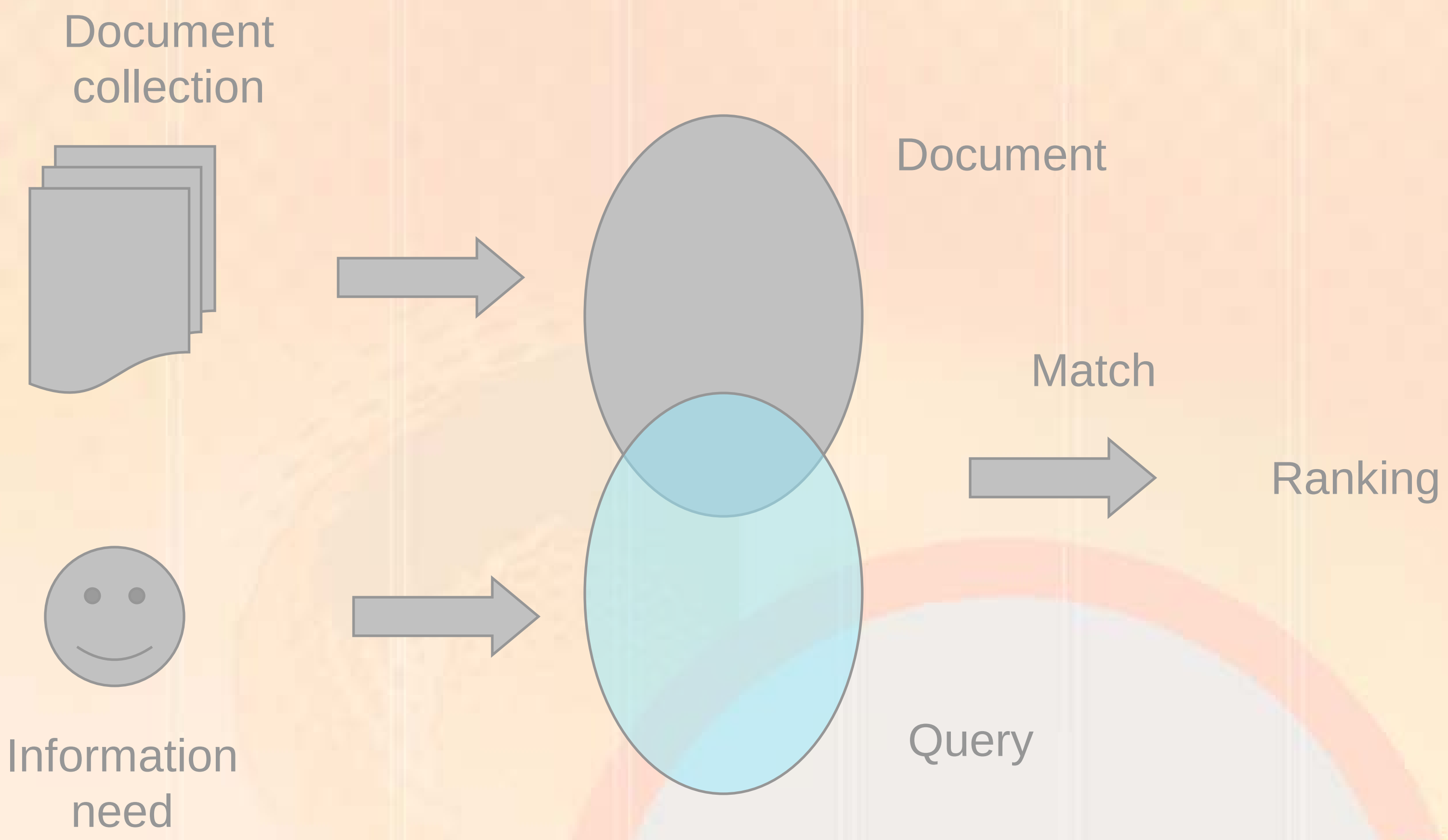
# Language Technology I

Information Retrieval

# Information Retrieval

- Traditional information retrieval is basically text search
  - *A collection of text documents*
  - *Documents are generally high-quality and designed to convey information*
  - *Documents are assumed to have no structure beyond words*
- Searches are generally based on meaningful phrases
- The goal is to find the document(s) that best match the search phrase, according to a search model

# Basic Model


Document
collection
Document
Match
Ranking
Information
need
Query

# Terminology

- Document
  - *Unit of text indexed in the system*
  - *Result of the retrieval*
- IR systems usually adopt index terms to process queries
- Index term:
  - *a keyword or group of selected words*
  - *any word (more general)*
- An inverted index is built for the chosen index terms
  - *D0 = "it is what it is", D1 = "what is it" and D2 = "it is a banana"*
  - *"a": {D2}*
  - *"banana": {D2}*
  - *"is": {D0, D1, D2}*
  - *"it": {D0, D1, D2}*
  - *"what": {D0, D1}*
- Query
  - *User's information need as a set of terms*

