

Information Retrieval Evaluation

(CS429)

Nazli Goharian

nazli@ir.iit.edu

Slides are *mostly* based on Information Retrieval Algorithms and Heuristics, Grossman, Frieder

Measuring Effectiveness

- An algorithm is deemed incorrect if it does not have a “right” answer.
- A heuristic tries to guess something close to the right answer. Heuristics are measured on “how close” they come to a right answer.
- IR techniques are essentially heuristics because we do not know the right answer.
- So we have to measure how *close* to the right answer we can come.

Experimental Evaluations

- Batch (ad hoc) processing evaluations
 - Set of queries are run against a static collection
 - Relevance judgments identified by human evaluators are used to evaluate system
- User-based evaluation
 - Complementary to batch processing evaluation
 - Evaluation of users as they perform search are used to evaluate system (time, clickthrough log analysis, frequency of use, interview,...)

Some of IR Evaluation Issues

- How/what data set should be used?
- How many queries (topics) should be evaluated?
- What metrics should be used to compare systems?
- How often should evaluation be repeated?

Existing Testbeds

- Cranfield : A small (megabytes) domain specific testbed with fixed documents and queries, along with an exhaustive set of relevance judgment
- TREC: Various data sets for different tasks
 - Most use 25-50 queries (topics)
 - Collections size (2GB, 10GB,, Terabyte (GOV2)- this is only half a TByte as of yet!)
 - No exhaustive relevance judgment

Existing Testbeds (Cont'd)

- GOV2 (Terabyte):
 - 25 million pages of web; 100-10,000 queries; 426 GB
- Genomics:
 - 162,259 documents from the 49 journals; 12.3 GB
- Text Classification datasets:
 - Reuters-21578 (newswires)
 - Reuters RCV1 (806,791 docs)
 - 20 Newsgroups (20,000 docs; 1000 doc per 20 categories)
 -

Relevance Information & Pooling

- TREC uses *pooling* to approximate the number of relevant documents and identify these documents, called *relevance judgments* (*qrels*)
- For this, TREC maintains a set of documents, queries, and a set of relevance judgments that list which documents should be retrieved for each query (topics)
- In *pooling*, only top documents returned by the participating systems are evaluated, and the rest of documents, even relevant, are deemed non-relevant

