HPI Hasso Plattner Institut

IT Systems Engineering | Universität Potsdam

Search Engines
Chapter 8 – Evaluating Search Engines

9.7.2009

Felix Naumann

Evaluation

- Evaluation is key to building *effective* and *efficient* search engines.
  - Drives advancement of search engines
    - When intuition fails
  - Measurement usually carried out in controlled laboratory experiments
    - To control the many factors
  - *Online* testing can also be done
- Effectiveness: Measures ability to find right information
  - Compare ranking to user relevance feedback
- Efficiency: Measures ability to do this quickly
  - Measure time and space requirements
- Effectiveness, efficiency, and *cost* are related
  - If we want a particular level of effectiveness and efficiency, this will determine the cost of the system configuration.
  - Efficiency and cost targets may impact effectiveness.
- Usual approach: Find techniques to improve effectiveness, then find fast implementations

Efficiency vs. Effectiveness vs. Cost

grep

Cheap, efficient, but ineffective

Library of Congress staff

Effective, but inefficient, and expensive

http://www.informationliteracy.org/builder/view/924

Overview

- Evaluation Corpus
- Logging
- Effectiveness Metrics
  - Efficiency Metrics
- Training & Testing

Evaluation Corpus

- Goals
  - Provide fixed experimental setting and ata
  - Ensure fair and repeatable experiments
- Text corpus is without queries and relevance judgement
  - Linguistics, machine translation, speech recognition
- Cranfield experiments
  - Test collection of
    - Documents
    - Queries
    - Relevance judgements
- Coprora change (in particular grow) over time
  - CACM, AP, GOV2 as examples

Evaluation Corpora

- CACM
  - Titles and abstracts from the Communications of the ACM from 1958-1979.
  - Queries and relevance judgments generated by computer scientists.
- AP
  - Associated Press newswire documents from 1988-1990 (from TREC disks 1-3).
  - Queries are the title fields from TREC topics 51-150. Topics and relevance judgments generated by government information analysts.
- GOV2
  - Web pages crawled from Web sites in the .gov domain during early 2004.
  - Queries are the title fields from TREC topics 701-850. Topics and relevance judgments generated by government analysts.

Test Collections

Collection	Number of documents	Size	Average number of words/doc.
CACM	3,204	2.2 Mb	64
AP	242,918	0.7 Gb	474
GOV2	25,205,179	426 Gb	1073

Tiny

Collection	Number of queries	Average number of words/query	Average number of relevant docs/query
CACM	64	13.0	16
AP	100	4.3	220
GOV2	150	3.1	180

Long queries

TREC Topic Example

```
<top>
        <num> Number: 794
        <title> pet therapy
        <desc> Description:
        How are pets or animals used in therapy for humans
        and what are  the benefits?
        <narr> Narrative:
        Relevant documents must include details of how pet-
        or animal-assisted therapy is or has been used.
        Relevant details include information about pet
        therapy programs, descriptions of the circumstances
        in which pet therapy is used, the benefits of this
        type of therapy, the degree of success of this
        therapy, and any laws or regulations governing it.
</top>
```

Short query

Long query

Criteria for relevance

Relevance Judgments

- Obtaining relevance judgments is an expensive, time-consuming process
  - Who does it?
  - What are the instructions?
  - What is the level of agreement?
- TREC judgments
  - Depend on task being evaluated
  - Generally binary
    - Thus, all documents containing same useful information are judged relevant: Focus on topical relevance
  - Sometimes levels of relevance:
    Not relevant | relevant | highly relevant
  - Agreement good because of “narrative”

Pooling

- Exhaustive judgments for all documents in a collection is not practical
- Pooling technique is used in TREC
  1. Top *k results (for TREC, k varied between 50 and* 200) from the rankings obtained by different search engines (or retrieval algorithms) are merged into a pool.
  2. Duplicates are removed.
  3. Documents are presented in some random order to the relevance judges.
- Produces a large number of relevance judgments for each query, although still incomplete.
  - Problem for new retrieval algorithms that find different documents

Overview

- Evaluation Corpus
- Logging
- Effectiveness Metrics
    - Efficiency Metrics
- Training & Testing

Query Logs

- Used for both tuning and evaluating search engines
  - also for various techniques such as query suggestion
- Many more queries than for test collections
  - But less precise
- Problem: Privacy (especially when shared)
- Typical contents
  - User identifier or user session identifier
    - Login, toolbar, cookie, …
  - Query terms – stored exactly as user entered
  - Ordered list of URLs of results, their ranks on the result list, and whether they were clicked on
  - Timestamp(s) – records the time of user events such as query submission and result-clicks

Query Logs

- Clicks are not relevance judgments.
  - Although they are highly correlated
  - Biased by a number of factors:
    rank on result list, snippet, general popularity
- Other indicators
  - Dwell time: time spent on a clicked result
  - Search exit action: result page, print page, timeout, enter other URL, …
- Can use clickthrough data to predict *preferences* between pairs of documents
  - Appropriate for tasks with multiple levels of relevance, focused on user relevance
  - Various strategies used to generate preferences

Example Click Policy

- *Skip Above and Skip Next*
  - Click data
    d_1
    d_2
    d_3 (clicked)
    d_4
  - Generated preferences

$$d_3 > d_2$$
$$d_3 > d_1$$
$$d_3 > d_4$$

Query Logs

- Click data can be aggregated to remove noise
- *Click distribution* information
  - Can be used to identify clicks that have a higher frequency than would be expected
  - High correlation with relevance

- *Click deviation CD(d, p)* for a result *d* in position *p*:

$$CD(d,p) = O(d,p) - E(p)$$

  - *O(d,p)*: observed click frequency for a document in a rank position p *over all instances of a given query*
  - *E(p)*: expected click frequency at rank p *averaged across all queries*
  - Use to filter clicks for preference-generation policies

Overview

- Evaluation Corpus
- Logging
- Effectiveness Metrics
  - Efficiency Metrics
- Training & Testing

Effectiveness Measures

- *A* is set of relevant documents
  - But we may not find all
- *B* is set of retrieved documents
  - But not all of them are relevant

	Relevant	Non-Relevant
Retrieved	$A \cap B$	$\overline{A} \cap B$
Not Retrieved	$A \cap \overline{B}$	$\overline{A} \cap \overline{B}$

$$Recall = \frac{|A \cap B|}{|A|}$$

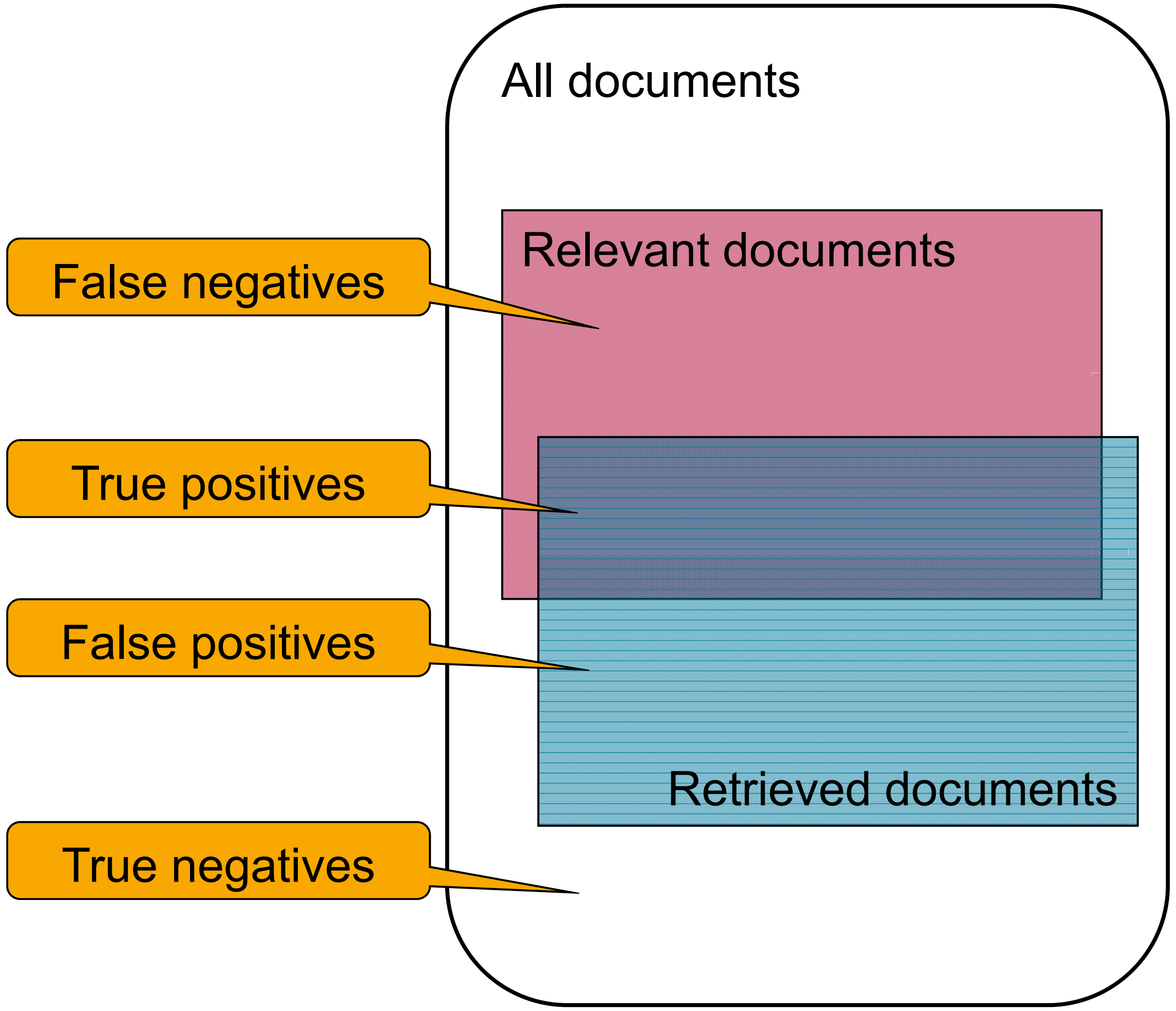
Proportion of relevant documents that are retrieved

