Information Retrieval (Handout Second Part)

Computer Science Tripos Part II

Simone Teufel

Natural Language and Information Processing (NLIP) Group
UNIVERSITY OF CAMBRIDGE
Simone.Teufel@cl.cam.ac.uk

Lent 2012

1. Lecture 5: Advanced Retrieval Models
   - Dimensionality Reduction (LSI)
   - The Probabilistic Model
   - Relevance Feedback (for VSM)
   - Query Expansion

Dimensionality Reduction

- Vectors in standard vector space are very sparse
- Orthogonal dimensions clearly wrong for near-synonyms *canine–dog*
- Different word senses are conflated into the same dimension
- One way to solve this: **dimensionality reduction**

Latent Semantic Analysis

- Hypothesis for LSA (Latent Semantic Analysis; Landauer): true semantic space has fewer dimensions than number of words observed.
- Extra dimensions are noise. Dropping them brings out **latent** semantic space
- Decompose document-term matrix into 3 matrices
- The central one only has *k* true dimensions (top eigenvalues of document-term matrix)

Linear Algebra: a reminder

- Eigenvalues λ and eigenvectors $\vec{x}$ of a matrix **A**:
  $\mathbf{A}\ \vec{x} = \lambda\vec{x}$
- Example:

$$\mathbf{A} = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 9 & 0 \\ 0 & 0 & 4 \end{pmatrix} \Rightarrow \vec{x_1} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \vec{x_2} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \vec{x_3} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

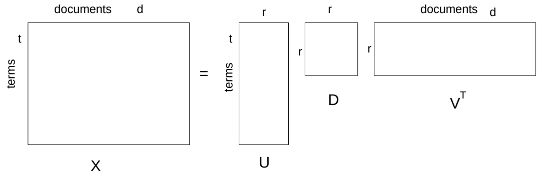
$$\lambda_1 = 9; \lambda_2 = 4; \lambda_3 = 2$$

- Eigenvalues are determined by solving the polynomial
  $\det(\mathbf{A} - \lambda\ \mathbf{I}) = 0$
  I is unit matrix (diagonal consists of 1s, 0s otherwise)

Eigenvector Decomposition

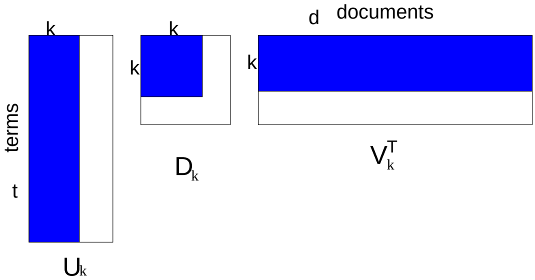
- We can decompose any square matrix C into 3 matrices
  $C = Q \Lambda Q^{-1}$
  such that Q represents the eigenvectors, and eigenvalues are listed in descending order in matrix Λ.
- Rectangular matrices need SVD (Singular Value Decomposition) for similar decomposition, because they have left and right singular vectors rather than eigenvectors.
- Left singular vectors of A are eigenvectors of AA^T.
- Right singular vectors of A are eigenvectors of $A^T A$.

Singular Value Decomposition



- r: rank of matrix; t: no of terms; d: no of documents
- D contains singular values (square roots of common eigenvalues for U and V) in descending order
- U contains left singular vectors of X in same ordering
- V contains right singular vectors of X in same ordering

Singular Value Decomposition



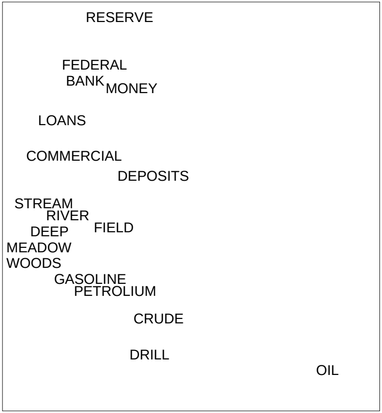
- Keep only first k (most dominant) singular values in D
- This results in two latent semantic spaces:
  - Reduced U_k represents terms in concept space
  - Reduced V_k represents documents in concept space

Dimensionality Reduction

Similarity calculations in LSI:

- Term–term similarity: $\mathrm{U}_k\mathrm{D}_k$
- Document–document similarity: $\mathrm{V}_k\mathrm{D}_k$
    - Folding in of query: $\vec{q_k} = V_k^{-1} D_k^T \vec{q}$ puts it in concept space
    - It can now be compared to other documents in concept space (cosine)
- Term–document similarity: compare vector in $\mathrm{U}_k\mathrm{D}_k^{\frac{1}{2}}$ with vector in $\mathrm{V}_k\mathrm{D}_k^{\frac{1}{2}}$
- Matrix D_k scales axes for comparison across spaces

Example: first 2 dimensions



from Griffiths, Steyvers, Tenenbaum (2007)

The probabilistic model

- Probability ranking principle: Present the documents by their estimated probability of relevance with respect to the information need: $P(R = 1|d, q)$ (van Rijsbergen, 1979)
- Bayes optimal decision rule: return only documents that are more likely to be relevant than nonrelevant: return d iff $P(R = 1|d, q) > P(R = 0|d, q)$
- Binary independence model (BIM): terms are independent from each other and are either present or not; relevance of a document is independent of the relevance of other documents
- $P(R = 1|\vec{x}, \vec{q}) = \frac{P(\vec{x}|R=1,\vec{q})P(R=1|\vec{q})}{P(\vec{x}|\vec{q})}$

Derivation of odds ratio

- Estimate odds $O(R|\vec{x},\vec{q}) = \frac{P(R=1|\vec{x},\vec{q})}{P(R=0|\vec{x},\vec{q})}$
- $O(R|\vec{x},\vec{q}) = \frac{P(R=1|\vec{q})}{P(R=0|\vec{q})} \cdot \frac{P(\vec{x}|R=1,\vec{q})}{P(\vec{x}|R=0,\vec{q})}$
- $= O(R|\vec{q}) \cdot \prod_{t=1}^{M} \frac{P(x_t|R=1,\vec{q})}{P(x_t|R=0,\vec{q})}$ (with term independence assumption)
- $= O(R|\vec{q}) \cdot \prod_{t:x_t=1} \frac{P(x_t=1|R=1,\vec{q})}{P(x_t=1|R=0,\vec{q})} \cdot \prod_{t:x_t=0} \frac{P(x_t=0|R=1,\vec{q})}{P(x_t=0|R=0,\vec{q})}$ (separating the terms)
- Notation:
  $p_t = P(x_t = 1|R = 1, \vec{q})$ – probability of a term appearing in a document relevant to query;
  $u_t = P(x_t = 1|R = 0, \vec{q})$ – probability of a term appearing in a nonrelevant document

Derivation of odds ratio, ctd

- Assumption: terms not occurring in query are equally likely to occur in relevant and nonrelevant documents: if $q_t = 0$ then $u_t = p_t$
- Then: $O(R|\vec{q},\vec{x}) = O(R|\vec{q}) \cdot \prod_{t:x_t=q_t=1} \frac{p_t}{u_t} \cdot \prod_{t:x_t=0,q_t=1} \frac{1-p_t}{1-u_t}$ (left product over query terms found in document; right product over query terms not found)
- $O(R|\vec{q},\vec{x}) = O(R|\vec{q}) \cdot \prod_{t:x_t=q_t=1} \frac{p_t(1-u_t)}{u_t(1-p_t)} \cdot \prod_{q_t=1} \frac{1-p_t}{1-u_t}$ (because we can multiply LHS with $\prod_{t:x_t=1,q_t=1} \frac{1-u_t}{1-p_t}$ and RHS with $\prod_{t:x_t=1,q_t=1} \frac{1-p_t}{1-u_t}$)
- For one particular query, right product is a constant (as is $O(R|\vec{q})$)

