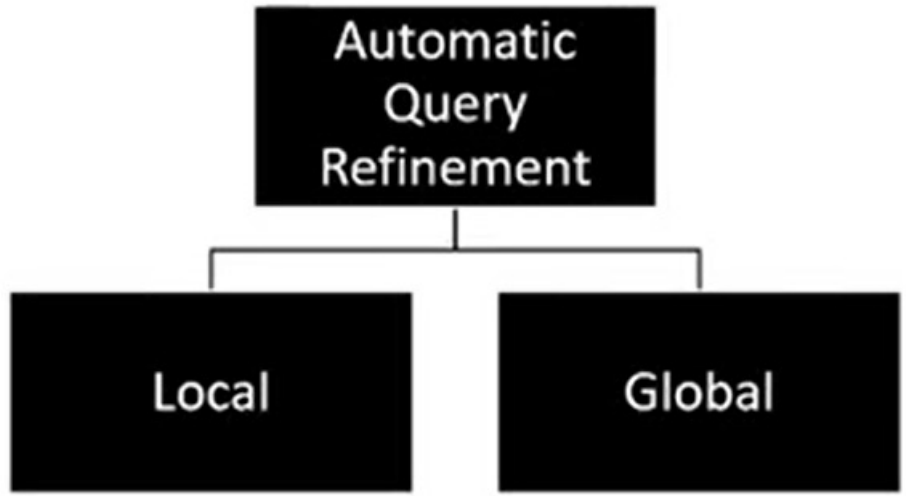
How to improve relevance?

Relevance feedback and Query expansion

The problem of synonymy

- What result do you expect for a query, “plane”?
- What is plane appears in this query, “plane from London to Ahmedabad”?
- So many synonyms which will work for web search
  - Flight
  - Aircraft
  - Airplane
  - Aeroplane
  - By Air
  - Fly
  - Flgt
  - Arcrft

How to ensure good results?



Use the **query** or the **results** for reformulating the query

We will study:

Relevance Feedback

Pseudorelevance

Indirect Relevance Feedback

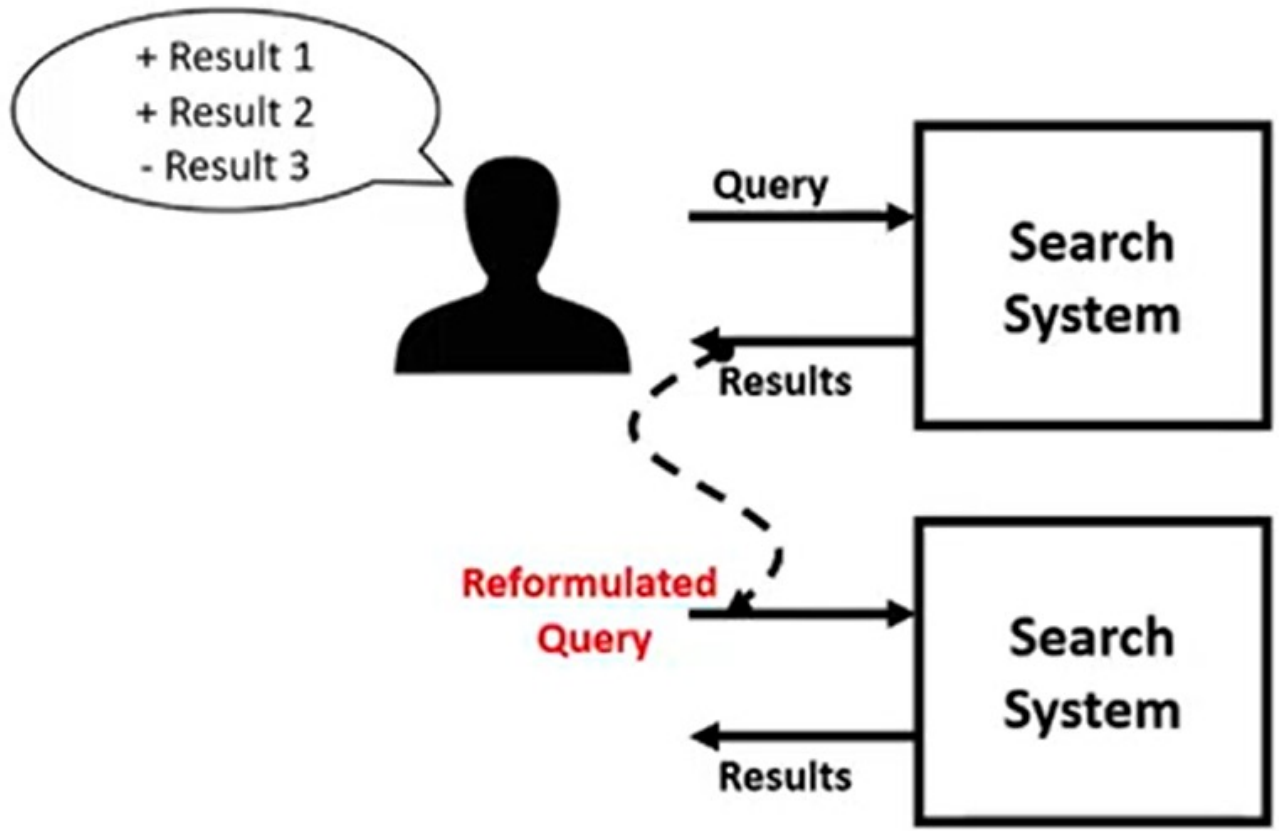
Do not use the **query** or the **results** for reformulating the query.

