



IN4325

Query refinement

Felipe Moraes (WIS, TU Delft)

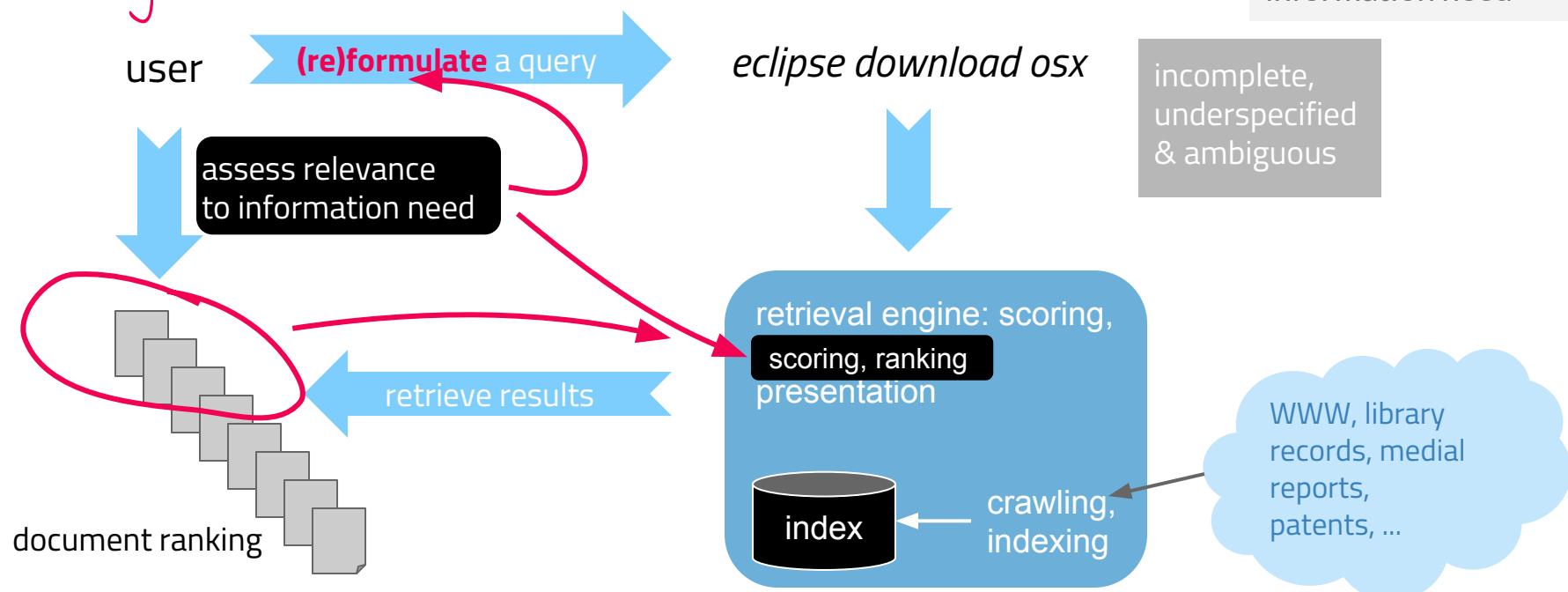
(slides: C. Hauff)

The big picture

The essence of IR

Information need: Looks like I need Eclipse for this job. Where can I download the latest beta version for macOS Sierra?

today's focus



Information need

Topic the user wants to know more about

Query

Translation of need into an input for the search engine

Relevance

A document is relevant if it (partially) provides answers to the information need

Information needs

Different categorizations exist:

- **Informational** vs. **transactional** vs. **navigational**
- Number of relevant documents wanted
- Tasks underlying the information need

Belkin's **Anomalous State of Knowledge**: users do not always know what exactly their information need is

Thus:

today's web search queries are 2-3 terms long

- Queries can represent different information needs (short, ambiguous, imprecise)
- A query may be a poor representation of the underlying information need

Query refinement techniques



Query refinement either automatically or through user interactions

- Query expansion
- (Pseudo-) relevance feedback
- Spelling correction
- Query autocompletion
- Query suggestions

Goal: produce a query that is a **better representation** of the information need (this in turn *should* lead to a better set of retrieved documents)

Query expansion



Semantic gap

Query expansion

Idea: instead of a user manually adding synonyms to her query, let the system help (automatically or semi-automatically) in order to decrease the **semantic gap**.

Global approaches (independent of the query)

- Query expansion with a domain-specific thesaurus (indexing vocabulary and simple relations) is common and successful for domain-specific corpora
- A generic “thesaurus” such as WordNet has not shown to be effective

Local approaches (relative to the retrieved documents)

- Relevance feedback
- Pseudo-relevance feedback
- Implicit feedback

Query expansion

Idea: instead of a query, let the system do a semi-automatic expansion

Global approach

- Query expansion based on vocabulary from domain-specific knowledge bases
- A generic "term" is expanded by a set of terms

Local approach

- Relevance feedback
- Pseudo-relevance feedback
- Implicit feedback

Alzheimer Disease MeSH Descriptor Data 2018				
	Details	Qualifiers	MeSH Tree Structures	Concepts
MeSH Heading	Alzheimer Disease			
Tree Number(s)	C10.228.140.380.100 C10.574.945.249 F03.615.400.100			
Unique ID	D000544			
Scope Note	A degenerative disease of the BRAIN characterized by the insidious onset of DEMENTIA . Impairment of MEMORY , judgment, attention span, and problem solving skills are followed by severe APRAXIAS and a global loss of cognitive abilities. The condition primarily occurs after age 60, and is marked pathologically by severe cortical atrophy and the triad of SENILE PLAQUES ; NEUROFIBRILLARY TANGLES ; and NEUROPIL THREADS . (From Adams et al., Principles of Neurology, 6th ed, pp1049-57)			
Entry Version	ALZHEIMER DIS			
Entry Term(s)	Acute Confusional Senile Dementia Alzheimer Dementia Alzheimer Disease, Early Onset Alzheimer Disease, Late Onset Alzheimer Sclerosis Alzheimer Syndrome Alzheimer Type Senile Dementia Alzheimer's Disease Alzheimer's Disease, Focal Onset Alzheimer-Type Dementia (ATD) Dementia, Alzheimer Type			

Query expansion

Idea: instead of a query, let the system do a semi-automatic search

Alzheimer Disease MeSH Descriptor Data 2018

Details

Qualifiers

MeSH Tree Structures

Concepts

WordNet Search - 3.1

- [WordNet home page](#) - [Glossary](#) - [Help](#)

Word to search for:

Display Options:

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Display options for sense: (gloss) "an example sentence"

Noun

- [S: \(n\) tank, army tank, armored combat vehicle, armoured combat vehicle](#)
(an enclosed armored military vehicle; has a cannon and moves on caterpillar treads)
- [S: \(n\) tank, storage tank](#) (a large (usually metallic) vessel for holding gases or liquids)
- [S: \(n\) tank, tankful](#) (as much as a tank will hold)
- [S: \(n\) tank car, tank](#) (a freight car that transports liquids or gases in bulk)
- [S: \(n\) cooler, tank](#) (a cell for violent prisoners)

Local approach

- Relevance feedback
- Pseudo-relevance feedback
- Implicit feedback

Query expansion

Idea: instead of a simple query, let the user provide a semi-automatic query

Global approach

- Query expansion based on vocabulary
- Domain-specific
- A generic approach

Local approach

- Relevance feedback
- Pseudo-relevance feedback
- Implicit feedback

Bing search results for "tank":

All Images Videos Maps News | My saves

18.900.000 Results Date Only English Region

Videos of tank

bing.com/videos

3:51 HD
Tank - Next Breath [Music Video]
YouTube · 28-2-2012 · 7M+ views

5:01 HD
Tank - Better For You [Official Video]
YouTube · 23-12-2015 · 2M+

4:55 HD
Tank - I Can't Make You Love Me [Official Music Video]
YouTube · 15-12-2010 · 51M+

See more videos of tank

World of Tanks | Epic Online Tank Game | Play for Free

https://worldoftanks.com

Furious 15-vs-15 Battles on Legendary Tanks, Over 500 War Vehicles are Ready to Roll Out. Join Multiplayer Tank Game with 150 Million Players Worldwide!

