PANIMALAR INSTITUTE OF TECHNOLOGY

DEPARTMENT OF CSE

R-2017

ACADEMIC YEAR	:2023-2024
BATCH	:2020-2024
YEAR/SEM	:IV/VIII
SUBJECTCODE/TITLE	:CS8080/INFORMATION RETRIEVAL TECHNIQUES

UNIT-2

MODELING AND RETRIEVAL EVALUATION

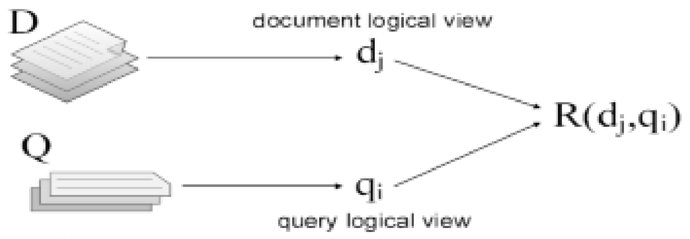
UNIT II
MODELING AND RETRIEVAL EVALUATION

Basic IR Models - Boolean Model - TF-IDF (Term Frequency/Inverse Document Frequency) Weighting - Vector Model – Probabilistic Model – Latent Semantic Indexing Model – Neural Network Model – Retrieval Evaluation – Retrieval Metrics – Precision and Recall – Reference Collection – User-based Evaluation – Relevance Feedback and Query Expansion – Explicit Relevance Feedback.

Modeling

- ✓ Modeling in IR is a complex process aimed at producing a ranking function.
- ✓ Ranking function: a function that assigns scores to documents with regard to a given query.
- ✓ This process consists of two main tasks:
    - The conception of a logical framework for representing documents and queries
    - The definition of a ranking function that allows quantifying the similarities among documents and queries
- ✓ IR systems usually adopt index terms to index and retrieve documents

IR Model Definition:



An IR model is a quadruple [D, Q, F, R(qi, dj)] where
1. D is a set of logical views for the documents in the collection
2. Q is a set of logical views for the user queries
3. F is a framework for modeling documents and queries
4. R(qi, dj) is a ranking function

Types of Information Retrieval (IR) Model

- ✓ An information model (IR) model can be classified into the following three models
    - Classical IR Model
    - Non-Classical IR Model
    - Alternative IR Model

Classical IR Model

It is the simplest and easy to implement IR model. This model is based on mathematical knowledge that was easily recognized and understood as well. Boolean, Vector and Probabilistic are the three classical IR models.

Non-Classical IR Model

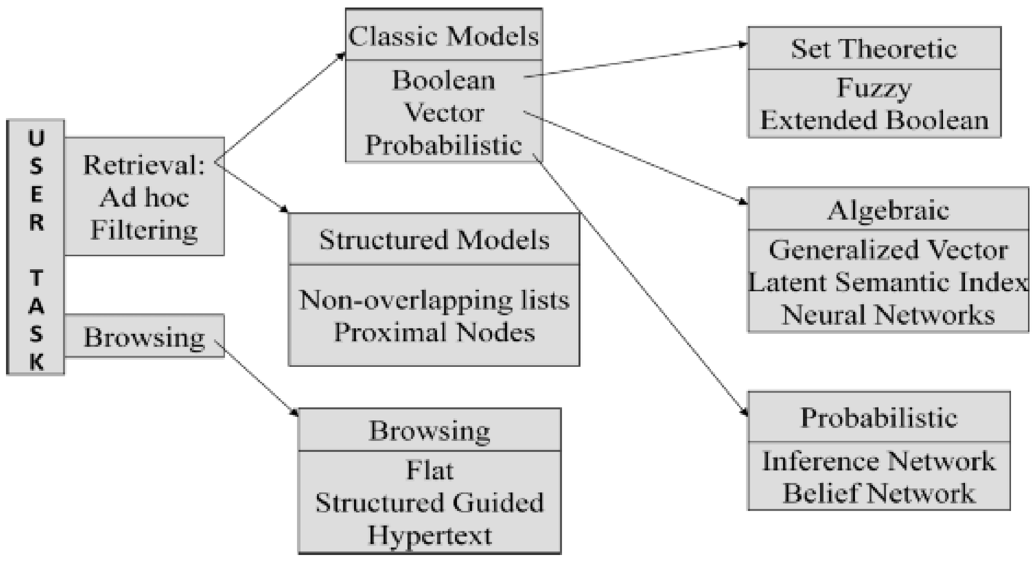
It is completely opposite to classical IR model. Such kind of IR models are based on principles other than similarity, probability, Boolean operations. Information logic model, situation theory model and interaction models are the examples of non-classical IR model.

Alternative IR Model

It is the enhancement of classical IR model making use of some specific techniques from some other fields. Cluster model, fuzzy model and latent semantic indexing (LSI) models are the example of alternative IR model.

Classic IR model:

- ✓ Each document is described by a set of representative keywords called index terms.
- ✓ Assign a numerical weights to distinct relevance between index terms.
- ✓ Three classic models:
    - Boolean,
    - Vector,
    - Probabilistic



BOOLEAN MODEL

The Boolean retrieval model is a model for information retrieval in which we can pose any query which is in the form of a Boolean expression of terms, that is, in which terms are combined with the operators AND, OR, and NOT. The model views each document as just a set of words. Based on a binary decision criterion without any notion of a grading scale. Boolean expressions have precise semantics.

It is the oldest information retrieval (IR) model. The model is based on set theory and the Boolean algebra, where documents are sets of terms and queries are Boolean expressions on terms. The Boolean model can be defined as −

- D − A set of words, i.e., the indexing terms present in a document. Here, each term is either present (1) or absent (0).
- Q − A Boolean expression, where terms are the index terms and operators are logical products − AND, logical sum − OR and logical difference − NOT
- F − Boolean algebra over sets of terms as well as over sets of documents

If we talk about the relevance feedback, then in Boolean IR model the Relevance prediction can be defined as follows −

- R − A document is predicted as relevant to the query expression if and only if it satisfies the query expression as −

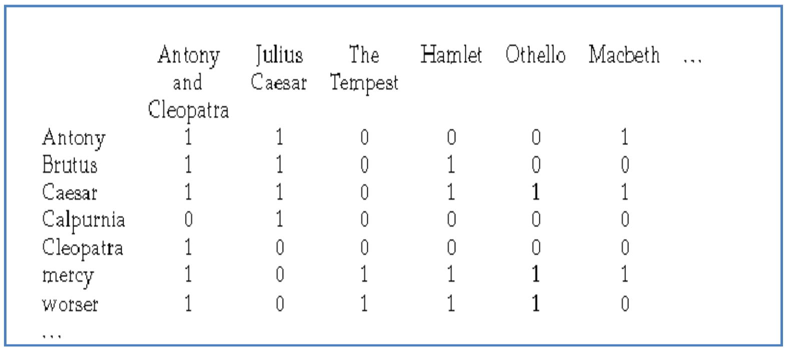
$$((text \vee information) \wedge rerieval \wedge \tilde{} \ theory)$$

We can explain this model by a query term as an unambiguous definition of a set of documents. For example, the query term *"economic"* defines the set of documents that are indexed with the term *"economic"*.

Now, what would be the result after combining terms with Boolean AND Operator? It will define a document set that is smaller than or equal to the document sets of any of the single terms. For example, the query with terms *"social"* and *"economic"* will produce the documents set of documents that are indexed with both the terms. In other words, document set with the intersection of both the sets.

Now, what would be the result after combining terms with Boolean OR operator? It will define a document set that is bigger than or equal to the document sets of any of the single terms. For example, the query with terms *"social"* or *"economic"* will produce the documents set of documents that are indexed with either the term *"social"* or *"economic"*. In other words, document set with the union of both the sets.

way to avoid linearly scanning the texts for each query is to index the documents in advance. Let us stick with Shakespeare's Collected Works, and use it to introduce the basics of the Boolean retrieval model. Suppose we record for each document – here a play of Shakespeare's – whether it contains each word out of all the words Shakespeare used (Shakespeare used about 32,000 different words). The result is a binary term-document incidence, as in Figure. Terms are the indexed units; they are usually words, and for the moment you can think of them as words.



	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
Antony	1	1	0	0	0	1	
Brutus	1	1	0	1	0	0	
Caesar	1	1	0	1	1	1	
Calpurnia	0	1	0	0	0	0	
Cleopatra	1	0	0	0	0	0	
mercy	1	0	1	1	1	1	
worser	1	0	1	1	1	0	
...							

Figure : A term-document incidence matrix. Matrix element (t, d) is 1 if the play in column d contains the word in row t, and is 0 otherwise.

To answer the query Brutus AND Caesar AND NOT Calpurnia, we take the vectors for Brutus, Caesar and Calpurnia, complement the last, and then do a bitwise AND:

110100 AND 110111 AND 101111 = 100100

Answer : The answers for this query are thus Antony and Cleopatra and Hamlet Let us now consider a more realistic scenario, simultaneously using the opportunity to introduce some terminology and notation. Suppose we have N = 1 million documents. **By documents we mean whatever units we have decided to build a retrieval system over**. They might be individual memos or chapters of a book. We will refer to the group of documents over which we perform retrieval as the **COLLECTION**. It is sometimes also referred to as a **Corpus.**

we assume an average of **6 bytes per word** including spaces and punctuation, then this is a document collection about 6 GB in size. Typically, there might be about **M = 500,000** distinct terms in these documents. There is nothing special about the numbers we have chosen, and they might vary by an order of magnitude or more, but they give us some idea of the dimensions of the kinds of problems we need to handle.

Advantages of the Boolean Mode

The advantages of the Boolean model are as follows −

- The simplest model, which is based on sets.
- Easy to understand and implement.
- It only retrieves exact matches
- It gives the user, a sense of control over the system.

Disadvantages of the Boolean Model

The disadvantages of the Boolean model are as follows −

- The model’s similarity function is Boolean. Hence, there would be no partial matches. This can be annoying for the users.
- In this model, the Boolean operator usage has much more influence than a critical word.

- The query language is expressive, but it is complicated too.
- No ranking for retrieved documents.

TF-IDF (Term Frequency/Inverse Document Frequency)

Term Frequency (tf_{ij})

It may be defined as the number of occurrences of w_i in d_j. The information that is captured by term frequency is how salient a word is within the given document or in other words we can say that the higher the term frequency the more that word is a good description of the content of that document.

Document Frequency (df_i)

It may be defined as the total number of documents in the collection in which w_i occurs. It is an indicator of informativeness. Semantically focused words will occur several times in the document unlike the semantically unfocused words.

Assign to each term in a document a weight for that term, that depends on the number of occurrences of the term in the document. We would like to compute a score between a query term t and a document d, based on the weight of t in d. The simplest approach is to assign the weight to be equal to the number of occurrences of term t in document d. This weighting scheme is referred to as term frequency and is denoted $tf_{t,d}$ with the subscripts denoting the term and the document in order.

Inverse document frequency

This is another form of document frequency weighting and often called idf weighting or inverse document frequency weighting. The important point of idf weighting is that the term's scarcity across the collection is a measure of its importance and importance is inversely proportional to frequency of occurrence.

Raw term frequency as above suffers from a critical problem: all terms are considered equally important when it comes to assessing relevancy on a query. In fact certain terms have little or no discriminating power in determining relevance. For instance, a collection of documents on the auto industry is likely to have the term auto in almost every document. A mechanism for attenuating the effect of terms that occur too often in the collection to be meaningful for relevance determination. An immediate idea is to scale down the term weights of terms with high collection frequency, defined to be the total number of occurrences of a term in the collection. The idea would be to reduce the tf weight of a term by a factor that grows with its collection frequency.

Mathematically,

$$idf_t = log\left(1 + \frac{N}{n_t}\right)$$

$$idf_t = log\left(\frac{N - n_t}{n_t}\right)$$

Here,

N = documents in the collection

n_t = documents containing term *t*

Tf-idf weighting

We now combine the definitions of term frequency and inverse document frequency, to produce a composite weight for each term in each document.

The tf-idf weighting scheme assigns to term t a weight in document d given by

$$\text{tf-idf}_{t,d} = \text{tf}_{t,d} \times \text{idf}_t.$$

In other words, tf-idf$_{t,d}$ assigns to term t a weight in document d that is

1. highest when t occurs many times within a small number of documents (thus lending high discriminating power to those documents);
2. lower when the term occurs fewer times in a document, or occurs in many documents (thus offering a less pronounced relevance signal);
3. lowest when the term occurs in virtually all documents.

VECTOR MODEL

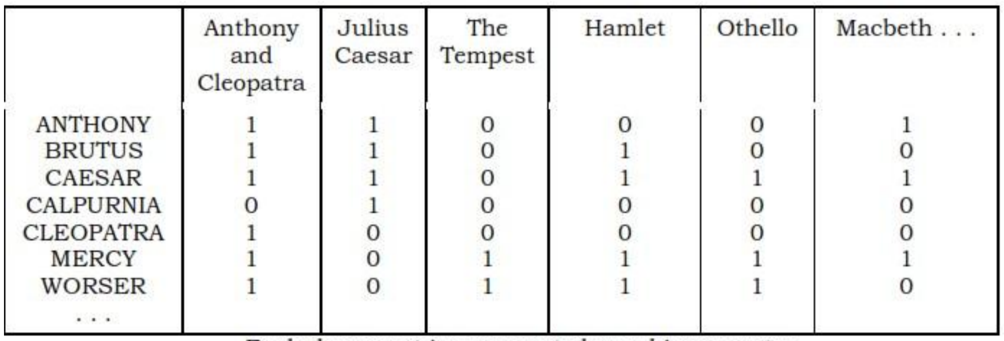
Assign non-binary weights to index terms in queries and in documents. Compute the similarity between documents and query. More precise than Boolean model.

Due to the disadvantages of the Boolean model, Gerard Salton and his colleagues suggested a model, which is based on Luhn's similarity criterion. The similarity criterion formulated by Luhn states, "the more two representations agreed in given elements and their distribution, the higher would be the probability of their representing similar information."