Derivation of odds ratio, ctd

- Now rank documents by log of left product:
  $RSV_d = \sum_{t:x_t=q_t=1} log \frac{p_t(1-u_t)}{u_t(1-p_t)}$
- $c_t = log \frac{p_t(1-u_t)}{u_t(1-p_t)} = log \frac{p_t}{1-p_t} + log \frac{1-u_t}{u_t}$
- Return documents with positive RSV (retrieval status value) scores

Estimating p_t in the real world

documents	relevant	non-relevant	total
term present x_t=1	s	$df_t - s$	df_t
term absent x_t=0	$S - s$	$(N - df_t) - (S - s)$	df_t
total	S	$N - S$	N

- $p_t = \frac{s}{S}$ and $u_t = \frac{df_t - s}{N - S}$
- Problem: initially we don't know S and s.
- Iterative estimation starts from assumption that $p_t = u_t$ is constant for all index terms (e.g. 0.5) and that $u_t = \frac{df_t}{N}$
- Use this model to partition documents into relevant and non-relevant, leading to better estimates for S and s
- Now recalculate p_t, u_t, thereby improving model, get next estimates of S and s – etc

Comparison VSM – BIM

- BIM forces us to initially guess separation of documents
- BIM does not take term frequency or document length into account (BM25/Okapi does)
- Like VSM, BIM also assumes independence of terms
- Some controversy over which performs better
- Overall, general preference for VSM due to its simplicity and good performance

Relevance Feedback

- Idea: If we knew which documents the user judged relevant, we could use the successful terms in those documents to weight other relevant documents higher.
- For instance by directly using them in a (modified) query
- So, simply ask the user which documents were good
- Then revise the query and present a second return set of documents.

The Roccio Algorithm

- Create a modified query $\vec{q_m}$
- from original query $\vec{q_0}$
- by taking into account the difference in term distributions of known relevant document set D_r and known nonrelevant document set D_{nr}.
- $$\vec{q_m} = \alpha \vec{q_0} + \beta \frac{1}{|D_r|} \sum_{\vec{d_j} \in D_r} \vec{d_j} - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d_j} \in D_{nr}} \vec{d_j}$$
- No resulting negative term weights → set to 0
- Starting from original query, move some distance away from centroid of irrelevant documents, and towards the centroid of the relevant documents

An example

Initial Query: “New Space satellite applications”

Return set:

+	.539	NASA hasn’t scrapped imaging spectrometer
+	.533	NASA scratches enviroment gear from satellite plan
	.528	Science Panel backs NASA satellite plan, but urges launches of smaller probes
	.526	A NASA Satellite project accomplishes incredible feat: staying within budget
	.525	Scientist who exposed global warming proposes satellites for climate research
	.524	Report Provides Support for the critics of using big satellites to study climate
	.516	Arianespace receives satellite launch pact from telesat canada
+	.509	Telecommunications tale of two companies

After Relevance Feedback:

*	.513	NASA scratches enviroment gear from satellite plan
*	.500	NASA hasn’t scrapped imaging spectrometer
	.493	When the Pentagon launches a secret satellite, space sleuths do some spy work . . .
	.493	NASA uses ’warm’ superconductors for fast circuit
*	.492	Telecommunications tale of two companies
	.491	Soviets may adapt parts of SS-20 missile for commercial use
	.490	Gaping gap: Pentagon lags in race to match the soviets in rocket launchers
	.490	Rescue of satellite by spaace agency to cost $90 Million

What did Roccio RF do?

It expanded the query with the following weights:

2.074	new
15.106	space
30.816	satellite
5.660	application
5.991	nasa
5.196	eos
4.196	launch
3.972	aster
3.516	instrument
3.446	arianespace
3.004	bundespost
2.806	ss
2.790	rocket
2.053	scientist
2.003	broadcast
1.172	earth
0.836	oil
0.646	measure

Assumptions behind RF

- Assumption 1: all relevant documents are similar to each other, and and all irrelevant documents are different to them
  - But: there are some inherently disjunctive queries
  - But: alternative vocabulary
- Assumption 2: user knows what they want and is willing to cooperate
  - Works best if we have data on many queries
  - If unwilling, use pseudo relevance feedback

Relevance Feedback: Making it work

- Positive feedback shown to be more important than negative feedback → keep γ low, e.g., α =1, $\beta = 0.85$, $\gamma = 0.15$
- Most useful in increasing recall in those situations where recall is important
- (Harman 1992) finds that using only a limited number of terms results in performance improvement; others disagree
- Fair evaluation means that we have to exclude documents already judged relevant by user from subsequent measurements
- Users don't like interruptions of their search process

Pseudo Relevance Feedback

- Simply assume the top N documents are relevant
- Then do relevance feedback as before
- Has been shown to improve results overall in TREC ad-hoc
- But: can be dangerous in some distributions of relevant documents, e.g., one large cluster of relevant documents and several smaller ones (which now have a lower chance of ever being retrieved).

Implicit Relevance Feedback

- Efforts to use other observable actions performed by the user and try to interpret them
    - do they click through (a good sign) – DirectHit search engine employs “clickstream mining”
    - and stay on a page long enough to possibly read it (even better)
- Research in its infancy, requires far larger-scale user studies than are currently available

Query Expansion

- Look for other sources of relevant terms for the query from outside the return set
- Ask user directly: search engine suggests related terms which users can co-opt
- How to generate these suggestions?
- They must come from a thesaurus (repository of substitutable terms)
- The use of large external collection of documents (off-line, pre-search time) is called global analysis
- Global analysis serves to automatically create a thesaurus (which also allows us to keep it dynamically up-to-date)
- Advantage: we need no help from user at search time

Term–term similarity-based thesaurus

- Start with term-document matrix A (weighted; rows length-normalised)
- Calculate AA^T, a term-term matrix that records in its cells $C_{u,v}$ how often terms u and v cooccur with each other in the same document.
- Some example output from Schuetze and Pedersen's (1997) automatic thesaurus (who use LSA and cosine):

word	nearest neighbours
bottomed	dip, copper, drops, topped, slide, trimmed
captivating	shimmer, stunningly, superbly, plucky, witty
lithographs	drawings, Picasso, Dali, sculptures, Gauguin
senses	grasp, psyche, truly, clumsy, naive, innate

- Quality can be a problem; polysemy introduces noise
- Overall, less successful than RF, but still active research area

Reading for Today (L5)

- Course textbook: chapters 9, 11, 18
- Chapter “Classic Information Retrieval Models” (2.5) in *Modern Information Retrieval* for a simpler description

2 Lecture 6: Evaluation of Retrieval Models

- Difficulties with IR Evaluation
- TREC; Test Collections
- Precision and Recall
- Metrics for Ranked Retrieval
  - Simplified PageRank
  - Full PageRank Formula
  - PageRank as a probabilistic Process
  - PageRank Computation

Evaluation in IR

- We want to know how well a retrieval sytem performs
- What is “performance” in an IR setting?
  - For a DBMS, performance is data retrieval time, since search is exact
  - For an IR system, search is inexact
    - still interested in retrieval time
    - also interested in retrieval accuracy
    - may be interested in other factors: ease of use, presentation of documents, help in formulating queries, . . .
- IR evaluation has focused primarily on retrieval accuracy: how good is a system at returning documents which are relevant to the user need?

