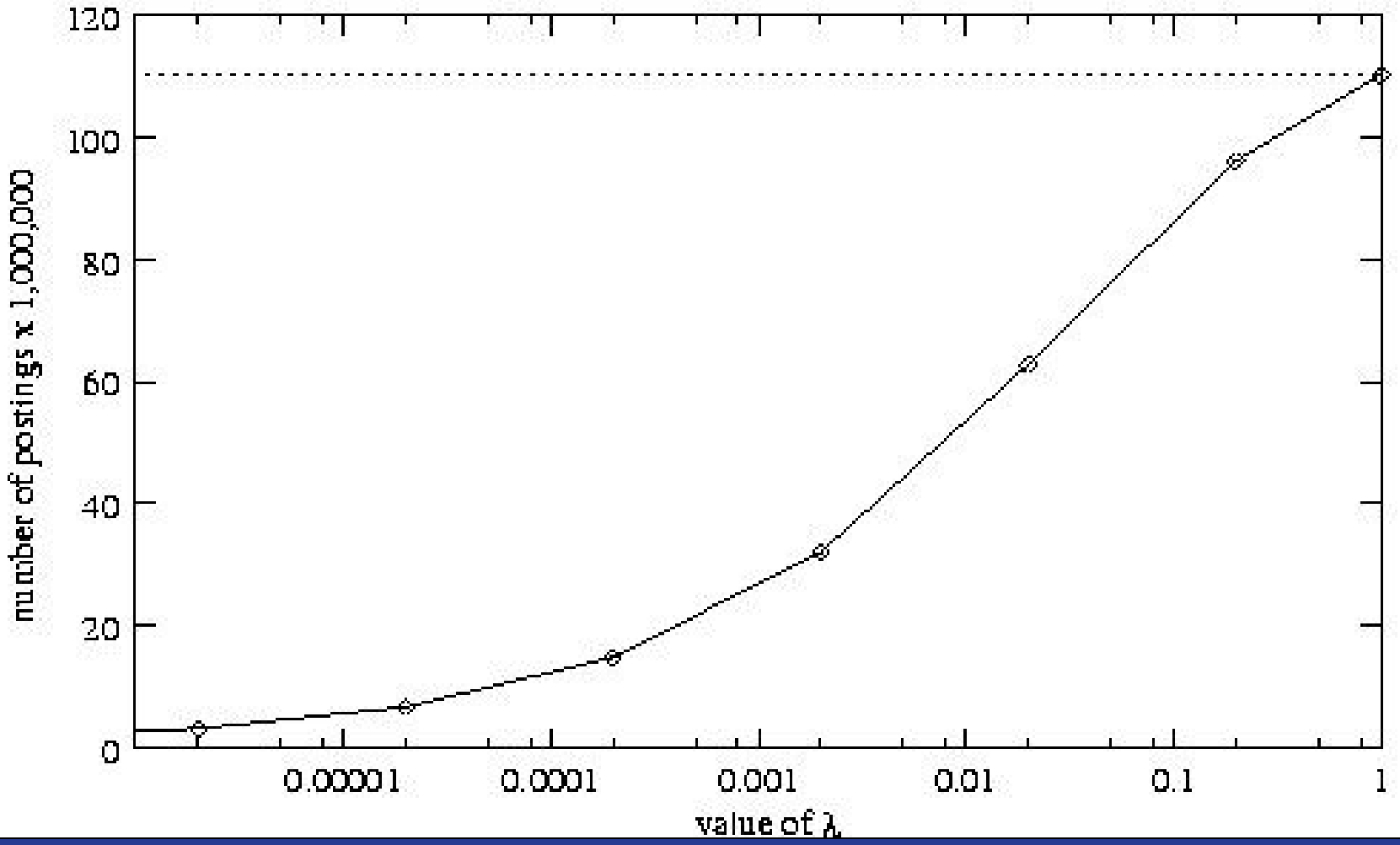
$$H(R,D) = -\sum_{T} P(T|R)\log((1-\lambda)P(T)+\lambda P(T|D))$$

only terms with non-zero $P(T|R)$ contribute to sum

So, what happens?



How much are we throwing away?