Eg:

Use *Thesaurus.*

Do Spelling Correction.

Relevance Feedback



An Example

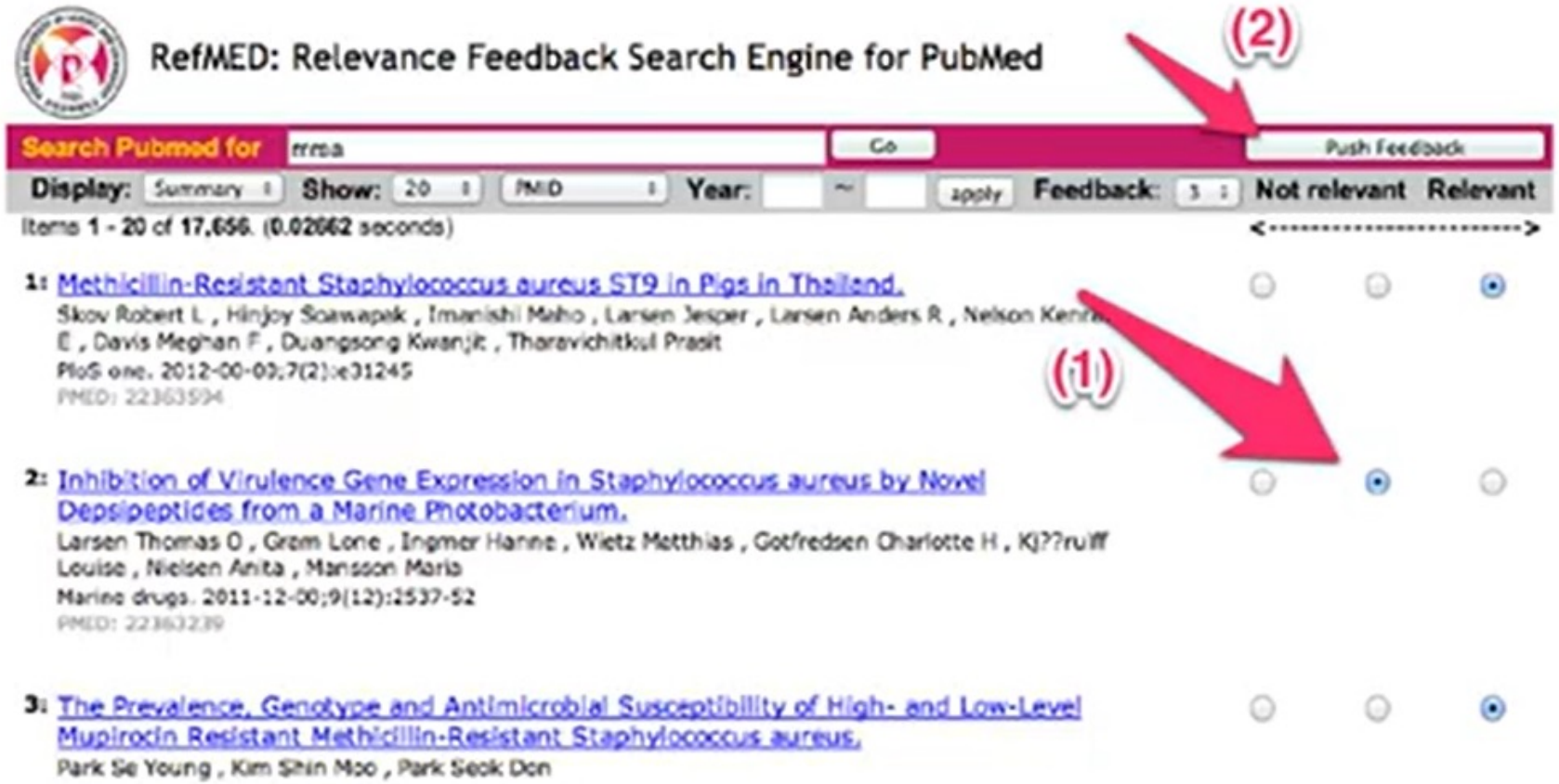


Image source: https://sites.google.com/site/postechdm/research

Interesting Characteristics

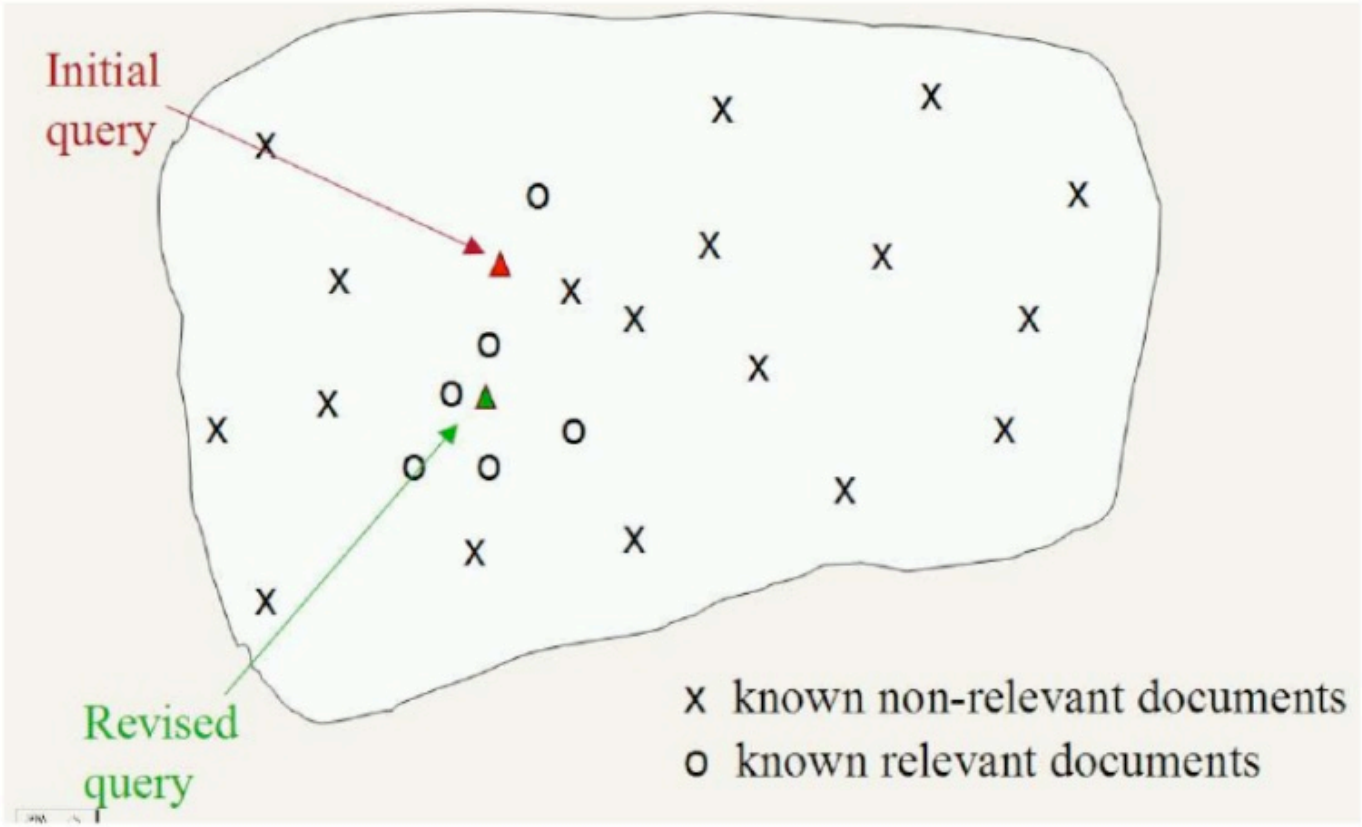
- Indexed content is unknown to the user.
- “Information Need” changes after looking at the results.
  - User visits youtube to listen to a specific set of songs.
  - After the first song, he changes his mind and listens to something else!

A Recap of Vector Space Models

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

Image Source: https://fox.cs.vt.edu/talks/1995/KY95/

Example application of Rocchio



Rocchio in practice

- In practice, however, we usually do not know the full set of relevant and non relevant sets.
- For example, a user might only label a few documents as relevant / non relevant.

Therefore, in practice Rocchio is often parameterised as follows:

$$\vec{q}_m = \alpha \vec{q}_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d}_j \in D_r} \vec{d}_j - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in D_{nr}} \vec{d}_j$$

where $\vec{q}_0$ is the original query vector; D_r and D_{nr} are the sets of known relevant and non relevant documents.

- α, β, and γ are weight parameters attached to each component.
- Reasonable values are $\alpha = 1.0$, $\beta = 0.75$, $\gamma = 0.15$
- Note: if final $\vec{q}_m$ has negative term weights, set to 0.

Rocchio in practice

- Represent query and documents as weighted vectors (e.g., tf–idf).
- Use Rocchio formula to compute new query vector (given some known relevant / non-relevant documents).
- Calculate cosine similarity between new query vector and documents.
- (E.g., supervision exercises 9.5 and 9.6 from the book).

- Rocchio has been shown useful for increasing recall.
- Contains aspects of positive and negative feedback.
- Positive feedback is much more valuable than negative (i.e., indications of what *is* relevant)
- Most systems set $\gamma < \beta$ or even $\gamma = 0$.