History

- Evaluation has been a key issue in IR since the 60's
  - consequence of the empirical approach taken to IR
- Early work compared manual vs. automatic indexing
- The TREC competitions (over the last decade) have been very influential

Difficulties with IR Evaluation

- "Relevance" is difficult to define precisely
  - who makes the judgement?
  - humans are not very consistent
- Information need may not be clear – so how can we determine if it's been satisfied?
- Difficult to separate the user from the system, especially in interactive retrieval
- Judgements depend on more than just document and query
- For large document collections, difficult to determine the set of relevant documents

Evaluation under Laboratory Conditions

- Evaluation has been used as an analytical tool in an experimental setting
  - e.g. to determine if one weighting scheme is better than another
  - implies control of experimental variables
- Abstraction of IR system from operational setup
- Largely ignored interaction with the user
- Concentration on measures like precision and recall using standard test collections

TREC

- Text Retrieval Conference
  - Established in 1992; annual conference
  - designed to evaluate large-scale IR
    (2 gigabyte document collections, up to a million documents)
  - Run by NIST (US technology agency)
  - In 1992 25 organisations – industrial and academic – participated
  - In 2003 93 groups participated from 22 different countries
  - http://trec.nist.gov/

Test Collections

- Test collections used to compare retrieval performance of systems / techniques
    - set of documents
    - set of queries (or topics)
        - typically text description of user need, or information request, from which final query is constructed
    - set of relevance judgements
- How to compare performance?
    - results (set of returned documents, usually ranked) compared using some performance measure
    - precision and recall most common measures
- Ideally use multiple test collections
    - performance can be collection-specific

Use of Test Collections

- Before TREC, IR testing was on a relatively small scale
- Earlier work tended to use the same test material to maintain comparability
- Large test collections (both queries and documents) are important
    - to ensure statistical significance of results
    - to convince commercial system operators of the validity of the results
- TREC tracks typically have hundreds of thousands of documents, and hundreds of topics

Sample TREC Query

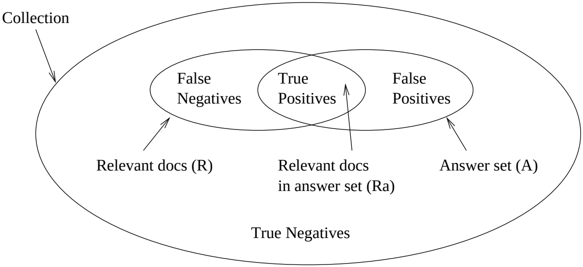
<num> Number: 508
<title> hair loss is a symptom of what diseases
<desc> Description:
Find diseases for which hair loss is a symptom.
<narr> Narrative:
A document is relevant if it positively connects the loss of head hair in humans with a specific disease. In this context, “thinning hair” and “hair loss” are synonymous. Loss of body and/or facial hair is irrelevant, as is hair loss caused by drug therapy.

TREC Relevance Judgements

Relevance Judgements

- Did the system return all possible relevant documents?
  - need a relevance judgement for every document in the collection, for every query/topic
  - at 30s a document/topic pair, would take 6,500 hours to judge 800,000 TREC documents for one topic
- TREC solution is pooling
  - select N runs per system
  - take the top K (usually 100) documents returned by each system (according to system's ranking) for those runs
  - then assume all relevant documents are in union and manually assess this set
  - pooling found not to be bias towards systems contributing to the pool

Precision and Recall for Document Retrieval



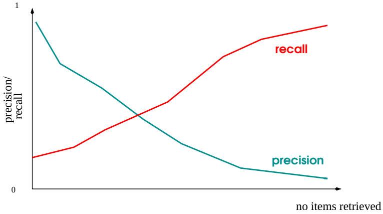
- Precision $= |\mathrm{Ra}|/|\mathrm{A}|$
  - precision $= \hat{P}(\mathrm{relevant}|\mathrm{retrieved})$
- Recall $= |\mathrm{Ra}|/|\mathrm{R}|$
  - recall $= \hat{P}(\mathrm{retrieved}|\mathrm{relevant})$

Another Representation

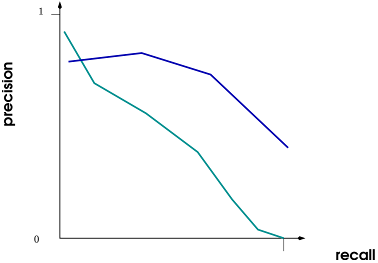
	relevant	not relevant
retrieved	A	B
not retrieved	C	D

- precision $= \text{A} \,/\, (\text{A}+\text{B})$ — $\hat{P}(\text{relevant}|\text{retrieved})$
- recall $= \text{A} \,/\, (\text{A}+\text{C})$ — $\hat{P}(\text{retrieved}|\text{relevant})$
- miss $= \text{C} \,/\, (\text{A}+\text{C})$ — $\hat{P}(\text{not-retrieved}|\text{relevant})$
- false alarm (fallout) $= \text{B} \,/\, (\text{B}+\text{D})$ — $\hat{P}(\text{retrieved}|\text{not-relevant})$

Recall-precision curve



Plot P and R as a function of how many docs are retrieved
Inverse relationship between P and R
P/R cross-over point is performance estimate



- Plot P as a function of R: precision–recall curve
- Area under normalised P-R curve is performance estimate

Recall-criticality and precision-criticality

- Inverse relationship between precision and recall forces general systems to go for compromise between them
- But some tasks particularly need good precision whereas others need good recall:

Precision-critical task	Recall-critical task
Little time available	Time matters less
A small set of relevant documents answers the information need	One cannot afford to miss a single document
Potentially many documents might fill the information need (redundantly)	Need to see *each* relevant document
Example: web search for factual information	Example: patent search

Single Value Measures

- F-score $= \frac{1}{\frac{1}{2}(\frac{1}{P}+\frac{1}{R})} = \frac{2PR}{P+R}$
- F-score is harmonic mean of P and R: inverse of average of inverses
- F-score is 1 when P = R = 1 and 0 when P or R are 0
- Penalises low values of P or R
  - it is very easy to obtain high precision (just return very few documents) or high recall (return all documents)
- Generalisation is E-measure $= \frac{1}{\alpha\frac{1}{P}+(1-\alpha)\frac{1}{R}}$
- $\alpha = \frac{1}{2}$ gives F-score

Metrics for Ranked Retrieval

- Precision and Recall well-defined for sets
- But matching can be defined as a matter of degree
  - Vector space model returns similarity score for each document
- How to evaluate the quality of the rank-ordering, as well as the number and proportion of relevant documents retrieved?