Tank (1984) - IMDb

www.imdb.com/title/tt0088224

Jaegers, assassins, and superheroes await you in our Winter Movie Guide. Plan your season and take note of the hotly anticipated indie, foreign, and documentary ...

5,6/10 ★★★★☆ (3,4K)
Category: Comedy
Cast: James Garner/Shirley Jones/C. Thomas... Content Rating: PG

TANK ARCHITECTURE AND INTERIOR DESIGN

tank.nl

TANK is an international design studio for architecture, interior design and branding. TANK realises bespoke and outstanding projects.

Images of tank

bing.com/images

See more images of tank

Related searches

- tank spellen
- tank wikipedia
- septic tank plaatsen
- septic tank verwijderen kosten
- sherman tank
- tanki tank
- speeleiland tanks
- war of tanks

WordNet has been tried and tested



Are the glosses useful for anything?

Ambiguous query: "Pluto"

20+% of Web search queries are single term queries. Very common in site search too!

Idea: give the user a choice between possible hypernyms if we do not have a query log

1. Look up WordNet synsets and **glosses**
2. Process the **glosses** (POS tagger, etc.) and keep the nouns as potential hypernyms
3. Apply **Hearst patterns** and retrieve the number of result pages per candidate
4. Consider the candidate with the highest score as hypernym

Noun

- [S: \(n\) Pluto](#) (a cartoon character created by Walt Disney)
- [S: \(n\) Pluto](#), [Dis](#), [Dis Pater](#), [Orcus](#) ((Roman mythology) god of the underworld; counterpart of Greek Hades)
- [S: \(n\) Pluto](#) (a large asteroid that was once thought to be the farthest known planet from the sun; it has an elliptical orbit) "*Pluto was discovered by Clyde Tombaugh in 1930*"



Bootstrapping how?

simple patterns to decide on taxonomic relations

- NP_0 such as $\{NP_1, NP_2, \dots, (and|or)\} NP_n$
 - "American cars such as" → Chevrolet, Pontiac
- Such NP as $\{NP, \}^* \{(or|and)\} NP$
 - "such colors as red or orange"
- $NP \{, NP\}^* \{, \}$ or other NP
- $NP \{, NP\}^* \{, \}$ and other NP
- $NP \{, \}$ including $\{NP,\}^* \{or|and\} NP$
- $NP \{, \}$ especially $\{NP,\}^* \{or|and\} NP$

WordNet has been tried and tested

1. WordNet glosses for query term “Pluto”

- **SYN1** a small planet and the farthes known planet from the sun; has the most elliptical orbit of all the planets
- **SYN2** (Greek mythology) the god of the underworld in ancient mythology; brother of Zeus and husband of Persephone
- **SYN3** a cartoon character created by Walt Disney

2. Candidate nouns

- **SYN1** planet, sun, orbit, planets
- **SYN2** Greek, god, underworld, mythology, brother, Zeus, husband, Persephone
- **SYN3** cartoon, character, Walt, Disney

3. Hearst patterns and page counts (shown for SYN1 only)

- “Pluto is a planet” (1550), “Pluto is planet” (145) “Pluto is a sun” (2), “Pluto is sun” (0)
- “Pluto is an orbit” (1), “Pluto is orbit” (1) “Pluto is a planets” (0), “Pluto is planets” (0)

4. Refinement offers: “Pluto planet”, “Pluto god”, “Pluto cartoon”

Pseudo-relevance feedback

PRF: we assume the top-ranked documents are relevant
RF: user indicates which top-ranked documents are relevant

information need

topic

query

search session



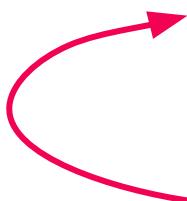
Overview

Search session
A set of searches conducted by a user to solve a (complex) search task within a limited amount of time.

Approach:

- User issues a short query
- System returns an initial set/ranked list of results
- **User marks some of the results relevant or non-relevant**
- System computes a **better representation** of the information need based on the feedback
- System displays revised set/ranked list of results

loop



training data of a search session

machine learning with very limited data

Insight: it is difficult to formulate a good query based on a complex information need, but it is relatively easy to **decide** whether the returned documents match the information need

(P)RF implementation is **retrieval model dependent**;
Strategy: words that occur more frequently in relevant than non-relevant documents are added to the query or increased in weight.

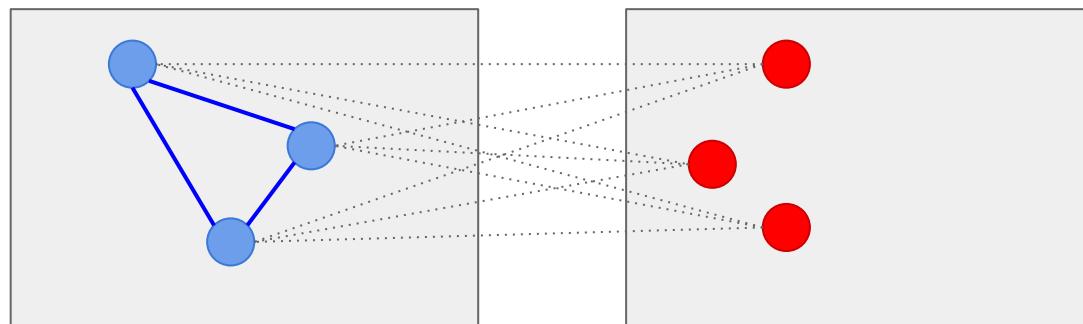
Relevance feedback builds on the cluster hypothesis

"Closely associated documents tend to be relevant to the same requests." (Keith van Rijsbergen, 1970s)



Common assumption of IR systems: relevant documents are more similar to each other than to non-relevant documents

association
between all
document pairs
(R-R), (R-NR)



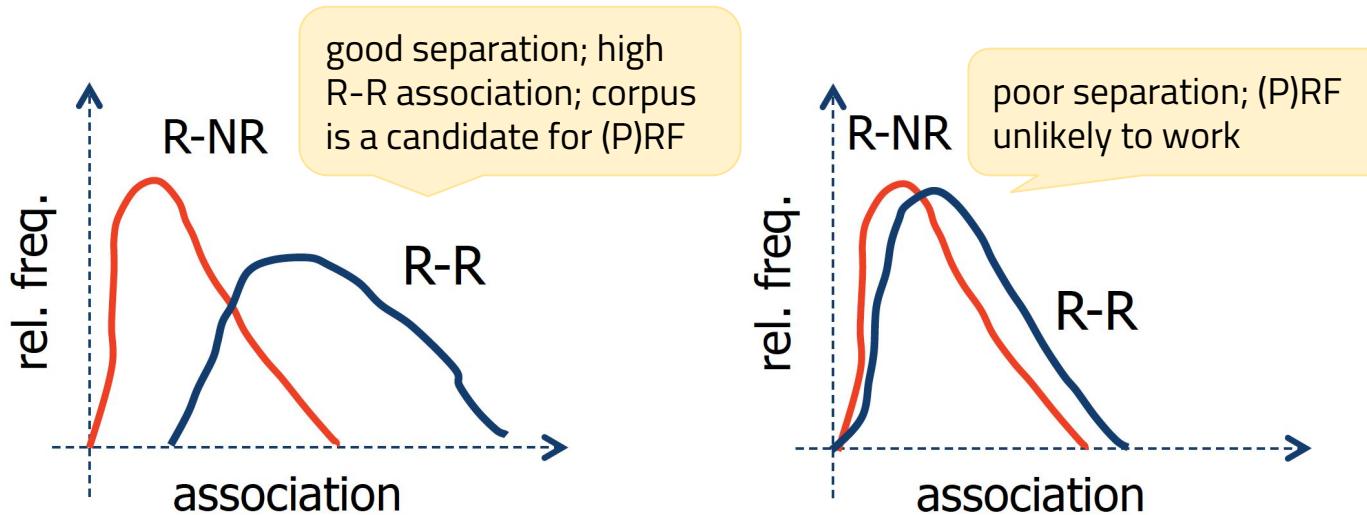
relevant documents (R)

non-relevant documents (NR)