Rocchio relevance feedback - Example

- Given:
  - Initial query = "cheap CDs cheap DVDs extremely cheap CDs".
  - d_1 = "CDs cheap software cheap CDs" is judged as relevant.
  - d_2 = "cheap thrills DVDs" is judged as nonrelevant
- What would the revised query vector be after relevance feedback?

Let us solve this together

Assume that we are using direct term frequency (with no scaling and no document frequency). There is no need to length-normalize vectors. Assume $\alpha = 1$, $\beta = 0.75$, $\gamma = 0.25$.

Rocchio relevance feedback - Example

Quiz: Can you complete the following table?

q_0 = "cheap CDs cheap DVDs extremely cheap CDs".

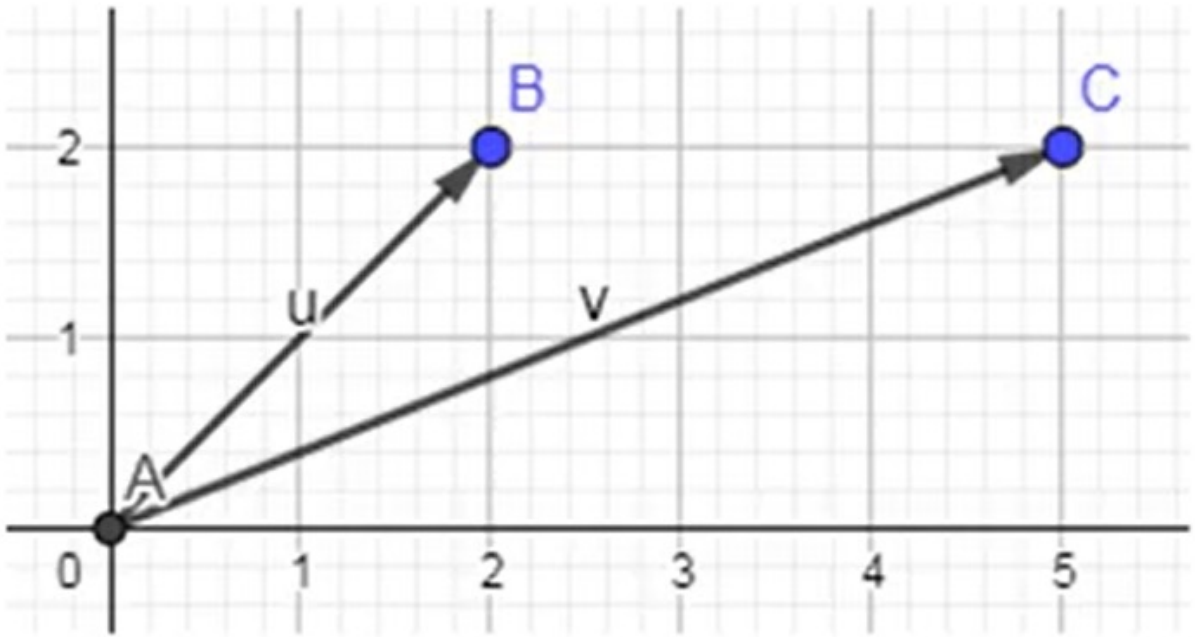
d_1 = "CDs cheap software cheap CDs".

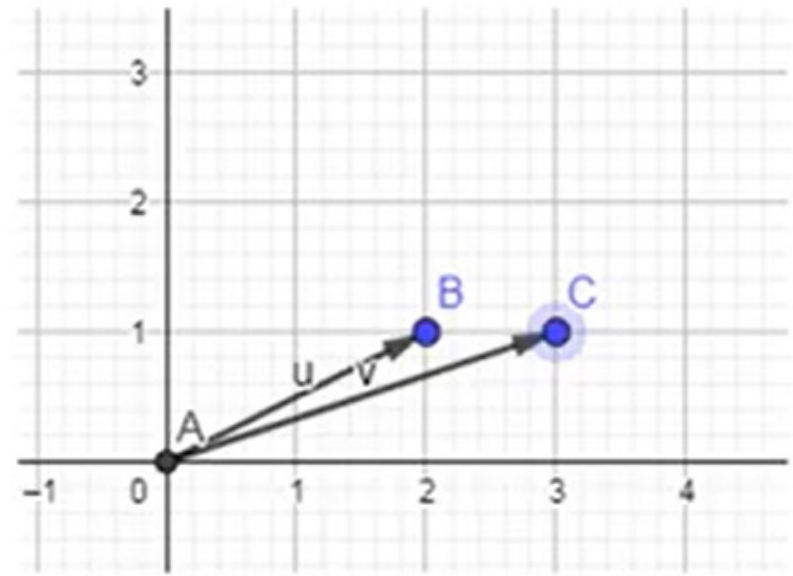
d_2 = "cheap thrills DVDs".

	cheap	CDs	DVDs	extremely	software	thrills
q_0	3	2	1	1	0	0
d_1	2	2	0	0	1	0
d_2	1	0	1	0	0	1

Moving Vectors

- Move (2,2) to (5,2) by adding 3 to x.





