Chapter 3

Modeling

Set-Based Model
Extended Boolean Model
Fuzzy Set Model
The Generalized Vector Model
Latent Semantic Indexing
Neural Network for IR

Set Theoretic Models

Set Theoretic Models

- The Boolean model imposes a **binary criterion** for deciding relevance
- The question of how to extend this model to associate **ranking** has attracted considerable attention in the past
- We now discuss three alternative set theoretic models:
  - Set-Based Model
  - Extended Boolean Model
  - Fuzzy Set Model

Set Theoretic Models

Set-Based Model

Set-Based Model

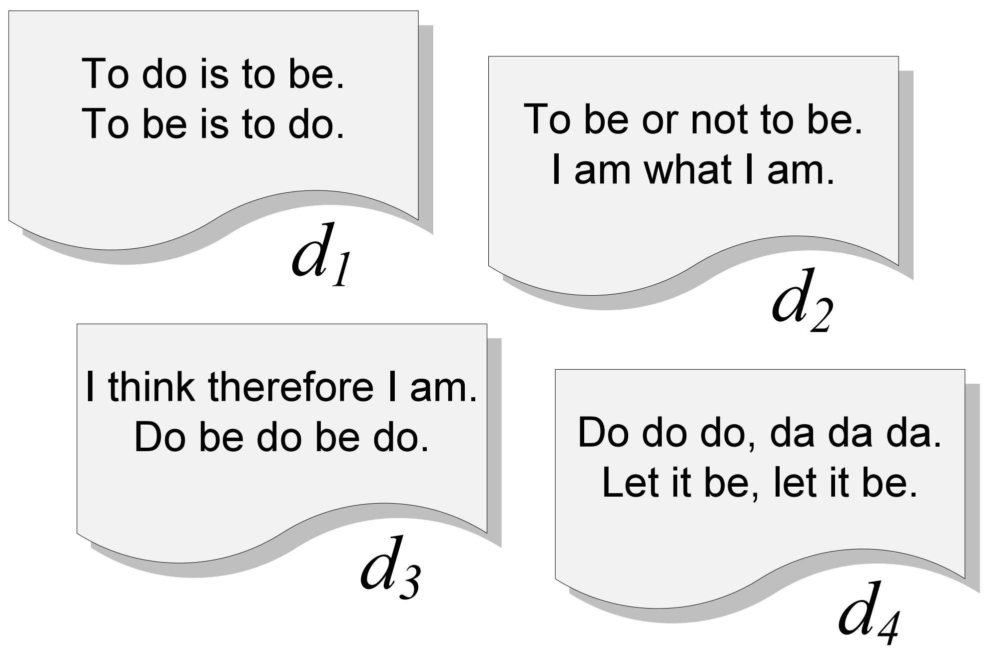
- A more recent approach (2005) that combines set theory with vectorial ranking
- The fundamental idea is to use mutual dependencies among index terms to improve results
- Term dependencies is captured through **termsets**, which are sets of correlated terms
- The approach leads to improved results with various collections
- The first IR model that effectively took advantage of term dependence

Termsets

- **Termset** is a concept used in place of the index terms
- A termset $S_i = \{k_a, k_b, ..., k_n\}$ is a subset of the terms in the collection
- If all index terms in S_i occur in a document d_j then we say that the termset S_i occurs in d_j
- There are 2^t termsets that might occur in the documents of a collection
  - However, most combinations of terms have no semantic meaning
  - Thus, the actual number of termsets in a collection is far smaller than 2^t

Termsets

- Let t be the number of terms of the collection
- Then, the set $V_S = \{S_1, S_2, ..., S_{2^t}\}$ is the **vocabulary-set** of the collection
- To illustrate, consider the document collection below

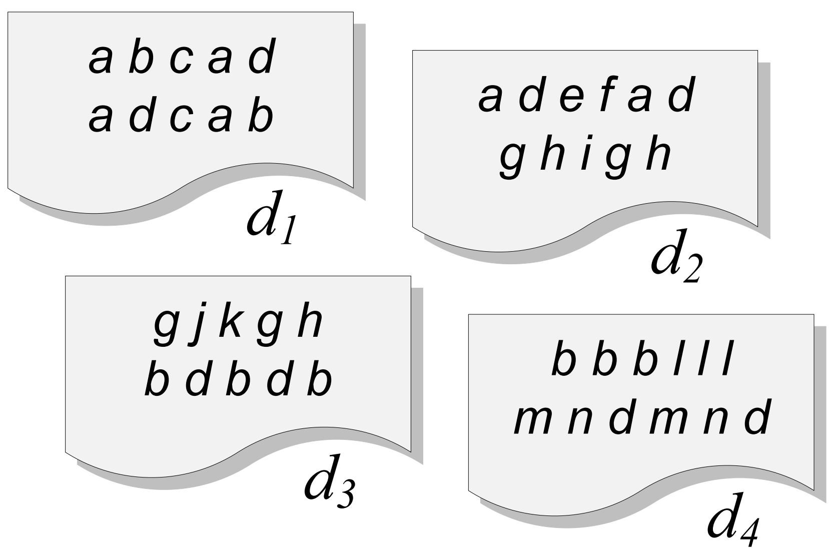


Termsets

- To simplify notation, let us define

k_a = to	k_d = be	k_g = I	k_j = think	k_m = let
k_b = do	k_e = or	k_h = am	k_k = therefore	k_n = it
k_c = is	k_f = not	k_i = what	k_l = da	