“amazon rain forest” again

$\lambda = 0.0000001$

word	*probability*
amazon	0.3367
rain	0.3365
forest	0.2896
ban	0.0370
brazil	0.0002

Serdyukov's expert model

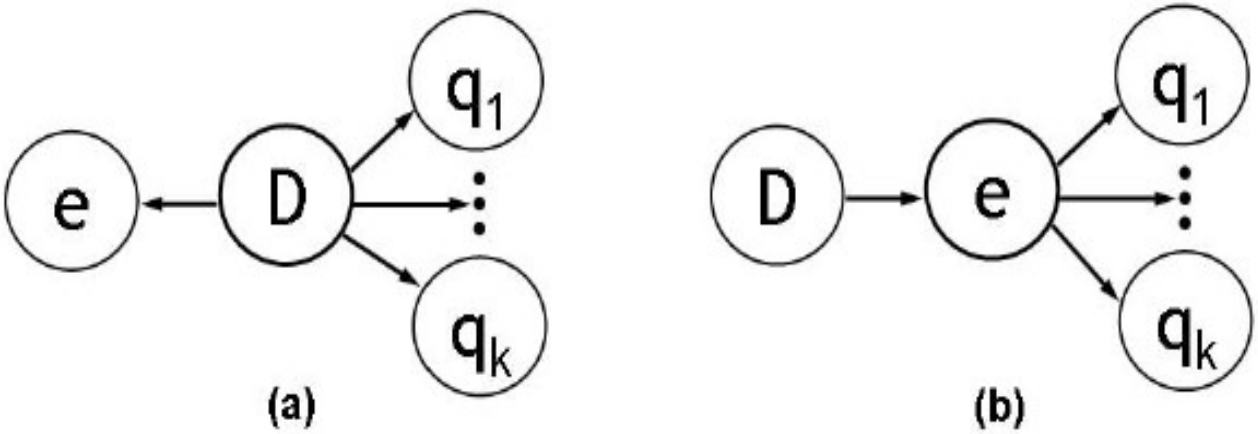
- Use an email archive to search for experts
- Experts both send and receive email on the topic they know well
- Each email is a mixture of the language models of each potential expert
  - i.e. because of in-line quotations

$$P_{rel}(T|D) = \sum_{e \in D} P(T|E=e)\,P(E=e|D)$$

Train expert models

Fix parameters

Serdyukov's expert model



Balog's model **Serdyukov's model**

Fig. 1. Dependence networks for two methods of estimating $P(e, q_1, ..., q_k)$

EM-training for expert search

Method	MAP	MRR	R-prec	P5	P10	P20
Method 1	0.1587	0.6550	0.2598	0.4285	0.4122	0.3341
Method 2	0.1712	0.6712	0.2755	0.4306	0.4304	0.3653

Table 1: Performance of expert ranking methods

- (Serdyukov and Hiemstra 2008)
  (table contains results from earlier experiments)

Probabilistic Latent Semantic Indexing

- Each document is a mixture of a number of latent models (or topics)
- We do not know what document discusses what topics

$$P_{rel}(T|D) = \sum_{m} P(T|M=m)\,P(M=m|D)$$

Only fix the number of models

Train latent models

Train model weights

Probabilistic Latent Semantic Indexing

- Related to Singular Value Decomposition
- Problems with over-training

  (Hofmann 1999)

Approach towards entity ranking

1. off-line preparation: index corpus with entity tagging. use NLP techniques to recognize entities if the are not tagged.
2. on-line, query dependent: building of an entity containment graph from top ranked retrieved documents
3. relevance propagation within the graph and output entities of interest in order of their relevance.