Rocchio relevance feedback - Example

Quiz: How to calculate the modified query vector, q_m?

d_1 is judged as relevant. d_2 is judged as nonrelevant.

Assume $\alpha = 1$, $\beta = 0.75$, $\gamma = 0.25$.

	cheap	CDs	DVDs	extremely	software	thrills
q_0	3	2	1	1	0	0
d_1	2	2	0	0	1	0
d_2	1	0	1	0	0	1
$q_m = q_0 + 0.75*d_1 - 0.25*d_2$						
q_m	4.25	3.5	0.75	1	0.75	0

Negative weight does not make sense. So, leave them as zero.

$$\vec{q}_m = \alpha\vec{q}_0 + \beta\frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r}\vec{d}_j - \gamma\frac{1}{|D_{nr}|}\sum_{\vec{d}_j \in D_{nr}}\vec{d}_j$$

Pseudo (Blind) Relevance Feedback

- No User Judgment.
- Assume that the top-k ranked documents are relevant.

Initial query = "cheap CDs cheap DVDs extremely cheap CDs".
d_1 = "CDs cheap software cheap CDs".
d_2 = "cheap thrills DVDs".
What would the revised query vector be after pseudo relevance feedback if top-1 document is considered as relevant?
Assume that we are using direct term frequency (with no scaling and no document frequency). There is no need to length-normalize vectors. Assume $\alpha = 1$, $\beta = 0.75$, $\gamma = 0.25$.

- May lead to query drift.

Indirect (Implicit) Relevance Feedback

- No asking for judgments from users.
- No automatic feedback such as assuming top-k documents as relevant.

Clickstream Mining

Global (User/Result-Independent) Query Refinement

- Automatic Thesaurus Generation
  - Fast = rapid
  - Tall = height?
  - Sound = noise?
  - Restaurant = Hotel = Motel?
- How to handle domain specific phrases?
- Slangs!
- …

How to automate the thesaurus generation?

Co-occurrence Analysis

MAINFRAMES

Mainframes are primarily referred to large computers with rapid, advanced processing capabilities that can execute and perform tasks equivalent to many Personal Computers (PCs) machines networked together. It is characterized with high quantity Random Access Memory (RAM), very large secondary storage devices, and high-speed processors to cater for the needs of the computers under its service.

Consisting of advanced components, mainframes have the capability of

MAINFRAMES

Mainframes usually are referred those computers with fast, advanced processing capabilities that could perform by itself tasks that may require a lot of Personal Computers (PC) Machines. Usually mainframes would have lots of RAMs, very large secondary storage devices, and very fast processors to cater for the needs of those computers under its service.

Due to the advanced components mainframes have, these computers

Source: https://web.stanford.edu/~jurafsky

Co-occurrence Analysis

- Term-Document Matrix
  - How often does individual terms appear in a document?
- Term-Term Matrix
  - How often terms co-occur?

Quiz: Which books are similar?

	Book1	Book2	Book3	Book4
cricket	400	10	355	3
football	5	5	4	4
hockey	9	330	10	200
tennis	2	6	12	4

Co-occurrence Analysis

- Two terms are similar if the term vectors are similar.

	Book1	Book2	Book3	Book4
boundary	400	310	355	389
four	515	225	390	400
movie	9	4	8	1
film	2	6	9	2

Remember, context is important!

Co-occurrence Analysis

- Assume a Boolean term-document matrix A.
- What does AA^T mean?

$$\begin{array}{c} \\ t_1 \\ t_2 \\ t_3 \\ t_4 \end{array}\!\!\begin{array}{c} \begin{array}{cccc} d_1 & d_2 & d_3 & d_4 \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \end{array} \begin{array}{c} \begin{array}{cccc} d_1 & d_2 & d_3 & d_4 \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \end{array} = \begin{array}{c} \\ t_1 \\ t_2 \\ t_3 \\ t_4 \end{array}\!\!\begin{array}{c} \begin{array}{cccc} t_1 & t_2 & t_3 & t_4 \end{array} \\ \begin{pmatrix} 2 & 2 & 0 & 0 \\ 2 & 2 & 0 & 0 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 2 & 2 \end{pmatrix} \end{array}$$

- Usually, weighted length-normalized tf in a sliding window is used to count co-occurrence.

$A =$

	D1	D2	D3	D4	D5	d6
T1	1	0	1	0	1	0
T2	1	1	0	1	0	0
T3	0	1	1	0	1	0
T4	0	1	0	0	1	0
T5	1	0	0	1	1	1
T6	1	0	1	0	1	0

Terms 1 & 6 appear in the same documents

$A^T =$

	T1	T2	T3	T4	T5	T6
D1	1	1	0	0	1	1
D2	0	1	1	1	0	0
D3	1	0	1	0	0	1
D4	0	1	0	0	1	0
D5	1	0	1	1	1	1
D6	0	0	0	0	1	0

$AA^T =$

	T1	T2	T3	T4	T5	T6
T1	3	1	2	1	2	3
T2	1	3	1	1	2	1
T3	2	1	3	2	1	2
T4	1	1	2	2	1	1
T5	2	2	1	1	4	2
T6	3	1	2	1	2	3

Term 6 seems to be best related to itself and term 1.