- Further, let the letters $a...f$ refer to the index terms $k_a...k_f$, respectively



Termsets

- Consider the query q as “to do be it”, i.e. $q = \{a, b, d, n\}$
- For this query, the vocabulary-set is as below

Termset	Set of Terms	Documents
S_a	$\{a\}$	$\{d_1, d_2\}$
S_b	$\{b\}$	$\{d_1, d_3, d_4\}$
S_d	$\{d\}$	$\{d_1, d_2, d_3, d_4\}$
S_n	$\{n\}$	$\{d_4\}$
S_{ab}	$\{a, b\}$	$\{d_1\}$
S_{ad}	$\{a, d\}$	$\{d_1, d_2\}$
S_{bd}	$\{b, d\}$	$\{d_1, d_3, d_4\}$
S_{bn}	$\{b, n\}$	$\{d_4\}$
S_{abd}	$\{a, b, d\}$	$\{d_1\}$
S_{bdn}	$\{b, d, n\}$	$\{d_4\}$

Notice that there are 11 termsets that occur in our collection, out of the maximum of 15 termsets that can be formed with the terms in q

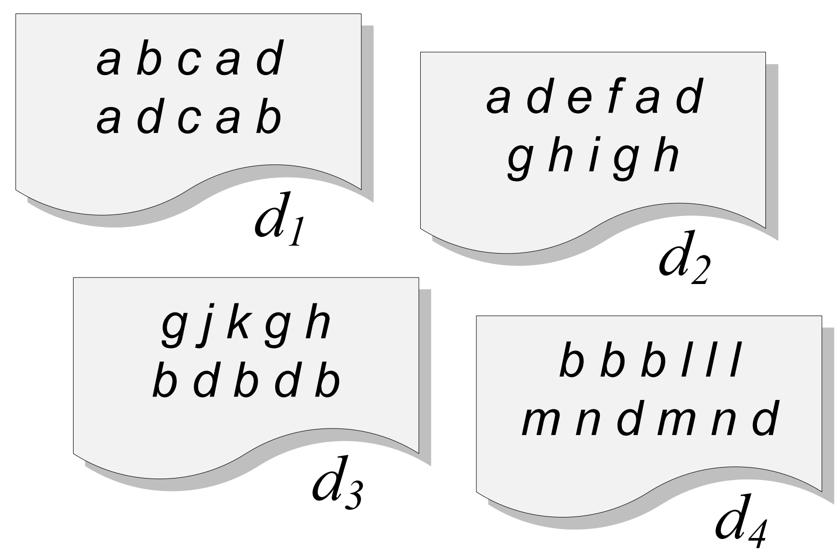
Termsets

- At query processing time, only the termsets generated by the query need to be considered
- To reduce the number of termsets for long queries, we use the concept of **frequent termsets**
- A termset composed of n terms is called an n-termset
- Let $\mathcal{N}_i$ be the number of documents in which S_i occurs
- An n-termset S_i is said to be **frequent** if $\mathcal{N}_i$ is greater than or equal to a given threshold
  - This implies that an n-termset is frequent if and only if all of its $(n-1)$-termsets are also frequent

Termsets

- Let the threshold on the frequency of termsets be 2
- To compute all frequent termsets for the query $q = \{a, b, d, n\}$ we proceed as follows

1. Compute the frequent 1-termsets and their inverted lists:
   - $S_a = \{d_1, d_2\}$
   - $S_b = \{d_1, d_3, d_4\}$
   - $S_d = \{d_1, d_2, d_3, d_4\}$
2. Combine the inverted lists to compute frequent 2-termsets:
   - $S_{ad} = \{d_1, d_2\}$
   - $S_{bd} = \{d_1, d_3, d_4\}$
3. Since there are no frequent 3-termsets, stop



Termsets

- Notice that there are only 5 frequent termsets in our collection
- Inverted lists for frequent n-termsets can be computed by starting with the inverted lists of frequent 1-termsets
  - Thus, the only indice that is required are the standard inverted lists used by any IR system
- This is reasonably fast for short queries up to 4-5 terms

Ranking Computation

- The ranking computation is based on the vector model, by using termsets instead of index terms
- Given a query q, let
  - $\{S_1, S_2, \ldots\}$ be the set of all termsets originated from q
  - $\mathcal{N}_i$ be the number of documents in which termset S_i occurs
  - N be the total number of documents in the collection
  - $\mathcal{F}_{i,j}$ be the frequency of termset S_i in document d_j
- For each pair $[S_i, d_j]$ we compute a weight $\mathcal{W}_{i,j}$ given by

