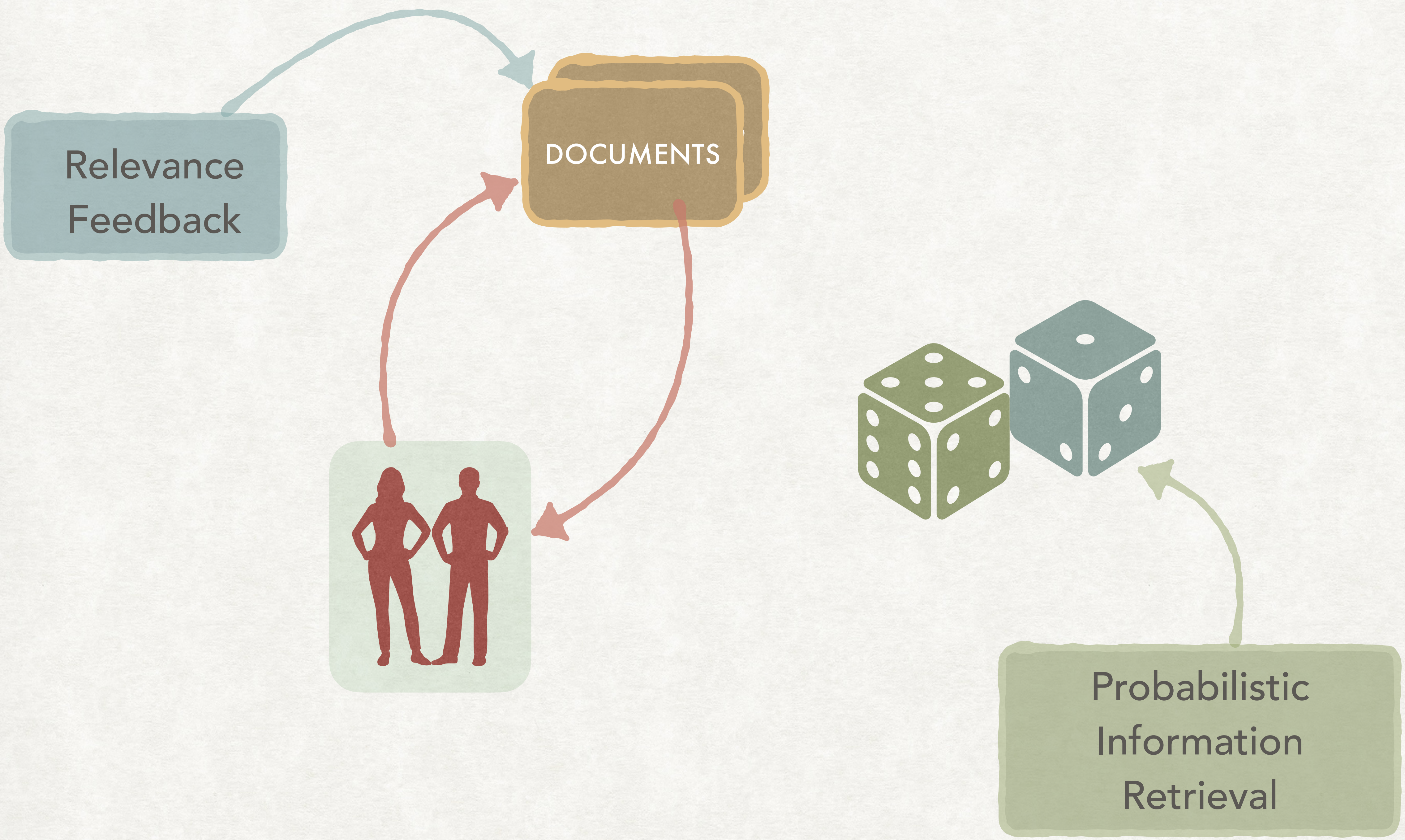
INFORMATION RETRIEVAL

Laura Nenzi
lnenzi@units.it

Lecture 8

LECTURE OUTLINE

*PROBABLY CONTAINS PROBABILITIES



RELEVANCE FEEDBACK

WHAT IS RELEVANCE FEEDBACK

RECEIVING FEEDBACK FROM THE USER

- The main idea is to involve the user in giving feedback on the initial set of results:
- The user issues a query.
- The system returns an initial set of results.
- The user decides which results are relevant and which are not.
- The system computes a new set of results based on the feedback received by the user.
- If necessary, repeat.

WHAT RELEVANCE FEEDBACK CAN SOLVE

AND WHAT IT CANNOT SOLVE

- Relevance feedback can help the user in refining the query without having him/her reformulate it manually.
- It is a *local method*, where the initial query is modified, in contrast to *global methods* that change the wording of the query (like spelling correction).
- Relevance feedback can be ineffective when in the case of
  - Misspelling (but we have seen spelling correction techniques).
  - Searching documents in another language.
  - Vocabulary mismatch between the user and the collection.