$$Precision = \frac{|A \cap B|}{|B|}$$

Proportion of retrieved documents that are relevant

- Works for Boolean retrieval (for now)
- Assumes we are interested in ALL relevant documents

Precision & Recall
(≈ correctness and completeness)



$$\text{Precision} = \frac{\text{True positives}}{\text{Retrieved documents}}$$

$$\text{Recall} = \frac{\text{True positives}}{\text{Relevant documents}}$$

Classification Errors

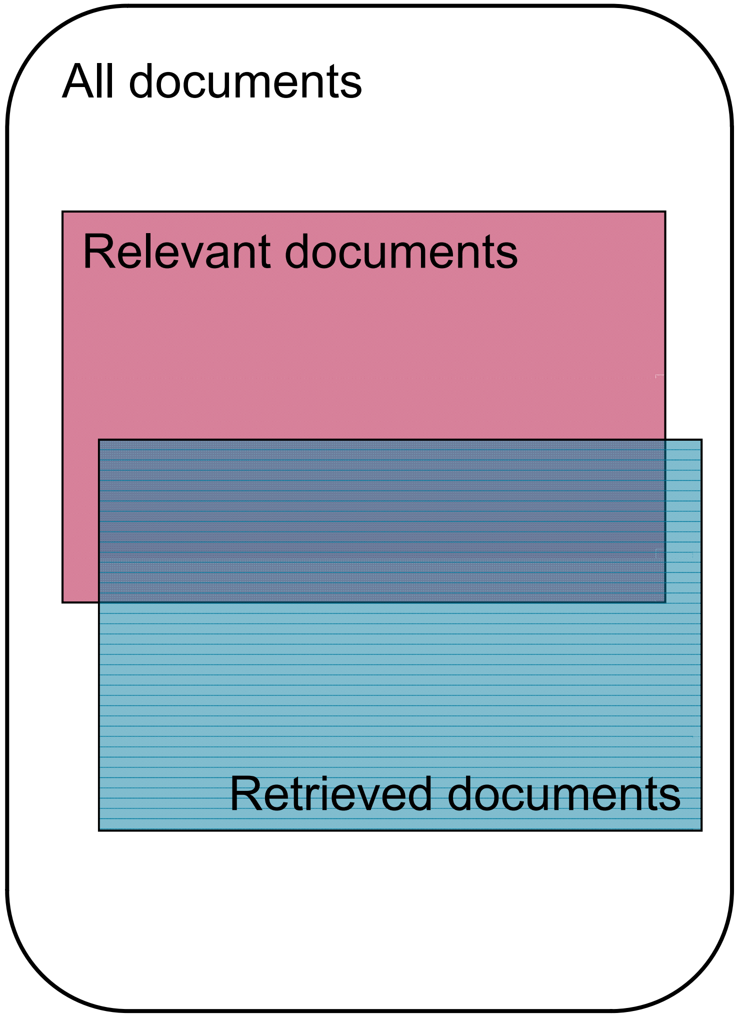
- *False Positive* (Type I error)
  - A non-relevant document is retrieved: $\overline{A} \cap B$

$$Fallout = \frac{|\overline{A} \cap B|}{|\overline{A}|}$$

  - Proportion of non-relevant documents retrieved
- *False Negative* (Type II error)
  - A relevant document is not retrieved: $A \cap \overline{B}$
  - 1- *Recall*
- *Precision* is used when probability that a positive result is correct is important
  - More meaningful to user
  - Fallout will always be tiny, because of so many irrelevant documents.

Perfect algorithms

- Find algorithm that maximizes precision.
  - Or minimizes classification errors in general (false positives and false negatives)
  - Return nothing!
- Find algorithm that maximizes recall.
  - Return everything!



F Measure

- *Harmonic mean* of recall and precision

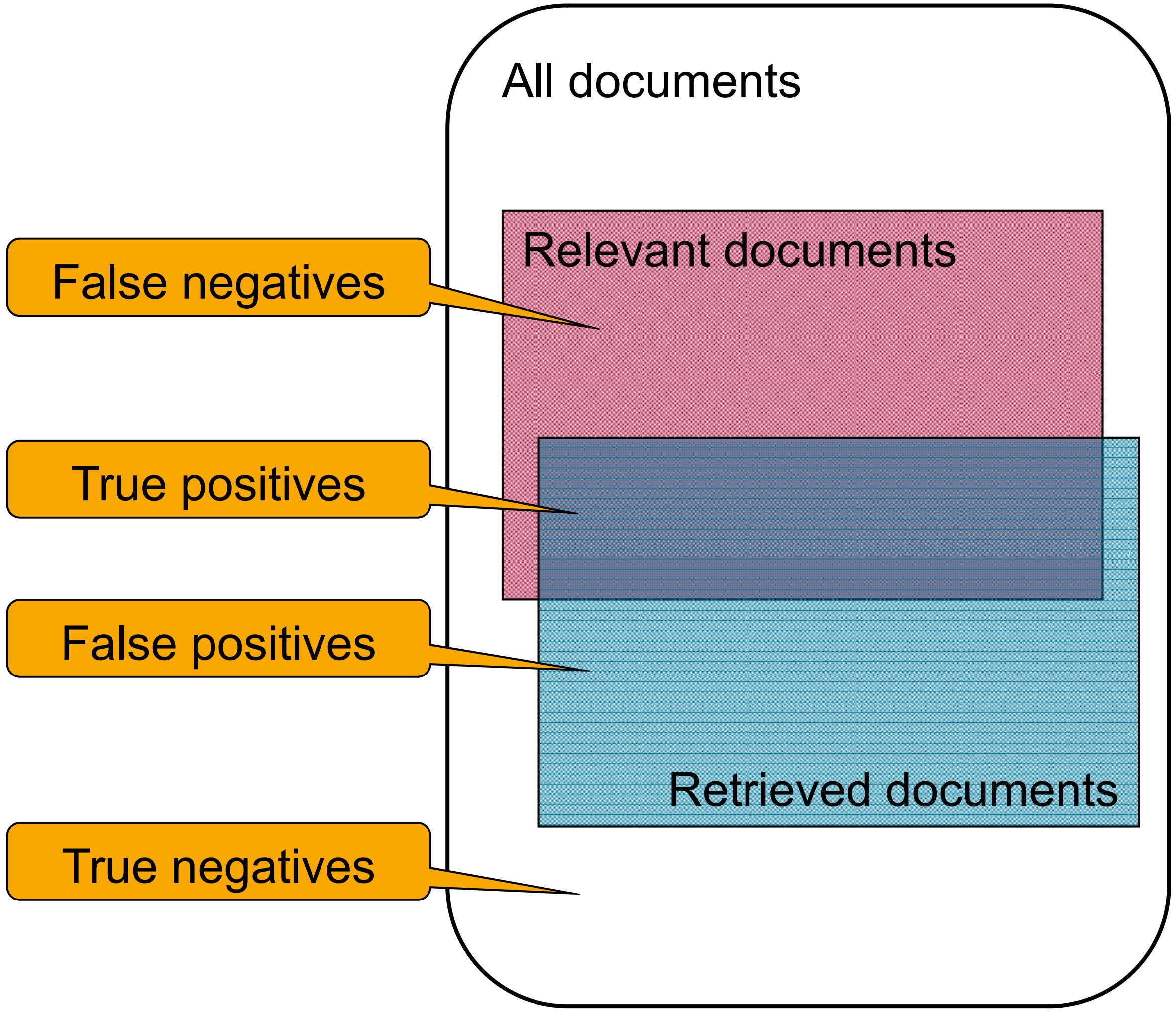
$$F = \frac{1}{\frac{1}{2}\left(\frac{1}{R} + \frac{1}{P}\right)} = \frac{2RP}{R+P}$$