Relevance feedback builds on the cluster hypothesis

"Closely associated documents tend to be relevant to the same requests." (Keith van Rijsbergen, 1970s)

Plot the relative frequency (binned) against the strength of association (usually cosine similarity)



Relevance feedback builds on the cluster hypothesis

Clustering methods should:

- Produce a **stable clustering**: no sudden changes when items are added/removed
- Be **tolerant to errors**: small errors should lead to small changes in clustering

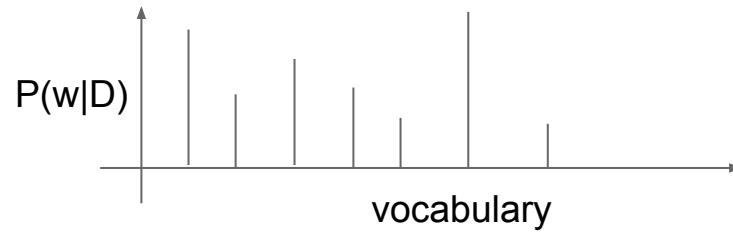
Clustering fails when:

- Subset of documents have very different important terms (semantics to the rescue!)
- Queries are inherently disjunctive
- Polysemy occurs

Pseudo-relevance feedback in language models

Language models

- **Unigram language model:** probability distribution over the words (*the vocabulary*) in a language (*the collection or document*)
- In IR, unigram LMs represent the *topical content*



- A LM representation of a document can be used to *generate new text* by sampling terms from the distribution (the text won't have a syntactic structure, but that's fine)

Language models

Smoothing

General idea: discount probabilities of **seen words**, assign extra probability mass to **unseen words** with a fallback model (the *collection language model*)

$$P(w | D) = \begin{cases} P_{smoothed}(w | D) & \text{if word } w \text{ is seen} \\ \alpha_d P(w | \mathbb{C}) & \text{otherwise} \end{cases}$$

Jelineck-Mercer (JM) smoothing: linear interpolation (amount of smoothing controlled) between ML and collection LM

$$P_\lambda(w | D) = (1 - \lambda)P_{ml}(w | D) + \lambda P(w | \mathbb{C}), \quad \lambda \in (0, 1)$$

Language models

Smoothing

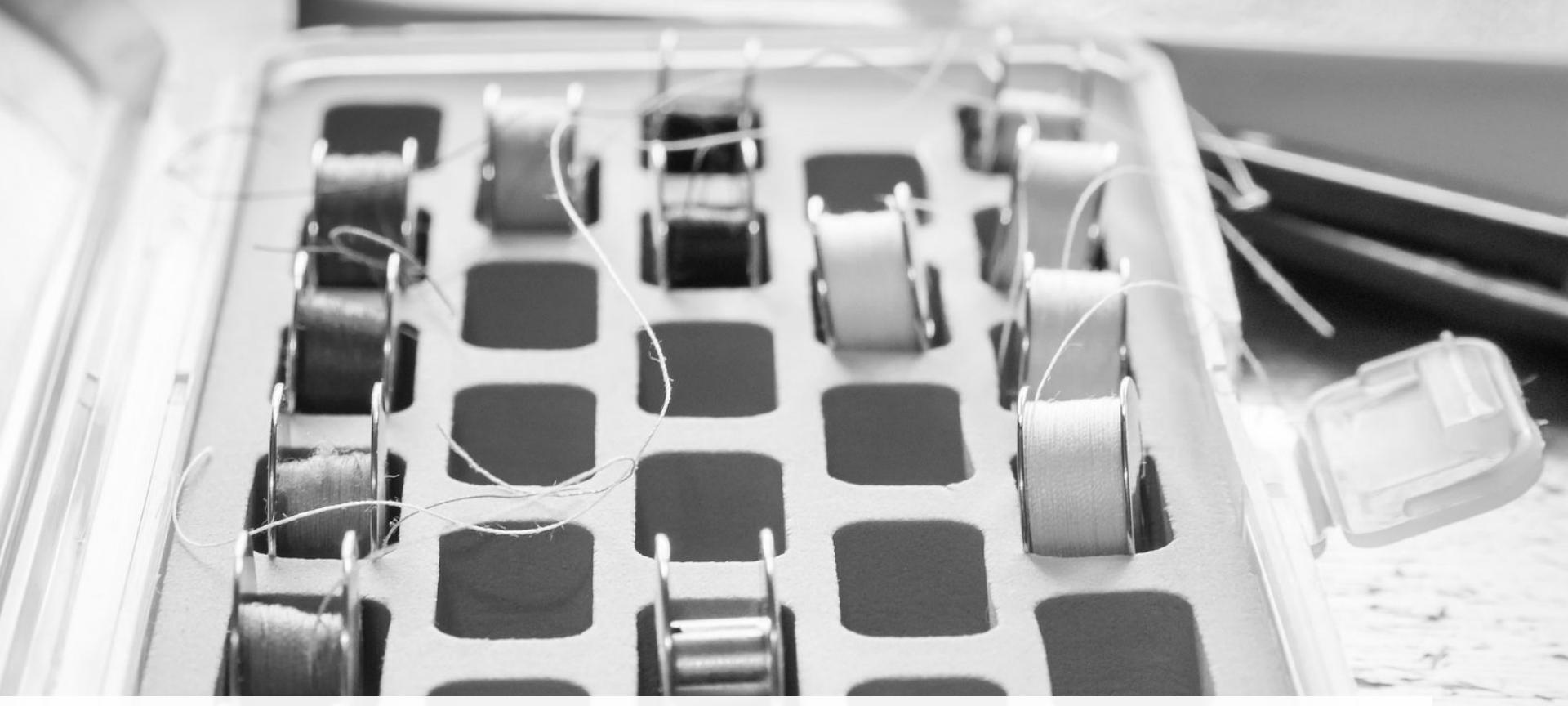
General idea: discount probabilities of **seen words**, assign extra probability mass to **unseen words** with a fallback model (the *collection language model*)

$$P(w|D) = \begin{cases} P_{smoothed}(w|D) & \text{if word } w \text{ is seen} \\ \alpha_d P(w|\mathbb{C}) & \text{otherwise} \end{cases}$$

Dirichlet smoothing: longer documents receive less smoothing

$$P_\mu(w|D) = \frac{c(w; D) + \mu P(w|\mathbb{C})}{\sum_w c(w; D) + \mu}, \text{ usually } \mu > 100$$

↑ "count" of term w in D



Model generalization: create a model/framework that contains existing models as special cases

Relevance models

- Query is a **fixed sample** in LM, with documents being ranked according to their prob. of generating the sample
- Relevance feedback does not come naturally to LM
 - BIM: adjust the weights of the relevance set
 - VSM: Rocchio
- Idea: instead of a fixed sample, consider the query to be a language model (the **relevance model**); it represents the topic covered by relevant documents
 - RF is now principled!

query text now a very small sample generated from the relevance model;
relevant documents are larger samples from the same model

Relevance models

Two options to use our relevance model R

- Option 1: Rank documents by $P(D|R)$
- Option 2: Rank documents according to their *similarity* between the document LM and the query (relevance) LM

not symmetric

Difficult for diverse (wrt. length, vocabulary) sets of documents.