Consider the following important points to understand more about the Vector Space Model −

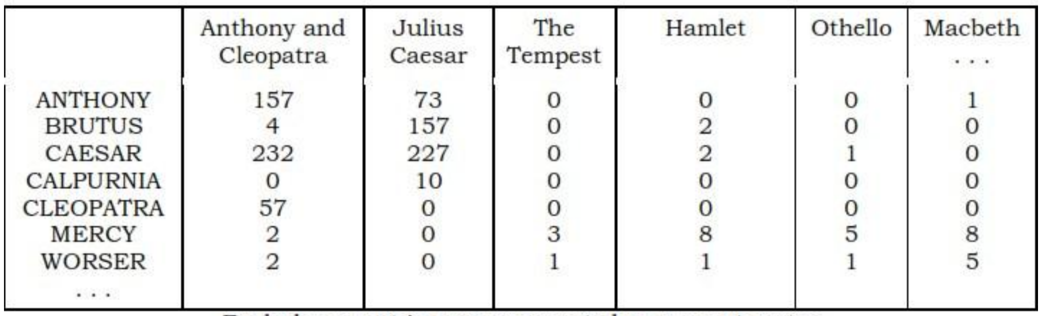
- The index representations (documents) and the queries are considered as vectors embedded in a high dimensional Euclidean space.
- The similarity measure of a document vector to a query vector is usually the cosine of the angle between them.

Binary values are changed into count values , later it is represented as weight matrix



	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth . . .
ANTHONY	1	1	0	0	0	1
BRUTUS	1	1	0	1	0	0
CAESAR	1	1	0	1	1	1
CALPURNIA	0	1	0	0	0	0
CLEOPATRA	1	0	0	0	0	0
MERCY	1	0	1	1	1	1
WORSER	1	0	1	1	1	0
. . .						

Each document is represented as a binary vector



	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth . . .
ANTHONY	157	73	0	0	0	1
BRUTUS	4	157	0	2	0	0
CAESAR	232	227	0	2	1	0
CALPURNIA	0	10	0	0	0	0
CLEOPATRA	57	0	0	0	0	0
MERCY	2	0	3	8	5	8
WORSER	2	0	1	1	1	5
. . .						

Each document is now represented as a count vector

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
ANTONY	5.255	3.18	0	0	0	0.35
BRUTUS	1.21	6.1	0	1	0	0
CAESAR	8.59	2.54	0	1.51	0.25	1
CALPURNIA	0	1.54	0	0	0	0
CLEOPATRA	2.85	0	0	0	0	0
MERCY	1.51	0	1.9	0.12	5.25	0.88
WORSER	1.37	0	0.11	4.15	0.25	1.95

Document represented by tf-idf weight vector

Binary →Count → Weight Matrix

Documents as Vectors

Each document is now represented as a real-valued vector of tf-idf weights ∈ R|V|.
So we have a |V|-dimensional real-valued vector space. Terms are axes of the space.
Documents are points or vectors in this space.
Very high-dimensional: tens of millions of dimensions when you apply this to web search engines. Each vector is very sparse - most entries are zero.

To represent the document as vector , We can consider
each document as → a vector
each term of doc → one component of vector
weight for each component is given by TFIDF(t, d) = TF(t, d) x IDF(t)

Queries as Vectors

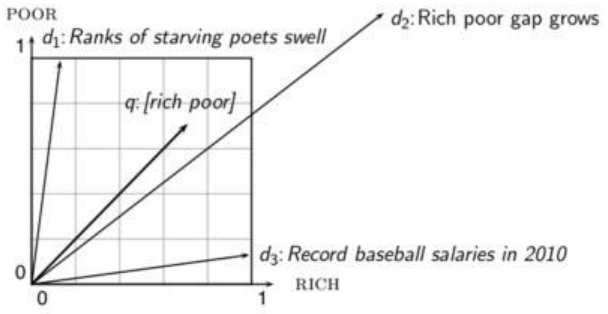
Key idea 1: Represent queries as vectors in same space
Key idea 2: Rank documents according to proximity to query in this space
proximity = similarity of vectors
proximity ≈ inverse of distance
Get away from Boolean model , Rank more relevant documents higher than less relevant documents

Formalizing Vector Space Proximity

We start by calculating Euclidean distance:

$$\left\| q - d_n \right\|_2$$

Euclidean distance is a bad idea . . . because Euclidean distance is large for vectors of different lengths


POOR
d_1: Ranks of starving poets swell
d_2: Rich poor gap grows
q: [rich poor]
d_3: Record baseball salaries in 2010
RICH
0
1
0
1

Cosine Similarity Measure Formula

Cosine is a normalized dot product, which can be calculated with the help of the following formula –

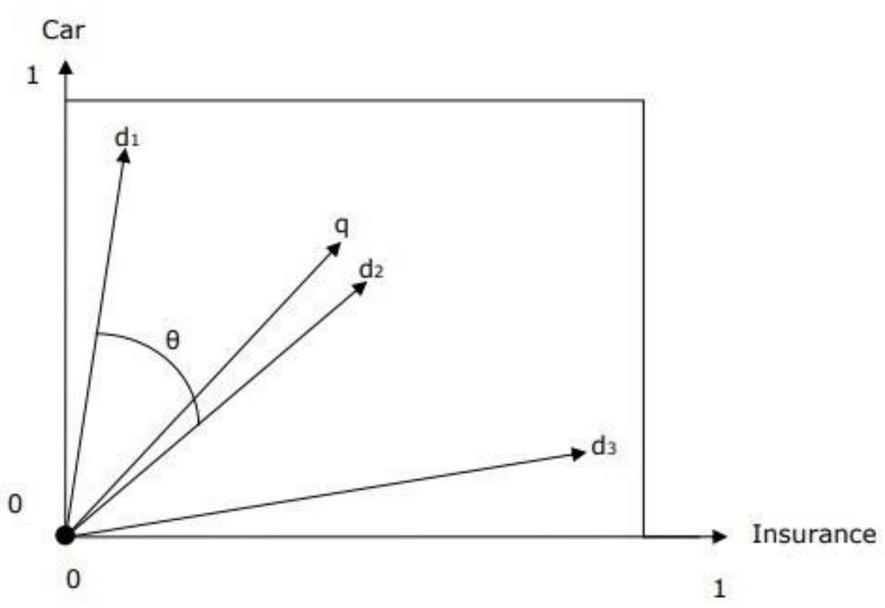
$$Score(\vec{d}\vec{q}) = \frac{\sum_{k=1}^{m} d_k \cdot q_k}{\sqrt{\sum_{k=1}^{m} (d_k)^2} \cdot \sqrt{\sum_{k=1}^{m} m (q_k)^2}}$$

$$Score(\vec{d}\vec{q}) = 1 \; when \; d = q$$

$$Score(\vec{d}\vec{q}) = 0 \; when \; d \; and \; q \; share \; no \; items$$

Vector Space Representation with Query and Document

The query and documents are represented by a two-dimensional vector space. The terms are ***car*** and ***insurance***. There is one query and three documents in the vector space.



The top ranked document in response to the terms car and insurance will be the document **d$_2$** because the angle between **q** and **d$_2$** is the smallest. The reason behind this is that both the concepts car and insurance are salient in d$_2$ and hence have the high weights. On the other side, **d$_1$** and **d$_3$** also mention both the terms but in each case, one of them is not a centrally important term in the document.

PROBABILISTIC MODEL

The probabilistic model tries to estimate the probability that the user will find the document dj relevant with ratio

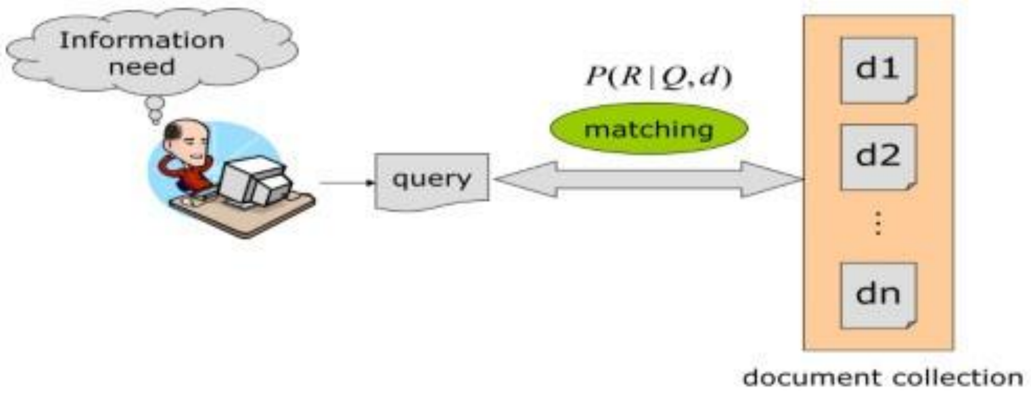
P(dj relevant to q) / P(dj non relevant to q)

Given a user query q, and the ideal answer set R of the relevant documents, the problem is to specify the properties for this set. Assumption (probabilistic principle): the probability of relevance depends on the query and document representations only; ideal answer set R should maximize the overall probability of relevance.

Given a query q, there exists a subset of the documents R which are relevant to q But membership of R is uncertain (not sure) , A Probabilistic retrieval model ranks documents in decreasing order of probability of relevance to the information need: P(R | q,d I). Users gives with ***information needs***, which they translate into ***query representations***. Similarly, there are ***documents***, which are converted into ***document representations*** . Given only a query, an IR system has an uncertain understanding of the information need. So IR is an uncertain process , Because,

- Information need to query
- Documents to index terms
- Query terms and index terms mismatch

Probability theory provides a principled foundation for such reasoning under uncertainty. This model provides how likely a document is relevant to an information need.



Document can be relevant and non relevant document, we can estimate the probability of a term t appearing in a relevant document P(t | R=1) . Probabilistic methods are one of the oldest but also one of the currently hottest topics in IR .

Basic probability theory

For events A , the probability of the event lies between 0= P(A) = 1 , For 2 events A and B

- Joint probability P(***A***, ***B***) of both events occurring
- Conditional probability P(***A***|***B***) of event ***A*** occurring given that event B has occurred
- Chain rule gives fundamental relationship between joint and conditional probabilities:

$$P(A,B) = P(A \cap B) = P(A|B)P(B) = P(B|A)P(A)$$

Similarly for the complement of an event P(A,B) :

$$P(\overline{A},B) = P(B|\overline{A})P(\overline{A})$$

Partition rule: if B can be divided into an exhaustive set of disjoint sub cases, then P(B) is the sum of the probabilities of the sub cases. A special case of this rule gives:

$$P(B) = P(A,B) + P(\overline{A},B)$$

- Bayes' Rule for inverting conditional probabilities:

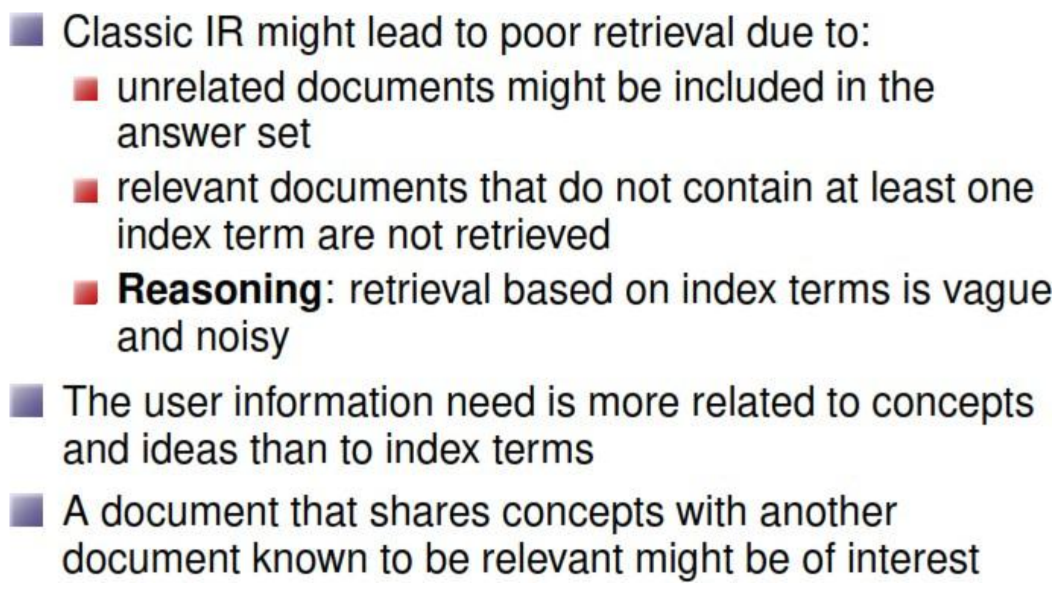
$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} = \left[\frac{P(B|A)}{\sum_{X\in\{A,\overline{A}\}} P(B|X)P(X)}\right] P(A)$$

  Can be thought of as a way of updating probabilities:
  - Start off with prior probability $P(A)$ (initial estimate of how likely event A is in the absence of any other information)
  - Derive a posterior probability $P(A|B)$ after having seen the evidence B, based on the likelihood of B occurring in the two cases that A does or does not hold
- Odds of an event (is the ratio of the probability of an event to the probability of its complement.) provide a kind of multiplier for how probabilities change:

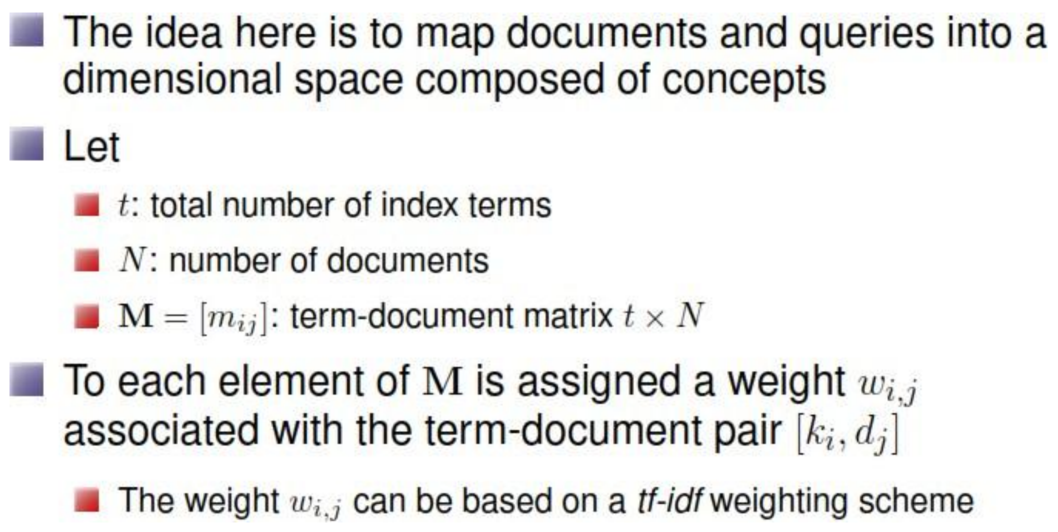
$$\textbf{Odds:} \quad O(A) = \frac{P(A)}{P(\overline{A})} = \frac{P(A)}{1-P(A)}$$

LATENT SEMANTIC INDEXING MODEL

Latent semantic indexing (**LSI**) is an indexing and retrieval method that uses a mathematical technique called singular value decomposition (SVD) to identify patterns in the relationships between the terms and concepts contained in an unstructured collection of text. LSI is based on the principle that words that are used in the same contexts tend to have similar meanings. A key feature of LSI is its ability to extract the conceptual content of a body of text by establishing associations between those terms that occur in similar contexts.