  - Harmonic mean emphasizes the importance of small values, whereas the arithmetic mean is affected more by outliers that are unusually large.
- More general form: Weighted harmonic mean

$$F_\alpha = \frac{RP}{\alpha R + (1-\alpha)P}$$

  - Thus, harmonic mean is $F_{1/2}$

Precision & Recall
(≈ correctness and completeness)

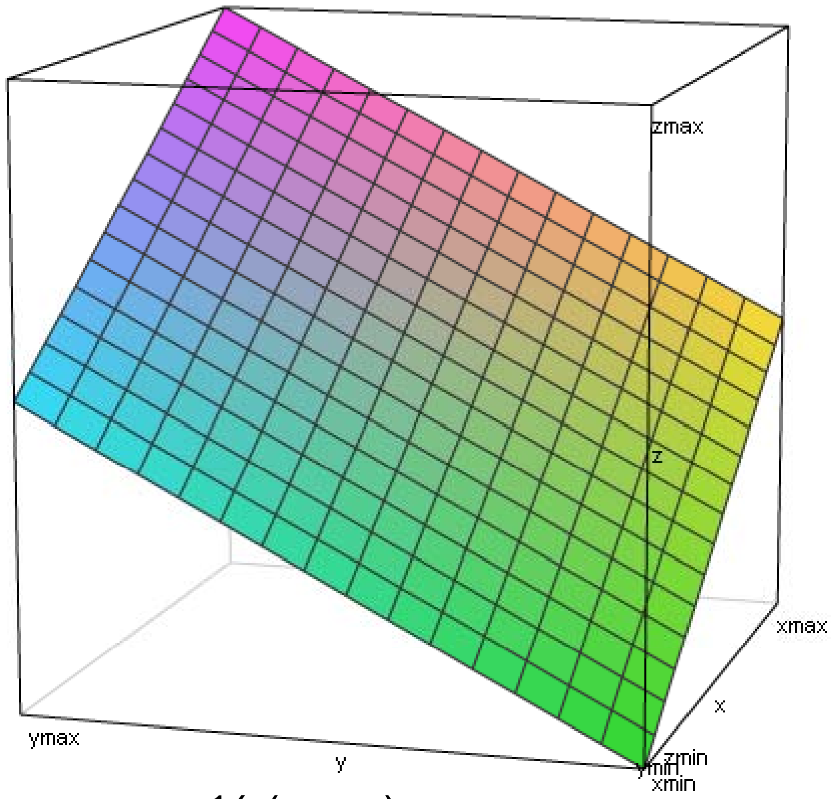


$$\text{Precision} = \frac{\text{True positives}}{\text{Retrieved documents}}$$

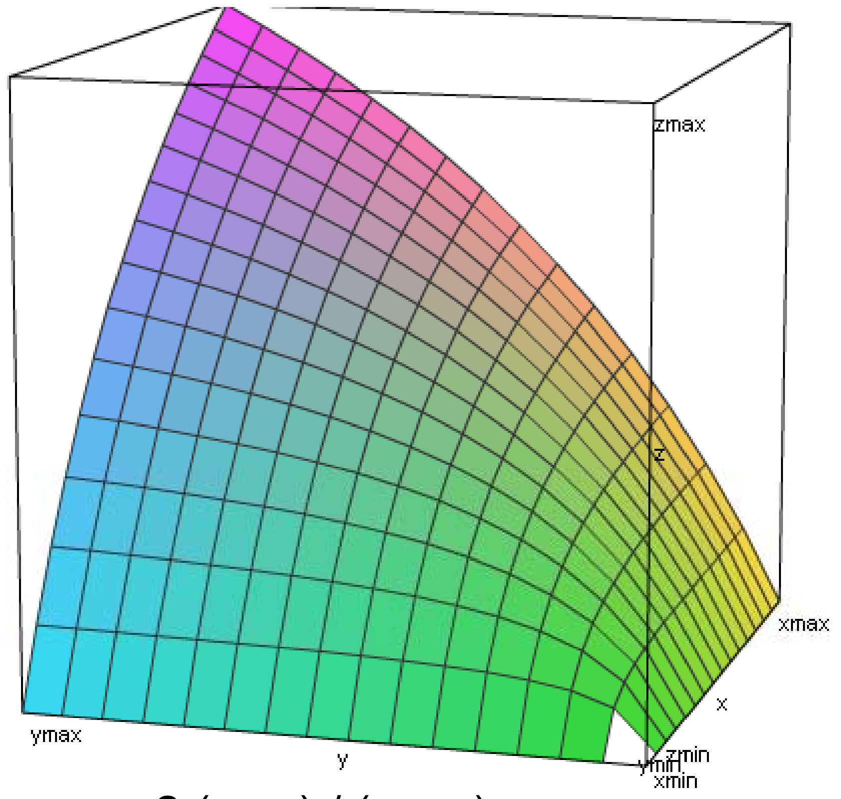
$$\text{Recall} = \frac{\text{True positives}}{\text{Relevant documents}}$$

$$\text{F-Measure} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Arithmetic mean („Average“) vs. Harmonic mean („F-Measure“)



$z = ½ (x + y)$



$z = 2 (x · y) / (x + y)$

Ranking Effectiveness

- Problem: Evaluate ranking, not just Boolean classification
- Idea: Calculate precision and recall at every rank position

■ ■ ■ ■ ■ ■ = the relevant documents

Ranking #1

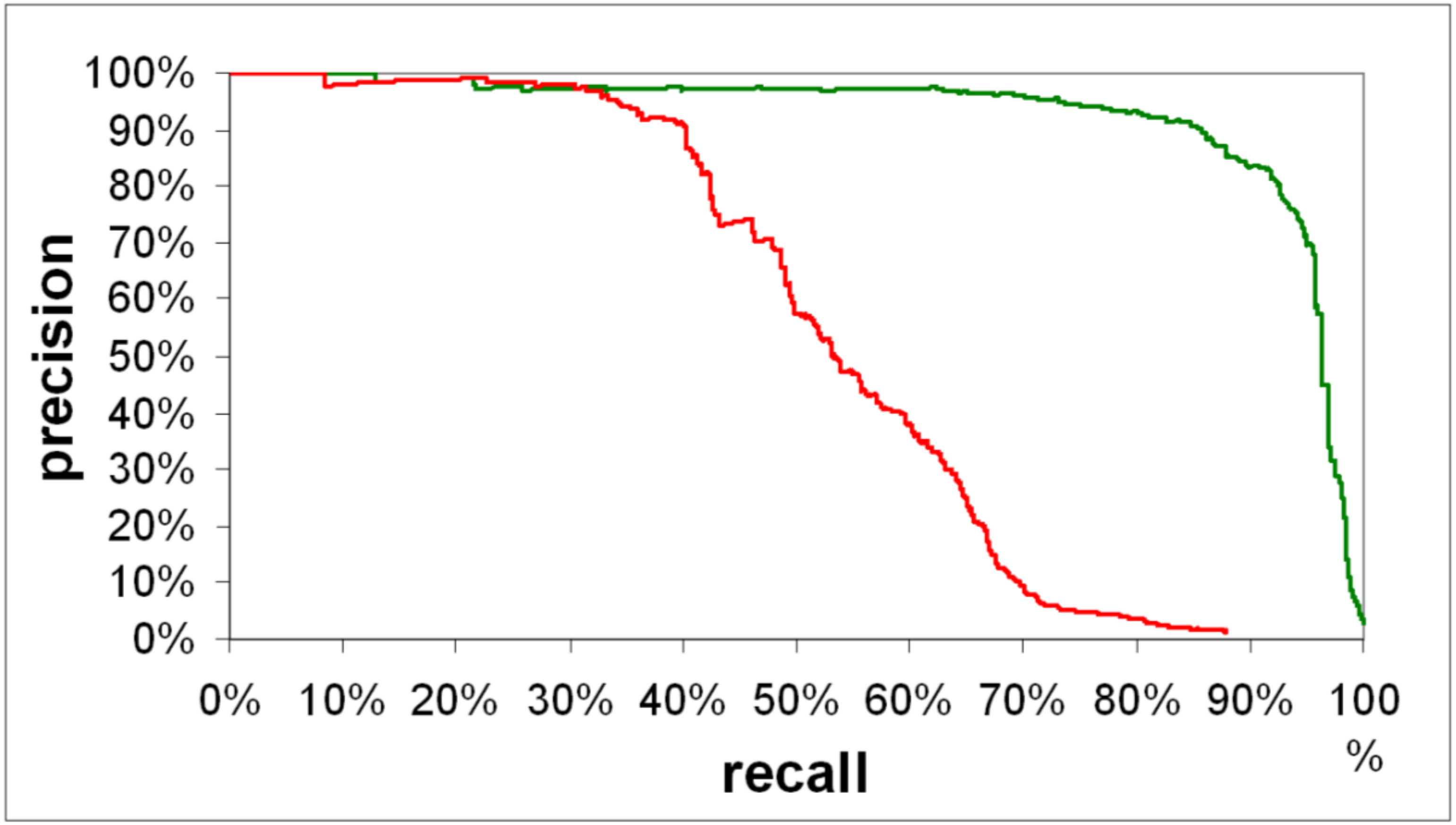
	■	□	■	■	■	■	□	□	□	■
Recall	0.17	0.17	0.33	0.5	0.67	0.83	0.83	0.83	0.83	1.0
Precision	1.0	0.5	0.67	0.75	0.8	0.83	0.71	0.63	0.56	0.6