# IR models

- **An IR model is characterized by three parameters:**
  - ***representations for documents and queries***
  - ***matching strategies for assessing the relevance of documents to a user query***
  - ***methods for ranking query output***
- **Classic models**
  - ***Boolean***
  - ***Vector space***
  - ***Probabilistic***

**Set Theoretic**

***Boolean model***

***Fuzzy model***

***Extended boolean model***

**Algebraic**

***Vector space model***

***Generalized vector model***

***Latent semantic index***

***Neural networks model***

**Probabilistic**

***Probabilistic model***

***Inference network***

***Belief network***

# IR models – basic concepts

- Each document represented by a set of representative keywords or index terms
- An index term is a document word useful for remembering the document main themes
- Traditionally, index terms were nouns because nouns have meaning by themselves
- Not all terms are equally useful for representing the document contents: less frequent terms allow identifying a narrower set of documents
- The *importance* of the index terms is represented by weights associated to them

# Boolean Model

- Based on set theory and Boolean algebra
  - *Documents are sets of terms*
  - *Queries are Boolean expressions on terms*
- **D:** set of words (indexing terms) present in a document
  - *each term is either present (1) or absent (0)*
- **Q:** A boolean expression
  - *terms are index terms*
  - *operators are AND, OR, and NOT*
- **Matching:** Boolean algebra over sets of terms and sets of documents
- No term weighting is allowed

# Boolean Model example

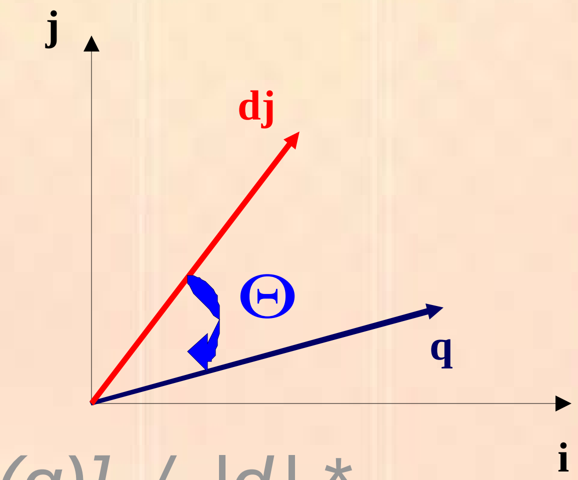
$$((text \vee information) \wedge retrieval \wedge \neg theory)$$

- *“Information Retrieval”* X
- *“Information Theory”*
- *“Modern Information Retrieval: Theory and Practice”*
- *“Text Compression”*

# Boolean Model Disadvantages

- Similarity function is boolean
  - *Exact-match only, no partial matches*
  - *Retrieved documents not ranked*
- All terms are equally important
  - *Boolean operator usage has much more influence than a critical word*
- Query language is expressive but complicated

# Vector Space Model




$vec(d_j) = (w_{1j}, w_{2j}, \ldots, w_{tj})$
$vec(q) = (w_{1q}, w_{2q}, \ldots, w_{tq})$
$Sim(q,d_j) \quad = \cos(\Theta)$
$\quad = [vec(d_j) \otimes vec(q)] \;/\; |d_j| *$
$|q| \quad = [\Sigma\, w_{ij} * w_{iq}] \;/\; |d_j| * |q|$

- $w_{ij}$ *is term's i weight in document j*
- Cosine is a normalized dot product
- Since $w_{ij} > 0$ and $w_{iq} > 0$, $0 \leq sim(q,d_j) \leq 1$
- A document is retrieved even if it matches the query terms only partially

# Term Weighting

- Higher weight = greater impact on cosine
- Want to give more weight to the more "important" or useful terms
- What is an important term?
  - *If we see it in a query, then its presence in a document means that the document is relevant to the query.*
  - *How can we model this?*

# Weights in the Vector Model

- $Sim(q,dj) = [\Sigma w_{ij} * w_{iq}] / |d_j| * |q|$
- How do we compute the weights $w_{ij}$ and $w_{iq}$?
- A good weight must take into account two effects:
  - quantification of intra-document contents (similarity)
    - *tf* factor, the *term frequency* within a document
  - quantification of inter-documents separation (dissimilarity)
    - *idf* factor, the *inverse document frequency*
- $wij = tf(i,j) * idf(i)$