Kullback-Leibler divergence ("KL divergence") measures the difference between two probability distributions P and Q:

$$KL(P||Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

"true distribution";
usually R

always positive (larger the more apart two distributions are); we use the **negative KL divergence** to rank

Relevance models

$$\begin{aligned} -KL(P||Q) &= -\sum_x P(x) \log \frac{P(x)}{Q(x)} \\ &= \sum_{w \in V} P(w|R) \log P(w|D) - \sum_{w \in V} P(w|R) \log P(w|R) \\ &= \sum_{w \in V} \frac{f_{w,Q}}{|Q|} \log P(w|D) \end{aligned}$$

We can ignore terms not in Q.

Maximum likelihood estimate of $P(w|R)$

$\log \frac{M}{N} = \log M - \log N$

relevance model

independent of the document, ignore for ranking purposes

Isn't this rank equivalent to query likelihood? Yes!

However: we have a **more general model**, we can estimate the relevance model in many ways!

Relevance models

$$= \sum_{w \in V} P(w|R) \log P(w|D) - \sum_{w \in V} P(w|R) \log P(w|R)$$

$$P(w|R) \approx P(w|q_1, q_2, \dots, q_n)$$

$$P(w|R) \approx \frac{P(w, q_1, q_2, \dots, q_n)}{P(q_1, q_2, \dots, q_n)}$$

if query terms are samples from the relevance model, an unseen word's probability should depend on the query terms

$$P(w, q_1, q_2, \dots, q_n) = \sum_{D \in \mathcal{C}} p(D) P(w, q_1, q_2, \dots, q_n | D)$$

set of language models

term independence assumption

$$P(w, q_1, q_2, \dots, q_n | D) = P(w|D) \prod_{i=1}^n P(q_i|D)$$

Relevance models

$$= \sum_{w \in V} P(w|R) \log P(w|D) - \sum_{w \in V} P(w|R) \log P(w|R)$$

$$P(w|R) \approx P(w|q_1, q_2, \dots, q_n)$$

$$P(w|R) \approx \frac{P(w, q_1, q_2, \dots, q_n)}{P(q_1, q_2, \dots, q_n)}$$

if query terms are samples from the relevance model, an unseen word's probability should depend on the query terms

$$P(w, q_1, q_2, \dots, q_n) = \sum_{D \in \mathcal{C}} \underline{P(D)} P(w|D) \prod_{i=1}^n P(q_i|D)$$

Requires **two passes for ranking**:

1. Rank documents using query likelihood to obtain the weights needed
2. Use KL-divergence to rank documents by comparing the relevance model and document model

prior probability
of a document

query likelihood score
of a document

i.e. pseudo-relevance feedback (formally in LM)

Relevance models

In the literature often referred to as RM1, RM2, **RM3**, RM4.

Once more ...

1. Rank documents using the query likelihood score for query Q .
2. Select some number of the top-ranked documents to be the set \mathcal{C} .
3. Calculate the relevance model probabilities $P(w|R)$ using the estimate for $P(w, q_1 \dots q_n)$.
4. Rank documents again using the KL-divergence score:¹³

$$\sum_w P(w|R) \log P(w|D)$$

All vocabulary terms? Just the 10-25 that have the highest probabilities?

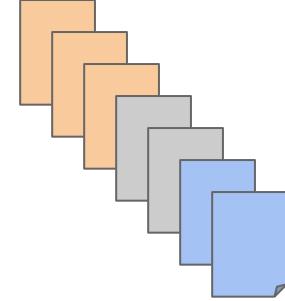
Actually, this model is well motivated but in practice a slight adaptation has turned out to give the best results (=RM3):

Interpolate the relevance model with the original query model to avoid **query drift**.

Query drift
The presence of aspects/topics not related to the query in the top-retrieved documents.

The whole collection? Just the top 10-50 ranked ones?

Relevance models and clustering



Nothing stops us from smoothing the document language model with **document clusters**:



$$\begin{aligned} P(w|D) &= \lambda P_{ML}(w|D) + (1 - \lambda)P(w|Cluster) \\ &= \lambda P_{ML}(w|D) + (1 - \lambda)[\beta P_{ML}(w|Cluster) + (1 - \beta)P_{ML}(w|Coll)] \end{aligned}$$

Many decisions: which clustering algorithm? How many clusters? Clustering (in)dependent of the queries?

Negative relevance feedback

$$S(Q, D) = -D(\theta_Q || \theta_D) = - \sum_{w \in V} p(w|\theta_Q) \log \frac{p(w|\theta_Q)}{p(w|\theta_D)}$$

Another common way of denoting a language model of the **Query** and **Document**

KL-divergence

Negative feedback is easy to integrate into the vector space model (remember Rocchio).

In language modeling, it is less natural to directly modify the relevance model (neg. probabilities are not possible).

Idea: $S(Q, D) = -D(\theta_Q || \theta_D) + \beta \cdot D(\theta_N || \theta_D)$

Create a single negative topic model and penalize the document score if it is similar to it..

Estimating relevance models based on external corpora

Idea: mixture of relevance models drawn from different corpora:

$$P(w|\hat{\theta}_Q) = \sum_{c \in \mathcal{C}} P(c)P(w|\theta_Q, c)$$

External corpus only

Mixture of external corpus and target corpus

	QL	RM3	BIGNEWS		GOV2		WEB	
			EE	MoRM	EE	MoRM	EE	MoRM
trec12	0.2502	0.3201	0.3204	0.3319	0.2709	0.3215	0.3092	0.3324
robust	0.2649	0.3214	0.3501	0.3530	0.2748	0.3207	0.3301	0.3352
wt10g	0.1982	0.2030	0.2256	0.2331	0.1999	0.1958	0.2452	0.2429

Mean average precision

collection	docs	terms
BIGNEWS	6,422,629	2,417,464
GOV2	25,205,179	49,917,419
WEB	19,200,000,000	-

Last words on query expansion

Has not been taken up by Web search engines

- WSEs cannot afford **computationally expensive** AQE techniques (millisecond response times required)
- AQE techniques **perform well on average**, but can cause severe degradation for some queries
- AQE tends to improve **recall** (instead of guaranteeing high precision), often less important for WSE
- **Users** may get confused (their query does not match the returned results)



Last words on query expansion

Common applications of automatic query expansion
(besides document ranking):

- **Question answering**: retrieving passages of documents containing answers to concrete questions, e.g. "When was Barack Obama born"?
- **Multimedia IR**: text-based search over media metadata (annotations, concepts, speech transcripts) as well as multimodal search
- **Information filtering**: monitoring a stream of documents and selecting those that are relevant to a user
- **Cross-language IR**: retrieving documents written in other languages than the query's language

Spell checking

Web search: “Did you mean ...”

Google

studiguid tu delft



Google

tudiguid tu delf



Google

studiguid td delf



Google

studiguid del



All

Images

Maps

Videos

Shopping

More

Settings

Tools

About 201 results (0,35 seconds)

Did you mean:

study guide studieguiden study guides studio mid del

Overview

extensions → extensions (insertion error)
poiner → pointer (deletion error)
marshmellow → marshmallow (substitution error)
brimingham → birmingham (transposition error)
doceration → decoration (2 substitution errors)

10-15% of Web search queries contain spelling errors;
most are single-character errors

Challenges: variety in type and severity of possible
spelling errors in queries (little context available); no
definite lexicon (**Heap's law**)

Generic spell checker:

- Create a spelling dictionary and suggest corrections for any word w not in it
- Suggestions based on similarity between dictionary words and w
 - Levenshtein edit distance
 - Soundex

Assumptions in practice:
- first letter is correct
- correct term has similar length

A hundred years old ...