Same recall and precision

Ranking #2

	□	■	□	□	■	■	■	□	■	■
Recall	0.0	0.17	0.17	0.17	0.33	0.5	0.67	0.67	0.83	1.0
Precision	0.0	0.5	0.33	0.25	0.4	0.5	0.57	0.5	0.56	0.6

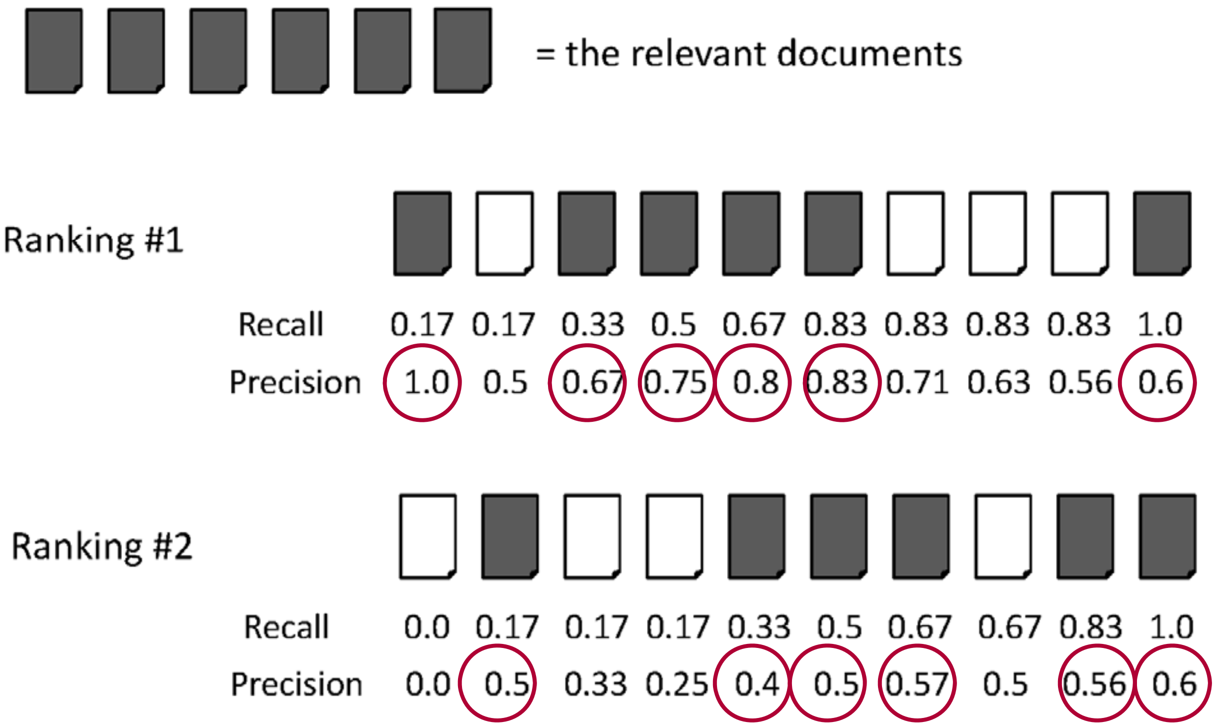
Precision / Recall diagrams



Summarizing a Ranking

- Problem: Long lists are unwieldy and difficult to compare.
- Three ideas
  1. Calculating recall and precision at small number of fixed rank positions.
     - Compare two rankings: If precision at position p is higher, recall is higher too.
     - "Precision at rank *p*"
       - Usually, *p*=10 or *p*=20
     - Ignores ranking after *p*; ignores ranking within 1 to *p*.
  2. Calculating precision at standard recall levels, from 0.0 to 1.0 in increments of 0.1
     - Requires *interpolation*
     - Later
  3. Averaging the precision values from the rank positions where a relevant document was retrieved

Average Precision



Ranking #1

Recall	0.17	0.17	0.33	0.5	0.67	0.83	0.83	0.83	0.83	1.0
Precision	1.0	0.5	0.67	0.75	0.8	0.83	0.71	0.63	0.56	0.6

Ranking #2

Recall	0.0	0.17	0.17	0.17	0.33	0.5	0.67	0.67	0.83	1.0
Precision	0.0	0.5	0.33	0.25	0.4	0.5	0.57	0.5	0.56	0.6

$$\text{Ranking \#1: } (1.0 + 0.67 + 0.75 + 0.8 + 0.83 + 0.6)/6 = 0.78$$

$$\text{Ranking \#2: } (0.5 + 0.4 + 0.5 + 0.57 + 0.56 + 0.6)/6 = 0.52$$

- Advantage: Reflects goal of finding all relevant documents but emphasizes top ranked documents

Averaging Across Queries

- Problem: Evaluate ranking algorithm, not just one ranking

■ ■ ■ ■ ■ = relevant documents for query 1

Ranking #1	■	□	■	□	□	■	□	□	■	■
Recall	0.2	0.2	0.4	0.4	0.4	0.6	0.6	0.6	0.8	1.0
Precision	1.0	0.5	0.67	0.5	0.4	0.5	0.43	0.38	0.44	0.5

■ ■ ■ = relevant documents for query 2

Ranking #2	□	■	□	□	■	□	■	□	□	□
Recall	0.0	0.33	0.33	0.33	0.67	0.67	1.0	1.0	1.0	1.0
Precision	0.0	0.5	0.33	0.25	0.4	0.33	0.43	0.38	0.33	0.3

Averaging

- Each ranking produces average precision
  - Take average of those numbers
- *Mean Average Precision* (MAP) (= average average precision)
  - Summarize rankings from multiple queries by averaging average precision
  - Most commonly used measure in research papers
  - Assumes user is interested in finding many relevant documents for each query
  - Requires many relevance judgments in text collection
- Later: Recall-precision graphs are also useful summaries

MAP

■ ■ ■ ■ ■ = relevant documents for query 1

Result #1	■	□	■	□	□	■	□	□	■	■
Recall	0.2	0.2	0.4	0.4	0.4	0.6	0.6	0.6	0.8	1.0
Precision	1.0	0.5	0.67	0.5	0.4	0.5	0.43	0.38	0.44	0.5

■ ■ ■ = relevant documents for query 2

Result #2	□	■	□	□	■	□	■	□	□	□
Recall	0.0	0.33	0.33	0.33	0.67	0.67	1.0	1.0	1.0	1.0
Precision	0.0	0.5	0.33	0.25	0.4	0.33	0.43	0.38	0.33	0.3

$$\textit{average precision query 1} = (1.0 + 0.67 + 0.5 + 0.44 + 0.5)/5 = 0.62$$
$$\textit{average precision query 2} = (0.5 + 0.4 + 0.43)/3 = 0.44$$