NLP tagging

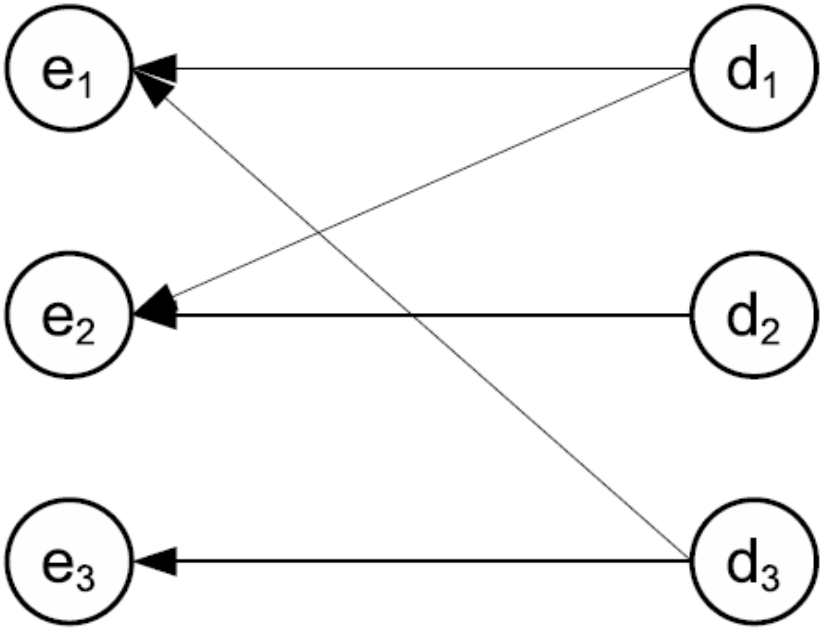
- XML fragment

<entry> Jorge Castillo (artist) Castillo greatly admired Pablo Picasso, and that influence shows his paintings, etchings, and lithographs ...

- tagged fragment

<entry><s><enamex.person>Jorge Castillo</enamex.person> <O.PUNC>(</O.PUNC> <O.NN>artist</O.NN><O.PUNC>) </O.PUNC> </s><s><enamex.person>Castillo</enamex.person> <O.RB>greatly</O.RB> <O.VBD>admired</O.VBD> <enamex.person>Pablo Picasso</enamex.person><O.PUNC>, </O.PUNC> <O.CC>and</O.CC><O.DT>that</O.DT> <O.NN>influence</O.NN> <O.VBZ>shows</O.VBZ><O.IN >in</

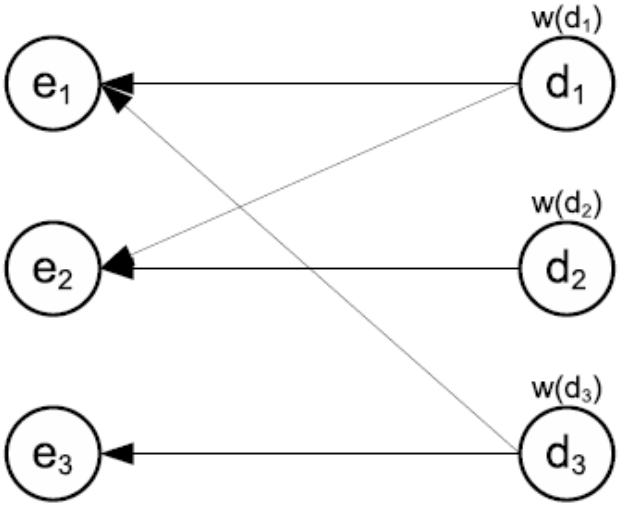
Including Further Entity Types



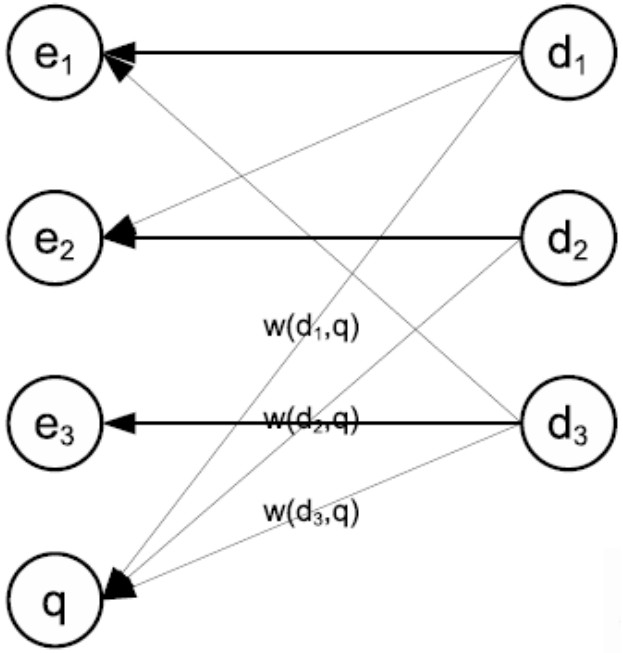
- We model with entity containment graphs the relationship between entities and documents.
- Documents and Entities are represented as vertices.
- Edges symbolize the containment relation.