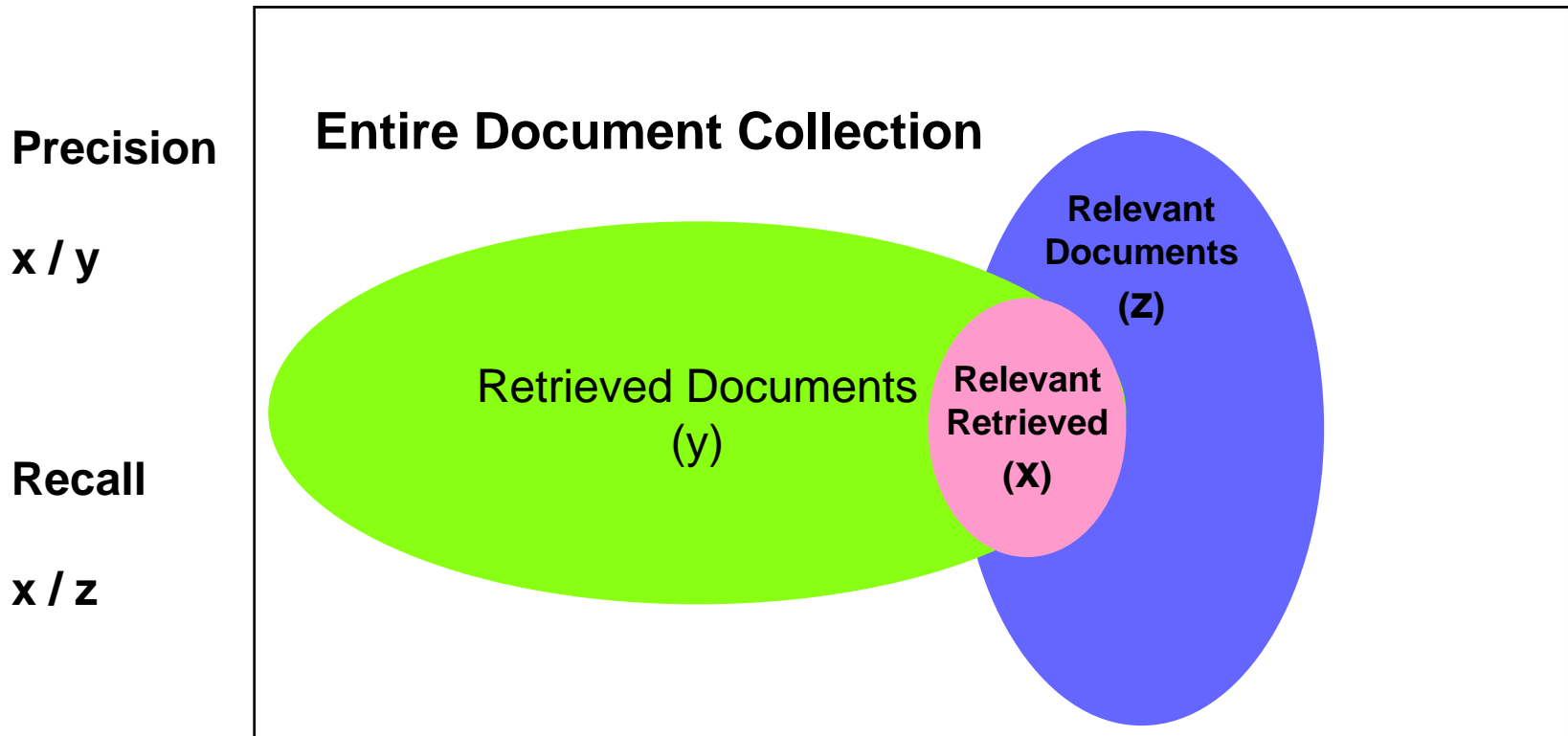
Problem...

- Building larger test collections along with complete relevance judgment is difficult or impossible as demands assessor time and many diverse retrieval runs.

Measures in Evaluating IR

- *Recall* is the fraction of relevant documents retrieved from the set of total relevant documents collection-wide.
- *Precision* is the fraction of relevant documents retrieved from the total number retrieved.

Precision / Recall



Precision / Recall

Example

- Consider a query that retrieves 10 documents.
- Lets say the result set is.
 - D1
 - D2
 - D3
 - D4
 - D5
 - D6
 - D7
 - D8
 - D9
 - D10
- With all 10 being relevant, Precision is 100%
- Having only 10 relevant in the whole collection, Recall is 100%

Example (continued)

- Now lets say that only documents two and five are relevant.
- Consider these results:
 - D1
 - D2**
 - D3
 - D4
 - D5**
 - D6
 - D7
 - D8
 - D9
 - D10
- Two out of 10 retrieved documents are relevant thus, precision is 20%. Recall is (2/total relevant) in entire collection.

Levels of Recall

- If we keep retrieving documents, we will ultimately retrieve all documents and achieve 100 percent recall.
- That means that we can keep retrieving documents until we reach $x\%$ of recall.

Levels of Recall (example)