$$\textit{mean average precision} = (0.62 + 0.44)/2 = 0.53$$

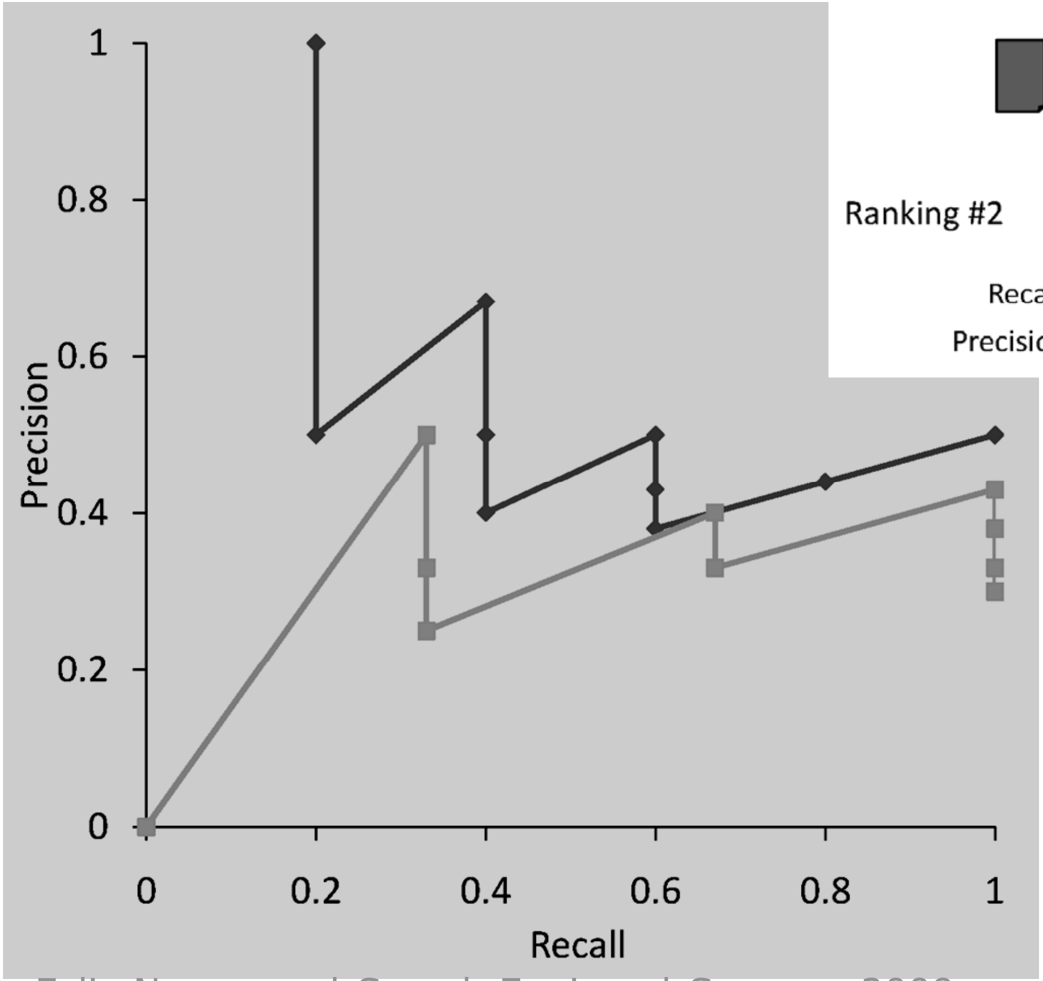
Recall-Precision Graph

■ ■ ■ ■ ■ = relevant documents for query 1

Ranking #1	■	□	■	□	□	■	□	□	■	■
Recall	0.2	0.2	0.4	0.4	0.4	0.6	0.6	0.6	0.8	1.0
Precision	1.0	0.5	0.67	0.5	0.4	0.5	0.43	0.38	0.44	0.5

■ ■ ■ = relevant documents for query 2

Ranking #2	□	■	□	□	■	□	■	□	□	□
Recall	0.0	0.33	0.33	0.33	0.67	0.67	1.0	1.0	1.0	1.0
Precision	0.0	0.5	0.33	0.25	0.4	0.33	0.43	0.38	0.33	0.3



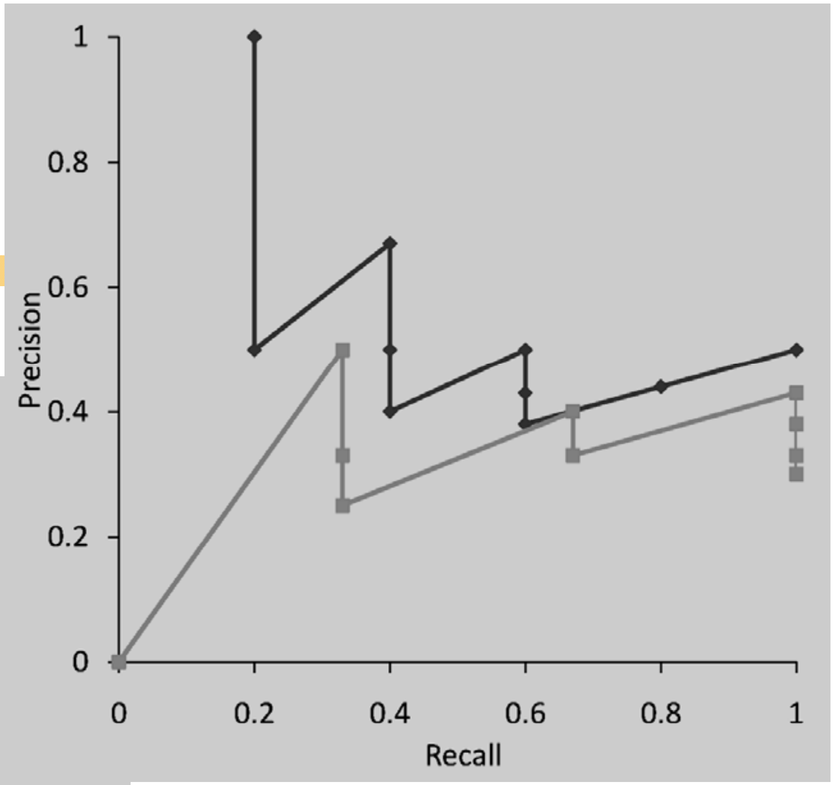
Interpolation

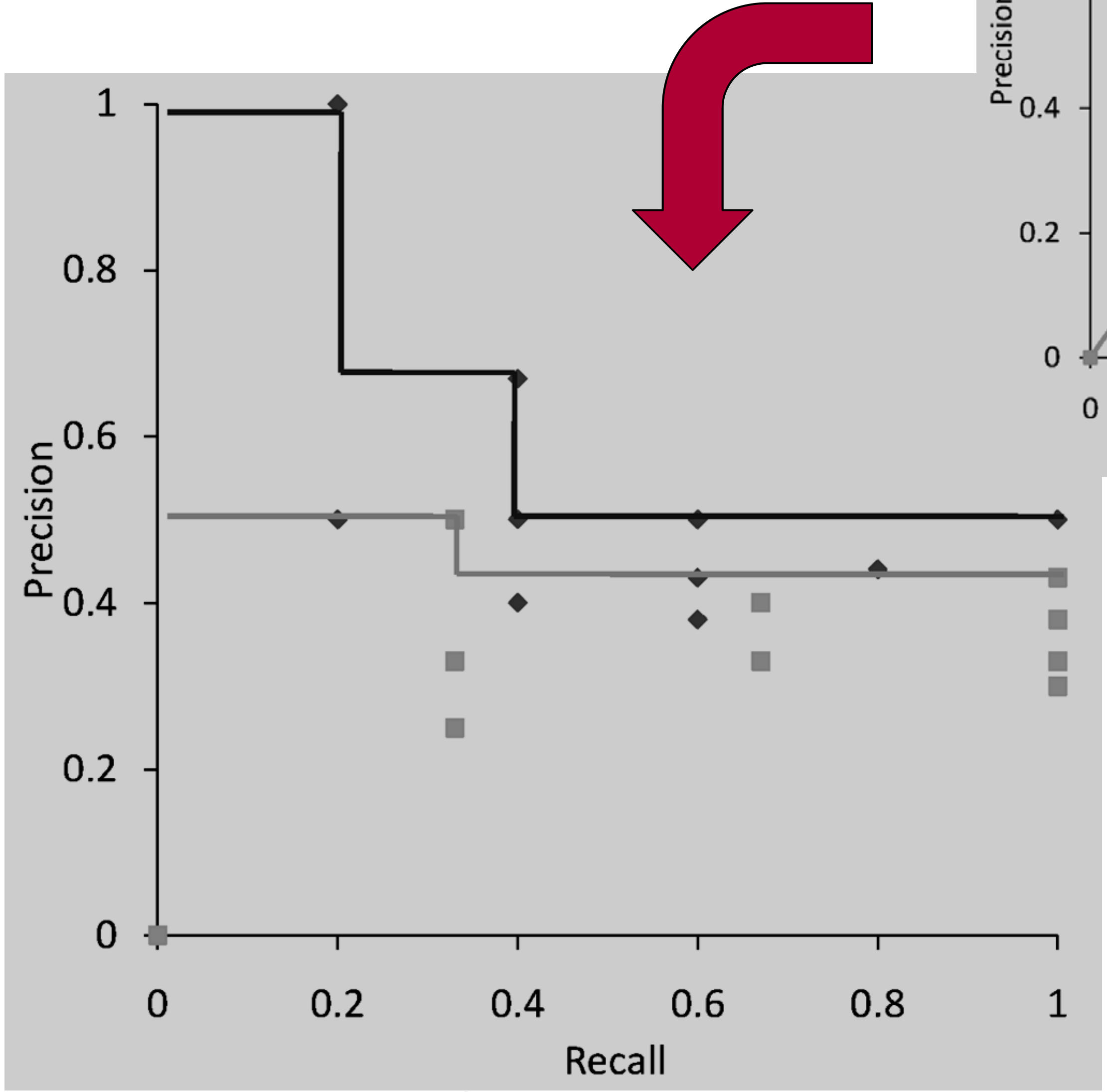
- Problem: Graphs have different shapes and are difficult to compare.
- To average graphs, calculate precision at standard recall levels:

$$P(R) = \max\{P' : R' \geq R \wedge (R', P') \in S\}$$

  - where *S* is the set of observed (*R*,*P*) points
- Defines precision at a recall level as the *maximum* precision observed in any recall-precision point at a higher recall level
  - Produces a step function
  - Defines precision at recall 0.0