THE ROCCHIO ALGORITHM

FEEDBACK FOR THE VECTOR SPACE MODEL

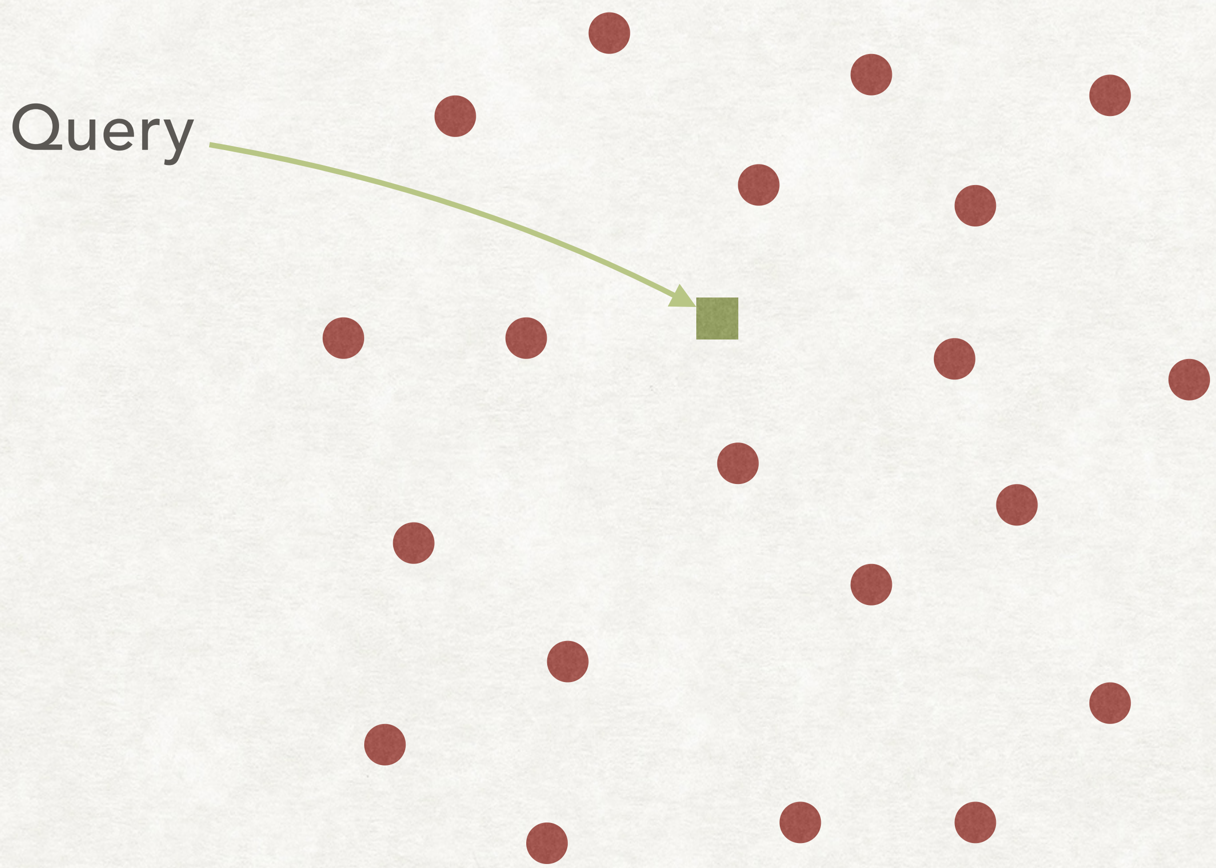
- It is possible to introduce relevance feedback in the vector space model
- We will see the Rocchio Algorithm (1971)
- It was introduced in the SMART (*System for the Mechanical Analysis and Retrieval of Text*) information retrieval system…
- …which is also where the vector space model was firstly developed