Soundex

Homophone: word that is pronounced the same way as another word but differs in meaning (e.g. *raise* vs. *rays*)

Soundex is a **phonetic encoding** originally employed for name matching

extenssions → E235
extensions → E235

Use the edit distance
of the soundex codes

1. Keep the first letter (in uppercase).
2. Replace these letters with hyphens: a, e, i, o, u, y, h, w.
3. Replace the other letters by numbers as follows:
 - 1: b, f, p, v
 - 2: c, g, j, k, q, s, x, z
 - 3: d, t
 - 4: l
 - 5: m, n
 - 6: r
4. Delete adjacent repeats of a number.
5. Delete the hyphens.
6. Keep the first three numbers or pad out with zeros.



Picking a spelling correction

A misspelled word can have several possible corrections

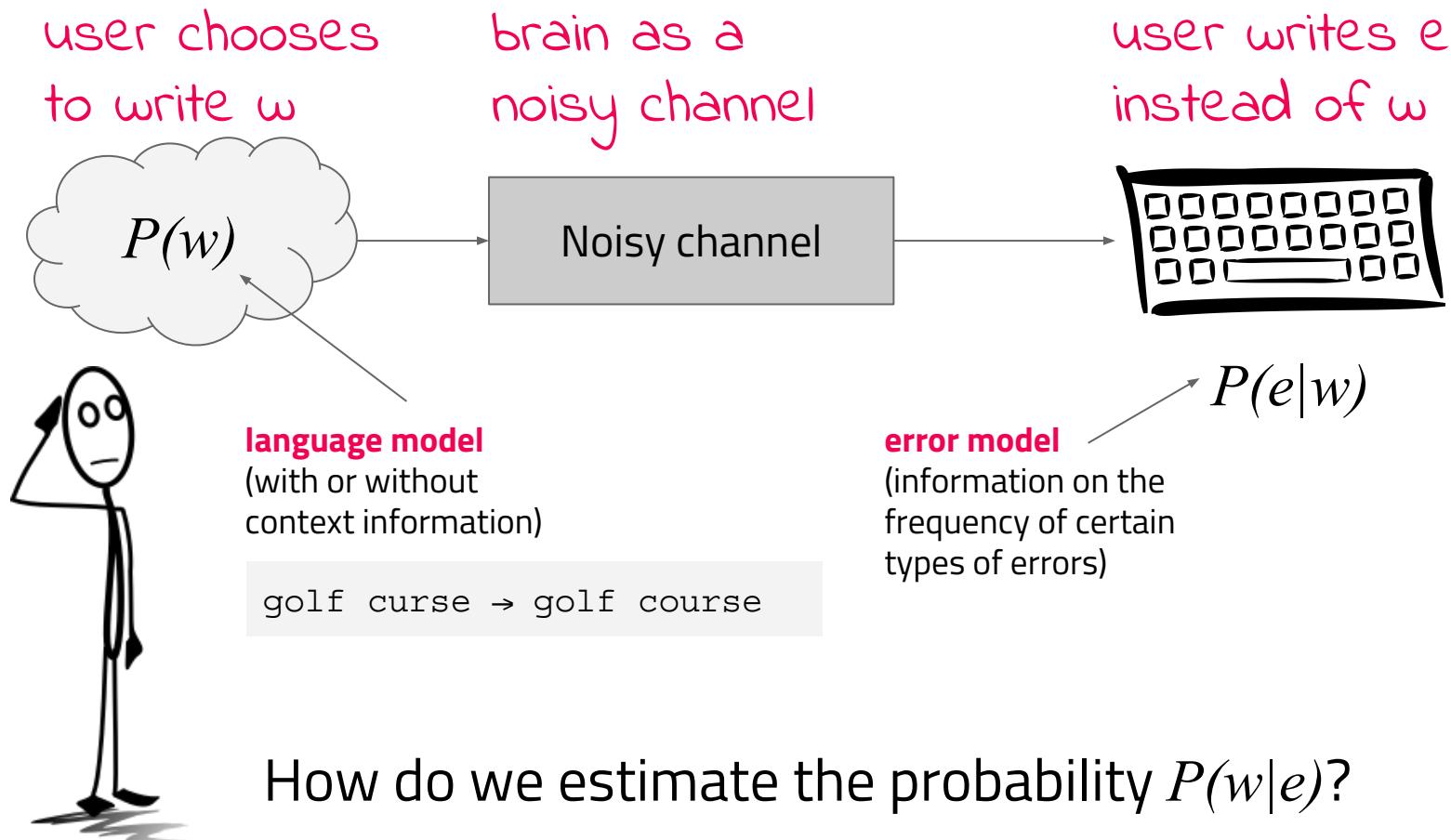
lawers → lowers, lawyers, layers, lasers

trial lawers → trial lowers, trial lawyers, ...

Ranking of spelling corrections:

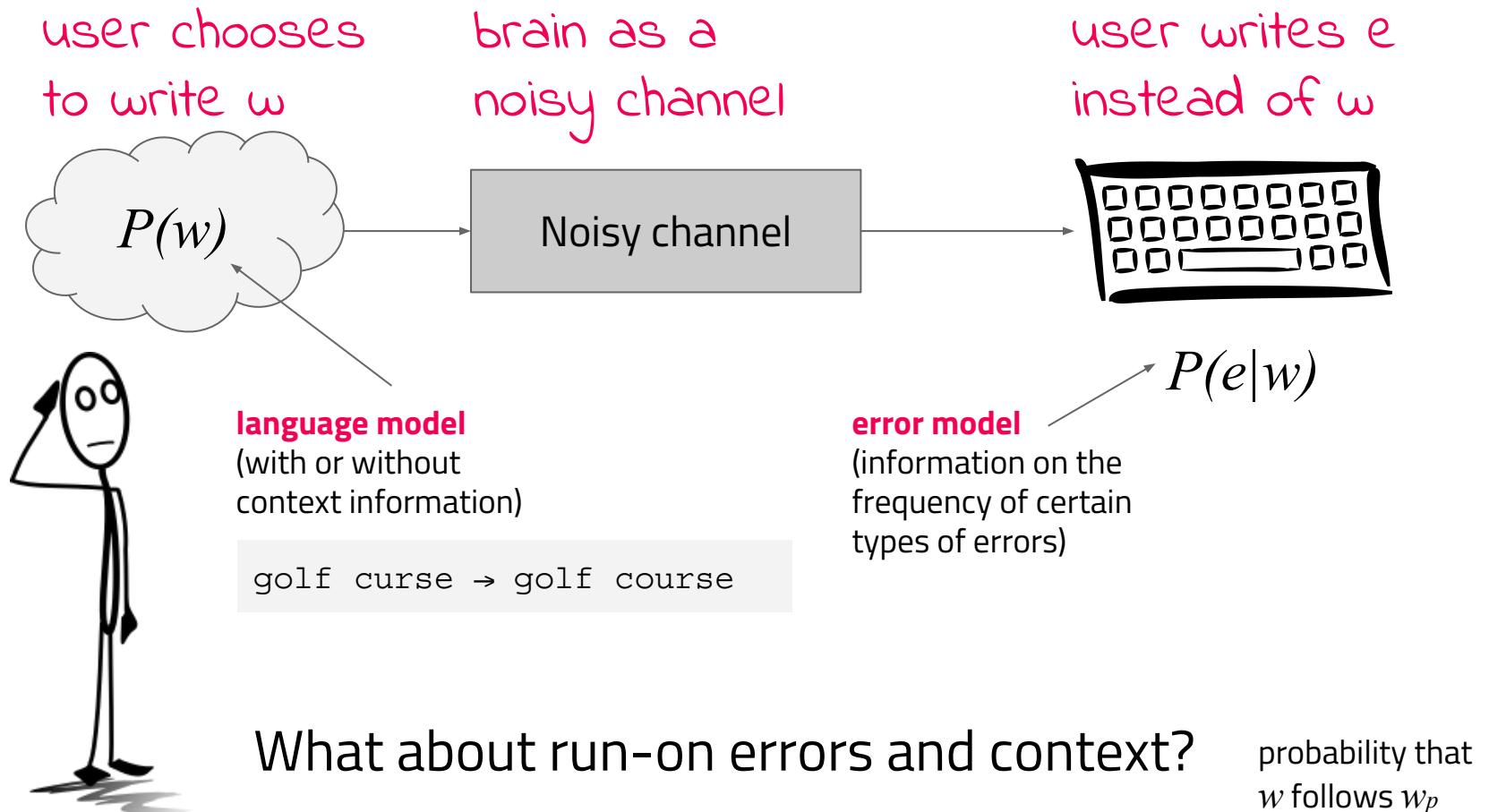
- Use of word frequency of occurrence in the language (**context** independent)
- Use of context and word frequencies leads to better results
- **Run-on errors:** word boundaries skipped or mistyped (whitespace can be treated as character)