Precision/Recall @ Rank

Rank	Doc
1	d_{12}
2	d_{123}
3	d_4
4	d_{57}
5	d_{157}
6	d_{222}
7	d_4
8	d_{26}
9	d_{77}
10	d_{90}

- Blue documents are relevant
  - P@n: P@3 $= 0.33$, P@5 $= 0.2$, P@8 $= 0.25$
  - R@n: R@3 $= 0.33$, R@5 $= 0.33$, R@8 $=$ 0.66
- Ranks chosen for reporting depend on expected quantity of documents retrieved
- Rank statistics give some indication of how quickly user will find relevant documents from ranked list
- But may want to abstract away from ranking, since size of ranking will depend on query and document set

Precision at Recall *r*

Rank	S1	S2
1	X	
2		X
3	X	
4		
5		X
6	X	X
7		X
8		X
9	X	
10	X	

→

	S1	S2
p @ r 0.2	1.0	0.5
p @ r 0.4	0.67	0.4
p @ r 0.6	0.5	0.5
p @ r 0.8	0.44	0.57
p @ r 1.0	0.5	0.63

Summary IR measures over several queries

- Want to average over queries
- Problem: queries have differing number of relevant documents
- Cannot use one single cut-off level for all queries
    - This would not allow systems to achieve the theoretically possible maximal values in all conditions
    - Example: if a query has 10 relevant documents
        - If cutoff > 10, $P < 1$ for all systems
        - If cutoff < 10, $R < 1$ for all systems
- Therefore, more complicated joint measures are required

Mean Average Precision (MAP)

- Also called “average precision at seen relevant documents”
- Determine precision at each point when a new relevant document gets retrieved
- Use P=0 for each relevant document that was not retrieved
- Determine average for each query, then average over queries

$$MAP = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{Q_j}\sum_{i=1}^{Q_j}P(doc_i)$$

with:

Q_j	number of relevant documents for query j
N	number of queries
$P(doc_i)$	precision at ith relevant document

Mean Average Precision: example $(MAP = \frac{0.564+0.623}{2} = 0.594)$

Query 1		
Rank		$P(doc_i)$
1	X	1.00
2		
3	X	0.67
4		
5		
6	X	0.50
7		
8		
9		
10	X	0.40
11		
12		
13		
14		
15		
16		
17		
18		
19		
20	X	0.25
AVG:		0.564

Query 2		
Rank		$P(doc_i)$
1	X	1.00
2		
3	X	0.67
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15	X	0.2
AVG:		0.623

11 point average precision

$$P_{11\_pt} = \frac{1}{11} \sum_{j=0}^{10} \frac{1}{N} \sum_{i=1}^{N} \tilde{P}_i(r_j)$$

with $\tilde{P}_i(r_j)$ the precision at the jth recall point in the ith query (out of N queries)

- Define 11 standard recall points $r_j = \frac{j}{10}$: $r_0 = 0,\ r_1 = 0.1 \ldots r_{10} = 1$
- We need $\tilde{P}_i(r_j)$; i.e. the precision at our recall points
- $P_i(R = r)$ is the precision at those points when recall changes (a new relevant document is retrieved)
- But $\tilde{P}_i(r_j)$ does not always coincide with a measurable data point r (only if number of relevant documents per query is divisible by 10)

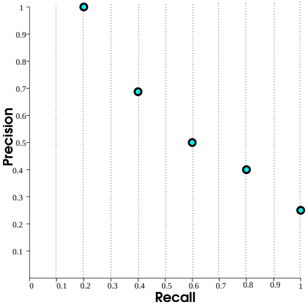
Interpolation

- Solution: interpolation

$$\tilde{P}_i(r_j) = \begin{cases} max(r_j \leq r < r_{j+1}) P_i(R = r) & \text{if } P_i(R = r) \text{ exists} \\ \tilde{P}_i(r_{j+1}) & \text{otherwise} \end{cases}$$

- Note that $P_i(R = 1)$ can always be measured.

11 point average precision: measured data points, Q1

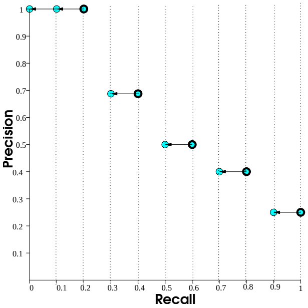


- Blue for Query 1
- Bold Circles measured

Query 1				
Rank		R	P	
1	X	0.2	1.00	$\tilde{P}_1(r_2) = 1.00$
2				
3	X	0.4	0.67	$\tilde{P}_1(r_4) = 0.67$
4				
5				
6	X	0.6	0.50	$\tilde{P}_1(r_6) = 0.50$
7				
8				
9				
10	X	0.8	0.40	$\tilde{P}_1(r_8)= 0.40$
11				
12				
13				
14				
15				
16				
17				
18				
19				
20	X	1.0	0.25	$\tilde{P}_1(r_{10}) = 0.25$

- Five r_js ($r_2, r_4, r_6, r_8, r_{10}$) coincide directly with

11 point average precision: interpolation, Q1



- Bold circles measured
- **thin circles interpolated**

Query 1			
Rank		R	P
1	X	.20	1.00
2			
3	X	.40	.67
4			
5			
6	X	.60	.50
7			
8			
9			
10	X	.80	.40
11			
12			
13			
14			
15			
16			
17			
18			
19			
20	X	1.00	.25

$\tilde{P}_1(r_0) = 1.00$

$\tilde{P}_1(r_1) = 1.00$

$\tilde{P}_1(r_2) = 1.00$

$\tilde{P}_1(r_3) = .67$

$\tilde{P}_1(r_4) = .67$

$\tilde{P}_1(r_5) = .50$

$\tilde{P}_1(r_6) = .50$

$\tilde{P}_1(r_7) = .40$

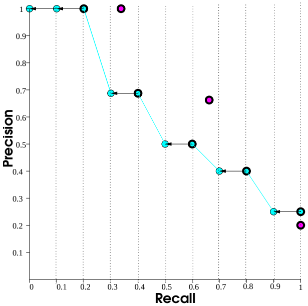
$\tilde{P}_1(r_8)= .40$

$\tilde{P}_1(r_9) = .25$

$\tilde{P}_1(r_{10}) = .25$

- The six other r_js (r_0, r_1, r_3, r_5, r_7, r_9) are interpolated

11 point average precision: measured data points, Q2



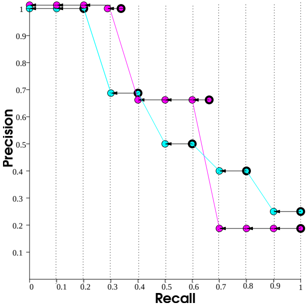
- Blue: Query 1; **Red: Query 2**
- **Bold circles measured**; thin circles interpol.

Query 2			
Rank	Relev.	R	P
1	X	.33	1.00
2			
3	X	.67	.67
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15	X	1.0	.2

$\tilde{P}_2(r_{10}) = .20$

- Only r_{10} coincides with a measured data point

11 point average precision: interpolation, Q2



- Blue: Query 1; **Red: Query 2**
- Bold circles measured; **thin circles interpol.**

$\tilde{P}_2(r_0) = 1.00$
$\tilde{P}_2(r_1) = 1.00$
$\tilde{P}_2(r_2) = 1.00$
$\tilde{P}_2(r_3) = 1.00$

$\tilde{P}_2(r_4) = .67$
$\tilde{P}_2(r_5) = .67$
$\tilde{P}_2(r_6) = .67$

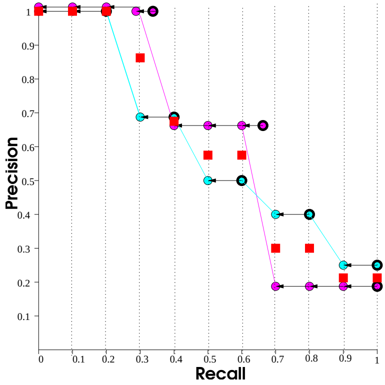
Query 2			
Rank	Relev.	R	P
1	X	.33	1.00
2			
3	X	.67	.67
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15	X	1.0	.2