ROCCHIO ALGORITHM: MAIN IDEA

MOVING THE QUERY VECTOR

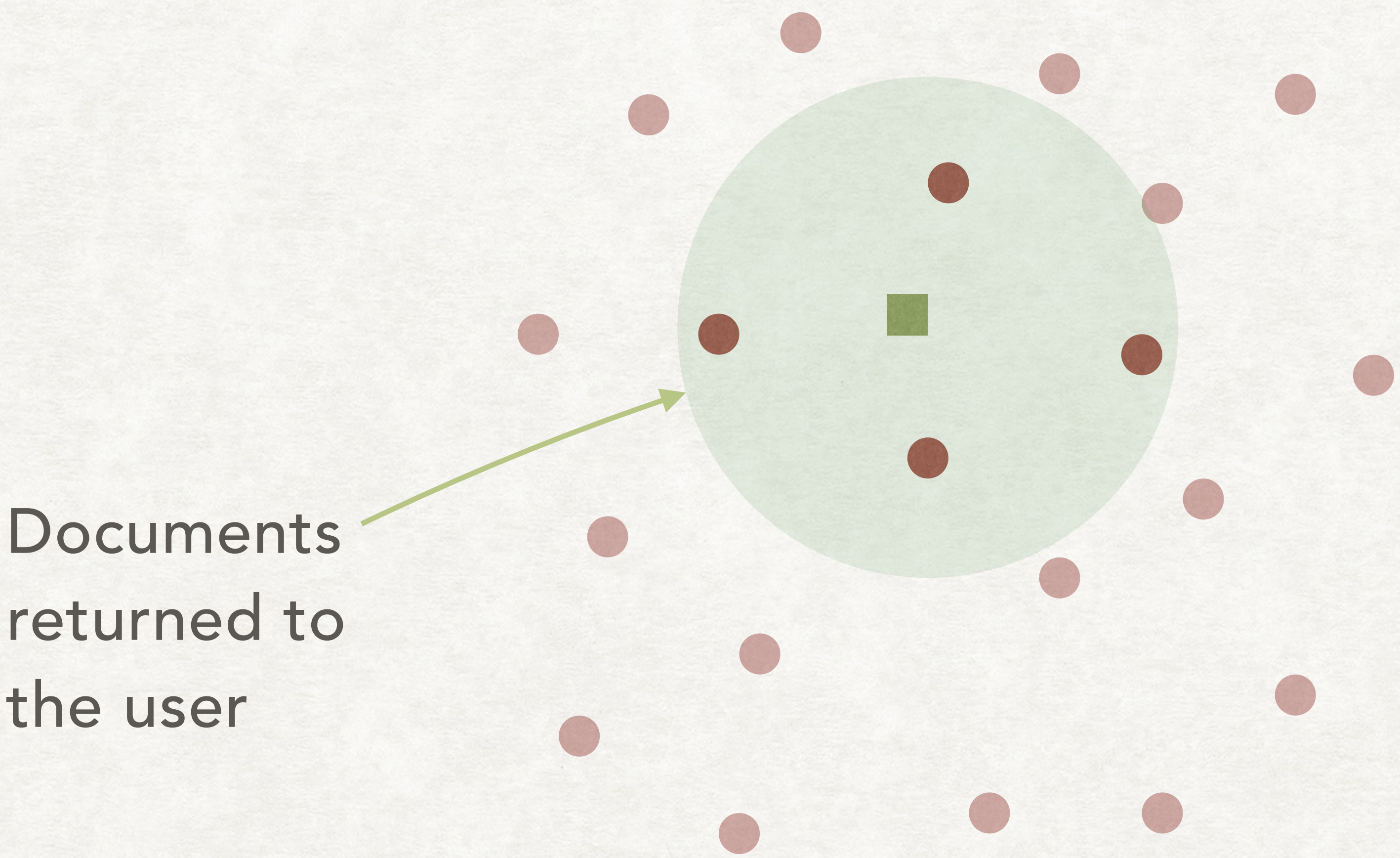
ROCCHIO ALGORITHM: MAIN IDEA

MOVING THE QUERY VECTOR



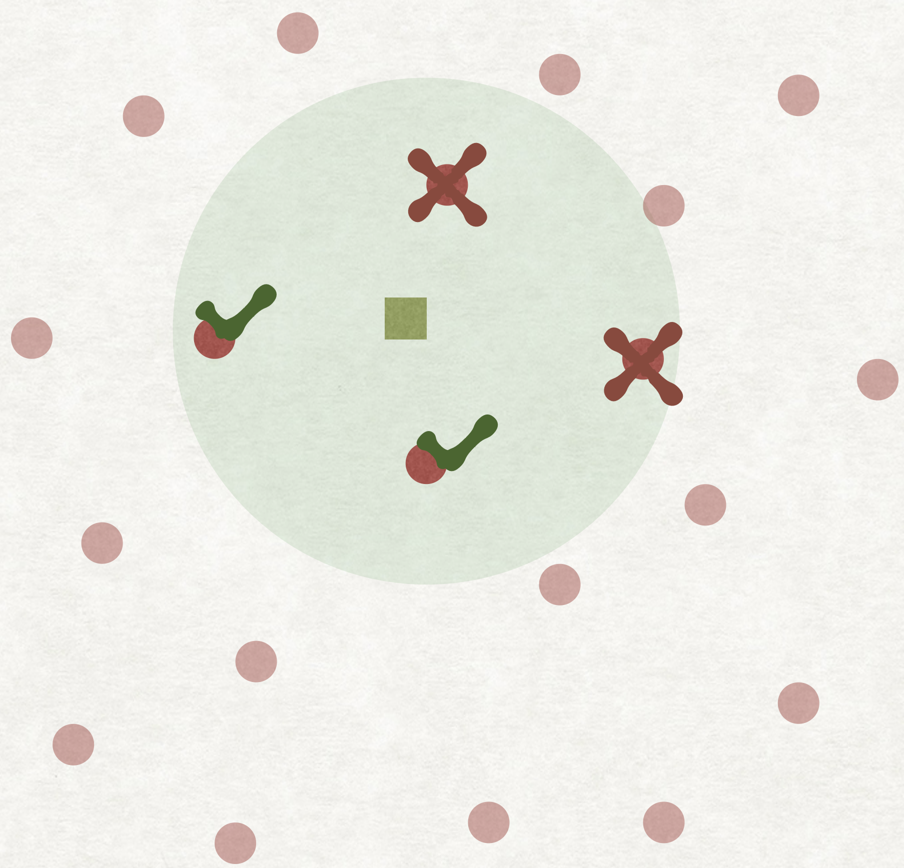
ROCCHIO ALGORITHM: MAIN IDEA

MOVING THE QUERY VECTOR



ROCCHIO ALGORITHM: MAIN IDEA

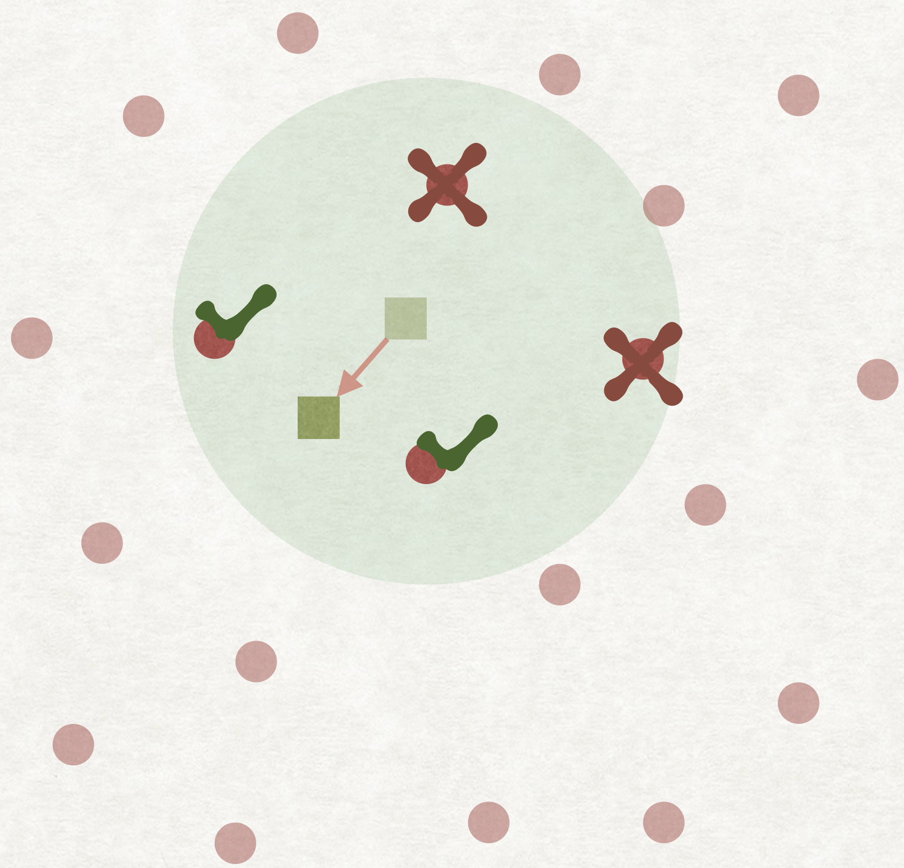
MOVING THE QUERY VECTOR



Feedback from the user

ROCCHIO ALGORITHM: MAIN IDEA

MOVING THE QUERY VECTOR



Revised query

ROCCHIO ALGORITHM: THEORY

- The user gives us two sets of documents:
  - The relevant documents C_r
  - The non-relevant documents C_{nr}
- We want to maximise the similarity of the query with the set of relevant documents…
- …while minimising it with respect to the set of non-relevant documents.