Precision and Recall

Precision: fraction of retrieved docs that are relevant = P(relevant|retrieved)

Recall: fraction of relevant docs that are retrieved

= P(retrieved|relevant)

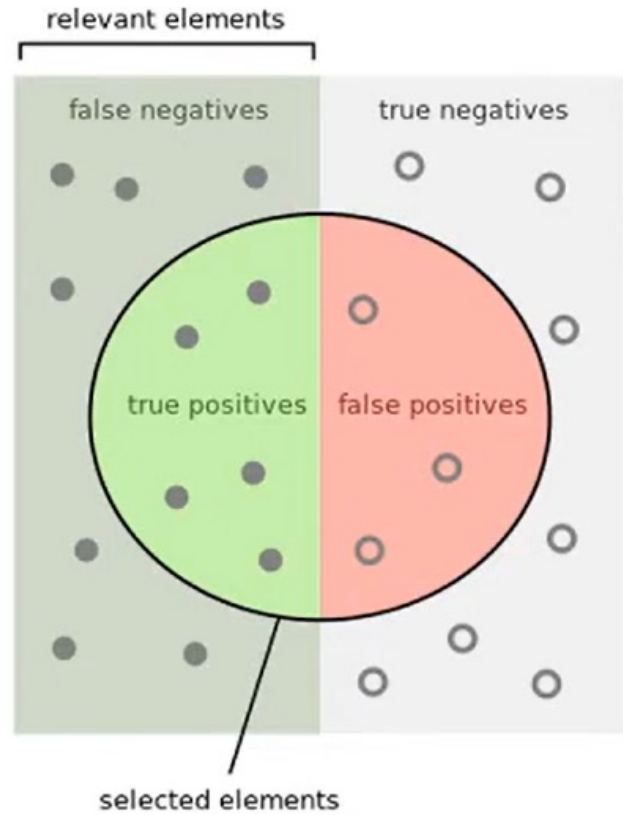
	Relevant	Nonrelevant
Retrieved	tp	fp
Not Retrieved	fn	tn

- Precision P = tp/(tp + fp)
- Recall R = tp/(tp + fn)

Precision and Recall

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$



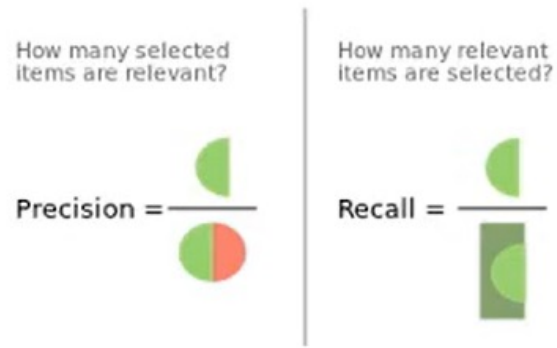
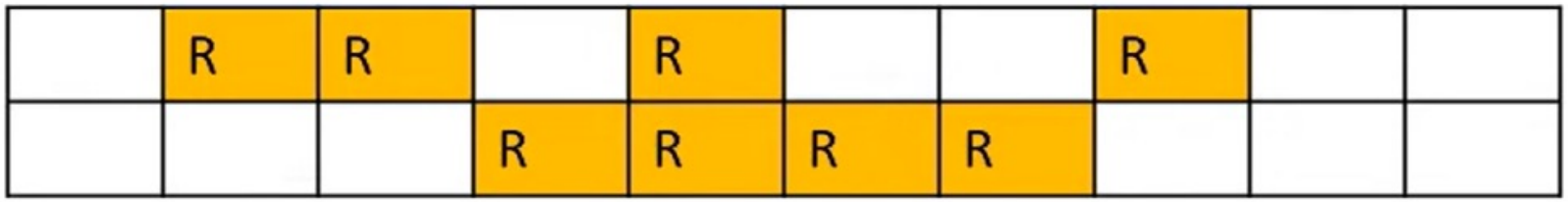


Image Source: Wikipedia

Precision and Recall

- An IR system retrieves the following 20 documents.
- There are 100 relevant documents in our collection.
- Hollow squares represent irrelevant documents.
- Solid squares with ‘R’ are relevant.



- What is Precision? Precision = 8/20.
- What is Recall? Recall = 8/100.

Quiz

- R refers to Relevant Document
- N refers to Nonrelevant Document.
- Collection has 10,000 documents.
- Assume that there are 8 relevant documents in total in the collection. Calculate Precision and Recall.