$\tilde{P}_2(r_7) = .20$
$\tilde{P}_2(r_8) = .20$
$\tilde{P}_2(r_9) = .20$

$\tilde{P}_2(r_{10}) = .20$

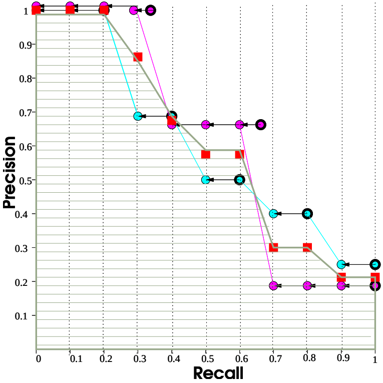
- 10 of the r_js are interpolated

11 point average precision: averaging



- Now average at each p_j
- over N (number of queries)
- → 11 averages

11 point average precision: area/result



- End result:
- 11 point average precision
- Approximation of area under prec. recall curve

IR Performance

- Difficult to raise performance in both precision and recall (precision/recall trade-off)
  - any improvement in precision typically results in a decrease in recall, and vice versa
- Even with small collections, difficult to raise performance beyond 40%/40% P/R level
- With larger collections 30%/30% is more likely
- Systems using statistically based natural language indexing provide respectable performance which is hard to beat

Summary

- Focused on evaluation for ad-hoc retrieval
  - other issues arise when evaluating different tracks, e.g. QA, although typically still use P/R-based measures
- Evaluation for interactive tasks is more involved
- Significance testing is an issue
  - could a good result have occurred by chance?
  - is the result robust across different document sets?
  - slowly becoming more common
  - underlying population distributions unknown, so apply non-parametric tests such as the sign test

Reading for Today (L6)

- Course Textbook chapter 6

3 Lecture 7: Web Retrieval

- Challenges for web search
- PageRank
- HITS: Hubs and Authorities

Challenges of Web Search

- Distributed data
  - data is stored on millions of machines with varying network characteristics
- Volatile data
  - new computers and data can be added and removed easily
  - dangling links and relocation problems
- Large volume
- Unstructured and redundant data
  - not all HTML pages are well structured
  - much of the Web is repeated (mirrored or copied)

Challenges of Web Search

- Quality of data
  - data can be false, invalid (e.g. out of date), SPAM
  - poorly written, can contain grammatical errors
- Heterogeneous data
  - multiple media types, multiple formats, different languages
- Unsophisticated users
  - information need may be unclear
  - may have difficulty formulating a useful query

Web Challenges – Size of Vocabulary

- Heap's law: $V = Kn^{\beta}$
    - β is typically between 0.4 and 0.6, so vocabulary size V grows roughly with the square root of the text size n
- 99% of distinct words in the VLC2 collection are not dictionary headwords (Hawking, Very Large Scale Information Retrieval)

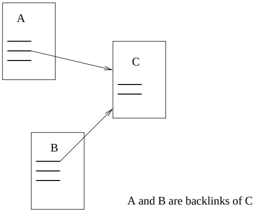
Link-Based Retrieval

- A characteristic of the Web is its hyperlink structure
- Web search engines exploit properties of the structure to try and overcome some of the web-specific challenges
- Basic idea: hyperlink structure can be used to infer the validity / popularity / importance of a page
  - similar to citation analysis in academic publishing
  - number of links to a page correspond with page's importance
  - links coming from an important page are indicators of other important pages
  - Anchor text describes the page
    - can be a useful source of text in addition to the text on the page itself, eg *Big Blue* → IBM

PageRank

- PageRank is *query-independent* and provides a global importance score for every page on the web
  - can be calculated once for all queries
  - but can't be tuned for any one particular query
- PageRank has a simple intuitive interpretation:
  - PageRank score for a page is the probability a random surfer would visit that page
- PageRank is/was used by Google
  - PageRank is combined with other measures such as TF×IDF

Link Structure for PageRank



A and B are backlinks of C

- Pages with many backlinks are typically more important than pages with few backlinks
- But pages with few backlinks can also be important
  - some links, e.g. from Yahoo, are more important than other links

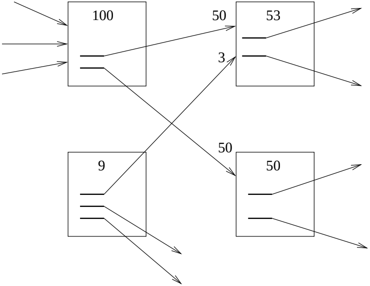
PageRank Scoring

- Consider a browser doing a random walk on the Web
  - start at a random page
  - at each step go to another page along one of the out-links, each link having equal probability
- Each page has a long-term visit rate (the "steady state")
  - use the visit rate as the score

Simplified PageRank

$$R(u) = d \sum_{v:v \rightarrow u} \frac{R(v)}{N_v}$$

u is a web page
N_v is the number of links from v



Teleporting

- Web is full of dead-ends
  - "long-term visit rate" doesn't make sense
- A page may have no in-links
- *Teleporting*: jump to any page on the Web at random (with equal probability $1/N$)
  - when there are no out-links use teleporting
  - otherwise use teleporting with probability α, or follow a link chosen at random with probability $(1-\alpha)$

PageRank

$$R(u) = (1-\alpha) \sum_{v:v \to u} \frac{R(v)}{N_v} + \alpha E(u)$$

- $E(u)$ is a prior distribution over web pages
- Typical value of α is 0.1
- $R(u)$ can be calculated using an iterative algorithm

Probabilistic Interpretation of PageRank

- PageRank models the behaviour of a ”random surfer”
- Surfer randomly clicks on links, sometimes jumping to any page at random based on E
- Probability of a random jump is α
- PageRank for a page is the probability that the random surfer finds himself on that page

Markov Chains

- A Markov chain consists of n *states* plus an $n \times n$ *transition probability matrix* **P**
- At each step, we are in exactly one of the states
- For $1 \leq i, j \leq n$, the matrix entry P_{ij} tells us the probability of j being the next state given the current state is i
- For all i, $\sum_{j=1}^{n} P_{ij} = 1$
- Markov chains are abstractions of random walks
    - crucial property is that the distribution over next states only depends on the current state, and not how the state was arrived at

Random Surfer as a Markov Chain

- Each state represents a web page; each transition probability represents the probability of moving from one page to another
  - transition probabilities include teleportation
- Let $\overline{x}^t$ be the probability vector for time t
  - x_i^t is the probability of being in state i at time t
- we can compute the surfer's distribution over the web pages at any time given only the initial distribution and the transition probability matrix **P**

$$x_i^t = \overline{x}^0 P^t$$

Ergodic Markov Chains

- A Markov chain is *ergodic* if the following two conditions hold:
    - For any two states i, j, there is an integer $k \geq 2$ such that there is a sequence of k states $s_1 = i, s_2, \ldots, s_k = j$ such that $\forall l, 1 \leq l \leq k-1$, the transition probability $P_{s_l, s_{l+1}} > 0$
    - There exists a time T_0 such that for all states j, and for all choices of start state i in the Markov chain, and for all $t > T_0$, the probability of being in state j at time t is > 0

Ergodic Markov Chains

- Theorem: *For any ergodic Markov chain, there is a unique steady-state probability distribution over the states,* $\overline{\pi}$, *such that if* $N(i,t)$ *is the number of visits to state* i *in* t *steps, then*

  $$\lim_{t \to \infty} \frac{N(i,t)}{t} = \pi(i),$$

  where $\pi(i) > 0$ *is the steady-state probability for state* i*.*

  (Introduction to IR, ch.21)