Interpolation



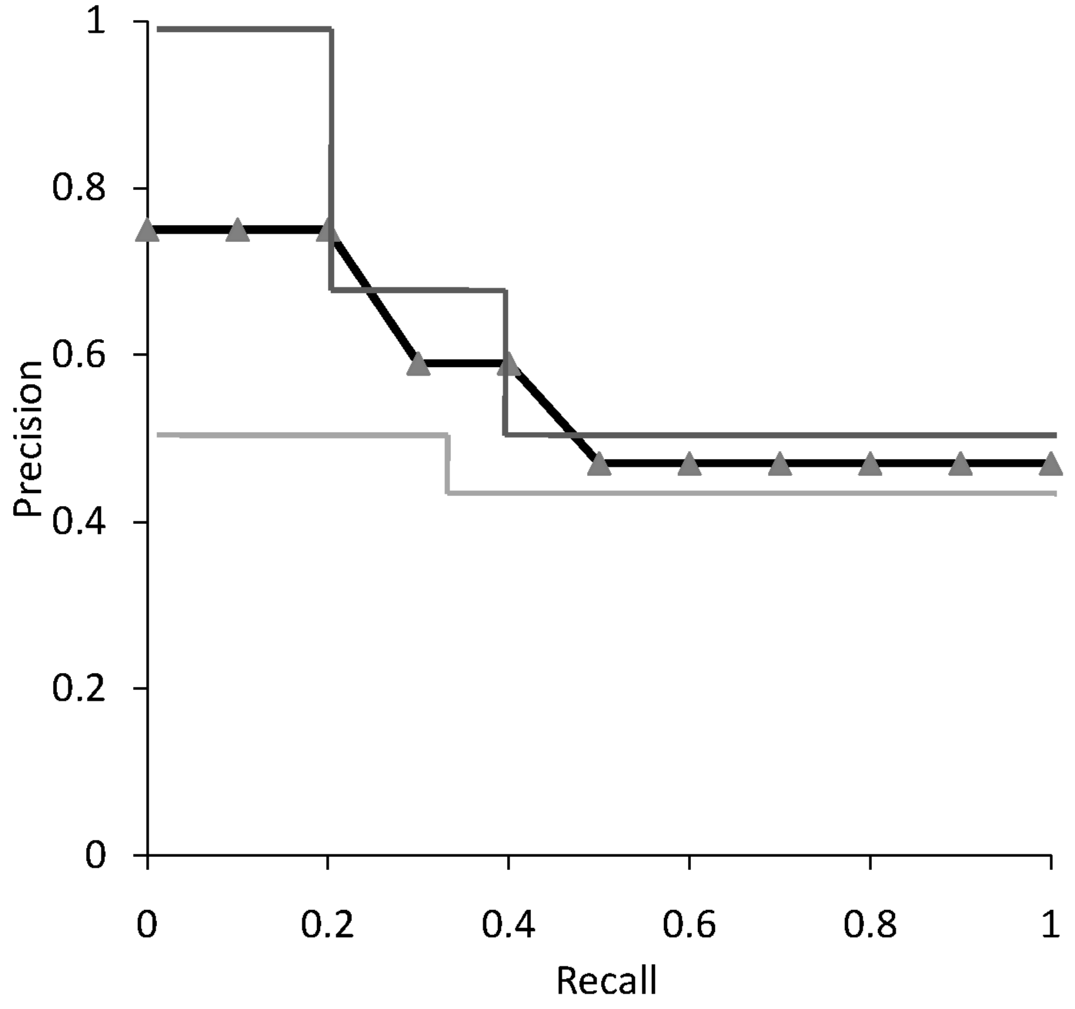


Average Precision at Standard Recall Levels

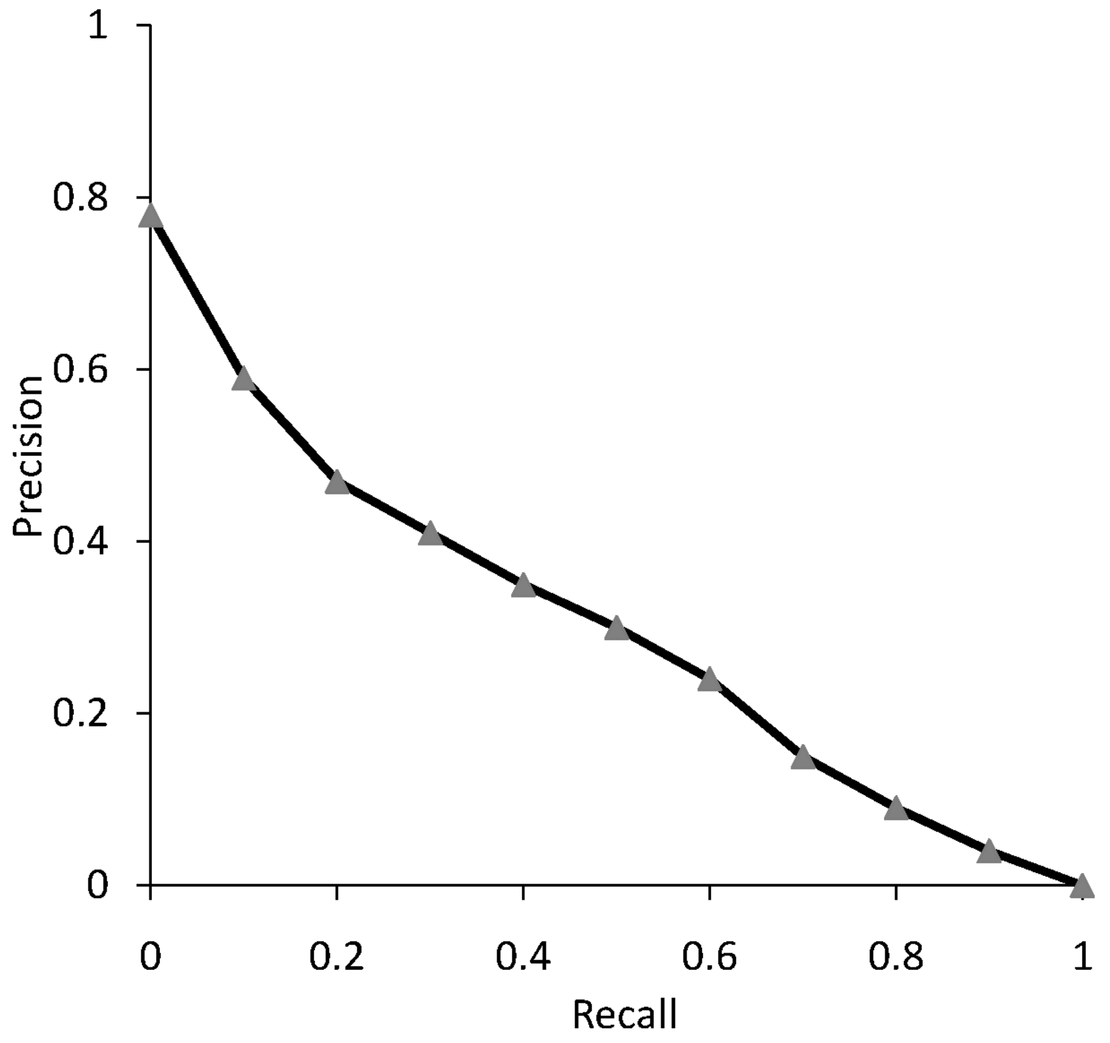
Recall	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Ranking 1	1.0	1.0	1.0	0.67	0.67	0.5	0.5	0.5	0.5	0.5	0.5
Ranking 2	0.5	0.5	0.5	0.5	0.43	0.43	0.43	0.43	0.43	0.43	0.43
Average	0.75	0.75	0.75	0.59	0.47	0.47	0.47	0.47	0.47	0.47	0.47