Noisy channel model



$$P(w|e) \propto P(e|w) \times P(w)$$

Noisy channel model



What about run-on errors and context?

$$P^{context}(w) = \lambda P(w) + (1 - \lambda)P(w|w_p)$$

probability that w follows w_p



Source of probabilities

$P(w)$

- **Query log** (mostly for Web search), though frequency alone is not enough (e.g. britny spears)
- High-quality **document corpus** (e.g. news corpus)
- **Wikipedia history diffs** (small edits are often corrections)
- Trusted **lexicon**

$P(e|w)$

- Simple: all errors with the same edit distance have the same probability
- Complex: some errors are more likely than others, e.g. based on keyboard layout, source language, phonetics, cognitive misconceptions

Iterative spelling correction based on query logs



Recall or precision: which metric is more important for a Web search query spell checker?

1. Tokenize the query.
2. For each token, a set of alternative words and pairs of words is found using an edit distance modified by weighting certain types of errors, as described earlier. The data structure that is searched for the alternatives contains words and pairs from both the query log and the trusted dictionary.
3. The noisy channel model is then used to select the best correction.
4. The process of looking for alternatives and finding the best correction is repeated until no better correction is found.

Any string appearing in the query log can be a valid correction, even if misspelled.

The correct spellings tend to be more correct than the misspellings. Small mistakes are more common than large mistakes.

albert einstein	4834
albert einstien	525
albert einstine	149
albert einsten	27
albert einsteins	25
albert einstain	11
albert einstain	10
albert eintein	9
albeart einstein	6
aolbert einstein	6
alber einstein	4
albert einseint	3
albert einsteirn	3
albert einsterin	3
albert eintien	3
alberto einstein	3
albrecht einstein	3
alvert einstein	3

Iterative spelling correction based on query logs

Ablation study

remove some feature(s)
and determine the system
effectiveness compared to
the compete setup.

Accuracy

	All queries	Valid	Misspelled
Nr. queries	1044	864	180
Full system	81.8	84.8	67.2
No lexicon	70.3	72.2	61.1
No query log	77.0	82.1	52.8
All edits equal	80.4	83.3	66.1
Unigrams only	54.7	57.4	41.7
1 iteration only	80.9	88.0	47.2
2 iterations only	81.3	84.4	66.7

Query autocomplete

Overview

inf	information	information r	information r
informatique	information	information ratio	information ratio
infomedics	information security officer	information retrieval	information retrieval
influenza	information technology	information radiators	information revolution
infinity	information bias	information risk theory	information risk
infographic	information ratio	information rights management	information rules
inflatie	information planet	information request	information radiators
inflatie 2017	information asset	information resources	information rights management
infinity war	information overload	information risk theory audience	information retrieval python
infacol	informationele positionering	information risk	information retrieval pdf
informatica actie	information icon	information retrieval vu	information retrieval techniques

logged in

Goals:

1. Reduce query entry time
2. Prepare results in advance of query submission
3. Help users formulate a more precise query

Task

Given the current prefix
(=query string the user
has typed in so far),
rank all possible
candidates* (=complete
queries).

Display the top ranked
candidates to the user.

*assume that we have that list available



Two strong baselines



Assumptions:

1. Access to a query log and document clicks
2. Access to a corpus
3. Access to a user's past queries

Most popular ranker

Query candidates are ranked according to their past popularity

Clicked documents ranker

Cosine similarity between a user's profile (previously clicked docs by that user) and the candidate query profile (previously clicked docs across all users for that query)

Task

Approaches	Query-log Evidence	Personalized	MRR				
			2	4	6	8	10
Sentence occurrence ranker (SO)	No	No	0.005▼	0.0456▼	0.0696▼	0.1003▼	0.1546▼
Most Popular ranker (MP)	Yes	No	0.0964	0.2146	0.2851	0.3248	0.3641
Time Ranker (TR)	Yes	No	0.0324▼	0.1236▼	0.1995▼	0.2707▼	0.3281▼
Most Popular Time ranker (MT)	Yes	No	0.0961	0.2249▲	0.3112▲	0.3684▲	0.4153▲
Terms occurrence ranker (TO)	Yes	No	0.0021▼	0.0326▼	0.0773▼	0.1163▼	0.1617▼
Near Words Ranker (NW)	Yes	No	0.0611▼	0.1576▼	0.2347▼	0.2972▼	0.3611
String Similarity Ranker (SS)	No	Yes	0.0137▼	0.0711▼	0.1628▼	0.1149▼	0.2069▼
WordNet Similarity Ranker (WR)	Yes	Yes	0.089▼	0.0302▼	0.0711▼	0.0908▼	0.1055▼
N-Gram Similarity Ranker (NR)	Yes	Yes	0.0837	0.2927▲	0.3693▲	0.4207▲	0.4602▲
Kernel Similarity Ranker (KR)	Yes	Yes	0.907	0.2876▲	0.3356▲	0.3923▲	0.4121▲
Clicked Documents Ranker (CR)	Yes	Yes	0.1442▲	0.2952▲	0.3462▲	0.3938▲	0.4183▲

Mean reciprocal rank

Prefix length

Table 1: Query auto-completion performance over the queries issued during the month of April '13 in our dataset, using the 11 presented ranking approaches. Statistically significant improvements/reductions in performance over the Most Popular ranker (MP) ($p < 0.05$ paired t-test) are denoted ▲ and ▼, respectively.

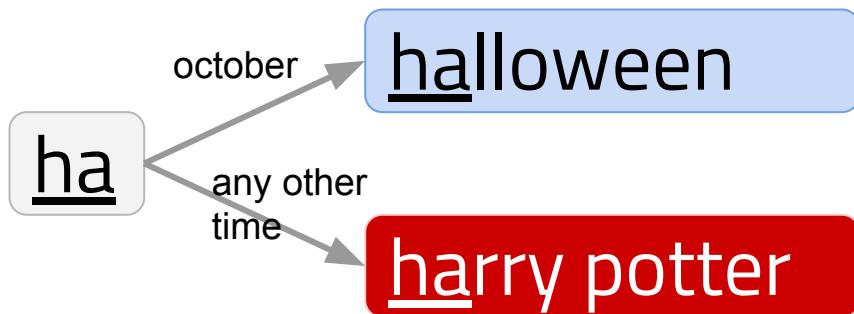
1,417,880 unique queries

November 2010 - March/April 2013

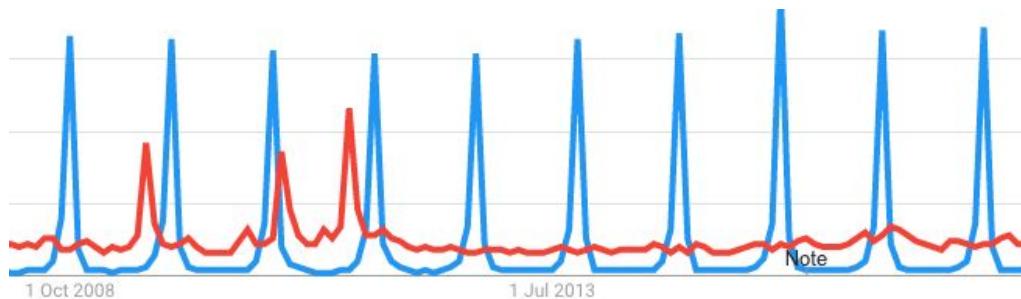
37,806 unique users

Medical search engine with 1.5M articles

Time-sensitive query autocomplete



Approach: apply time-series modeling
and rank candidates according to their
forecasted frequencies



Rare prefixes

Query logs are a good source for *frequent* query prefixes.