- $\pi(i)$ is the PageRank for state/web page i

Eigenvectors of the Transition Matrix

- The *left eigenvectors* of the transition probability matrix P are N-vectors $\overline{\pi}$ such that

$$\overline{\pi}\, P = \lambda\, \overline{\pi}$$

- We want the eigenvector with eigenvalue 1 (this is known as the *principal* left eigenvector of the matrix P, and it has the largest eigenvalue)
- This makes π the steady-state distribution we're looking for

PageRank Computation

- There are many ways to calculate the principal left eigenvector of the transition matrix
- One simple way:
    - Start with any distribution, eg $\overline{x} = (1, 0, \ldots, 0)$
    - After one step, distribution is $x\ P$
    - After two steps, distribution is $x\ P^2$
    - For large k, $x\ P^k = a$, where a is the steady state
    - Algorithm: keep multiplying x by P until the product looks stable

Personalised PageRank

- Putting all the probability mass from E onto a single page produces a personalised importance ranking relative to that page
- E gives the probabilities of jumping to pages via a random jump
- Putting all the mass on one page emphasises pages "close to" that page

HITS

- Hypertext Induced Topic Search (Kleinberg)
  - “Hyperlinks encode a considerable amount of latent human judgement”
  - “The creator of page p, by including a link to page q, has in some measure *conferred authority* on q”
- Example: consider the query ”Harvard”
  - `www.harvard.edu` may not use *Harvard* most often
  - but many pages containing the term *Harvard* will point at `www.harvard.edu`
- But some links are created for reasons other than conferral of authority, e.g. navigational purposes, advertisements
- Need also to balance criteria of *relevance* and *popularity*
  - e.g. lots of pages point at `www.google.com`

Hubs and Authorities (for a given query)

- An authority is a page which has many relevant pages pointing at it
  - authorities are likely to be relevant (precision)
  - there should be overlap between the sets of pages which point at authorities
- A hub is a page which links to many authorities
  - hubs help find relevant pages (recall)
  - hubs "pull-together" authorities on a common topic
  - hubs allow us to ignore non-relevant pages with a high *in-degree*
- Relationship between hubs and authorities is mutually reinforcing:
  - a good hub points to many good authorities
  - a good authority is pointed at by many good hubs

Finding Hubs and Authorities

- Suppose we are given some query σ
- We wish to find authoritative pages with respect to σ, restricting computation to a relatively small set of pages:
  - recover top-*n* pages using some search engine: the *root set*
  - add pages which link to the root set and pages which the root set link: the *base set*
- Base set might contain a few thousand documents, with many authorities
  - how do we find the authorities?

Finding Hubs and Authorities

- Each page p has a hub weight h_p and authority weight a_p
- Initially set all weights to 1
- Update weights iteratively:

$$h_p \leftarrow \sum_{q:p \to q} a_q$$

$$a_p \leftarrow \sum_{q:q \to p} h_q$$

  - $p \to q$ means p points at q
  - weights are normalised after each iteration
  - can prove this algorithm converges
- Pages for a given query can then be weighted by their hub and authority weights

Calculating Hub and Authority Weights

```
Loop($G$,$k$):
  $G$: a collection of $n$ linked pages
  $K$: a natural number
  Let $z$ denote the vector $(1,1,1,...,1) \in \mathcal{R}^n$
  Set $\overline{a}_0 := z$
  Set $\overline{h}_0 := z$
  For $i = 1,2,...,k$
    Update $\overline{a}_{i-1}$ obtaining new weights $\overline{a}_i^{'}$
    Update $\overline{h}_{i-1}$ obtaining new weights $\overline{h}_i^{'}$
    Normalise $\overline{a}_i^{'}$ obtaining $\overline{a}_i$
    Normalise $\overline{h}_i^{'}$ obtaining $\overline{h}_i$
  Return $(\overline{a}_k,\overline{h}_k)$
```

Example Results for HITS

Query	Top Authorities	
censorship	.378 http://www.eff.org/	The Electronic Frontier Foundation
	.344 http://www.eff.org/blueribbon.html	Campaign for online free speech
	.238 http://www.cdt.org/	Center for democracy & technology
	.235 http://www.vtw.org/	Voters telecommunications watch
search engines	.346 http://www.yahoo.com/	Yahoo
	.291 http://www.excite.com/	Excite
	.239 http://www.mckinley.com/	Welcome to Magellan
	.231 http://www.lycos.com/	Lycos home page
	.231 http://www.altavista.digital.com	AltaVista
Gates	.643 http://www.roadahead.com/	Bill Gates: The Road Ahead
	.458 http://www.microsoft.com/	Welcome to Microsoft
	.440 http://www.microsoft.com/corpinfo	

Reading for Today (L7)

- Course Textbook, chapter 21.

Additional (research papers):

- Authoritative Sources in a Hyperlinked Environment (1999), Jon Kleinberg, Journal of the ACM
- The PageRank Citation Ranking: Bringing Order to the Web (1998), Lawrence Page et al.
- The Anatomy of a Large-Scale Hypertextual Web Search Engine, Sergey Brin and Lawrence Page

available online

4 Lecture 8: Question Answering
- QA Task Definition
- QA Evaluation Metrics
- Three QA systems
- Named Entity Recognition and Answer Types
  - Comparison of Logical Form
  - Searching for Variants
- A data-driven approach

Question Answering: Task definition in TREC-QA

- QA Track since TREC-1999: Open-domain factual textual QA
- Task requirements (in comparison with IR):
  1. Input: NL questions, not keyword-based queries
  2. Output: answers, not documents
- Rules:
  - All runs completely automatic
  - Frozen systems once questions received; answers back to TREC within one week
  - Answers may be extracted or automatically generated from material in document collection only
  - The use of external resources (dictionaries, ontologies, WWW) is allowed
  - Each returned answer is checked manually by TREC-QA (no comparison to gold standard)

TREC QA: Example questions

TREC-8	How many calories are there in a Big Mac? Where is the Taj Mahal?
TREC-9	Who invented the paper clip? How much folic acid should an expectant mother take daily? Who is Colin Powell?
TREC-10	What is an atom? How much does the human adult female brain weigh? When did Hawaii become a state?
TREC-11	Name 20 countries that produce coffee.

Questions in TREC

- Type of question: reason, definition, list of instances, context-sensitive to previous questions (TREC-10)
- Source of question: invented for evaluation (TREC-8); since TREC-9 mined from logs (Encarta, Excite)
    - → strong impact on task: more realistic questions are harder on assessors and systems, but more representative for training
- Type of answer string: 250 Bytes (TREC-8/9, since TREC-12); 50 Bytes (TREC-8–10); exact since TREC-11
- Guarantee of existence of answer: no longer given since TREC-10

Examples of answer strings

What river in the US is known as the Big Muddy?