- Recall-precision graph plotted by simply joining the average precision points at the standard recall levels

Average Recall-Precision Graph



Graph for 50 Queries (becomes smoother)

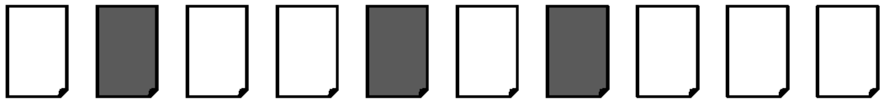


Focusing on Top Documents

- Users tend to look at only the top part of the ranked result list to find relevant documents.
  - First 1 or 2 result pages
- Some search tasks have only one relevant document
  - e.g., navigational search, question answering
- Recall not appropriate
  - Instead need to measure how well the search engine does at retrieving relevant documents at very high ranks

Focusing on Top Documents

- Precision at Rank *p*
  - *p* typically 5, 10, 20
  - Easy to compute, average over queries, understand
  - Not sensitive to rank positions less than *p*
    - Single relevant document can be ranked anywhere.
- Reciprocal Rank
  - Reciprocal (Kehrwert) of the rank at which the first relevant document is retrieved
  - *Mean Reciprocal Rank (MRR)* is the average of the reciprocal ranks over a set of queries
  - Very sensitive to rank position, regards only first relevant document
  - Reciprocal rank: 1/2



Discounted Cumulative Gain

- Popular measure for evaluating web search and related tasks
- Two assumptions
  1. Highly relevant documents are more useful than marginally relevant document
  2. The lower the ranked position of a relevant document, the less useful it is for the user, since it is less likely to be examined
- Uses *graded relevance* as a measure of the usefulness, or *gain,* from examining a document
- Gain is accumulated starting at the top of the ranking and may be reduced, or *discounted*, at lower ranks
- Typical discount is 1/*log(rank)*
  - With base 2, the discount at rank 4 is 1/2, and at rank 8 it is 1/3

Discounted Cumulative Gain

- *DCG* is the total gain accumulated at a particular rank *p*:

$$DCG_p = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2 i}$$

  - Where rel_i is graded relevance of document at rank *i*.
  - Can use binary values (0,1)
  - Can use “Bad” = 0 to “Perfect” = 5
- Alternative formulation:

$$DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i}-1}{log(1+i)}$$

  - Used by some web search companies
  - Same for binary grades
  - Emphasis on retrieving highly relevant documents

DCG Example

- 10 ranked documents judged on 0-3 relevance scale (gain):

  3, 2, 3, 0, 0, 1, 2, 2, 3, 0
- Discounted gain

  3, 2/1, 3/1.59, 0, 0, 1/2.59, 2/2.81, 2/3, 3/3.17, 0

  = 3, 2, 1.89, 0, 0, 0.39, 0.71, 0.67, 0.95, 0
- Discounted Cumulative Gain at each position

  3, 5, 6.89, 6.89, 6.89, 7.28, 7.99, 8.66, 9.61, 9.61

- DCG numbers are averaged across a set of queries at specific rank values
  - e.g., DCG at rank 5 is 6.89 and at rank 10 is 9.61

Normalized DCG

- DCG values are often *normalized* by comparing the DCG at each rank with the DCG value for the *perfect ranking.*
  - Makes averaging easier for queries with different numbers of relevant documents
- Example
  - Original result 3, 2, 3, 0, 0, 1, 2, 2, 3, 0
  - Original DCG values
    3, 5, 6.89, 6.89, 6.89, 7.28, 7.99, 8.66, 9.61, 9.61
  - Perfect ranking for the ten results:3, 3, 3, 2, 2, 2, 1, 0, 0, 0
  - Ideal DCG values:
    3, 6, 7.89, 8.89, 9.75, 10.52, 10.88, 10.88, 10.88, 10.88
  - NDCG values (divide actual by ideal):
    1, 0.83, 0.87, 0.76, 0.71, 0.69, 0.73, 0.8, 0.88, 0.88
    - NDCG $\leq$ 1 at any rank position

Using Preferences

- Idea: Use preferences (e.g., from query logs) to evaluate ranking
  - Compare preference ranking to actual ranking
- Two rankings described using preferences can be compared using the *Kendall tau coefficient (τ):*

$$\tau = \frac{P - Q}{P + Q}$$

  - *P* is the number of preferences that agree and *Q* is the number that disagree
  - τ = 1: all preferences agree
  - τ = -1: all preferences disagree
  - Use only known set of preferences (partial ranking)
- For preferences derived from binary relevance judgments, can use *BPREF*

BPREF

- BPREF = Binary PREFerence
- For a query with *R* relevant documents, only the first *R* non-relevant documents are considered

$$BPREF = \frac{1}{R}\sum_{d_r}(1 - \frac{N_{d_r}}{R})$$

  - d_r is a relevant document, and N_{dr} gives the number of non-relevant documents that are ranked higher than d_r.
  - Equivalent to using R x R preferences and N_{dr} counts number of disagreeing preferences
- Alternative definition similar to Kendall tau:

$$BPREF = \frac{P}{P+Q}$$

Overview

- Evaluation Corpus
- Logging
- Effectiveness Metrics
  - Efficiency Metrics
- Training & Testing

Efficiency Metrics

- Elapsed indexing time
  - Measures the amount of time necessary to build a document index on a particular system.
- Indexing processor time
  - Measures the CPU seconds used in building a document index. This is similar to elapsed time, but does not count time waiting for I/O or speed gains from parallelism.
- Query throughput
  - Number of queries processed per second.
- Query latency
  - The amount of time a user must wait after issuing a query before receiving a response, measured in milliseconds. This can be measured using the mean, but is often more instructive when used with the median or a percentile bound.
- Indexing temporary space
  - Amount of temporary disk space used while creating an index.
- Index size
  - Amount of storage necessary to store the index files.

Query throughput

- Most popular metric
- Reflects common problems
  - Capacity planning: Determine if more hardware is necessary
  - Determine whether system meets current requirements
- But: Latency not considered
  - Less than 150ms = instantaneous
- Latency and throughput are conflicting goals
  - Personal chef vs. Restaurant
- Introducing latency allows system to optimize
  - Reorganize queries for faster batch execution
- Search engines: Throughput is not a variable
  - Every query must be handled
  - Optimize for latency and hardware cost

Overview

- Evaluation Corpus
- Logging
- Effectiveness Metrics
  - Efficiency Metrics
- Training & Testing

Significance Tests

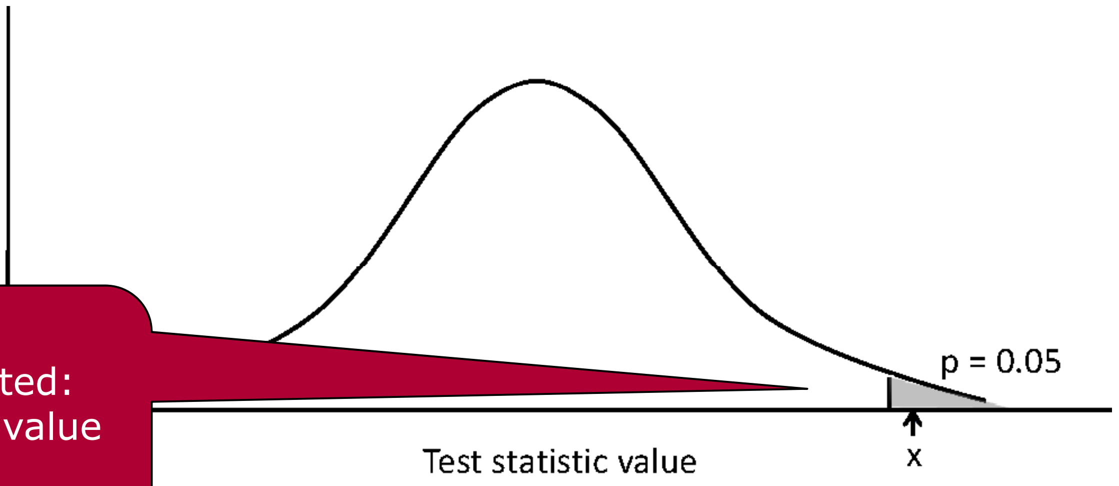
- Given the results from a number of queries, how can we conclude that ranking algorithm A is better than algorithm B?
  - Using some effectiveness metric
- A significance test enables us to reject the *null hypothesis* (no difference) in favor of the *alternative hypothesis* (B is better than A).
  - A is baseline, B is new and improved version
  - The *power* of a test is the probability that the test will reject the null hypothesis correctly.
  - Increasing the number of queries in the experiment also increases power of test.
- Example: If B is better than A in 90% of 200 queries, how confident can we be that B is better in general?
  - For that effectiveness measure
- Significance test can yield false positives and false negatives.