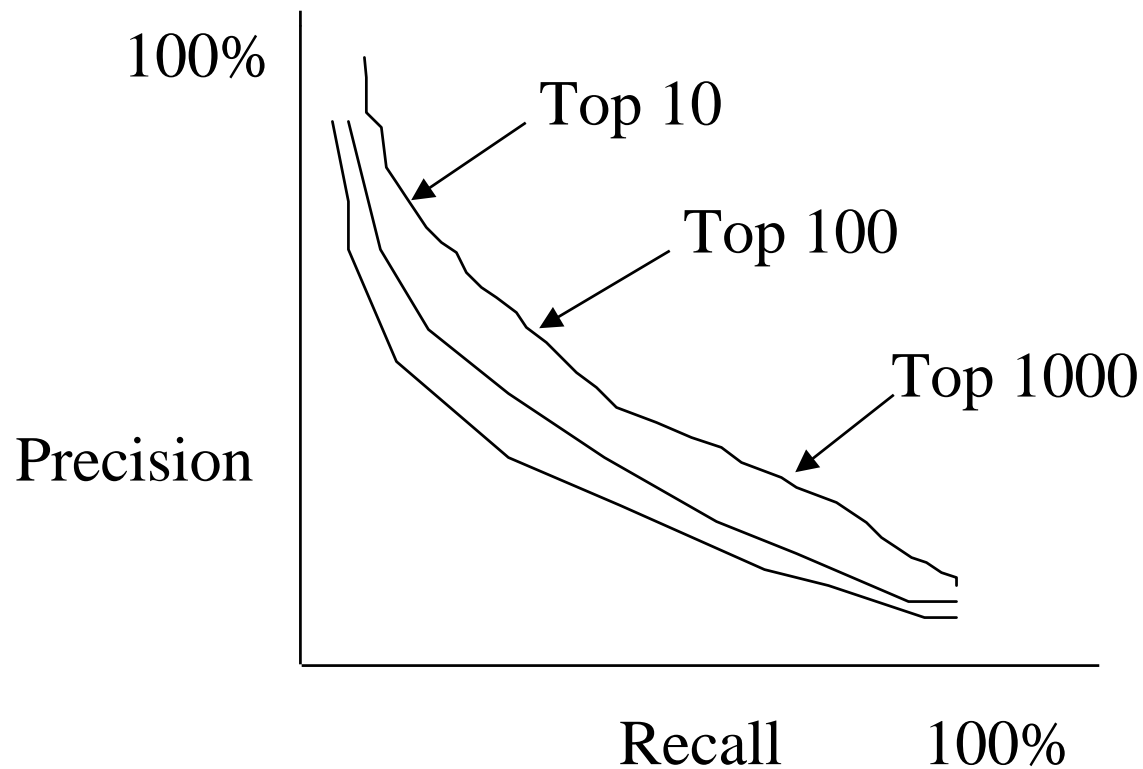
- Retrieve top 2000 documents.
- Five relevant documents exist and are also retrieved.

Document	DocId	Recall	Precision
– 100	A	.20	.01
– 200	B	.40	.01
– 500	C	.60	.006
– 1000	D	.80	.004
– 1500	E	1.0	.003

Recall / Precision Graph

- Compute precision at .1, .2, .3, ..., 1.0 levels of recall.
- Optimal graph would have straight line -- precision always at 1, recall always at 1.
- Typically, as recall increases, precision drops.

Precision/Recall Tradeoff



More Measures...

- *F Measure* – trade off precision versus recall

$$F \text{ Measure} = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

- Balanced *F Measure* considers equal weight on Precision and Recall:

$$F_{\beta=1} = \frac{2PR}{P + R}$$

More Measures...

- **MAP** (Mean average Precision)
 - **Average Precision** – Mean of the precision scores for a single query after each relevant document is retrieved, where relevant documents not retrieved have P of zero.
 - * Commonly 10-points of recall is used!
 - **MAP** is the mean of average precisions for a query batch
- **P@10** - Precision at 10 documents retrieved (in web searching). Problem: the cut-off 10 represents many different recall levels for different queries - also **P@1**. (**P@x**)
- **R-Precision** – Precision after R documents are retrieved; where R is number of relevant documents for a given query.

Example

- For Q1 - D2 and D5 are relevant:
D1, **D2**, D3 not judged, D4, **D5**, D6, D7, D8, D9, D10
- For Q2 - D1, D2 and D5 are only relevant:
D1, **D2**, D3 not judged, D4, **D5**, D6, D7, D8, D9, D10

P of Q1: 20%

AP of Q1: $(1/2 + 2/5)/2 = 0.45$

P of Q2: 30%

AP of Q2: $(1+1+3/5)/3 = 0.87$

MAP of system: $(AP_{q_1} + AP_{q_2})/2 = (0.45 + 0.87)/2 = 0.66$

P@1 for Q1: 0; P@1 for Q2: 100%;

R-Precision Q1: 50%; Q2: 67%

More Measures...