# TF and IDF Factors

- Let:
  - N *be the total number of docs in the collection*
  - $n_i$ *be the number of docs which contain* $k_i$
  - freq(i,j) *raw frequency of* $k_i$ *within* $d_j$

- A normalized *tf* factor is given by

  $$f(i,j) = freq(i,j) / max(freq(l,j))$$

  - *the maximum is computed over all terms which occur within the document* $d_j$

- The *idf* factor is computed as

  $$idf(i) = log(N / n_i)$$

  - *the* log *is used to make the values of* tf *and* idf *comparable.*

# Vector Space Model, Summarized

- The best term-weighting schemes tf-idf weights:
- $w_{ij} = f(i,j) * \log(N/n_i)$
- For the query term weights, a suggestion is
- $w_{iq} = (0.5 + [0.5 * freq(i,q) / max(freq(l,q)]) * \log(N / n_i)$

- This model is very good in practice:
  - tf-idf works well with general collections
  - Simple and fast to compute
  - Vector model is usually as good as the known ranking alternatives

# Pros & Cons of Vector Model

- Advantages:
  - term-weighting improves quality of the answer set
  - partial matching allows retrieval of docs that approximate the query conditions
  - cosine ranking formula sorts documents according to degree of similarity to the query
- Disadvantages:
  - assumes independence of index terms; not clear if this is a good or bad assumption

# Comparison of Classic Models

- **Boolean model** does not provide for partial matches and is considered to be the weakest classic model
- Some experiments indicate that the **vector model** outperforms the third alternative, the **probabilistic model**, in general
  - Recent IR research has focused on improving probabilistic models – but these haven’t made their way to Web search
- Generally we use a variation of the vector model in most text search systems

## Why evaluate IR systems?

- *There are many retrieval models/ algorithms/ systems, which one is the best?*
- *What is the best component for:*
  - *Ranking function (dot-product, cosine, …)*
  - *Term selection (stopword removal, stemming…)*
  - *Term weighting (TF, TF-IDF,…)*
- *How far down the ranked list will a user need to look to find some/all relevant documents?*

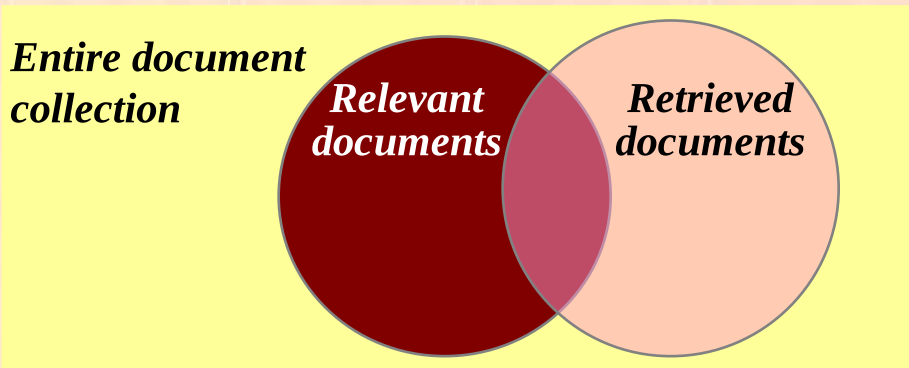
# Difficulties in Evaluating IR Systems

- *Effectiveness is related to the* **relevancy** *of retrieved items.*
- *Relevancy is not typically binary but continuous.*
- *Even if relevancy is binary, it can be a difficult judgment to make.*
- *Relevancy, from a human standpoint, is:*
  - *Subjective: Depends upon a specific user's judgment.*
  - *Situational: Relates to user's current needs.*
  - *Cognitive: Depends on human perception and behavior.*
  - *Dynamic: Changes over time.*

# Human Labeled Corpora (Gold Standard)

- *Start with a corpus of documents.*
- *Collect a set of queries for this corpus.*
- *Have one or more human experts exhaustively label the relevant documents for each query.*
- *Typically assumes binary relevance judgments.*
- *Requires considerable human effort for large document/query corpora.*

# Precision and Recall




| | retrieved | not retrieved |
|---|---|---|
| irrelevant | retrieved & irrelevant | Not retrieved & irrelevant |
| relevant | retrieved & relevant | not retrieved but relevant |

$$recall = \frac{\text{Number of relevant documents retrieved}}{\text{Total number of relevant documents}}$$

$$precision = \frac{\text{Number of relevant documents retrieved}}{\text{Total number of documents retrieved}}$$