- Classic IR might lead to poor retrieval due to:
  - unrelated documents might be included in the answer set
  - relevant documents that do not contain at least one index term are not retrieved
  - **Reasoning**: retrieval based on index terms is vague and noisy
- The user information need is more related to concepts and ideas than to index terms
- A document that shares concepts with another document known to be relevant might be of interest



- The idea here is to map documents and queries into a dimensional space composed of concepts
- Let
  - t: total number of index terms
  - N: number of documents
  - $\mathbf{M} = [m_{ij}]$: term-document matrix $t \times N$
- To each element of $\mathbf{M}$ is assigned a weight $w_{i,j}$ associated with the term-document pair $[k_i, d_j]$
  - The weight $w_{i,j}$ can be based on a *tf-idf* weighting scheme

- The matrix $\mathbf{M} = [m_{ij}]$ can be decomposed into three components using singular value decomposition

$$\mathbf{M} = \mathbf{K} \cdot \mathbf{S} \cdot \mathbf{D}^T$$

- were
  - $\mathbf{K}$ is the matrix of eigenvectors derived from $\mathbf{C} = \mathbf{M} \cdot \mathbf{M}^T$
  - $\mathbf{D}^T$ is the matrix of eigenvectors derived from $\mathbf{M}^T \cdot \mathbf{M}$
  - $\mathbf{S}$ is an $r \times r$ diagonal matrix of singular values where $r = \min(t, N)$ is the rank of $\mathbf{M}$

Computing an Example

- Let $\mathbf{M}^T = [m_{ij}]$ be given by

	K_1	K_2	K_3	$q \bullet d_j$
d_1	2	0	1	5
d_2	1	0	0	1
d_3	0	1	3	11
d_4	2	0	0	2
d_5	1	2	4	17
d_6	1	2	0	5
d_7	0	5	0	10
q	1	2	3	

- Compute the matrices $\mathbf{K}$, $\mathbf{S}$, and $\mathbf{D}^t$

- In the matrix $\mathbf{S}$, consider that only the s largest singular values are selected
- Keep the corresponding columns in $\mathbf{K}$ and $\mathbf{D}^T$
- The resultant matrix is called $\mathbf{M}_s$ and is given by

$$\mathbf{M}_s = \mathbf{K}_s \cdot \mathbf{S}_s \cdot \mathbf{D}_s^T$$

- where s, $s < r$, is the dimensionality of a reduced concept space
- The parameter s should be
  - large enough to allow fitting the characteristics of the data
  - small enough to filter out the non-relevant representational details

Latent Ranking

- The relationship between any two documents in s can be obtained from the $\mathbf{M}_s^T \cdot \mathbf{M}_s$ matrix given by

$$\begin{aligned}
\mathbf{M}_s^T \cdot \mathbf{M}_s &= (\mathbf{K}_s \cdot \mathbf{S}_s \cdot \mathbf{D}_s^T)^T \cdot \mathbf{K}_s \cdot \mathbf{S}_s \cdot \mathbf{D}_s^T \\
&= \mathbf{D}_s \cdot \mathbf{S}_s \cdot \mathbf{K}_s^T \cdot \mathbf{K}_s \cdot \mathbf{S}_s \cdot \mathbf{D}_s^T \\
&= \mathbf{D}_s \cdot \mathbf{S}_s \cdot \mathbf{S}_s \cdot \mathbf{D}_s^T \\
&= (\mathbf{D}_s \cdot \mathbf{S}_s) \cdot (\mathbf{D}_s \cdot \mathbf{S}_s)^T
\end{aligned}$$

- In the above matrix, the (i, j) element quantifies the relationship between documents d_i and d_j

- The user query can be modelled as a pseudo-document in the original M matrix
- Assume the query is modelled as the document numbered 0 in the M matrix
- The matrix $\mathrm{M}_s^T \cdot \mathrm{M}_s$ quantifies the relationship between any two documents in the reduced concept space
- The first row of this matrix provides the rank of all the documents with regard to the user query

Problems with Lexical Semantics

- Ambiguity and association in natural language
  - **Polysemy**: Words often have a **multitude of meanings** and different types of usage (more severe in very heterogeneous collections).
  - The vector space model is unable to discriminate between different meanings of the same word.

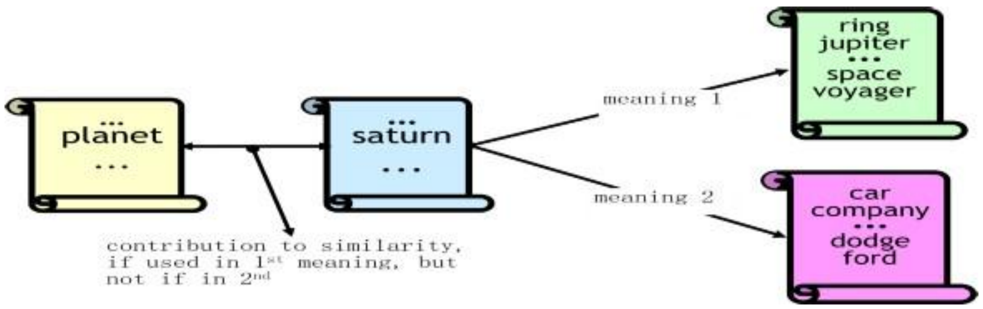
$$\mathrm{sim}_{\mathrm{true}}(d, q) < \cos(\angle(\vec{d}, \vec{q}))$$

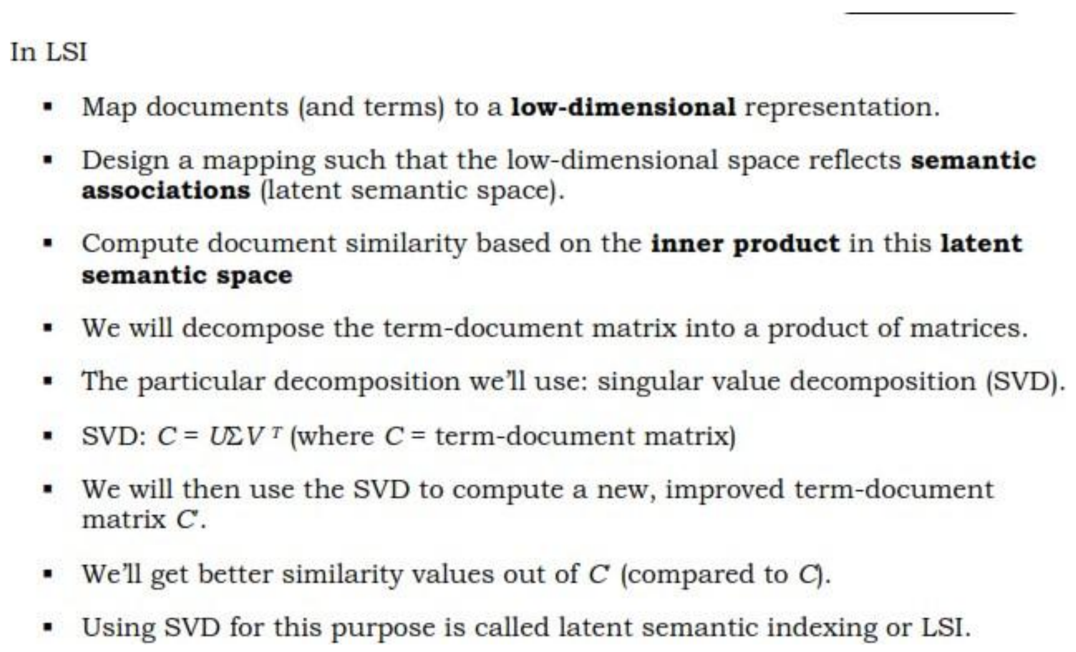
  - Synonymy: Different terms may have an identical or a similar meaning (weaker: words indicating the same topic).
  - No associations between words are made in the vector space representation.

$$\mathrm{sim}_{\mathrm{true}}(d, q) > \cos(\angle(\vec{d}, \vec{q}))$$

Polysemy and Context

- Document similarity on single word level: polysemy and context
- LSI Perform a **low-rank approximation** of **document-term matrix**) .





In LSI

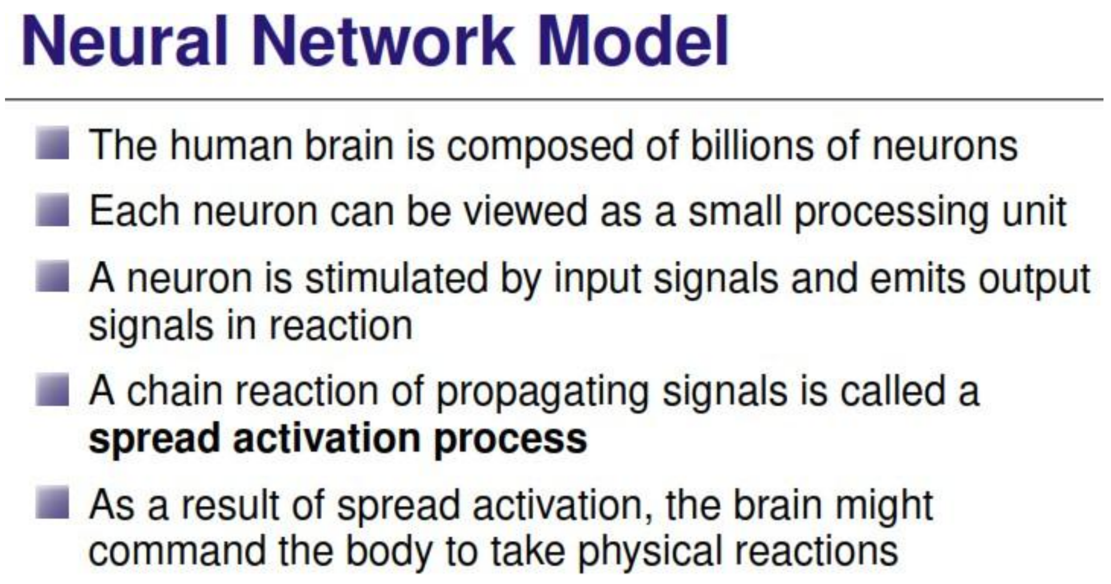
- Map documents (and terms) to a **low-dimensional** representation.
- Design a mapping such that the low-dimensional space reflects **semantic associations** (latent semantic space).
- Compute document similarity based on the **inner product** in this **latent semantic space**
- We will decompose the term-document matrix into a product of matrices.
- The particular decomposition we'll use: singular value decomposition (SVD).
- SVD: $C = U\Sigma V^T$ (where C = term-document matrix)
- We will then use the SVD to compute a new, improved term-document matrix C'.
- We'll get better similarity values out of C' (compared to C).
- Using SVD for this purpose is called latent semantic indexing or LSI.

NEURAL NETWORK MODEL

Neural Network Definition

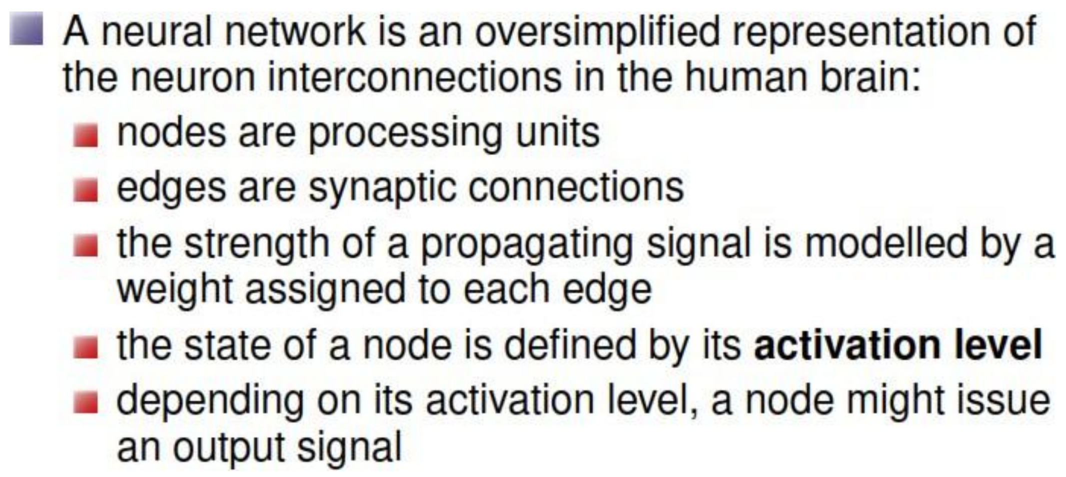
Neural networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. They interpret sensory data through a kind of machine perception, labeling or clustering raw input. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text or time series, must be translated.