ROCCHIO ALGORITHM: THEORY

This can be formalised as defining the *optimal* query $\vec{q}_{opt}$ as:

$$\vec{q}_{opt} = \arg\max_{\vec{q}} [\text{sim}(\vec{q}, C_r) - \text{sim}(\vec{q}, C_{nr})]$$

If we use cosine similarity, we can reformulate the definition as:

$$\vec{q}_{opt} = \frac{1}{|C_r|} \sum_{\vec{d} \in C_r} \vec{d} - \frac{1}{|C_{nr}|} \sum_{\vec{d} \in C_{nr}} \vec{d}$$

Centroid of relevant documents

Centroid of non-relevant documents

ROCCHIO ALGORITHM

However, we usually do not have knowledge of the relevance of *all* documents in the system. Instead we have:

- a set D_r of *known relevant* documents
- a set D_{nr} of *known non-relevant* documents

We also have the original query $\vec{q}_0$ performed by the user.

We can perform a linear combination of:

- The centroid of D_r
- The centroid of D_{nr}
- The original query $\vec{q}_0$

ROCCHIO ALGORITHM

In the Rocchio algorithm the query is updated as follows:

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

Original query

Centroid of the
known relevant documents

Centroid of the known
non-relevant documents

If one of the components of $\vec{q}_m$ is less than 0, we set it to 0
(all documents have non-negative coordinates)

ROCCHIO ALGORITHM

SELECTING THE WEIGHTS

- We need to select reasonable weights α, β, and γ:
- Positive feedback is more valuable than negative feedback, so usually $\gamma < \beta$.
- Reasonable values might be $\alpha = 1$, $\beta = 0.75$, and $\gamma = 0.15$.
- It is also possible to have only positive feedback with $\gamma = 0$.

PSEUDO-RELEVANCE FEEDBACK

NOW WITHOUT THE USER

- It is possible to perform a relevance feedback without the user…
- …even before he/she receives the results of the first query.
- Perform the query $\vec{q}$ as usual.
- Consider the first k retrieved documents in the ranking as relevant.
- Perform relevance feedback using this assumption.
- Might provide better results, but the retrieved documents might drift the query in an unwanted direction.

PROBABILISTIC INFORMATION RETRIEVAL

PROBABILISTIC IR

MAIN IDEAS

- If we know some relevant and some non-relevant documents for a query we can estimate the probability of a document to be relevant given the terms it contains, $P(R = 1 | d, q)$.
- This is the main idea of a probabilistic model of IR: estimate probabilities of a document being relevant with respect to a query based on its content.
- There will be some assumptions to simplify the computation of this probability…
- …and some estimates: we do not known most of the probabilities involved!

A QUICK REVIEW

BASICS OF PROBABILITY THEORY

- The probability of A and B can be written as a conditional probability:
  $P(A, B) = P(A \mid B)P(B) = P(B \mid A)P(A)$

- The probability of B and A plus the probability of B and not A is simply the probability of B:
  $P(B) = P(B, A) + P(B, \overline{A})$

- The *odds* of an event A is defined as:
  $$O(A) = \frac{P(A)}{P(\overline{A})} = \frac{P(A)}{1 - P(A)}$$

A QUICK REVIEW

BASICS OF PROBABILITY THEORY

- The classical Bayes' rule is:
- $$P(A|B) = \frac{P(B|A)P(A)}{P(B)} = \frac{P(B|A)}{\sum_{X \in [A,\bar{A}]} P(B|X)P(X)} P(A)$$
- Which can be interpreted as:
  - Given the prior probability $P(A)$ of A…
  - …how we can update it based on the evidence B, thus obtaining a posterior probability $P(A|B)$.

PROBABILITY RANKING PRINCIPLE (PRP)

AND THE BASIS FOR PROBABILISTIC IR

For each document we consider the random variable $R_{d,q}$ (or R for short) , representing wether a document is relevant (R=1) or not (R=0) w.r.t. a query q

We want to rank documents according to their probability of being relevant to a given query q:

$$P(R = 1 \mid d, q)$$

Probability of having something relevant

Given that the document is d and the query is q

1/0 LOSS

AN THE OPTIMAL DECISION RULE

The simples case:

- Penalty when we retrieve a document that is not relevant.
- Penalty when we miss a relevant document.
- The penalty is the same in all cases, there are no costs associated to retrieving documents.

If we need to rank documents then we rank them by decreasing $P(R = 1 \mid d, q)$.

If we need to return a *set* of documents we return all then ones where $P(R = 1 \mid d, q) > P(R = 0 \mid d, q)$.

It can be proved that this choice minimise the *expected loss* under the 1/0 loss.