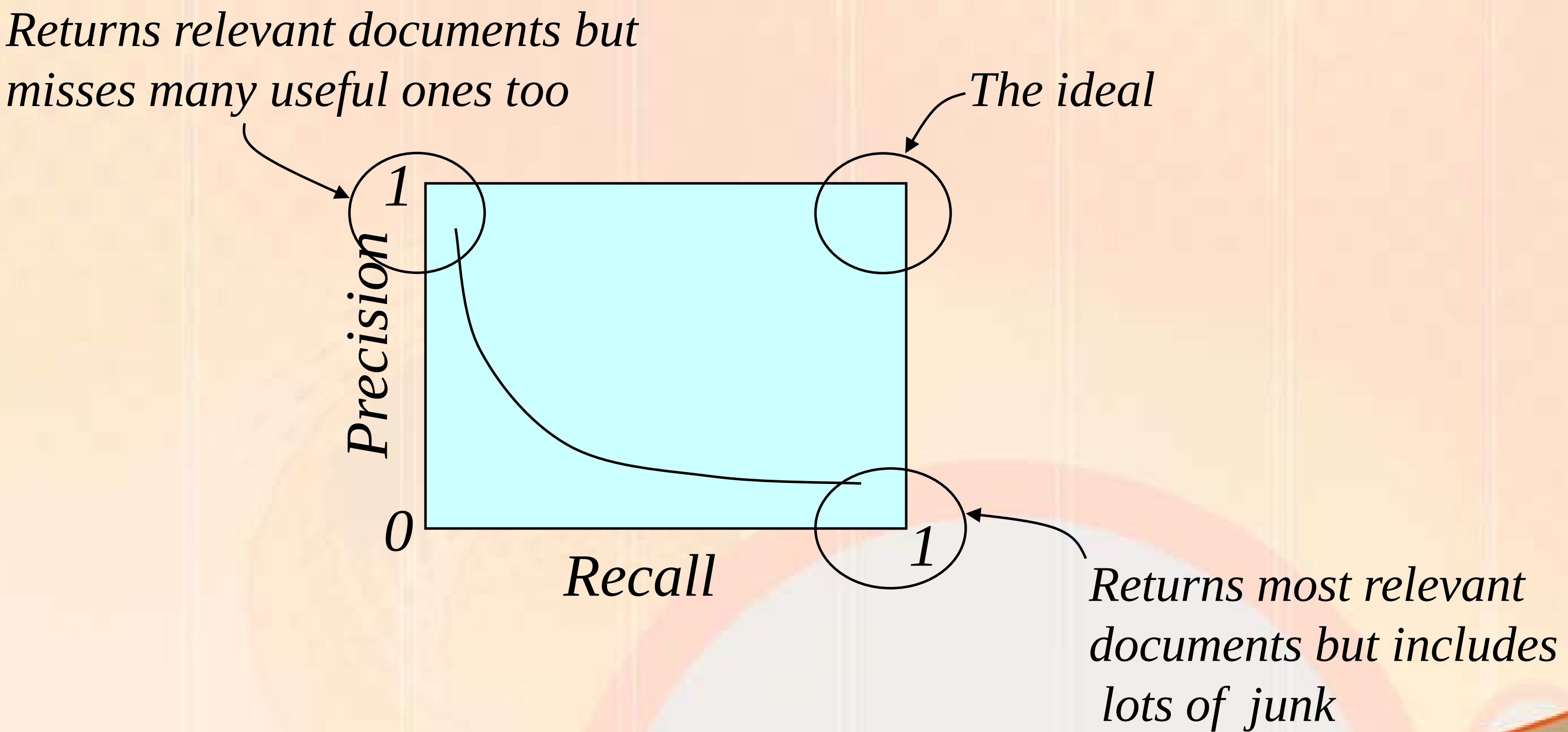
## Precision and Recall

- *Precision*
  - *The ability to retrieve top-ranked documents that are mostly relevant.*
- *Recall*
  - *The ability of the search to find* ***all*** *of the relevant items in the corpus.*

# Determining Recall is Difficult

- *Total number of  relevant items is sometimes not available:*
  - *Sample across the database and perform relevance judgment on these items.*
  - *Apply different retrieval algorithms to the same database for the same query. The aggregate of relevant items is taken as the total relevant set.*

# Trade-off between Recall and Precision



# F-Measure

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

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

- *Compared to arithmetic mean, both need to be high for harmonic mean to be high.*

# E Measure (parameterized F Measure)

- *A variant of F measure that allows weighting emphasis on precision over recall:*

$$E = \frac{(1+\beta^2)P}{\beta^2 P + R} = \frac{(1+\beta^2)}{\frac{\beta^2}{R} + \frac{1}{P}}$$

- *Value of β controls trade-off:*
  - β *= 1: Equally weight precision and recall (E=F).*
  - β *> 1: Weight recall more.*
  - β *< 1: Weight precision more.*