Retrieved Documents:

RRNNN NNN**R**N **R**NNN**R** NNNN**R**

Quiz

- R refers to Relevant Document
- N refers to Nonrelevant Document.
- Collection has 10,000 documents.
- Assume that there are 8 relevant documents in total in the collection. Calculate Precision and Recall.
- Retrieved Documents:

RRNNN NNN**R**N **R**NNN**R** NNNN**R**

Relevant Docs = 6

Non-R = 14

Retrieved = 20

Precision = $\frac{6}{20}$

Recall = $\frac{6}{8}$

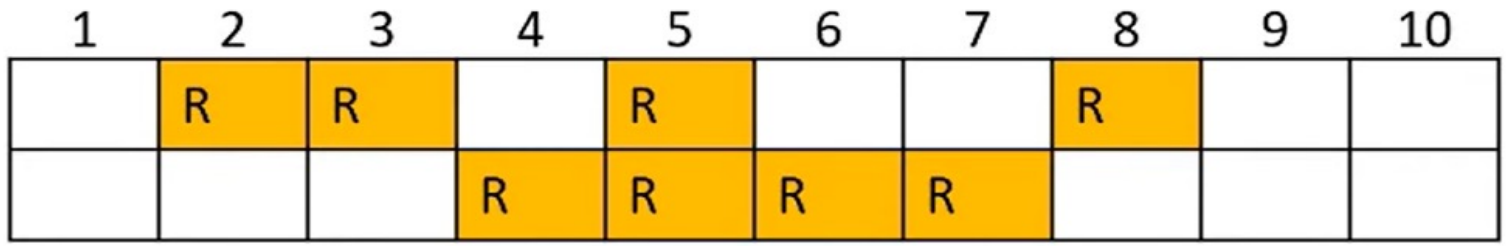
F-Measure

- One measure of performance that takes into account both recall and precision.
- Harmonic mean of recall and precision:

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

Compute Precision and Recall

- Case 1:

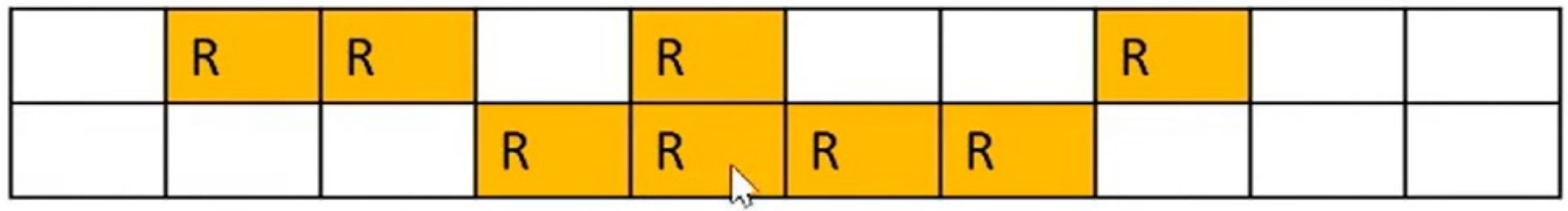


- Case 2:

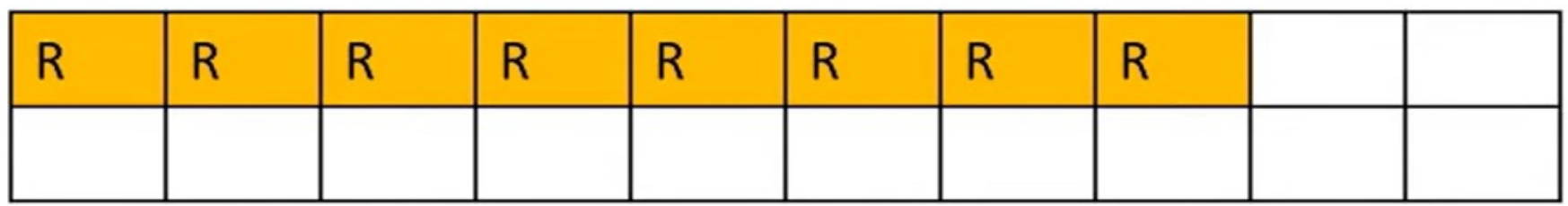
20 documents retrieved. Assume that there are 100 relevant documents.

Compute Precision and Recall

- Case 1: Precision = 8/20, Recall = 8/100



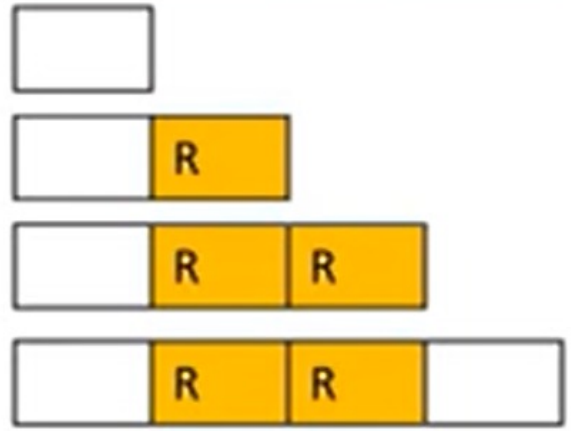
- Case 2: Precision = 8/20, Recall = 8/100



Precision@k

- We cut-off results at k and compute precision.

- P@1 = 0
- P@2 = ½
- P@3 = 2/3
- P@4 = 2/4



Disadvantage: If there are only 4 relevant documents in entire collection, and if we retrieve 10 documents, max precision achievable is only 0.4.