RETRIEVAL COSTS

MORE THAN THE 1/0 LOSS

We can also have a more complex model for costs:

- C_1 is the cost of retrieving a relevant document.
- C_0 is the cost of retrieving a non-relevant document

Then to select the document to be retrieved d we must the one where for all *non-retrieved* documents d' it holds that:

$$\underbrace{C_1 \cdot P(R=1 \mid d, q) + C_0 \cdot P(R=0 \mid d, q)}_{\text{Weighted cost of retrieving } d} \leq \underbrace{C_1 \cdot P(R=1 \mid d', q) + C_0 \cdot P(R=0 \mid d', q)}_{\text{Weighted cost of retrieving } d'}$$

THE BINARY INDEPENDENCE MODEL

THE BINARY INDEPENDENCE MODEL

OR "BIM"

Binary — Or "Boolean". Each document (and query) is represented as a vector $\vec{x} = (x_1, \ldots, x_M)$ where $x_i = 1$ if the term is present and $x_i = 0$ otherwise

Independence — We assume that all terms occurs in a document independently.

Not a correct assumption, but "it works"

Additionally, we assume the relevant of a document to be independent on the relevance of other documents.
This is not true in practice: e.g., duplicate and near-duplicate documents are not independent.

ESTIMATION OF THE PROBABILITY

$$P(R = 1 \mid d, q)$$

In our model this is given by

WE DO NOT KNOW THE EXACT VALUE, WE WILL NEED TO PROVIDE ESTIMATES!

$$P(R = 1 \mid \vec{x}, \vec{q})$$

By Bayes' rule

$$\frac{P(\vec{x} \mid R = 1, \vec{q})\, P(R = 1 \mid \vec{q})}{P(\vec{x} \mid \vec{q})}$$

Probability for a document with representation $\vec{x}$ is retrieved given that a relevant document for the query q is retrieved

Probability of retrieving a relevant document for the query q

ESTIMATION OF THE PROBABILITY

$$P(R = 0 \mid d, q)$$

In out model this is given by

$$P(R = 0 \mid \vec{x}, \vec{q})$$

By Bayes' rule

$$\frac{P(\vec{x} \mid R = 0, \vec{q}) \, P(R = 0 \mid \vec{q})}{P(\vec{x} \mid \vec{q})}$$

Probability for a document with representation $\vec{x}$ is retrieved given that a **non**-relevant document for the query q is retrieved

Probability of retrieving a **non**-relevant document for the query q

DO WE REALLY NEED TO KNOW THE PROBABILITY?

FOR RANKING ODDS ARE SUFFICIENT

For the purpose of ranking, we can use a monotone function of the probability.
For example, the odds of R given $\vec{x}$ and $\vec{q}$:

$$O(R \mid \vec{x}, \vec{q}) = \frac{P(R = 1 \mid \vec{x}, \vec{q})}{P(R = 0 \mid \vec{x}, \vec{q})}$$

$$\frac{\dfrac{P(\vec{x} \mid R = 1, \vec{q})\, P(R = 1 \mid \vec{q})}{P(\vec{x} \mid \vec{q})}}{\dfrac{P(\vec{x} \mid R = 0, \vec{q})\, P(R = 0 \mid \vec{q})}{P(\vec{x} \mid \vec{q})}} \rightarrow \frac{P(\vec{x} \mid R = 1, \vec{q})\, P(R = 1 \mid \vec{q})}{P(\vec{x} \mid R = 0, \vec{q})\, P(R = 0 \mid \vec{q})}$$

CAN WE SIMPLIFY IT FURTHER?

RANKING AND PROBABILITIES

$$\frac{P(\vec{x} \mid R=1,\vec{q})\ P(R=1 \mid \vec{q})}{P(\vec{x} \mid R=0,\vec{q})\ P(R=0 \mid \vec{q})}$$

Depends on the document

The same for all documents

↓

Does not affect the ranking

↓

We can remove it

We now have to estimate:

$$\frac{P(\vec{x} \mid R=1,\vec{q})}{P(\vec{x} \mid R=0,\vec{q})}$$

USING THE BIM

$$\frac{P(\vec{x} \mid R = 1, \vec{q})}{P(\vec{x} \mid R = 0, \vec{q})}$$

We can now employ the independence assumption: each of the terms is assumed to appear independently from the others

$$\frac{P(x_1 \mid R = 1, \vec{q})}{P(x_1 \mid R = 0, \vec{q})} \times \frac{P(x_2 \mid R = 1, \vec{q})}{P(x_2 \mid R = 0, \vec{q})} \times \cdots \times \frac{P(x_M \mid R = 1, \vec{q})}{P(x_M \mid R = 0, \vec{q})}$$

Which means the the value to estimate is now:

$$\prod_{i=1}^{M} \frac{P(x_i \mid R = 1, \vec{q})}{P(x_i \mid R = 0, \vec{q})}$$

SPLITTING UP FURTHER

$$\prod_{i=1}^{M} \frac{P(x_i \mid R = 1, \vec{q})}{P(x_i \mid R = 0, \vec{q})}$$