Neural networks help us cluster and classify. You can think of them as a clustering and classification layer on top of the data you store and manage. They help to group unlabeled data according to similarities among the example inputs, and they classify data when they have a labeled dataset to train on. (Neural networks can also extract features that are fed to other algorithms for clustering and classification; so you can think of deep neural networks as components of larger machine-learning applications involving algorithms for reinforcement learning, classification and regression.)



Neural Network Model

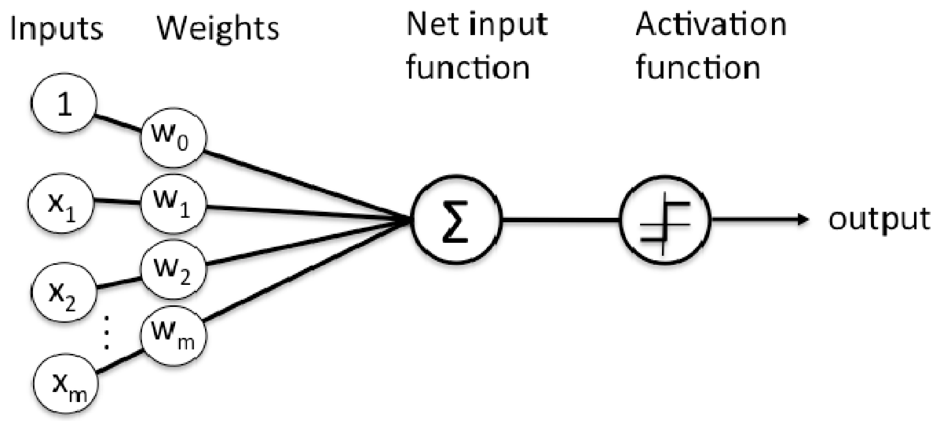
- The human brain is composed of billions of neurons
- Each neuron can be viewed as a small processing unit
- A neuron is stimulated by input signals and emits output signals in reaction
- A chain reaction of propagating signals is called a **spread activation process**
- As a result of spread activation, the brain might command the body to take physical reactions



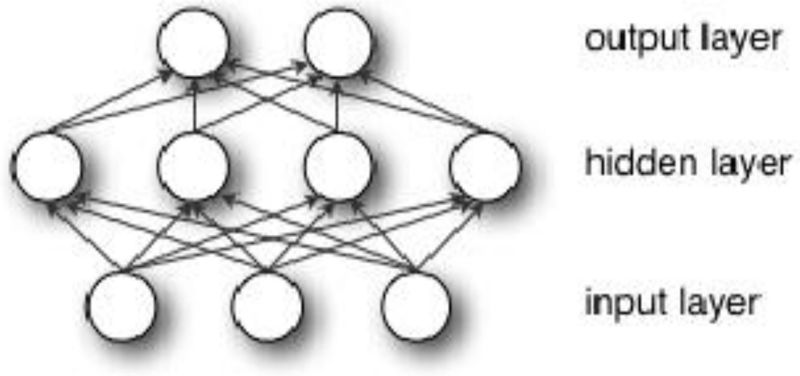
- A neural network is an oversimplified representation of the neuron interconnections in the human brain:
  - nodes are processing units
  - edges are synaptic connections
  - the strength of a propagating signal is modelled by a weight assigned to each edge
  - the state of a node is defined by its **activation level**
  - depending on its activation level, a node might issue an output signal

Neural Network Elements

Deep learning is the name we use for “stacked neural networks”; that is, networks composed of several layers. The layers are made of *nodes*. A node is just a place where computation happens, loosely patterned on a neuron in the human brain, which fires when it encounters sufficient stimuli. A node combines input from the data with a set of coefficients, or weights, that either amplify or dampen that input, thereby assigning significance to inputs with regard to the task the algorithm is trying to learn; e.g. which input is most helpful is classifying data without error? These input-weight products are summed and then the sum is passed through a node's so-called activation function, to determine whether and to what extent that signal should progress further through the network to affect the ultimate outcome, say, an act of classification. If the signals passes through, the neuron has been "activated." Here's a diagram of what one node might look like.



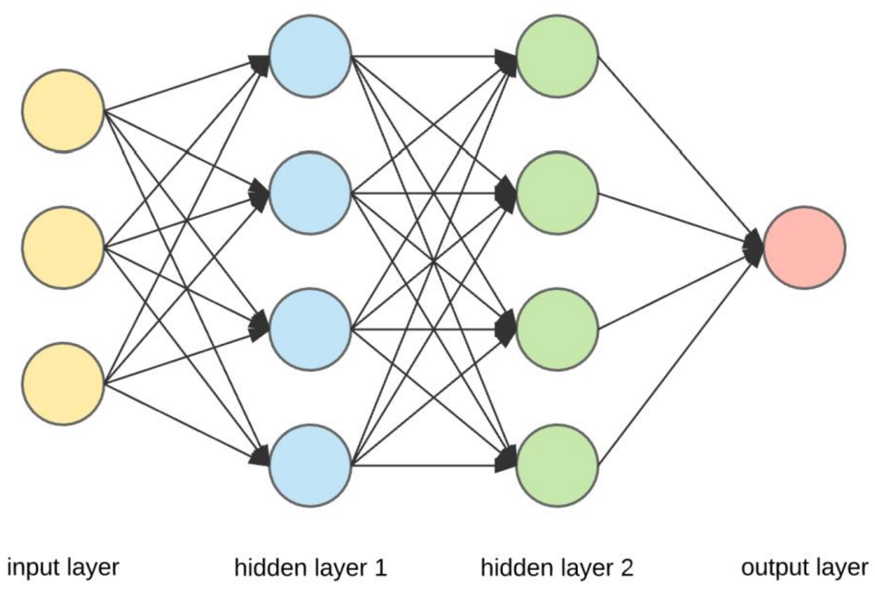
A node layer is a row of those neuron-like switches that turn on or off as the input is fed through the net. Eac   layer's output is simultaneously the subsequent layer's input, starting from an initial input layer receiving your data.



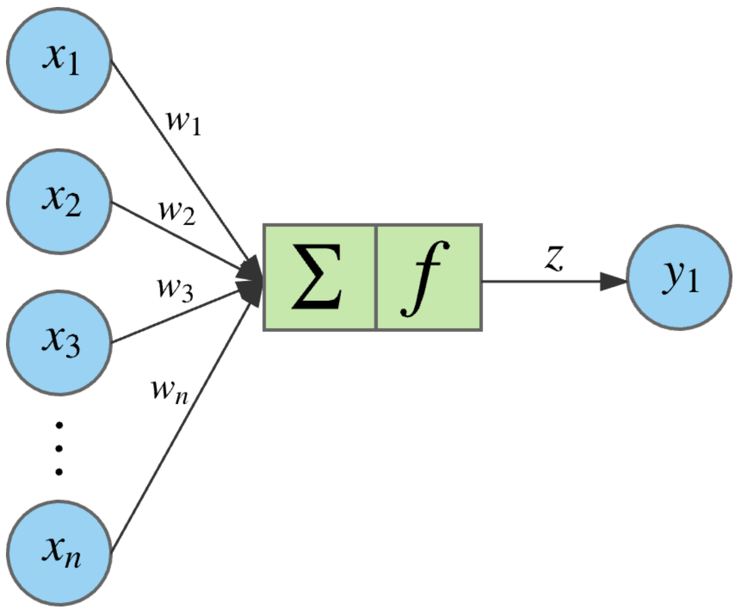
Pairing the model's adjustable weights with input features is how we assign significance to those features with regard to how the neural network classifies and clusters input.

What is an Artificial Neu al Network Model?

A multi-layer, fully-connected neural network containing an input layer, hidden layers, and an output layer is called an artificial neural network or ANN. The image below represents an ANN.



If you see carefully, you will notice that each node in one layer is connected to every node in the layer next to it. As you increase the number of hidden layers, the network becomes deeper. Let's see what an individual node in the output or hidden layer looks like.



As you can see, the node gets many inputs. It sums up all the weights and passes it on as output, via a non-linear activation function. This output of the node becomes the input of the node in the next layer.

Glossary of Artificial Neural Network Model

Let's look at the basic terms you should know when it comes to an artificial neural network model.

Inputs

The data first fed into the neural network from the source is called the input. Its goal is to give the network data to make a decision or prediction about the information fed into it. The neural network model usually accepts real value sets of inputs and it should be fed into a neuron in the input layer.

Training set

The inputs for which you already know the correct outputs are called training sets. These are used to help the neural network get trained and memorize the result for the given input set.

Outputs

Every neural network generates an output as a prediction or decision about the data fed into it. This output is in the form of real values set or Boolean decisions. Only of the neurons in the output layer generates the output value.

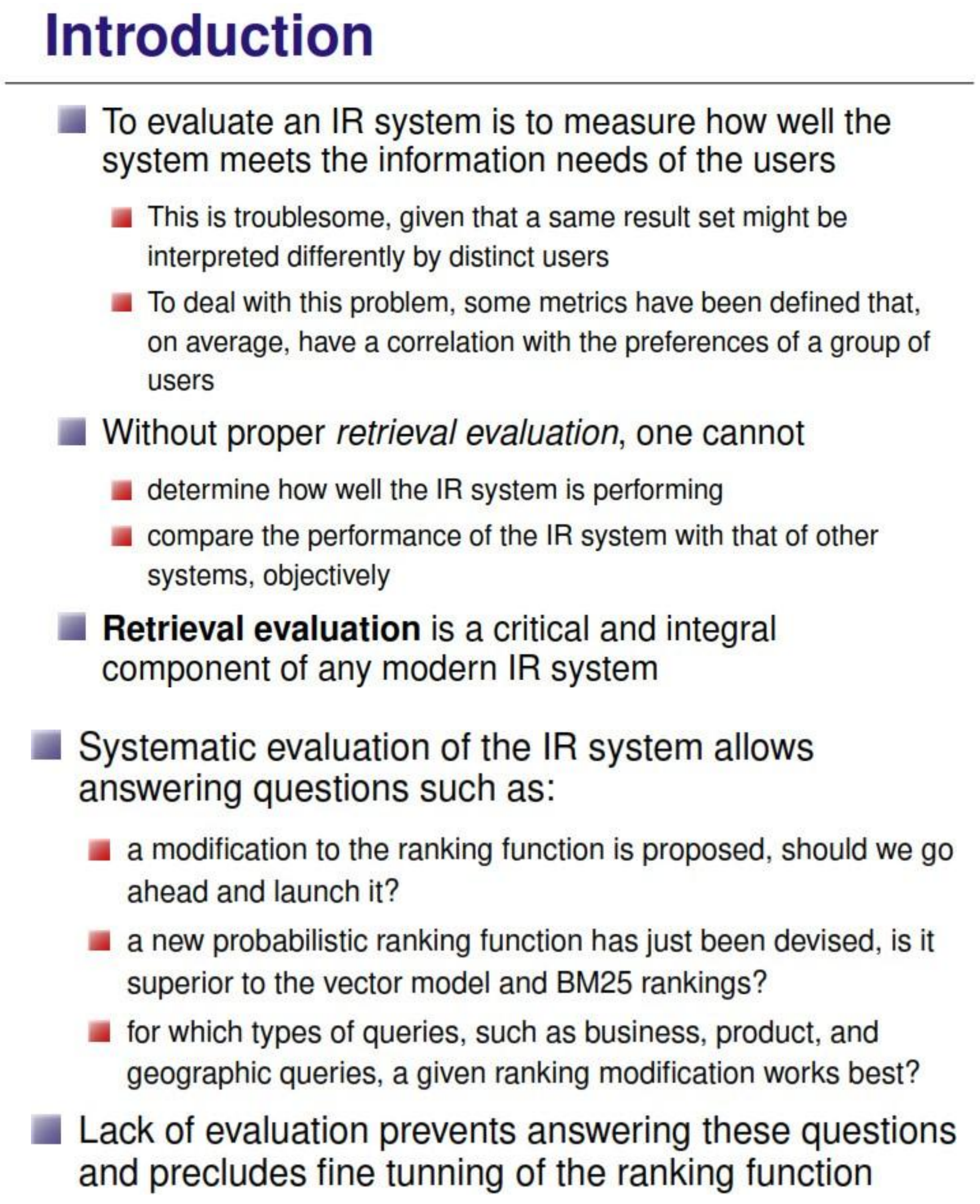
Neuron

Also known as a perceptron, a neuron is the basic unit of a neural network. It accepts an input value and generates an output based on it. As discussed before, every neuron receives a part of the input and passes it through the non-linear activation function to the node in the next layer. These activation functions can be TanH, sigmoid, or ReLu. The non-linear feature of these functions helps to train the network.

Weight space

Every neuron has a numeric weight. When it delivers input to another note, its weight is totaled with the others to generate an output. By making small changes to these weights are how neural networks are trained. The fine-tuning of weights helps determine the correct set of weights and biases that would generate the best outcome. This is where back propagation comes in.