Modelling query-dependent scores

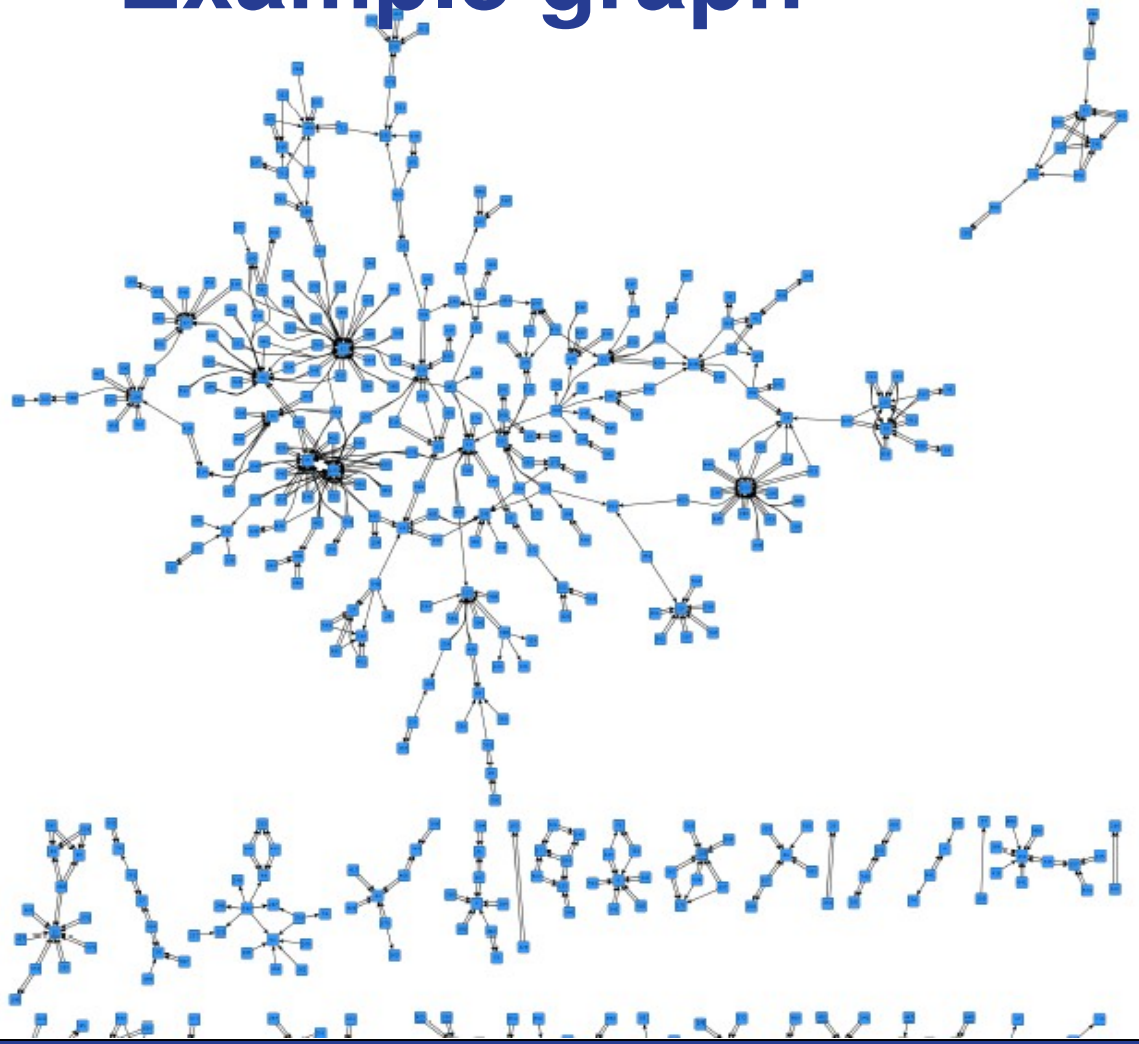
- Model 1: vertex weights



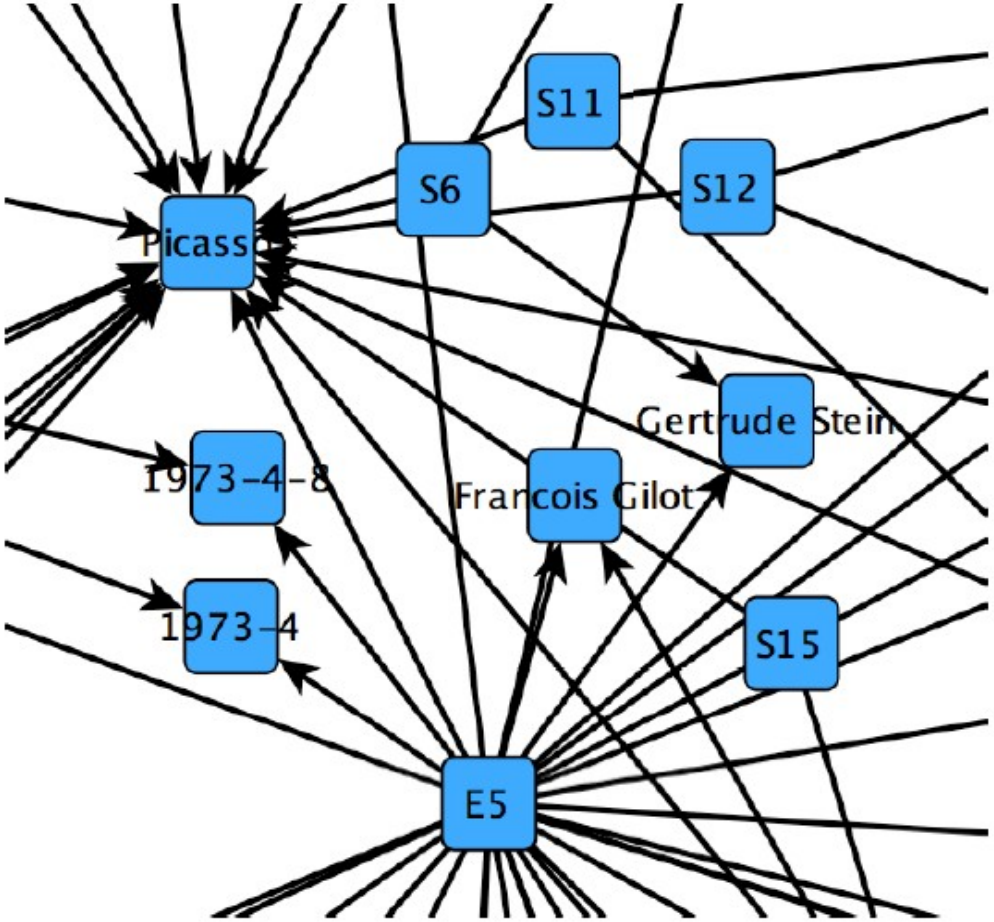
- Model 2: additional query node and edge weights



Example graph



Entity identity



- identity check: Is Gilot the same person as Francois Gilot?
- precision: How do we model the occurrence of April 8, 1973 and 1973?

Probabilistic random walk

- The mutually recursive definition describes a walk over the different type of edges in the graph: query–doc, doc–doc, doc–ent, ent–ent.

Probabilistic Random Walk

$$P(e) = \lambda_1 \sum_d P(e|d)P(d) + \lambda_2 \sum_{e'} P(e|e')P(e')$$

$$P(d) = \lambda_0 P(d|q) + \lambda_1 \sum_e P(d|e)P(e) + \lambda_2 \sum_{d'} P(d|d')P(d')$$

Experimental Results

Performance overview of the relevance propagation models:

Model	*unweighted*	*weighted*
MAX		0.352
IDG	0.342	0.371
HITS	0.343	0.376
PRW	0.340	0.386

(Rode et al. 2007)

Advanced models conclusion

- Relevance models: query expansion using initial ranked list
- Expectation Maximization Training: estimate the probability of unseen events
- Random walks: find most central entity/document

References

- Krisztian Balog, Leif Azzopardi, and Maarten de Rijke. Formal models for expert nding in enterprise corpora. *Proceedings of SIGIR 2006*
- Djoerd Hiemstra, Stephen Robertson and Hugo Zaragoza. Parsimonious Language Models for Information Retrieval", In *Proceedings of SIGIR 2004*
- Thomas Hofmann, *Probabilistic latent semantic indexing, Proceedings of SIGIR 1999*
- Victor Lavrenko and Bruce Croft. Relevance based language models. *Proceedings of SIGIR 2001*
- Henning Rode, Pavel Serdyukov, Djoerd Hiemstra, and Hugo Zaragoza, “Entity Ranking on Graphs: Studies on Expert Finding”, Technical Report 07-81, CTIT, 2007
- Pavel Serdyukov and Djoerd Hiemstra, Modeling documents as mixtures of persons for expert finding, In *Proceedings of ECIR* 2008

Acknowledgments

- Some slides were kindly provided by:
  - Pavel Serdyukov
  - Henning Rode