Each x_i can only assume two values:
0 if the i^{th} term is not present
1 if the i^{th} term is present

$$\prod_{i:x_i=1} \frac{P(x_i = 1 \mid R = 1, \vec{q})}{P(x_i = 1 \mid R = 0, \vec{q})} \cdot \prod_{i:x_i=0} \frac{P(x_i = 0 \mid R = 1, \vec{q})}{P(x_i = 0 \mid R = 0, \vec{q})}$$

For the terms in the document

For the terms not in the document

HOW MANY PROBABILITIES TO ESTIMATE?

$$\prod_{i:x_i=1} \frac{P(x_i = 1 | R = 1, \vec{q})}{P(x_i = 1 | R = 0, \vec{q})} \cdot \prod_{i:x_i=0} \frac{P(x_i = 0 | R = 1, \vec{q})}{P(x_i = 0 | R = 0, \vec{q})}$$

For each term we need only to estimate four probabilities:

	Document relevant	Document not relevant
Term present	p_i	u_i
Tern absent	$1 - p_i$	$1 - u_i$

SIMPLIFYING FURTHER

$$\prod_{i:x_i=1} \frac{p_i}{u_i} \cdot \prod_{i:x_i=0} \frac{1-p_i}{1-u_i}$$

Let us assume that all query terms **not** in the query appears equally in relevant and non-relevant documents. That is, $p_i = u_i$ when $q_i = 0$.

We can remove the factors for all terms not in the query, obtaining:

$$\prod_{i:x_i=1;q_i=1} \frac{p_i}{u_i} \cdot \prod_{i:x_i=0;q_i=1} \frac{1-p_i}{1-u_i}$$

SIMPLIFYING FURTHER

$$\prod_{i:x_i=1;q_i=1} \frac{p_i}{u_i} \cdot \prod_{i:x_i=0;q_i=1} \frac{1-p_i}{1-u_i}$$

We now multiply everything by

Each term is actually 1.

$$\prod_{i:x_i=1;q_i=1} \frac{1-p_i}{1-u_i} \cdot \frac{1-u_i}{1-p_i}$$

By rearranging the factors we obtain:

$$\prod_{i:x_i=1;q_i=1} \frac{p_i}{u_i} \frac{1-u_i}{1-p_i} \cdot \prod_{i:q_i=1} \frac{1-p_i}{1-u_i}$$

SIMPLIFYING FURTHER

$$\prod_{i:x_i=1;q_i=1} \frac{p_i}{u_i}\frac{1-u_i}{1-p_i} \cdot \prod_{i:q_i=1} \frac{1-p_i}{1-u_i}$$

This part does **not** depend on the document! We can remove it

$$\prod_{i:x_i=1;q_i=1} \frac{p_i}{u_i}\frac{1-u_i}{1-p_i}$$

RATIO OF ODDS

$$\prod_{i:x_i=1;q_i=1} \frac{p_i}{u_i}\frac{1-u_i}{1-p_i}$$

Each factor can be seen as two odds:

$$\frac{p_i}{1-p_i}$$

Odds of the term appearing in the document if the document is relevant

$$\frac{1-u_i}{u_i}$$

Inverse odds of the term appearing in the document if the document is **not** relevant

RETRIEVAL STATUS VALUE

The **Retrieval Status Value (RSV) of a document** d is defined as the logarithm of the quantity that we now have:

$$\mathrm{RSV}_d = \log \left(\prod_{i:x_i=1;q_i=1} \frac{p_i}{u_i} \frac{1-u_i}{1-p_i} \right)$$

$$= \sum_{i:x_i=1;q_i=1} \log \frac{p_i}{u_i} \frac{1-u_i}{1-p_i}$$

RETRIEVAL STATUS VALUE

Consider each term of the sum:

$$c_i = \log \frac{p_i}{u_i} \frac{1 - u_i}{1 - p_i}$$

Which can be rewritten as a **log odds ratio**:

$$c_i = \log \frac{p_i}{1 - p_i} + \log \frac{1 - u_i}{u_i}$$

c_i can be considered the **weight** of the i^{th} term of the dictionary, and can be pre-computed (like other measures like the inverse document frequency)

RETRIEVAL STATUS VALUE

At the end the RSV of a document d can be written as:

$$\text{RSV}_d = \sum_{i:x_i=q_i=1} c_i$$

Which algorithmically, can be described as:

> *To compute the RSV of a document d, sum the weight c_i of each term contained in both the document and the query*

We now need a way to estimate the various probabilities to (pre-)compute all c_i.

PROBABILITY ESTIMATION IN PRACTICE

ESTIMATION FOR NON-RELEVANT DOCUMENTS

- We assume that non-relevant documents are a majority inside the collection.