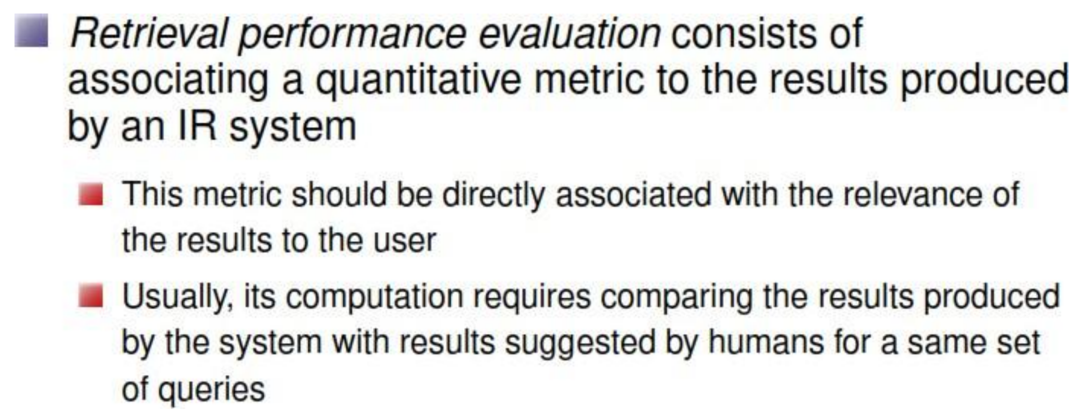
RETRIEVAL EVALUATION



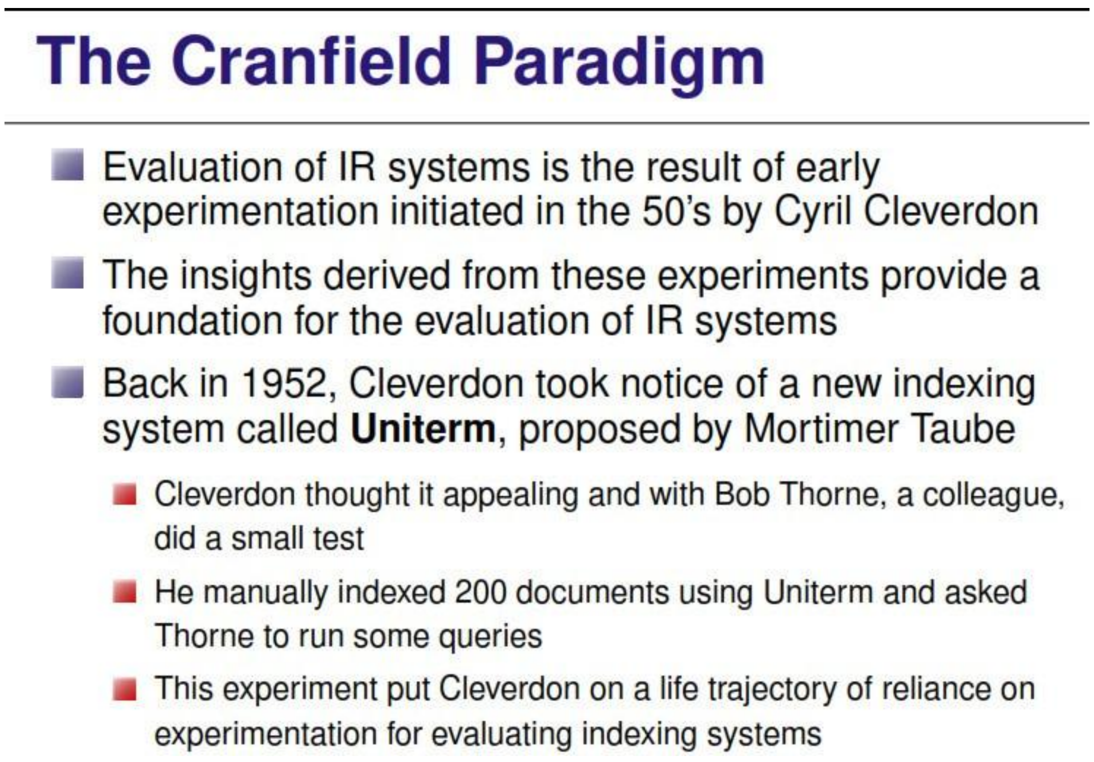
Introduction

- To evaluate an IR system is to measure how well the system meets the information needs of the users
  - This is troublesome, given that a same result set might be interpreted differently by distinct users
  - To deal with this problem, some metrics have been defined that, on average, have a correlation with the preferences of a group of users
- Without proper *retrieval evaluation*, one cannot
  - determine how well the IR system is performing
  - compare the performance of the IR system with that of other systems, objectively
- **Retrieval evaluation** is a critical and integral component of any modern IR system

- Systematic evaluation of the IR system allows answering questions such as:
  - a modification to the ranking function is proposed, should we go ahead and launch it?
  - a new probabilistic ranking function has just been devised, is it superior to the vector model and BM25 rankings?
  - for which types of queries, such as business, product, and geographic queries, a given ranking modification works best?
- Lack of evaluation prevents answering these questions and precludes fine tunning of the ranking function



- *Retrieval performance evaluation* consists of associating a quantitative metric to the results produced by an IR system
  - This metric should be directly associated with the relevance of the results to the user
  - Usually, its computation requires comparing the results produced by the system with results suggested by humans for a same set of queries

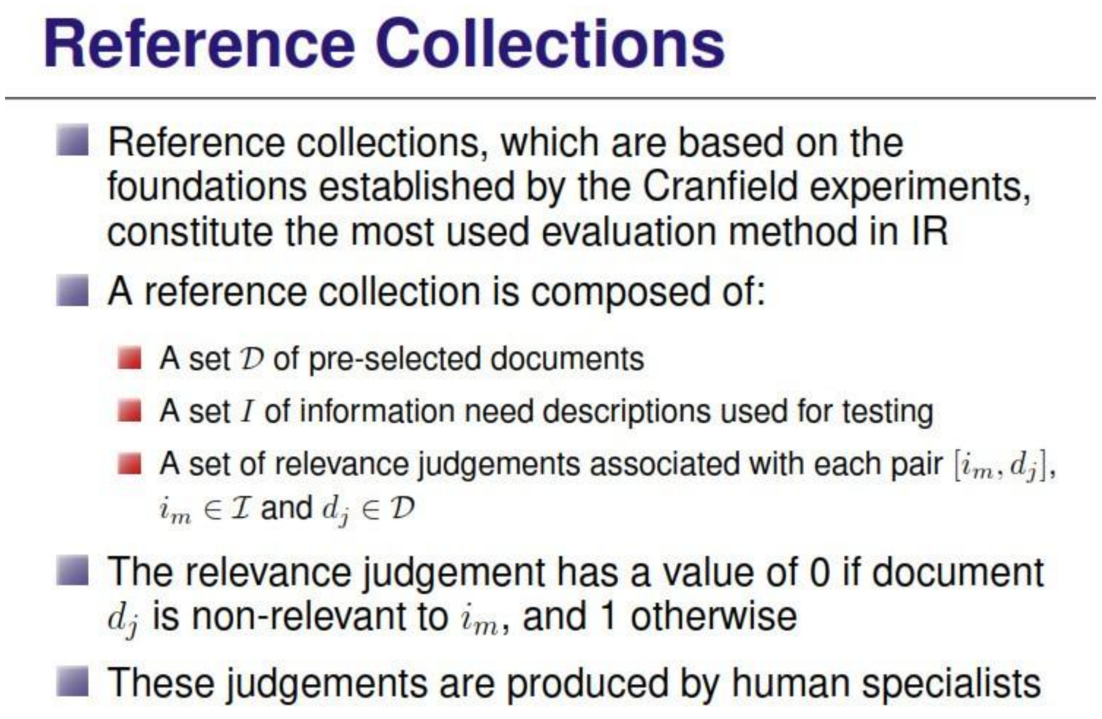


The Cranfield Paradigm

- Evaluation of IR systems is the result of early experimentation initiated in the 50's by Cyril Cleverdon
- The insights derived from these experiments provide a foundation for the evaluation of IR systems
- Back in 1952, Cleverdon took notice of a new indexing system called **Uniterm**, proposed by Mortimer Taube
  - Cleverdon thought it appealing and with Bob Thorne, a colleague, did a small test
  - He manually indexed 200 documents using Uniterm and asked Thorne to run some queries
  - This experiment put Cleverdon on a life trajectory of reliance on experimentation for evaluating indexing systems

- Cleverdon obtained a grant from the National Science Foundation to compare distinct indexing systems
- These experiments provided interesting insights, that culminated in the modern metrics of precision and recall
  - **Recall ratio:** the fraction of relevant documents retrieved
  - **Precision ration:** the fraction of documents retrieved that are relevant
- For instance, it became clear that, in practical situations, the majority of searches does not require high recall
- Instead, the vast majority of the users require just a few relevant answers

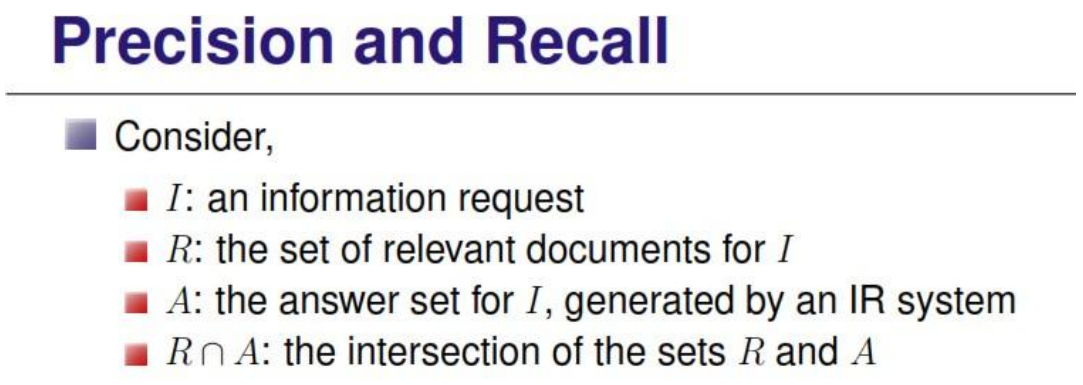
- The next step was to devise a set of experiments that would allow evaluating each indexing system in isolation more thoroughly
- The result was a **test reference collection** composed of documents, queries, and relevance judgements
  - It became known as the *Cranfield-2* collection
- The reference collection allows using the same set of documents and queries to evaluate different ranking systems
- The uniformity of this setup allows quick evaluation of new ranking functions



Reference Collections

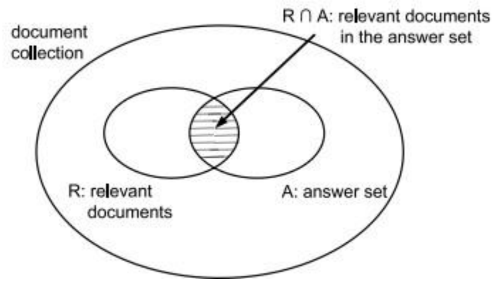
- Reference collections, which are based on the foundations established by the Cranfield experiments, constitute the most used evaluation method in IR
- A reference collection is composed of:
  - A set $\mathcal{D}$ of pre-selected documents
  - A set I of information need descriptions used for testing
  - A set of relevance judgements associated with each pair $[i_m, d_j]$, $i_m \in \mathcal{I}$ and $d_j \in \mathcal{D}$
- The relevance judgement has a value of 0 if document d_j is non-relevant to i_m, and 1 otherwise
- These judgements are produced by human specialists

RETRIEVAL METRICS - PRECISION AND RECALL



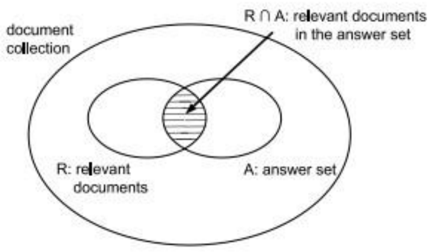
Precision and Recall

- Consider,
  - I: an information request
  - R: the set of relevant documents for I
  - A: the answer set for I, generated by an IR system
  - $R \cap A$: the intersection of the sets R and A



- The recall and precision measures are defined as follows
  - **Recall** is the fraction of the relevant documents (the set R) which has been retrieved i.e.,

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



  - **Precision** is the fraction of the retrieved documents (the set A) which is relevant i.e.,

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

- The definition of precision and recall assumes that all docs in the set A have been examined
- However, the user is not usually presented with all docs in the answer set A at once
  - User sees a ranked set of documents and examines them starting from the top
- Thus, precision and recall vary as the user proceeds with their examination of the set A
- Most appropriate then is to plot a **curve of precision versus recall**

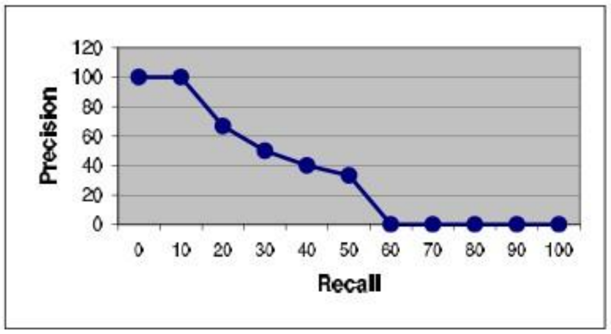
- Consider a reference collection and a set of test queries
- Let R_{q_1} be the set of relevant docs for a query q_1:
  - $R_{q_1} = \{d_3, d_5, d_9, d_{25}, d_{39}, d_{44}, d_{56}, d_{71}, d_{89}, d_{123}\}$
- Consider a new IR algorithm that yields the following answer to q_1 (relevant docs are marked with a bullet):

01. d_{123} •	06. d_9 •	11. d_{38}
02. d_{84}	07. d_{511}	12. d_{48}
03. d_{56} •	08. d_{129}	13. d_{250}
04. d_6	09. d_{187}	14. d_{113}
05. d_8	10. d_{25} •	15. d_3 •

- If we examine this ranking, we observe that
  - The document d_{123}, ranked as number 1, is relevant
    - This document corresponds to 10% of all relevant documents
    - Thus, we say that we have a precision of 100% at 10% recall
  - The document d_{56}, ranked as number 3, is the next relevant
    - At this point, two documents out of three are relevant, and two of the ten relevant documents have been seen
    - Thus, we say that we have a precision of 66.6% at 20% recall

01. d_{123} •	06. d_9 •	11. d_{38}
02. d_{84}	07. d_{511}	12. d_{48}
03. d_{56} •	08. d_{129}	13. d_{250}
04. d_6	09. d_{187}	14. d_{113}
05. d_8	10. d_{25} •	15. d_3 •

- If we proceed with our examination of the ranking generated, we can plot a curve of precision versus recall as follows:



Recall	Precision
0	100
10	100
20	66.6
30	50
40	40
50	33.3
60	0
70	0
80	0
90	0
100	0