System A:	the Mississippi
System B:	Known as Big Muddy, the Mississippi is the longest
System C:	as Big Muddy , the Mississippi is the longest
System D:	messed with . Known as Big Muddy , the Mississip
System E:	Mississippi is the longest river in the US
System F:	the Mississippi is the longest river in the US
System G:	the Mississippi is the longest river(Mississippi)
System H:	has brought the Mississippi to its lowest
System I:	ipes.In Life on the Mississippi,Mark Twain wrote t
System K:	Southeast;Mississippi;Mark Twain;officials began
System L:	Known; Mississippi; US,; Minnessota; Cult Mexico
System M:	Mud Island,; Mississippi; “The; history; Memphis

Decreasing quality of answers

Manual checking of answers

- Systems return [docid, answer-string] pairs; mean answer pool per question judged: 309 pairs
- Answers judged in the context of the associated document
- "Objectively" wrong answers okay if document supports them
  - Taj Mahal
- Considerable disagreement in terms of absolute evaluation metrics
- But relative MRRs (rankings) across systems very stable

Ambiguous answers are disqualified

Ambiguous answers are judged as “incorrect”:

`What is the capital of the Kosovo?`

```
250B answer:
protestors called for intervention to end the ‘‘Albanian
uprising’’. At Vucitrn, 20 miles northwest of Pristina, five
demonstrators were reported injured, apparently in clashes
with police. Violent clashes were also repo
```

Supportedness of answers

Answers need to be supported by the document context → the second answer is "unsupported":

`What is the name of the late Phillippine President Marco's wife?`

- `Ferdinand Marcos and his wife Imelda...` → [supported]
- `Imelda Marcos really liked shoes...` → [unsupported]

MRR: Mean reciprocal rank

- Task is precision-oriented: only look at top 5 answers
- Score for individual question i is the reciprocal rank r_i where the first correct answer appeared (0 if no correct answer in top 5 returns).

$$RR_i = \frac{1}{r_i}$$

- Possible reciprocal ranks per question: [0, 0.2, 0.25, 0.33, 0.5, 1]
- Score of a run (MRR) is mean over n questions:

$$MRR = \frac{1}{n}\sum_{i=1}^{n} RR_i$$

Example: Mean reciprocal rank

162: What is the capital of Kosovo?

1	18 April, 1995, UK GMT Kosovo capital
2	Albanians say no to peace talks in Pr
3	0 miles west of Pristina, five demon
4	Kosovo is located in south and south
5	The provincial capital of the Kosovo

$$\rightarrow RR_{162} = \frac{1}{3}$$

23: Who invented the paper clip?

1	embrace Johan Vaaler, as the true invento
2	seems puzzling that it was not invented e
3	paper clip. Nobel invented many useful th
4	modern-shaped paper clip was patented in A
5	g Johan Valerand, leaping over Norway, in

$$\rightarrow RR_{23} = 1$$

2: What was the monetary value of the Nobel Peace Prize in 1989?

1	The Nobel poll is temporarily disabled. 1994
2	perience and scientific reality, and applied
3	Curies were awarded the Nobel Prize together
4	the so-called beta-value. $40,000 more than
5	that is much greater than the variation in mean

$$\rightarrow RR_{2} = 0$$

$$\rightarrow MRR = \frac{\frac{4}{3}}{3} = .444$$

Other QA evaluation metrics used in TREC

- Average accuracy since 2003: only one answer per question allowed; accuracy is $\frac{Answers\ correct}{Total\ Answers}$
- Confidence-weighted score: systems submit one answer per question and order them according to the confidence they have in the answer (with their best answer first in the file)

$$\frac{1}{Q}\sum_{i=1}^{Q}\frac{\#correct\ in\ first\ i}{i}$$

(Q being the number of questions). This evaluation metric (which is similar to Mean Average Precision) was to reward systems for their confidence in their answers, as answers high up in the file participate in many calculations.

Results

- In TREC-8, 9, 10 best systems returned MMR of .65–.70 for 50B answers, answering around 70–80% of all questions
- In 55% of the cases where answer was found in the first 5 answers, this answer was in rank 1
- Accuracy of best system in TREC-10's list task had an accuracy of .75
- The best confidence-weighted score in TREC-11 achieved was .856 (NIL-prec .578, NIL recall .804)
- TREC-12 (exact task): Best performance was an accuracy of .700

QA systems

- Overview of three QA systems:
- Cymphony system (TREC-8)
  - NE plus answer type detection
  - Shallow parsing to analyse structure of questions
- SMU (TREC-9)
  - Matching of logical form
  - Feedback loops
- Microsoft (TREC-10)
  - Answer redundancy and answer harvesting
  - Claim: "Large amounts of data make intelligent processing unnecessary."

Overall algorithm

- Question Processing
  - Shallow parse
  - Determine expected answer type
  - Question expansion
- Document Processing
  - Tokenise, POS-tag, NE-index
- Text Matcher (= Answer production)
  - Intersect search engine results with NE
  - Rank answers

Named entity recognition

- Over 80% of 200 TREC-8 questions ask for a named entity (NE)
- NE employed by most successful systems in TREC (Verhees and Tice, 2000)
- MUC NE types: person, organisation, location, time, date, money, percent
- Textract covers additional types, e.g.:
  - number, fraction, decimal, ordinal, math equation
  - weight, length, temperature, angle, area, capacity, speed, rate
  - address, email, phone, fax, telex, www
- Textract subclassifies known types, e.g., organisation → company, government agency, school

Expected answer type

Who won the 1998 Nobel Peace Prize?

Expected answer type: PERSON
Key words: won, 1998, Nobel, Peace, Prize

Why did David Koresh ask the FBI for a word processor?

Expected answer type: REASON
Key words: David, Koresh, ask, FBI, word, processor

Question Expansion:

Expected answer type: [because | because of | due to | thanks to | since | in order to | to VP]
Key words: [ask|asks|asked|asking, David, Koresh, FBI, word, processor]

FST rules for expected answer type

R1: Name NP(city | country | company) → CITY|COUNTRY|COMPANY
VG[name] NP[a country] that VG[is developing] NP[a magnetic levitation railway system]

R2: Name NP(person_w) → PERSON
VG[Name] NP[the first private citizen] VG[to fly] PP[in space]
("citizen" belongs to word class `person_w`).

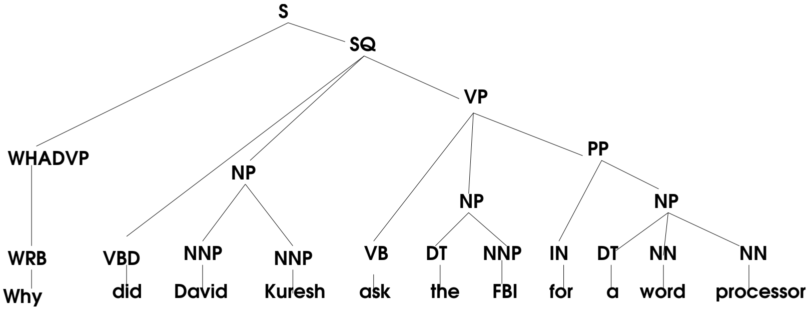
R3: CATCH-ALL: proper noun
Name a film that has won the Golden Bear in the Berlin Film Festival.

Direct matching of question words

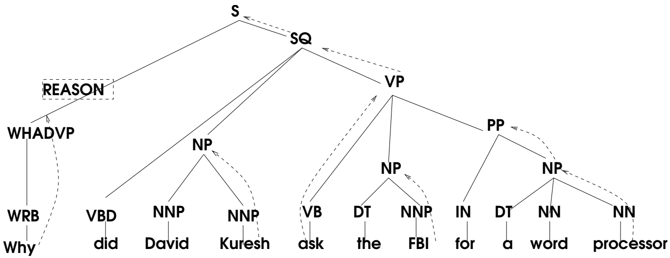
who/whom →	PERSON
when →	TIME/DATE
where/what place →	LOCATION
what time (of day) →	TIME
what day (of the week) →	DAY
what/which month →	MONTH
how often →	FREQUENCY
...	