Interpolated Precision

- We cut-off results at k^{th} relevance level.

- (Interpolated) P@1 = 0.5

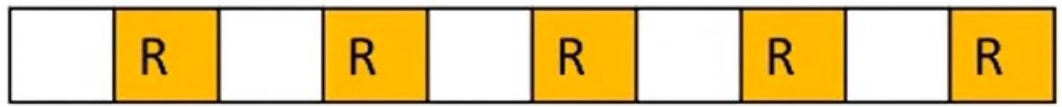
- (Interpolated) P@2 = 2/3

Interpolated Average Precision = (0.5 + 0.66) / 2 = 0.58

(if we are only interested in 2 levels of relevance)

What is the Average Precision?

- Case 1:



  - Average Precision = $\frac{½ + ½ + ½ + ½ + ½}{5}$
  - If there were 10 relevant documents, and we retrieved only five,
    - AP (at relevance level of 10) = $\frac{½ + ½ + ½ + ½ + ½ + 0 + 0 + 0 + 0 + 0}{10}$
- Case 2:

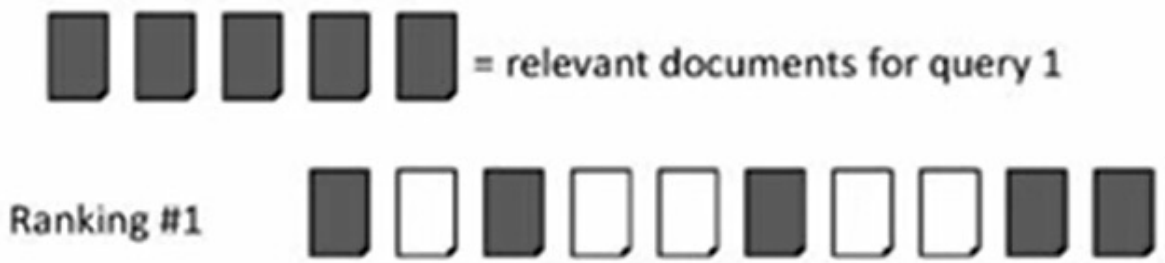
- What is AP at relevance level of 4? Assume there were 6 relevant documents in our collection.
  - AP = $\frac{1/3 + 1/3 + 1/3 + 0}{4}$

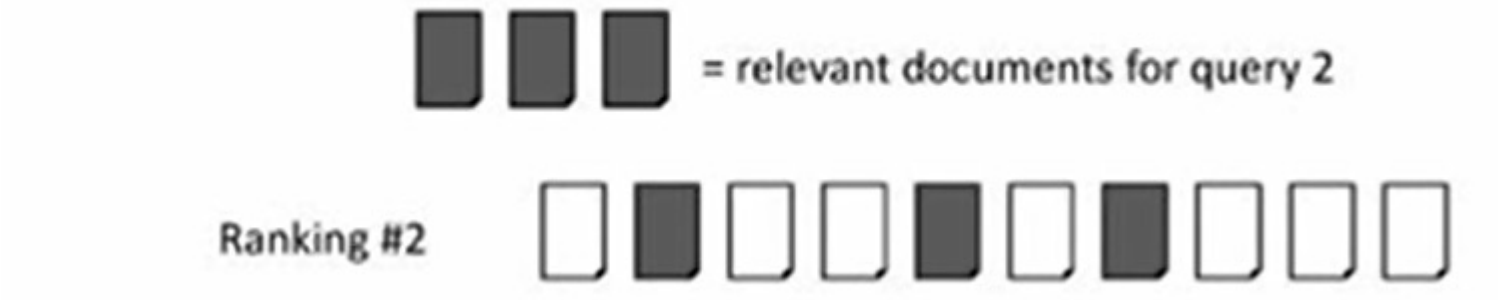
Mean Average Precision

MAP computes Average Precision for all relevance levels for a set of queries.

$$\mathrm{MAP}(Q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} \mathrm{Precision}(R_{jk})$$

Mean Average Precision: example





= 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

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

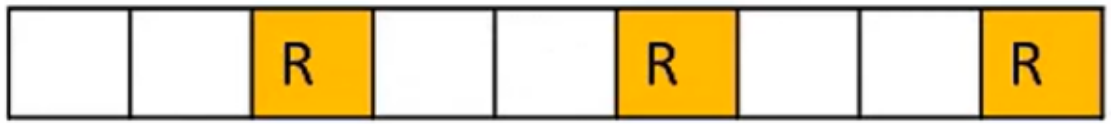
$$mean\ average\ precision = (0.62 + 0.44)/2 = 0.53$$

Compute MAP

- Query1:

Only 5 relevant docs in corpus.

- Query2:



- Query3:

Only 3 relevant docs in corpus.

Compute MAP

- Query1:

Only 5 relevant docs in corpus.

- Query2:

- Query3:

Only 3 relevant docs in corpus.

- Compute MAP.

$$MAP = (1/2 + 1/3 + 1/3)/3$$