- Consider now a second query q_2 whose set of relevant answers is given by

$$R_{q_2} = \{d_3, d_{56}, d_{129}\}$$

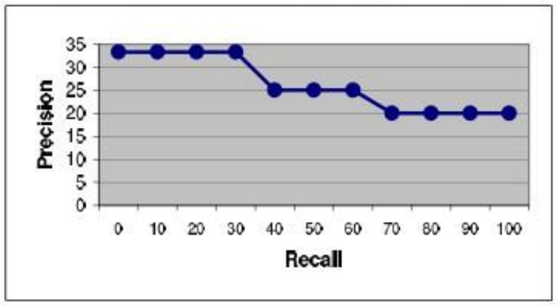
- The previous IR algorithm processes the query q_2 and returns a ranking, as follows

01. d_{425}	06. d_{615}	11. d_{193}
02. d_{87}	07. d_{512}	12. d_{715}
03. d_{56} •	08. d_{129} •	13. d_{810}
04. d_{32}	09. d_4	14. d_5
05. d_{124}	10. d_{130}	15. d_3 •

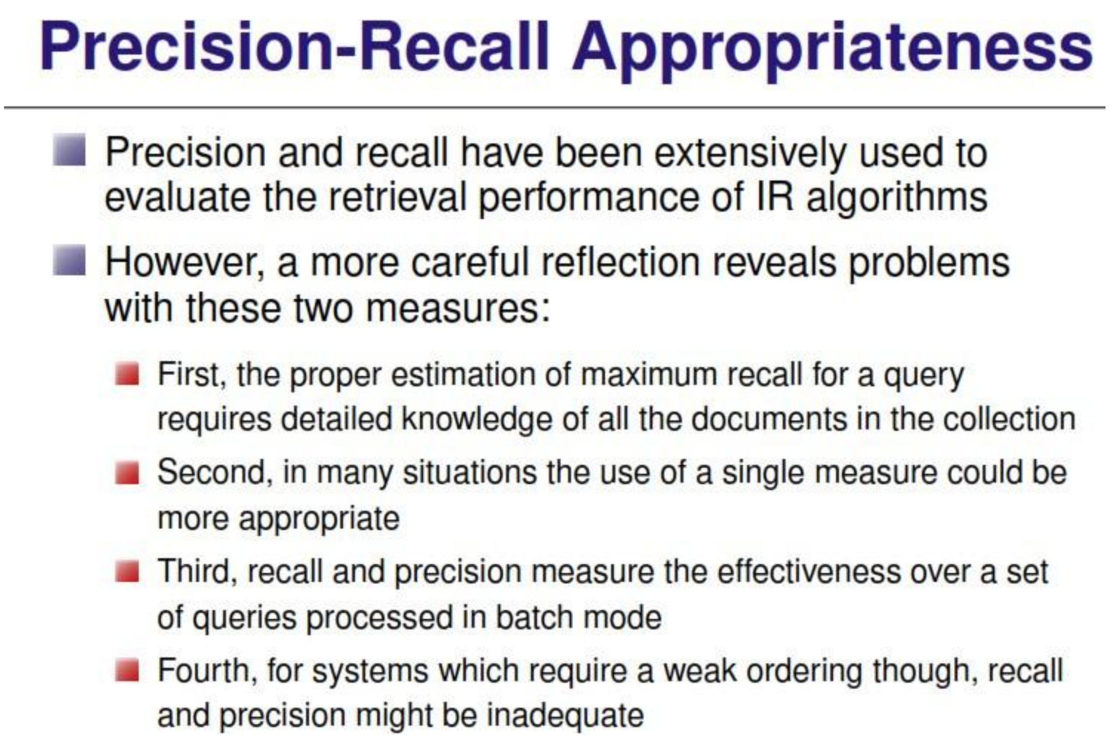
- If we examine this ranking, we observe
  - The first relevant document is d_{56}
    - It provides a recall and precision levels equal to 33.3%
  - The second relevant document is d_{129}
    - It provides a recall level of 66.6% (with precision equal to 25%)
  - The third relevant document is d_3
    - It provides a recall level of 100% (with precision equal to 20%)

01. d_{425}	06. d_{615}	11. d_{193}
02. d_{87}	07. d_{512}	12. d_{715}
03. d_{56} •	08. d_{129} •	13. d_{810}
04. d_{32}	09. d_4	14. d_5
05. d_{124}	10. d_{130}	15. d_3 •

- The precision figures at the 11 standard recall levels are interpolated as follows
- Let r_j, $j \in \{0, 1, 2, \ldots, 10\}$, be a reference to the j-th standard recall level
- Then, $\boxed{P(r_j) = max_{\forall r \mid r_j \leq r} \quad P(r)}$
- In our last example, this interpolation rule yields the precision and recall figures illustrated below



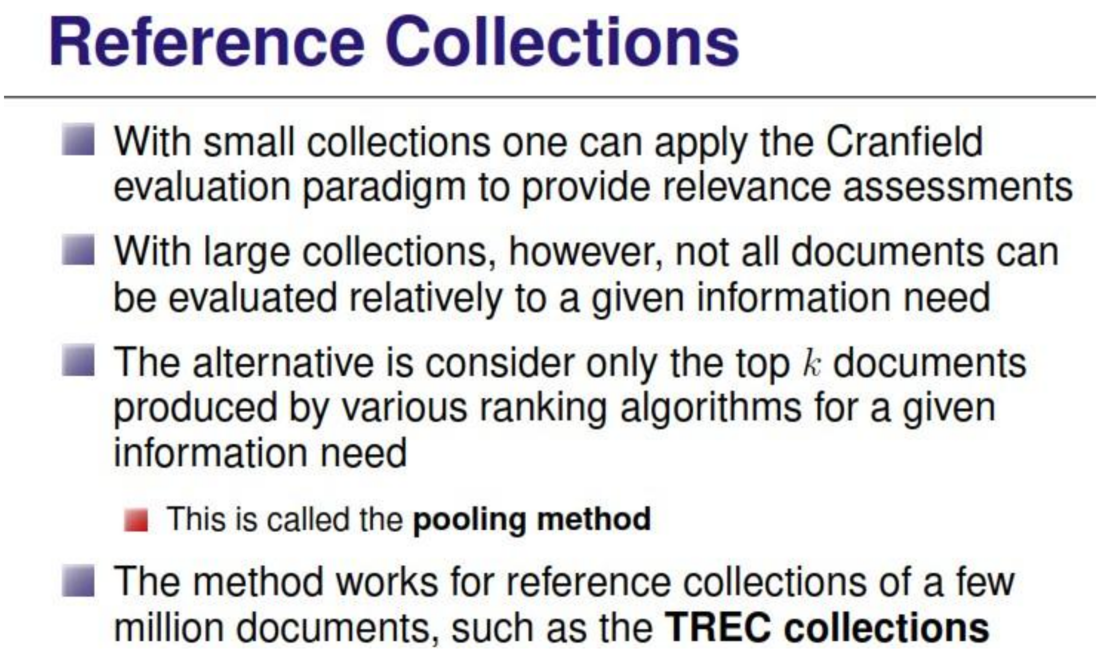
Recall	Precision
0	33.3
10	33.3
20	33.3
30	33.3
40	25
50	25
60	25
70	20
80	20
90	20
100	20



Precision-Recall Appropriateness

- Precision and recall have been extensively used to evaluate the retrieval performance of IR algorithms
- However, a more careful reflection reveals problems with these two measures:
  - First, the proper estimation of maximum recall for a query requires detailed knowledge of all the documents in the collection
  - Second, in many situations the use of a single measure could be more appropriate
  - Third, recall and precision measure the effectiveness over a set of queries processed in batch mode
  - Fourth, for systems which require a weak ordering though, recall and precision might be inadequate

REFERENCE COLLECTION



Reference Collections

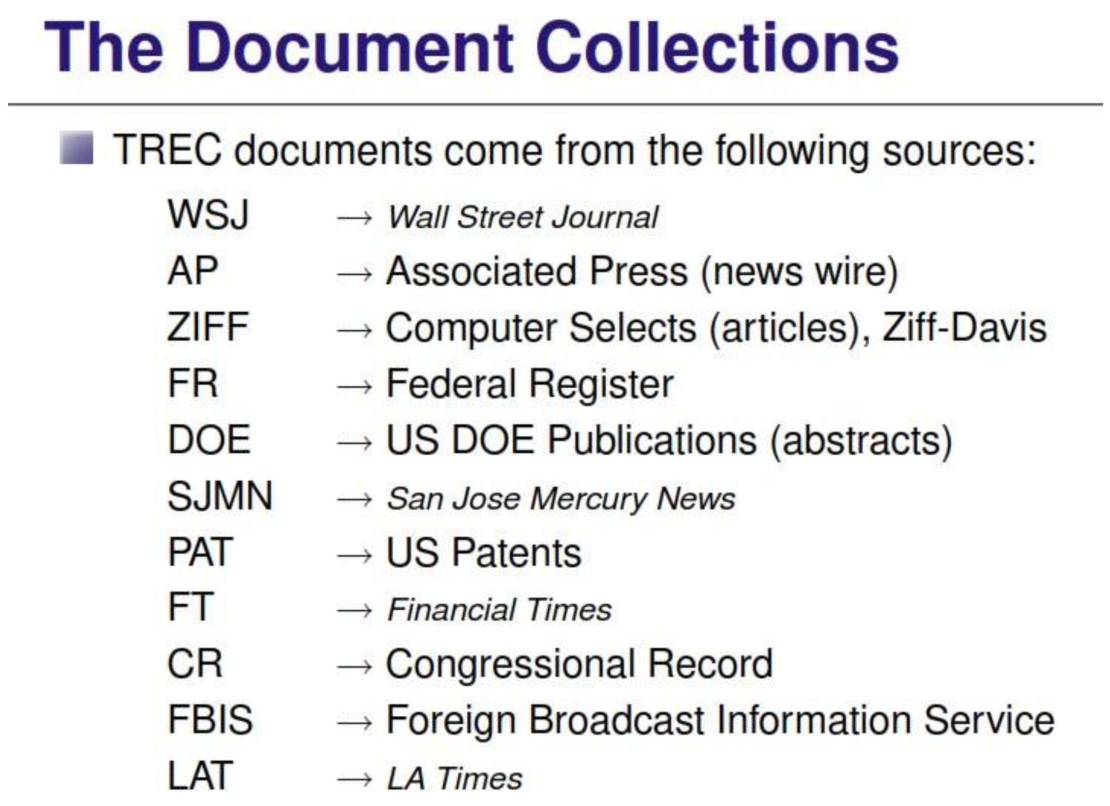
- With small collections one can apply the Cranfield evaluation paradigm to provide relevance assessments
- With large collections, however, not all documents can be evaluated relatively to a given information need
- The alternative is consider only the top k documents produced by various ranking algorithms for a given information need
  - This is called the **pooling method**
- The method works for reference collections of a few million documents, such as the **TREC collections**

The TREC Conferences

- TREC is an yearly promoted conference dedicated to experimentation with large test collections
- For each TREC conference, a set of experiments is designed
- The research groups that participate in the conference use these experiments to compare their retrieval systems
- As with most test collections, a TREC collection is composed of three parts:
  - the documents
  - the example information requests (called **topics**)
  - a set of relevant documents for each example information request

The Document Collections

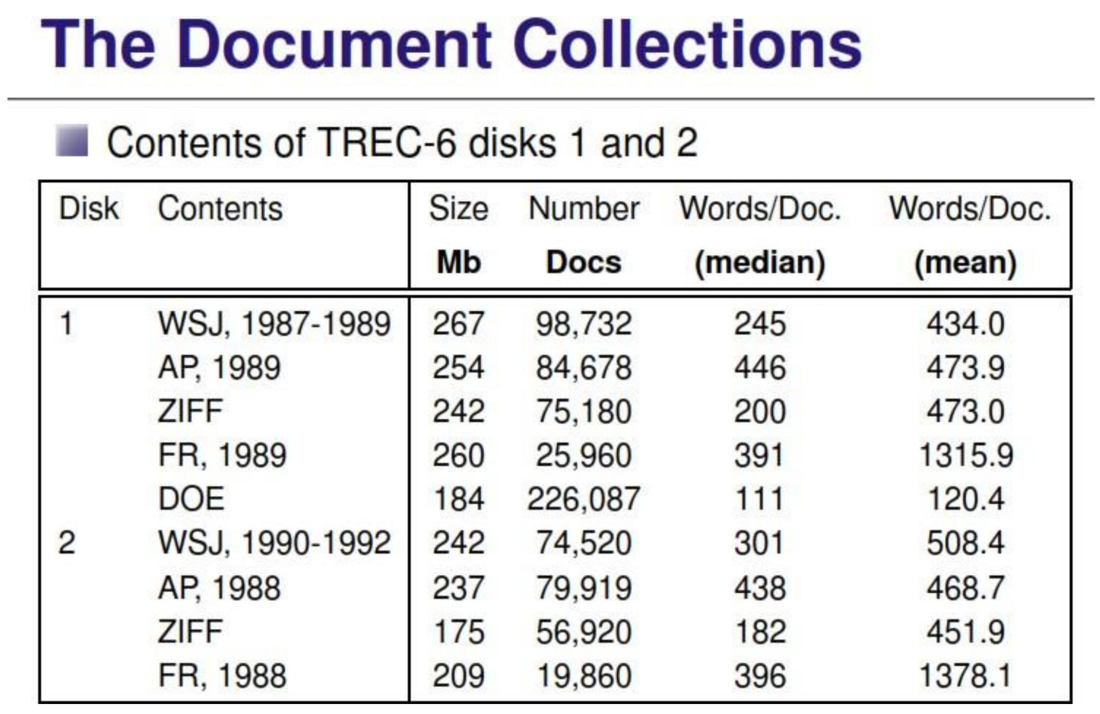
- The main TREC collection has been growing steadily over the years
- The TREC-3 collection has roughly 2 gigabytes
- The TREC-6 collection has roughly 5.8 gigabytes
  - It is distributed in 5 CD-ROM disks of roughly 1 gigabyte of compressed text each
  - Its 5 disks were also used at the TREC-7 and TREC-8 conferences
- The *Terabyte test collection* introduced at TREC-15, also referred to as GOV2, includes 25 million Web documents crawled from sites in the ".gov" domain



The Document Collections

- TREC documents come from the following sources:

WSJ	→ *Wall Street Journal*
AP	→ Associated Press (news wire)
ZIFF	→ Computer Selects (articles), Ziff-Davis
FR	→ Federal Register
DOE	→ US DOE Publications (abstracts)
SJMN	→ *San Jose Mercury News*
PAT	→ US Patents
FT	→ *Financial Times*
CR	→ Congressional Record
FBIS	→ Foreign Broadcast Information Service
LAT	→ *LA Times*



The Document Collections

- Contents of TREC-6 disks 1 and 2

Disk	Contents	Size	Number	Words/Doc.	Words/Doc.
		Mb	**Docs**	**(median)**	**(mean)**
1	WSJ, 1987-1989	267	98,732	245	434.0
	AP, 1989	254	84,678	446	473.9
	ZIFF	242	75,180	200	473.0
	FR, 1989	260	25,960	391	1315.9
	DOE	184	226,087	111	120.4
2	WSJ, 1990-1992	242	74,520	301	508.4
	AP, 1988	237	79,919	438	468.7
	ZIFF	175	56,920	182	451.9
	FR, 1988	209	19,860	396	1378.1

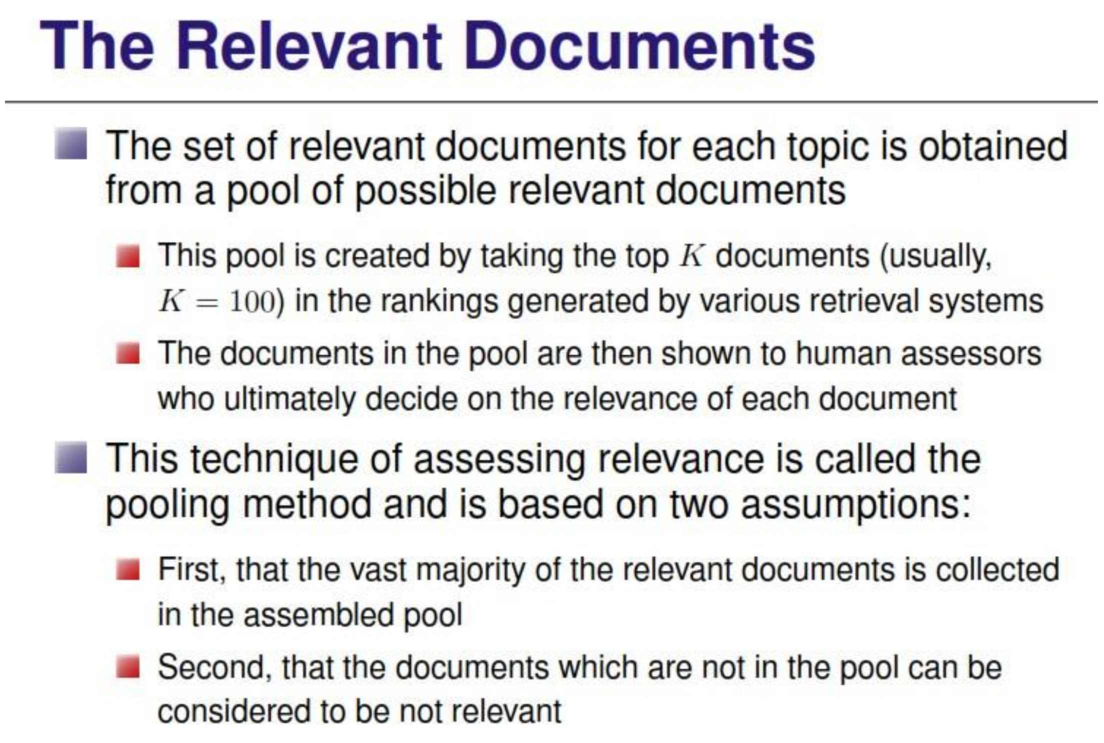
The TREC Web Collections

- A Web Retrieval track was introduced at TREC-9
  - The VLC2 collection is from an Internet Archive crawl of 1997
  - WT2g and WT10g are subsets of the VLC2 collection
  - .GOV is from a crawl of the .gov Internet done in 2002
  - .GOV2 is the result of a joint NIST/UWaterloo effort in 2004

Collection	# Docs	Avg Doc Size	Collection Size
VLC2 (WT100g)	18,571,671	5.7 KBytes	100 GBytes
WT2g	247,491	8.9 KBytes	2.1 GBytes
WT10g	1,692,096	6.2 KBytes	10 GBytes
.GOV	1,247,753	15.2 KBytes	18 GBytes
.GOV2	27 million	15 KBytes	400 GBytes

Information Requests Topics

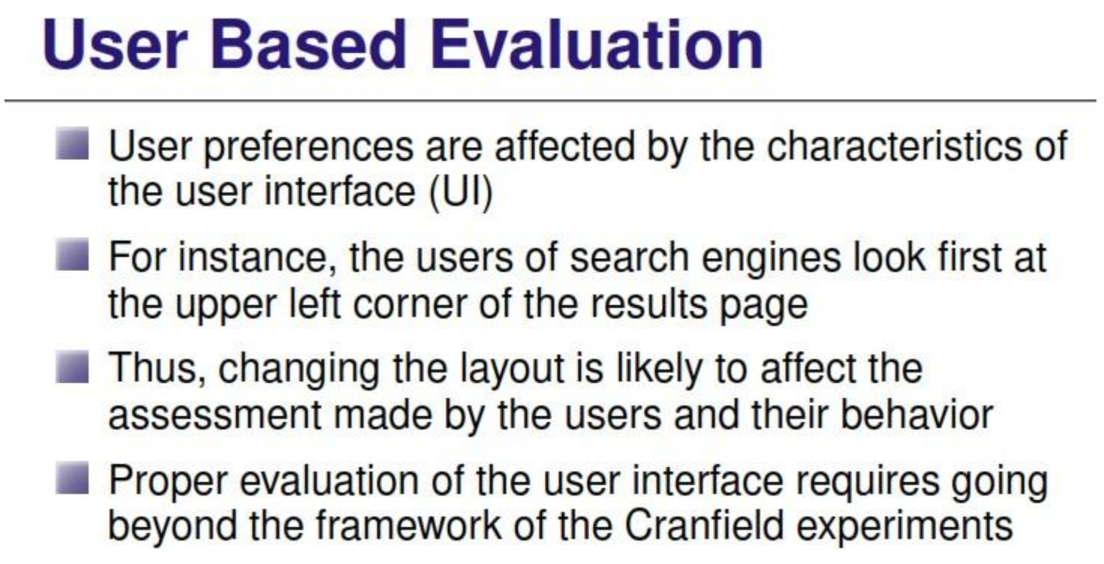
- Each TREC collection includes a set of example **information requests**
- Each request is a description of an information need in natural language
- In the TREC nomenclature, each test information request is referred to as a **topic**



The Relevant Documents

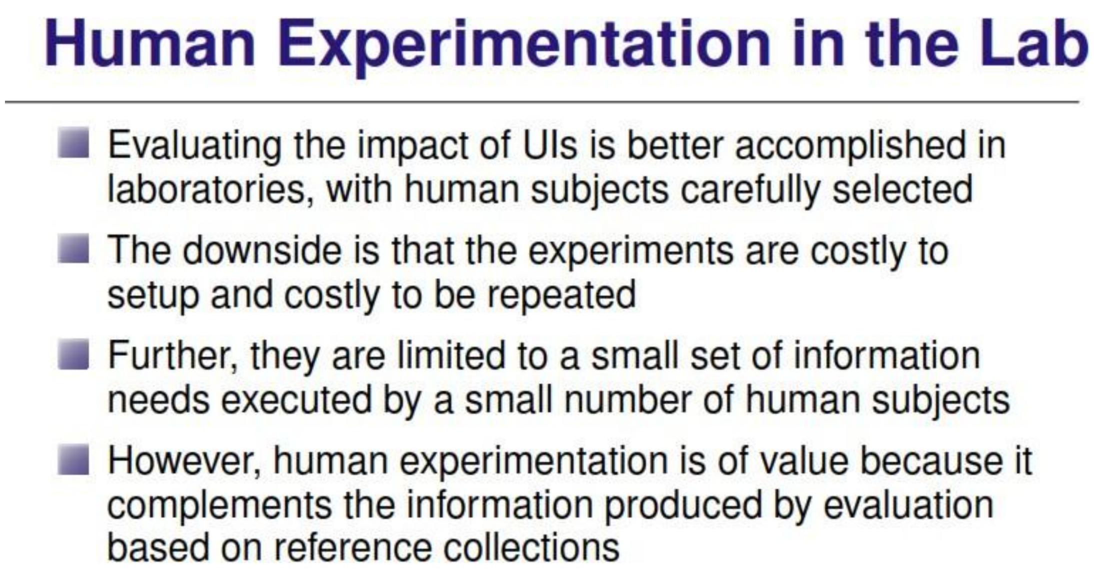
- The set of relevant documents for each topic is obtained from a pool of possible relevant documents
  - This pool is created by taking the top K documents (usually, $K = 100$) in the rankings generated by various retrieval systems
  - The documents in the pool are then shown to human assessors who ultimately decide on the relevance of each document
- This technique of assessing relevance is called the pooling method and is based on two assumptions:
  - First, that the vast majority of the relevant documents is collected in the assembled pool
  - Second, that the documents which are not in the pool can be considered to be not relevant

USER-BASED EVALUATION



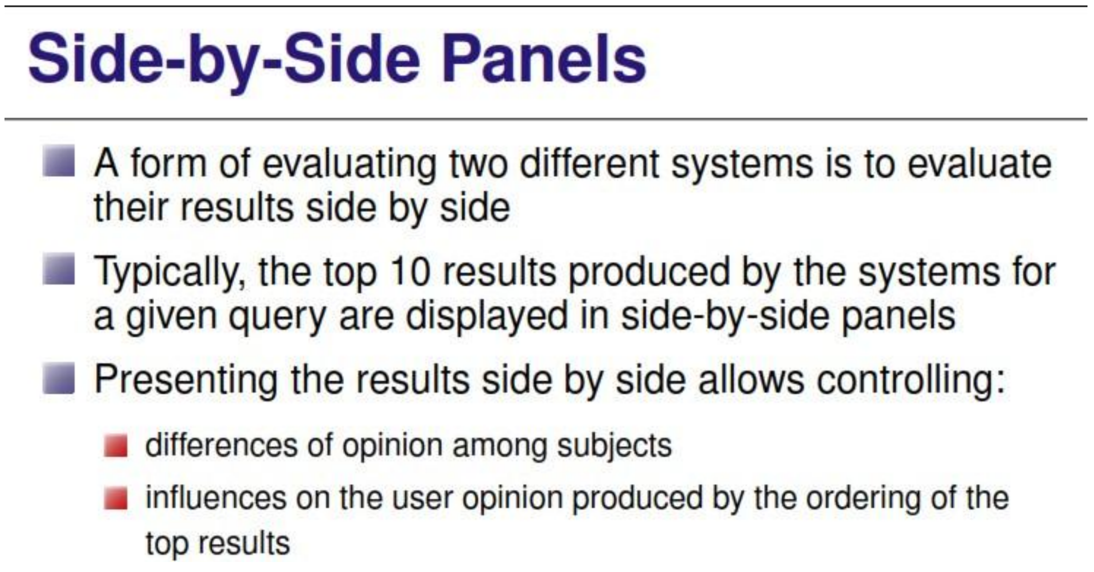
User Based Evaluation

- User preferences are affected by the characteristics of the user interface (UI)
- For instance, the users of search engines look first at the upper left corner of the results page
- Thus, changing the layout is likely to affect the assessment made by the users and their behavior
- Proper evaluation of the user interface requires going beyond the framework of the Cranfield experiments



Human Experimentation in the Lab

- Evaluating the impact of UIs is better accomplished in laboratories, with human subjects carefully selected
- The downside is that the experiments are costly to setup and costly to be repeated
- Further, they are limited to a small set of information needs executed by a small number of human subjects
- However, human experimentation is of value because it complements the information produced by evaluation based on reference collections

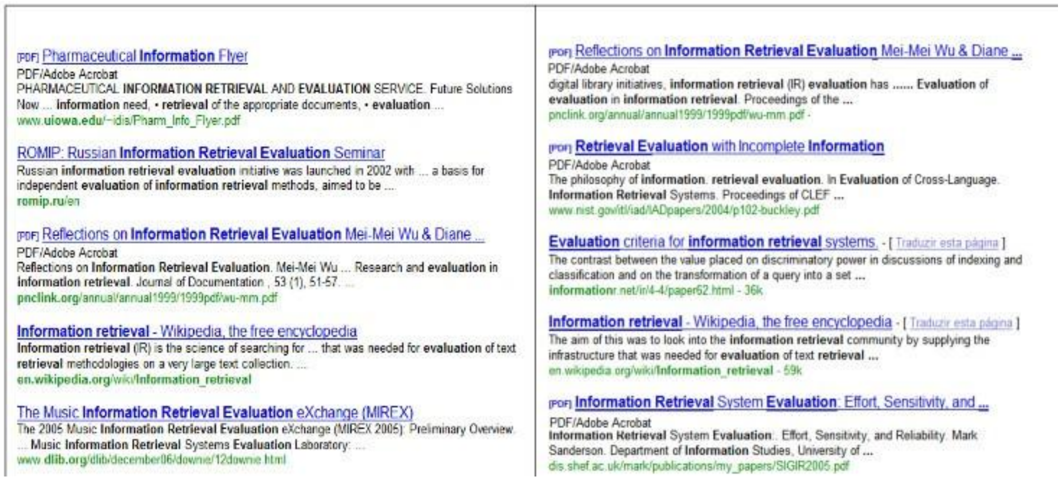


Side-by-Side Panels

- A form of evaluating two different systems is to evaluate their results side by side
- Typically, the top 10 results produced by the systems for a given query are displayed in side-by-side panels
- Presenting the results side by side allows controlling:
  - differences of opinion among subjects
  - influences on the user opinion produced by the ordering of the top results