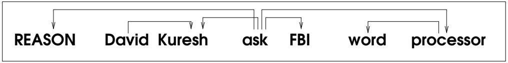
This classification happens only if the previous rule-based classification did not return unambiguous results.

Derivation of logical forms



Derivation of logical forms





Variants I + II: Morphological and Lexical

Morphological Variants (+40%):

- *Who invented the paper clip?* — Main verb "invent", ANSWER-TYPE "who" (subject) → add keyword "inventor"

Lexical Variants (+52%; used in 129 questions):

- *How far is the moon?* — "far" is an attribute of "distance"
- *Who killed Martin Luther King?* — "killer" = "assassin"

Variants III: Paraphrases

Semantic alternations and paraphrases, abductive reasoning (+8%; used in 175 questions)

- *How hot does the inside of an active volcano get?*
- Answer in "lava fragments belched out of the mountain were as hot as 300 degrees Fahrenheit"
- Facts needed in abductive chain:
    - volcano IS-A mountain; lava PART-OF volcano

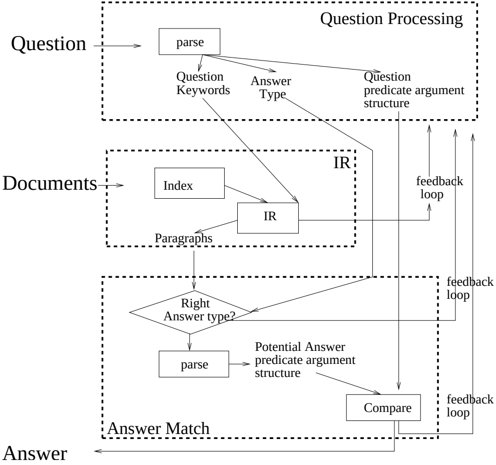
Combination of all variant feedback loops increases results considerably (+76%)

And the winner was. . . (repeatedly). . .

- The Southern Methodist University (SMU) system (Harabagiu et al.), a deep processing system (clear winner in most years, usually with a big gap to second contender)
- Machinery beyond answer type determination:
  1. Variants/feedback loops
  2. logical form-based comparison between answer candidate and question
- System was also very good at justifying its answers

	MRR lenient	MRR strict
Short answer	.599	.580
Long answer	.778	.760

Overview of SMU system



At the other end of the spectrum: just use (a lot of) data

- Circumvent difficult NLP problems by using more data
- The web has 2 billion indexed pages
- Deep reasoning is only necessary if search ground is restricted
- The larger the search ground, the greater the chance of finding answers with a simple relationship between question string and answer string:
  Who killed Abraham Lincoln?

DOC 1	John Wilkes Booth is perhaps America's most infamous assassin. He is best known for having fired the bullet that ended Abraham Lincoln's life.	TREC
DOC 2	John Wilkes Booth killed Abraham Lincoln.	web

The Microsoft system: Methods

1. Question processing is minimal: reordering of words, removal of question words, morphological variations
2. Matching done by Web query (google):
   - Extract potential answer strings from top 100 summaries returned
3. Answer generation is simplistic:
   - Weight answer strings (frequency, fit of match) – learned from TREC-9
   - Shuffle together answer strings
   - Back-projection into TREC corpus: keywords + answers to traditional IR engine
4. Improvement: Expected answer type filter (24% improvement)
   - No full-fledged named entity recognition

Query string generation

Rewrite module outputs a set of 3-tupels:

- Search string
- Position in text where answer is expected: LEFT|RIGHT|NULL
- Confidence score (quality of template)

Who is the world's richest man married to?

[+is the world's richest man married to	LEFT	5]
[the +is world's richest man married to	LEFT	5]
[the world's +is richest man married to	RIGHT	5]
[the world's richest +is man married to	RIGHT	5]
[the world's richest man +is married to	RIGHT	5]
[the world's richest man married +is to	RIGHT	5]
[the world's richest man married to +is	RIGHT	5]
[world's richest man married	NULL	2]
[world's AND richest AND married	NULL	1]

String weighting

- Obtain 1-grams, 2-grams, 3-grams from google short summaries
- Score each n-gram n according to the weight r_q of query q that retrieved it
- Sum weights across all summaries containing the ngram n (this set is called S_n)

$$w_n = \sum_{n \in S_n} r_q$$

w_n: weight of ngram n
S_n: set of all retrieved summaries which contain n
r_q: rewrite weight of query q

Answer string generation

- Merge similar answers (ABC + BCD → ABCD)
  - Assemble longer answers from answer fragments
  - Weight of new n-gram is maximum of constituent weights
  - Greedy algorithm, starting from top-scoring candidate
  - Stop when no further ngram tiles can be detected
  - But: cannot cluster "redwoods" and "redwood trees"
- Back-projection of answer
  - Send keywords + answers to traditional IR engine indexed over TREC documents
  - Report matching documents back as "support"
- Always return NIL on 5th position

The Microsoft system: Examples

- Success stories:

Question	Answer	TREC document
What is the birthstone for June?	Pearl	for two weeks during June (the pearl is the birth-stone for those born in that month)
What is the rainiest place on Earth?	Mount Wailaleale	and even Pago Pago, noted for its prodigious showers, gets only about 196 inches annually (The titleholder, according to the National Geographic Society, is Mount Wailaleale in Hawaii, where about 460 inches of rain falls each year).

  The MS system (and none of the deep systems) answered these questions.

- Time sensitivity of questions: `Q1202: Who is the Governor of Alaska?`

Microsoft system: Discussion

- Results: mid-range (.347 MRR, 49% no answer)
- Development time of less than a month
- Produced "exact strings" before TREC-11 demanded it: average returned length 14.6 bytes
- Does this system undermine of QA as a gauge for NL understanding?
  - If TREC wants to measure straight performance on factual question task, less NLP might be needed than previously thought
  - But if TREC wants to use QA as test bed for text understanding, "harder" questions might now be needed
- And still: the really good systems are still the ones that do deep NLP processing!

Summary

- Open domain, factual question answering
- TREC: Source of questions matters (web logs v. introspection)
- Mean reciprocal rank main evaluation measure
- MRR of best systems 0.68 - 0.58
- Best systems answer about 75% of questions in the first 5 guesses, and get the correct answer at position 1.5 on avg $(\frac{1}{.66})$
- System technology
  - NE plus answer type detection (Cymphony)
  - Matching of logical form, Feedback loops (SMU)
  - Answer redundancy and answer harvesting (Microsoft)

Reading for Today (L8)

- Course textbook chapter 8

Additional reading:

- Teufel (2007): Chapter *An Overview of evaluation methods in TREC Ad-hoc Information Retrieval and TREC Question Answering*. In: L. Dybkjaer, H. Hemsen, W. Minker (Eds.) Evaluation of Text and Speech Systems. Springer, Dordrecht, The Netherlands.
- Ellen Voorhees (1999): The TREC-8 Question Answering Track Report, Proceedings of TREC
- R. Srihari and W. Li (1999): "Information-extraction supported question answering", TREC-8 Proceedings
- S. Harabagiu et al (2001), "The role of lexico-semantic feedback in open-domain textual question-answering", ACL-2001
- E. Brill et al (2001), "Data intensive question answering", TREC-10 Proceedings