- Thus, we approximate the probability for non-relevant documents with statistics computed using the entire collection.

- Usually $\log \frac{1 - u_i}{u_i} = \log \frac{N - \mathrm{df}_i}{\mathrm{df_i}}$ for a term i.

- Which is approximately $\log \frac{N}{\mathrm{df}_i}$, which is actually the inverse document frequency idf_i for the term i.

ESTIMATION FOR RELEVANT DOCUMENTS

- Estimation for relevant documents is more complex. There are multiple approaches used in practice:

- We can estimate the probabilities by looking at statistics on a set of relevant documents that we have obtained in some way.

- We can put all probabilities equal to 0.5. With this estimate and assuming idf_i for non-relevant documents, this approximation is the sum of the idf_i for all query terms that occurs in the document.

- Another possibility is using some collection level statistics, for example obtaining $p_i = \frac{1}{3} + \frac{2}{3}\frac{\mathrm{df}_i}{N}$.

COMBINATION WITH RELEVANCE FEEDBACK

We can combine relevance feedback to help us estimate the probability used in computing the RSV_d:

1. Start with probabilities estimated as before
2. Retrive a set V of documents
3. The user classifies the documents retrieved and gives us a set of relevant documents: $VR = \{d \in V : R_{d,q} = 1\}$
4. Re-compute our estimates for p_i and u_i

COMBINATION WITH RELEVANCE FEEDBACK

RE-COMPUTING ESTIMATES

If VR is large enough we can use the following updating:

For each i let VR_i be the set of relevant documents containing the i^{th} term:

$$p_i = \frac{|VR_i|}{|VR|} \qquad u_i = \frac{\mathrm{df}_i - |VR_i|}{N - |VR|}$$

However in most case the set of documents evaluated by the user is not large, so we use a “smoothed” version:

$$p_i = \frac{|VR_i| + \frac{1}{2}}{|VR| + 1} \qquad u_i = \frac{\mathrm{df}_i - |VR_i| + \frac{1}{2}}{N - |VR| + 1}$$

COMBINATION WITH RELEVANCE FEEDBACK

PSEUDO-RELEVANCE FEEDBACK

We can extend the previous model to allow for pseudo-relevance feedback.

Select the first k highest ranked documents, consider them as a set V

Consider all of them relevant, and update the probability accordingly (simply substituting VR with V in the previous equations):

$$p_i = \frac{|V_i| + \frac{1}{2}}{|V| + 1} \qquad u_i = \frac{\mathrm{df}_i - |V_i| + \frac{1}{2}}{N - |V| + 1}$$

Repeat until the ranking converges

OKAPI BM25

OKAPI BM25

AKA BM25 WEIGHTING OR OKAPI WEIGHTING

This model is non-binary, since it takes into account the *frequency* of the terms inside the document.

We start with:

$$\mathrm{RSV}_d = \sum_{t \in q} \mathrm{idf}_t$$

Recall that this is the formula that we obtain with one of our estimates.

We now need a way to add information about the term frequencies

OKAPI BM25

AKA BM25 WEIGHTING OR OKAPI WEIGHTING

Let L_d be the length of the document and L_{avg} the average length of the documents in the collection.

$$\mathrm{RSV}_d = \sum_{t \in q} \mathrm{idf}_t \cdot \frac{(k_1 + 1)\mathrm{tf}_{t,d}}{k_1((1-b) + b \cdot \frac{L_d}{L_{avg}}) + \mathrm{tf}_{t,d}}$$

k_1 and b are two parameters, with $b \in [0,1]$ and $k_1 \geq 0$, usually $k_1 \in [1.2, 2.0]$

OKAPI BM25

AKA BM25 WEIGHTING OR OKAPI WEIGHTING

Let us break up the formula in its components

How much to consider term frequency,
With $k_1 = 0$ we have the binary model

$$\mathrm{RSV}_d = \sum_{t \in q} \mathrm{idf}_t \cdot \frac{(k_1 + 1)\,\mathrm{tf}_{t,d}}{k_1\left((1-b) + b \cdot \frac{L_d}{L_{avg}}\right) + \mathrm{tf}_{t,d}}$$

How much to normalise with respect to length,
regulated by b, with $b = 0$: no normalisation,
with $b = 1$, full scaling by document length

BAYESIAN NETWORKS IN IR

MAIN IDEAS

- Bayesian Networks can model dependencies between terms or documents (contrarily to the assumption of the BIM).
- Different models with different topologies exist.
- The model decomposes into two parts: a document collection network and a query network.
- The document collection network is large, but can be pre-computed: it maps from documents to terms
- The query network is relatively small but a new network needs to be built each time a query comes in, and then attached to the document network