The pool of candidate queries is usually drawn from a **pre-built prefix trie** (exact matching).

What happens if that does not yield any query candidates?

Idea: mine popular **query candidate suffixes** (popular n-grams) and generate **synthetic suggestion candidates** (prefix+suffix) that have never been observed in the log

Rare prefixes

Query logs are a good source of rare prefixes.

The pool of candidate prefixes is finite.
pre-built prefix trie (example)

What happens if that doesn't work?
candidates?

Idea: mine popular **queries** (n-grams) and generate **candidates** (prefix+suffix observed in the log)

what to cook with chicken and broccoli and bacon
what to cook with chicken and broccoli *and bacon*
what to cook with chicken and broccoli *and noodles*
what to cook with chicken and broccoli *and brown sugar*
what to cook with chicken and broccoli *and garlic*
what to cook with chicken and broccoli *and orange juice*
what to cook with chicken and broccoli *and beans*
what to cook with chicken and broccoli *and onions*
what to cook with chicken and broccoli *and ham soup* *

cheapest flights from seattle to dc
cheapest flights from seattle *to dc*
cheapest flights from seattle *to washington dc*
cheapest flights from seattle *to bermuda*
cheapest flights from seattle *to bahamas*
cheapest flights from seattle *to aruba*
cheapest flights from seattle *to punta cana*
cheapest flights from seattle *to airport* *
cheapest flights from seattle *to miami*

Rare prefixes: candidates generation

1. For each query in the query log, generate all possible n-grams from the end of the query

amsterdam schiphol airport → airport, schiphol airport, amsterdam schiphol airport

2. Aggregate the n-grams across all queries and keep the most popular ones (precomputed)

AOL query log

3. For a given query prefix, extract the end-term

+most popular ranker candidates

4. Match all suffixes that start with the end-term and create synthetic suggestion candidates

Top suffixes	Top 2-word suffixes	Top 3-word suffixes
com	for sale	federal credit union
org	yahoo com	new york city
net	myspace com	in new york
gov	google com	or no deal
pictures	new york	disney channel com
lyrics	real estate	my space com
edu	of america	in new jersey
sale	high school	homes for sale
games	new jersey	department of corrections
florida	space com	chamber of commerce
for sale	aol com	bath and beyond
us	s com	in las vegas

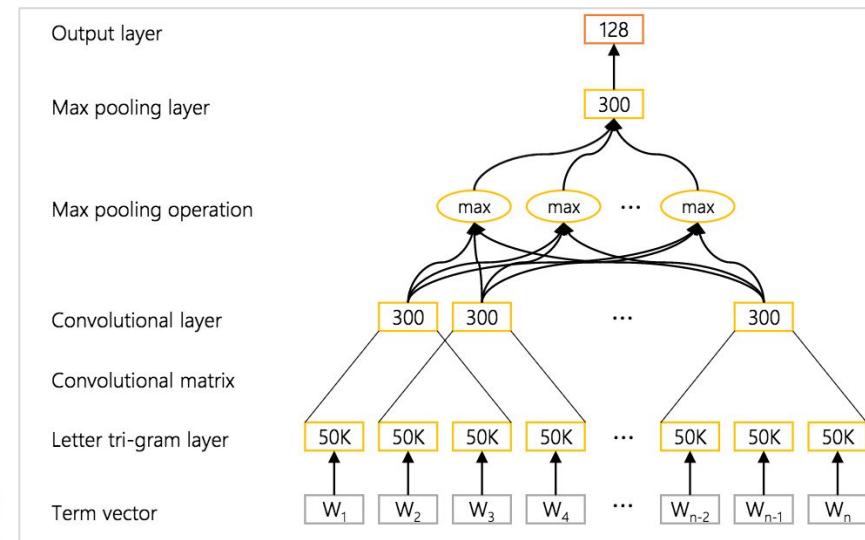
Rare prefixes: ranking features

Supervised ranking model: features are computed for every query prefix and suggestion candidate (synthetic or previously observed)

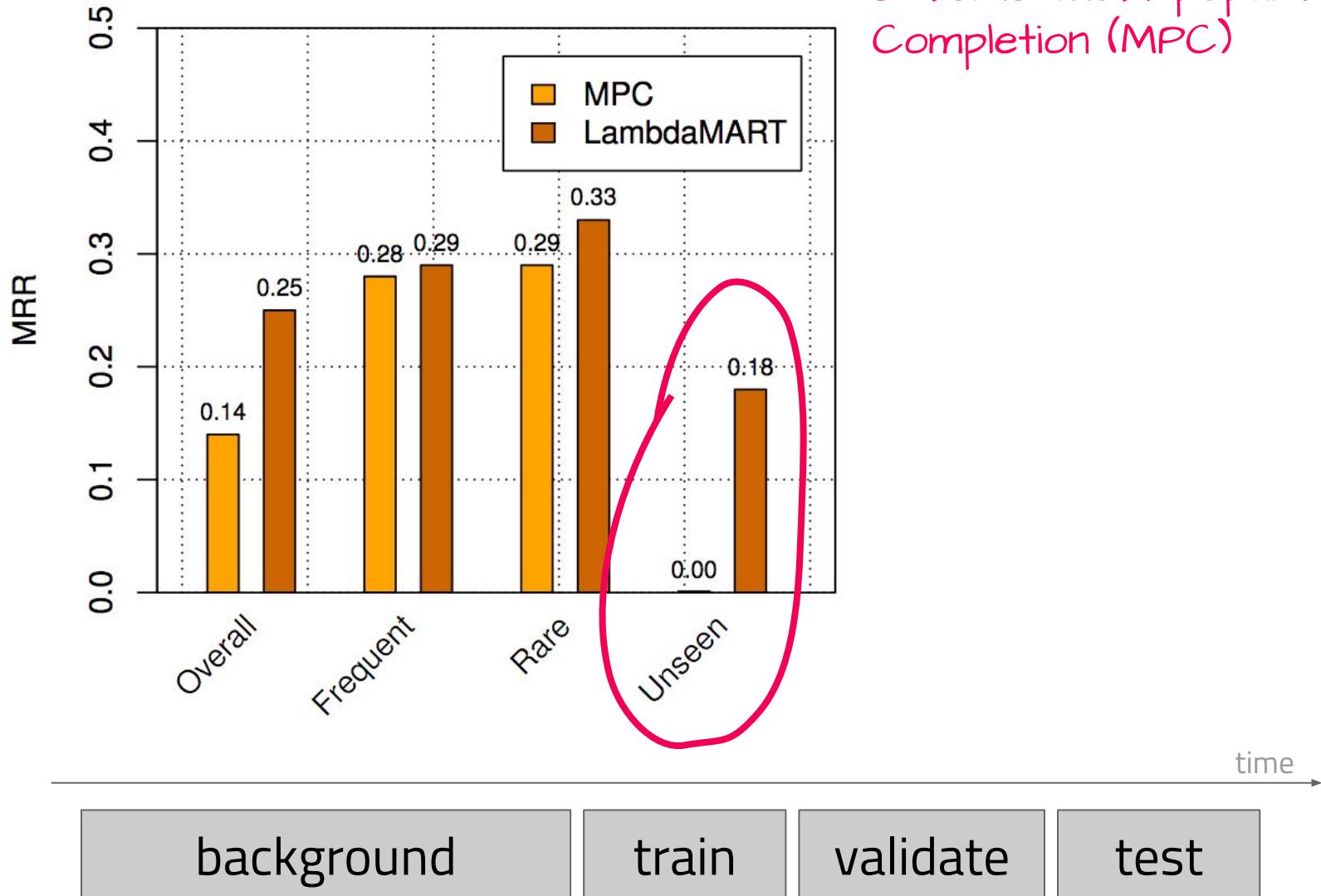
Main features for **LambdaMART**:

- Query log frequencies of N-grams appearing in a candidate suggestion
- **Convolutional latent semantic model**
(training on prefix/suffix pairs generated from sampled queries)

$$clsmsim(\bar{p}, \bar{s}) = \text{cosine}(y_1, y_2) = \frac{y_1^T y_2}{\|y_1\| \|y_2\|}$$



Rare prefixes: results



Rare prefixes: results

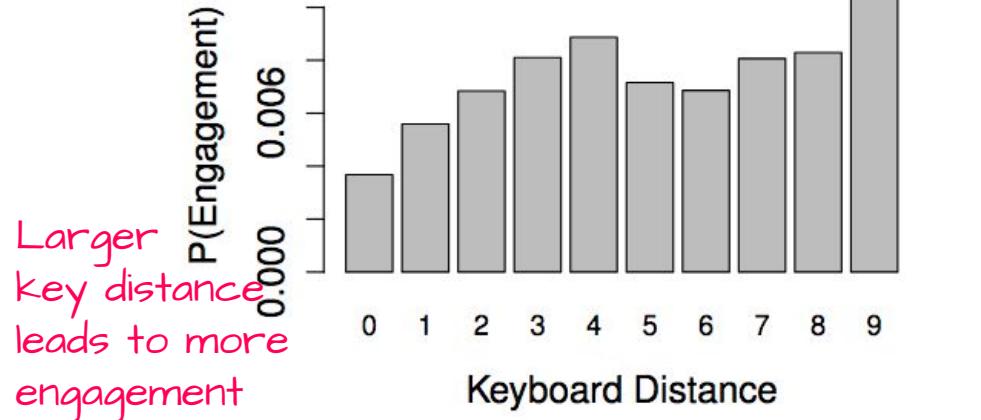
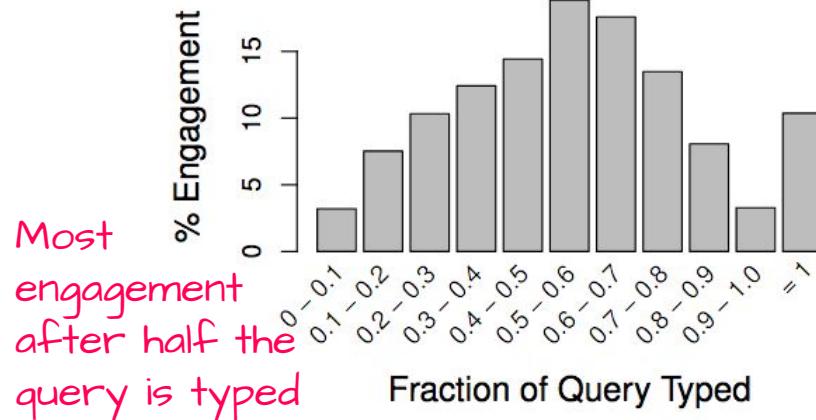
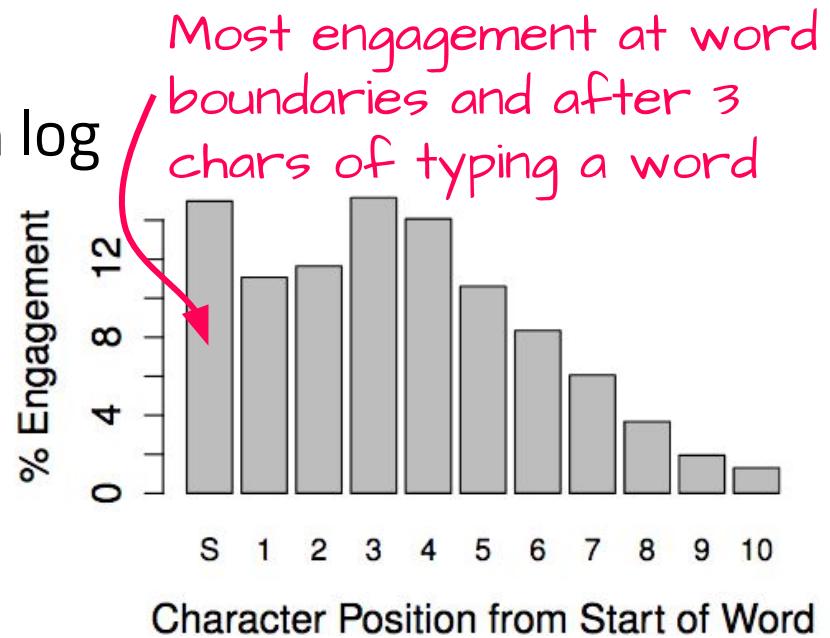
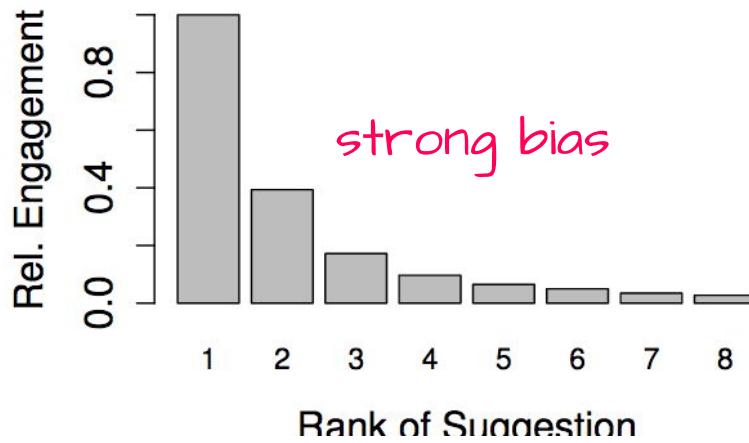
Bing trade secrets

Models	AOL		Bing
	MRR	% Improv.	% Improv.
Full-query based candidates only			
MostPopularCompletion	0.1446	-	-
LambdaMART Model (n -gram features = no, CLSM feature = no)	0.1445	-0.1	-1.7*
LambdaMART Model (n -gram features = yes, CLSM feature = no)	0.1427	-1.4*	-1.2*
LambdaMART Model (n -gram features = no, CLSM feature = yes)	0.1445	-0.1	-1.2*
LambdaMART Model (n -gram features = yes, CLSM feature = yes)	0.1432	-1.0*	-1.5*
Full-query based candidates + Suffix based candidates (Top 10K suffixes)			
MostPopularCompletion	0.1446	-	-
LambdaMART Model (n -gram features = no, CLSM feature = no)	0.2116	+46.3*	+32.8*
LambdaMART Model (n -gram features = yes, CLSM feature = no)	0.2326	+60.8*	+42.6*
LambdaMART Model (n -gram features = no, CLSM feature = yes)	0.2249	+55.5*	+40.1*
LambdaMART Model (n -gram features = yes, CLSM feature = yes)	0.2339	+61.7*	+43.8*

An **example** that shows how hard we (the IR community) have to work to yield significant gains from deep learning approaches. Gains are possible, but not guaranteed.

User engagement with query completion

1.6M queries from Bing's search log



That's it for query refinement!

Slack: in43252019.slack.com
Email: in4325-ewi@tudelft.nl