- **bpref** - (binary preference-based measure)
 - *Bpref* measure [2004], unlike MAP, P@10, and R-Precision, only uses information from judged documents.
 - A function of how frequently relevant documents are retrieved before non-relevant documents.

$$bpref = \frac{1}{R} \sum_r 1 - \frac{|n \text{ ranked higher than } r|}{R}$$

Measures (Cont'd)

[ACM SIGIR 2004]:

- When comparing systems over test collections with complete judgments, **MAP** and **bpref** are reported to be equivalent
- With incomplete judgments, **bpref** is shown to be more stable

bpref Example

- Retrieved result set with D2 and D5 being relevant:

D1

D2

D3 not judged

D4

D5

D6

D7

D8

D9

D10

R=2;

$$\text{bpref} = 1/2 [1 - (1/2)]$$

bpref Example

- Retrieved result set with D2 and D5 being relevant:

D1

D2

D3 not judged

D4 not judged

D5

D6

D7

D8

D9

D10

R=2;

$$\text{bpref} = 1/2 [(1 - 1/2) + (1 - 1/2)]$$

bpref Example

- D2, D5 and D7 are relevant:

D1

D2

D3 not judged

D4 not judged

D5

D6

D7

D8

D9

D10

R=3;

$$\text{bpref} = 1/3 [(1 - 1/3) + (1 - 1/3) + (1 - 2/3)]$$

bpref Example

- D2, D4, D6 and D9 are relevant:

D1

D2

D3

D4

D6

D7

D8

D9

D10

R=4;

$$\text{bpref} = 1/4 [(1 - 1/4) + (1 - 2/4) + (1 - 2/4)]$$

Search Tasks

- Precision-Oriented (such as in web search)
 - P@1
 - P@10
- Recall-Oriented (such as analyst task)
 - number of relevant documents that can be identified in a time frame. Usually 5 minutes time frame is chosen.

Evaluating Web Search Engines

- Dynamic environment (Facts):
 - Collection grows/changes rapidly and indices are constantly updated
 - User interests and popular queries change
 - Web queries are typically short (1-3 terms), thus difficult to capture users' need
 - Search algorithms are continually refined
 - Users only view top 10 results for 85% of their queries
 - Users do not revise their query after the first try for 75% of their queries
 - Majority of queries occur only a few times (55% occurs less than 5 times)
 - Top queries are changing over time too.

Evaluating Web Search Engines (Cont'd)

- Web is too large to calculate recall, thus need measures that are not recall-based
- Hundreds of millions of queries per day, thus need large sample of queries to represent the population of even one day
- Repeat evaluations frequently

Evaluating Various Search tasks

- TREC evaluation paradigm, using *Pooling*, has shown success for specific user task of *topical information (ad hoc)*.
- Other users tasks:
 - *Navigational*: finding specific sites
 - *Transactional*: finding specific item (buy books, etc.)
- ➔ Not dealing with set of relevant documents but with rather a single correct answer!

Known-item Search Evaluation

- Ranking the best site or item being searched
 - find a single known resource for a given query. Closer the rank of the item to the top, better for the user.
 - Evaluation Metric: **Mean Reciprocal Ranking (MRR)**
 - Weight of item (correct answer) in location 1 is 1
 - Weight of item in location n is 1/n

$$MRR = \frac{\sum_{q=1}^n \frac{1}{rankq}}{n}$$

Known-Item Search & MRR

$$MRR = \frac{\sum_{q=1}^n \frac{1}{rankq}}{n}$$

Example:

- MRR=0.25 means on average the system finds the known-item in position number 4 of result set.
- MRR= 0.75 means finding the item between ranks 1 and 2 on average.

Cost of Manual Evaluation

Search engines: 5

Queries: 300

Top documents: 20

Time to evaluate each result: 30 seconds (optimistic)

➔ $(300q * 20r * 5s) = 30,000$ results to evaluate

➔ 10.4 days to complete the task (not sleeping!)

➔ 31 days (8-hour working days) to complete

➔➔ Not scalable to dynamic env. such as Web!

(Research in progress!)