Side-by-Side Panels

- Side by side panels for Yahoo! and Google
  - Top 5 answers produced by each search engine, with regard to the query *"information retrieval evaluation"*



[PDF] Pharmaceutical **Information** Flyer
PDF/Adobe Acrobat
PHARMACEUTICAL **INFORMATION RETRIEVAL** AND **EVALUATION** SERVICE. Future Solutions Now ... **information** need, • **retrieval** of the appropriate documents, • **evaluation** ...
www.**uiowa.edu**/~idis/Pharm_Info_Flyer.pdf

ROMIP: Russian **Information Retrieval Evaluation** Seminar
Russian **information retrieval evaluation** initiative was launched in 2002 with ... a basis for independent **evaluation** of **information retrieval** methods, aimed to be ...
romip.ru/en

[PDF] Reflections on **Information Retrieval Evaluation** Mei-Mei Wu & Diane ...
PDF/Adobe Acrobat
Reflections on **Information Retrieval Evaluation**. Mei-Mei Wu ... Research and **evaluation** in **information retrieval**. Journal of Documentation , 53 (1), 51-57. ...
pnclink.org/annual/annual1999/1999pdf/wu-mm.pdf

Information retrieval - Wikipedia, the free encyclopedia
Information retrieval (IR) is the science of searching for ... that was needed for **evaluation** of text **retrieval** methodologies on a very large text collection. ...
en.wikipedia.org/wiki/**Information_retrieval**

The Music **Information Retrieval Evaluation** eXchange (MIREX)
The 2005 Music **Information Retrieval Evaluation** eXchange (MIREX 2005): Preliminary Overview. ... Music **Information Retrieval** Systems **Evaluation** Laboratory: ...
www.**dlib.org**/dlib/december06/downie/12downie.html

[PDF] Reflections on **Information Retrieval Evaluation** Mei-Mei Wu & Diane ...
PDF/Adobe Acrobat
digital library initiatives, **information retrieval** (IR) **evaluation** has ...... **Evaluation** of **evaluation** in **information retrieval**. Proceedings of the ...
pnclink.org/annual/annual1999/1999pdf/wu-mm.pdf -

[PDF] **Retrieval Evaluation** with Incomplete **Information**
PDF/Adobe Acrobat
The philosophy of **information**. **retrieval evaluation**. In **Evaluation** of Cross-Language. **Information Retrieval** Systems. Proceedings of CLEF ...
www.nist.gov/itl/iad/IADpapers/2004/p102-buckley.pdf

Evaluation criteria for **information retrieval** systems. - [Traduzir esta página]
The contrast between the value placed on discriminatory power in discussions of indexing and classification and on the transformation of a query into a set ...
informationr.net/ir/4-4/paper62.html - 36k

Information retrieval - Wikipedia, the free encyclopedia - [Traduzir esta página]
The aim of this was to look into the **information retrieval** community by supplying the infrastructure that was needed for **evaluation** of text **retrieval** ...
en.wikipedia.org/wiki/**Information_retrieval** - 59k

[PDF] **Information Retrieval** System **Evaluation**: Effort, Sensitivity, and ...
PDF/Adobe Acrobat
Information Retrieval System **Evaluation**:. Effort, Sensitivity, and Reliability. Mark Sanderson. Department of **Information** Studies, University of ...
dis.shef.ac.uk/mark/publications/my_papers/SIGIR2005.pdf

- The side-by-side experiment is simply a judgement on which side provides better results for a given query
  - By recording the interactions of the users, we can infer which of the answer sets are preferred to the query
- Side by side panels can be used for quick comparison of distinct search engines

RELEVANCE FEEDBACK AND QUERY EXPANSION

Most users find it difficult to formulate queries that are well designed for retrieval purposes Yet, most users often need to reformulate their queries to obtain the results of their interest Thus, the first query formulation should be treated as an initial attempt to retrieve relevant information. Documents initially retrieved could be analyzed for relevance and used to improve initial query

The process of query modification is commonly referred as **relevance feedback**, when the user provides information on relevant documents to a query, or **query expansion**, when information related to the query is used to expand it We refer to both of them as feedback methods Two basic approaches of feedback methods:

1. **explicit feedback**, in which the information for query reformulation is provided directly by the users, and
2. **implicit feedback**, in which the information for query reformulation is implicitly derived by the system

A Framework for Feedback Methods

Consider a set of documents D_r that are known to be relevant to the current query q. In relevance feedback, the documents in D_r are used to transform q into a modified query q_m. However, obtaining information on documents relevant to a query requires the direct interference of the user. Most users are unwilling to provide this information, particularly in the Web.

Because of this high cost, the idea of relevance feedback has been relaxed over the years Instead of asking the users for the relevant documents, we could: Look at documents they have clicked on; or Look at terms belonging to the top documents in the result set. In both cases, it is expect that the feedback cycle will produce results of higher quality.

A **feedback cycle** is composed of two basic steps:

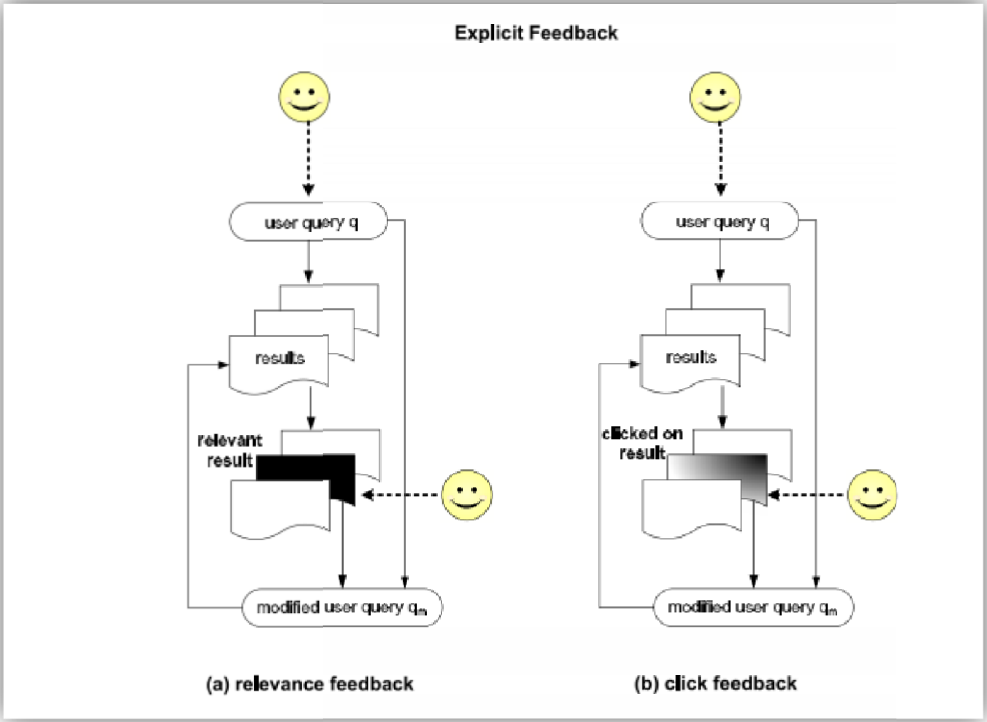
- Determine feedback information that is either related or expected to be related to the original query q and
- Determine how to transform query q to take this information effectively into account

✓ The first step can be accomplished in two distinct ways:

- Obtain the feedback information explicitly from the users
- Obtain the feedback information implicitly from the query results or from external sources such as a **thesaurus.**

EXPLICIT RELEVANCE FEEDBACK

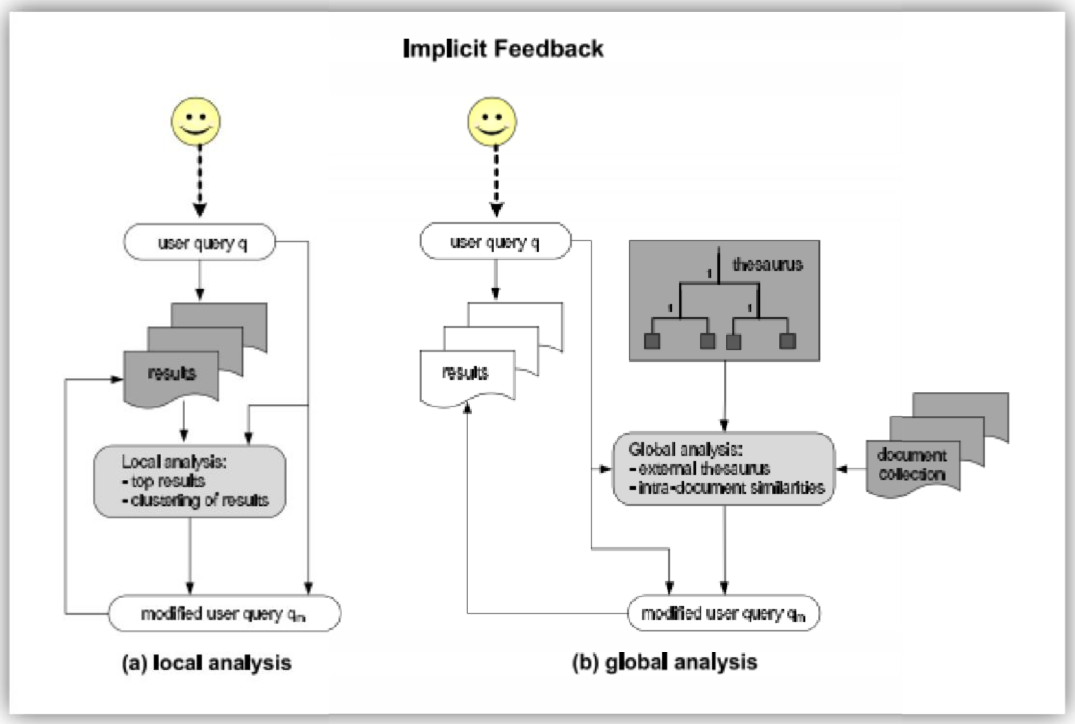
In an **explicit relevance feedback** cycle, the feedback information is provided directly by the users However, collecting feedback information is expensive and time consuming In the Web, **user clicks** on search results constitute a new source of feedback information A click indicate a document that is of interest to the user in the context of the current query Notice that a click does not necessarily indicate a document that is relevant to the query.



In an **implicit relevance feedback** cycle, the feedback information is derived implicitly by the system There are two basic approaches for compiling implicit feedback information:

local analysis, which derives the feedback information from the top ranked documents in the result set.

global analysis, which derives the feedback information from external sources such as a thesaurus.



Classic Relevance Feedback

In a classic relevance feedback cycle, the user is presented with a list of the retrieved documents Then, the user examines them and marks those that are relevant In practice, only the top 10 (or 20) ranked documents need to be examined The main idea consists of selecting important terms from the documents that have been identified as relevant, and enhancing the importance of these terms in a new query formulation.

Expected effect: the new query will be moved towards the relevant docs and away from the non-relevant ones Early experiments have shown good improvements in precision for small test collections Relevance feedback presents the following characteristics:

- it shields the user from the details of the query reformulation process (all the user has to provide is a relevance judgement)
- it breaks down the whole searching task into a sequence of small steps which are easier to grasp.

PART-A

1 Identify probabilistic Information Retrieval.
2 Analyze the Boolean model.
3 Construct the Vector space model representation.
4 List the classes of retrieval model.
5 Define Retrieval model.
6 Explain language modelling with example.
7 Illustrate similarity measure.
8 Analyze the problems in lexical semantics.
9 Demonstrate language model and Naïve Bayes.
10 Formulate the Bayesian rule.
11 What is meant by sparse vector?
12 Design an Inverted file with an example.
13 Evaluate the goals of LSI.
14 What is smoothing and stemming?
15 List probabilistic approaches of vector model.
16 What is meant Zone Index?
17 Interpret cosine similarity measure.
18 Analyze relevance feedback and pseudo relevance feedback.
19 List the steps involved in preprocessing.
20 Generalize on why distance is not preferred compared to angle.

PART-B

1 i) Express what is Boolean retrieval model. (4)
ii) Describe the document preprocessing steps in detail (9)
2 Illustrate the Vector space retrieval model with example (13)
3 Describe about basic concepts of Cosine similarity. (13)
4 Develop on example to implement term weighting .(min docs = 5) (13)
5 i) Tabulate the common preprocessing steps. (4)
ii) Discuss the Boolean retrieval in detail with diagram.(9)
6 i) Discuss in detail about term frequency and Inverse Document Frequency. (7)
ii)Compute TF-IDF .given a document containing terms with the given frequencies A(3) ,B(2), C(1).Assume document collections 10,000 and document frequencies of these terms are A(50), B(1300), C(250) (6)
7 i)Explain Latent Semantic Indexing and latent semantic space with an illustration. (9)
ii) Analyze the use of LSI in Information Retrieval. What is its need in synonyms and semantic relatedness.(4)
8 i)Examine, how to form a binary term - document incidence matrix (7)
ii) Give an example for the above. (6)
9 Describe document preprocessing and its stages in detail. (13)
10 i) Discuss the structure of inverted indices and the basic Boolean Retrieval model (7)
ii)Discuss the searching process in inverted file (6)
11 i) Why do we need sparse vectors? (4)
ii)Estimate sparse vectors and its efficiency with diagram.(9)
12 i) Analyze the language model based IR and its probabilistic representation. (7)
ii)Compare Language model vs Naive Bayes and Language model vs Vector space model (6)

13 i) Explain in detail about binary independence model for Probability Ranking Principle(PRP). (7)
ii) Analyze how the query generation probability for query likelihood model can be estimated.(6)
14 i) Apply how Probabilistic approaches to Information Retrieval are done. (7)
ii) Illustrate the following
a) Probabilistic relevance feedback. (2)
b) Pseudo relevance feedback. (2)
c) Indirect relevance feedback (2)

PART-C

1 Compose the information Retrieval services of the internet with suitable design. (15)
2 Assess the best Language model to computational linguistics for investigating the use of software to translate text or speech from one language to another. (15)
3 Contrast the uses of probabilistic IR in indexing the search in the internet. (15)
4 Create a Relevance feedback mechanism for your college website search in the internet. (15)