Significance Tests – Procedure

1. Compute the effectiveness measure for every query for both rankings.
2. Compute a test statistic based on a comparison of the effectiveness measures for each query. The test statistic depends on the significance test, and is simply a quantity calculated from the sample data that is used to decide whether or not the null hypothesis should be rejected.
3. The test statistic is used to compute a *P-value*, which is the probability that a test statistic value at least that extreme could be observed if the null hypothesis were true. Small P-values suggest that the null hypothesis may be false.
4. The null hypothesis (no difference) is rejected in favor of the alternate hypothesis (i.e. B is more effective than A) if the P-value is $\leq \alpha$, the *significance level*. Values for α are small, typically .05 and .1, to reduce the chance of false negatives.

Significance Tests

- Summary: If the probability of getting a specific test statistic value is very small assuming the null hypothesis is true, we reject that hypothesis and conclude that ranking algorithm B is more effective than the baseline algorithm A.
- One-sided test (one-tailed test=) because we want to show only that B is better than A.
- Distribution for the possible values of a test statistic assuming the null hypothesis
  - shaded area is *region of rejection*



If test yields value *x*, null hypothesis would be rejected: Probability of getting that value (or higher) is less than significance level of 0.05.

Example Experimental Results

Query	A	B	B-A
1	25	35	10
2	43	84	41
3	39	15	-24
4	75	75	0
5	43	68	25
6	15	85	70
7	20	80	60
8	52	50	-2
9	49	58	9
10	50	75	25

E.g., average precision, scaled to 0-100

t-Test

- Assumption is that the difference between the effectiveness values is a sample from a normal distribution
- Null hypothesis is that the mean of the distribution of differences is zero
- Test statistic $$t = \frac{\overline{B-A}}{\sigma_{B-A}} . \sqrt{N}$$
  - $\overline{B-A}$ is average difference
  - σ_{B-A} is standard deviation of differences
  - *N* is number of queries
  - for the example,

$$\overline{B-A} = 21.4,\ \sigma_{B-A} = 29.1,\ t = 2.33,\ \text{p-value}{=}.02$$

Probability that a test statistic value at least that extreme could be observed if the null hypothesis were true.

  - 0.02 < 0.05, thus significant.

Sign Test

- Ignores magnitude of differences
- Null hypothesis for this test is that
  - P(B > A) = P(A > B) = ½
  - Number of pairs where B is "better" than A would be the same as the number of pairs where A is "better" than B
  - Danger: Even only small (non-noticeable) differences count.
    - Use threshold of 5% difference
- Test statistic is number of pairs where B > A
- For example data
  - Test statistic is 7, p-value = 0.17
  - Cannot reject null hypothesis

Chance of observing seven successes in one ten trials if success probability is ½.

Setting Parameter Values

- Retrieval models often contain parameters that must be tuned to get best performance for specific types of data and queries
  - Retrieval model
  - Hundreds of feature weights

$$\sum_{i \in Q} \log \frac{(r_i+0.5)/(R-r_i+0.5)}{(n_i-r_i+0.5)/(N-n_i-R+r_i+0.5)} \cdot \frac{(k_1+1)f_i}{K+f_i} \cdot \frac{(k_2+1)qf_i}{k_2+qf_i}$$

- For experiments:
  - Use *training* and *test* data sets
  - If less data available, use *cross-validation* by partitioning the data into *K* subsets
  - Using training and test data avoids *overfitting* – when parameter values do not generalize well to other data

Finding Parameter Values

- Many techniques used to find optimal parameter values given training data
  - Standard problem in machine learning
- In IR, often explore the space of possible parameter values by *brute force*
  - Requires large number of retrieval runs with small variations in parameter values (*parameter sweep)*
- Active research area

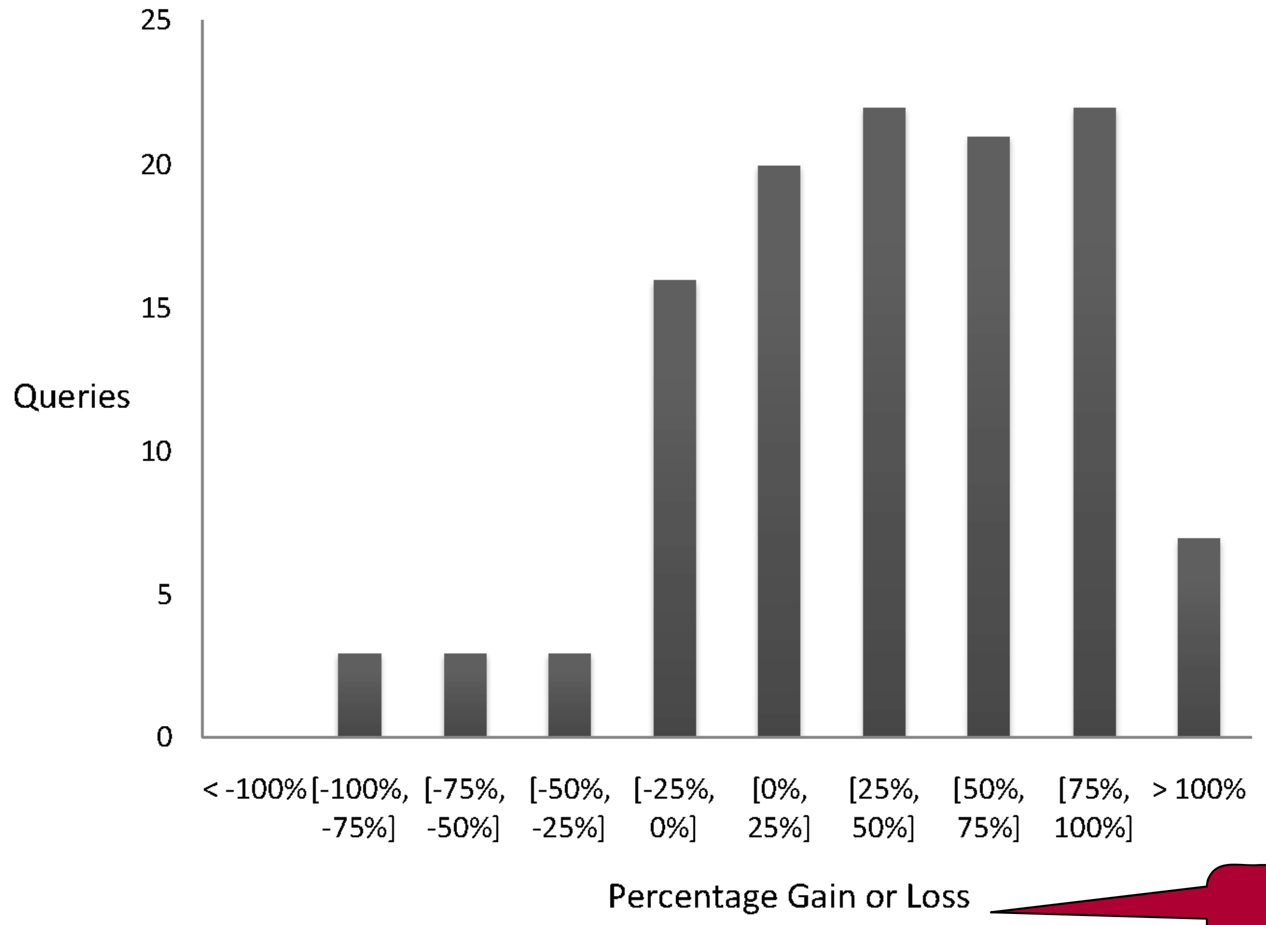
Online Testing

- Test (or even train) using live traffic on a search engine
  - Similar to logging
- Benefits
  - Real users
  - Less bias (real, accurate test collections are difficult to build)
  - Large amounts of test data
    - For free
- Drawbacks:
  - Noisy data
  - Can degrade user experience
- Often done on small proportion (1-5%) of live traffic

Summary

- No single measure is the correct one for any application
  - Choose measures appropriate for task
  - Use a combination
    - Mean average precision - single number summary, popular measure, pooled relevance judgments.
    - Average NDCG - single number summary for each rank level, emphasizes top ranked documents, relevance judgments only needed to a specific rank depth (typically to 10).
    - Recall-precision graph - conveys more information than a single number measure, pooled relevance judgments.
    - Average precision at rank 10 - emphasizes top ranked documents, easy to understand, relevance judgments limited to top 10.
  - Shows different aspects of the system effectiveness
- Use significance tests (t-test)
- Analyze performance of individual queries

Query Summary



E.g., using MAP