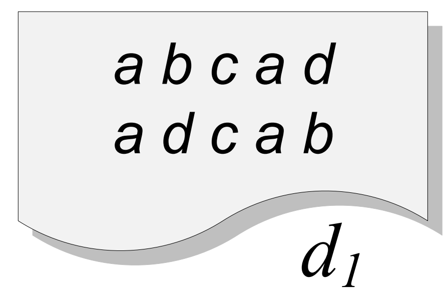
$$\mathcal{W}_{i,j} = \begin{cases} (1 + \log \mathcal{F}_{i,j}) \log(1 + \frac{N}{\mathcal{N}_i}) & \texttt{if } \mathcal{F}_{i,j} > 0 \\ 0 & \mathcal{F}_{i,j} = 0 \end{cases}$$

- We also compute a $\mathcal{W}_{i,q}$ value for each pair $[S_i, q]$

Ranking Computation

- The weights of interest considering the query $q = \{a, b, d, n\}$ and the document d_1 are (assuming minimum threshold frequency of 1)

Termset	Weight	
S_a	$\mathcal{W}_{a,1}$	$1 + \log 4 * \log(1 + 4/2) = 4.75$
S_b	$\mathcal{W}_{b,1}$	$1 + \log 2 * \log(1 + 4/3) = 2.44$
S_d	$\mathcal{W}_{d,1}$	$1 + \log 2 * \log(1 + 4/4) = 2.00$
S_n	$\mathcal{W}_{n,1}$	$0 * \log(1 + 4/1) = 0.00$
S_{ab}	$\mathcal{W}_{ab,1}$	$1 + \log 2 * \log(1 + 4/1) = 4.64$
S_{ad}	$\mathcal{W}_{ad,1}$	$1 + \log 2 * \log(1 + 4/2) = 3.17$
S_{bd}	$\mathcal{W}_{bd,1}$	$1 + \log 2 * \log(1 + 4/3) = 2.44$
S_{bn}	$\mathcal{W}_{bn,1}$	$0 * \log(1 + 4/1) = 0.00$
S_{dn}	$\mathcal{W}_{dn,1}$	$0 * \log(1 + 4/1) = 0.00$
S_{abd}	$\mathcal{W}_{abd,1}$	$1 + \log 2 * \log(1 + 4/1) = 4.64$
S_{bdn}	$\mathcal{W}_{bdn,1}$	$0 * \log(1 + 4/1) = 0.00$



Ranking Computation

- A document d_j and a query q are represented as vectors in a 2^t-dimensional space of termsets

$$\begin{aligned} \vec{d_j} &= (\mathcal{W}_{1,j}, \mathcal{W}_{2,j}, \ldots, \mathcal{W}_{2^t,j}) \\ \vec{q} &= (\mathcal{W}_{1,q}, \mathcal{W}_{2,q}, \ldots, \mathcal{W}_{2^t,q}) \end{aligned}$$

- The rank of d_j to the query q is computed as follows

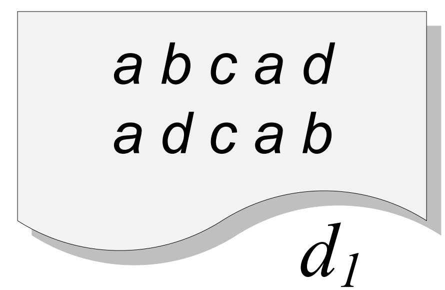
$$sim(d_j, q) = \frac{\vec{d_j} \bullet \vec{q}}{|\vec{d_j}| \times |\vec{q}|} = \frac{\sum_{S_i} \mathcal{W}_{i,j} \times \mathcal{W}_{i,q}}{|\vec{d_j}| \times |\vec{q}|}$$

- For termsets that are not in the query q, $\mathcal{W}_{i,q} = 0$

Ranking Computation

- The document norm $|\vec{d}_j|$ is hard to compute in the space of termsets
- Thus, its computation is restricted to 1-termsets
- Let again $q = \{a, b, d, n\}$ and d_1
- The document norm in terms of 1-termsets is given by

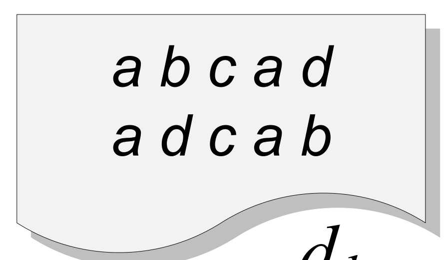
$$
\begin{aligned}
|\vec{d}_1| &= \sqrt{\mathcal{W}_{a,1}^2 + \mathcal{W}_{b,1}^2 + \mathcal{W}_{c,1}^2 + \mathcal{W}_{d,1}^2} \\
&= \sqrt{4.75^2 + 2.44^2 + 4.64^2 + 2.00^2} \\
&= 7.35
\end{aligned}
$$



Ranking Computation

- To compute the rank of d_1, we need to consider the seven termsets S_a, S_b, S_d, S_{ab}, S_{ad}, S_{bd}, and S_{abd}
- The rank of d_1 is then given by

$$
\begin{aligned}
sim(d_1, q) &= (\mathcal{W}_{a,1} * \mathcal{W}_{a,q} + \mathcal{W}_{b,1} * \mathcal{W}_{b,q} + \mathcal{W}_{d,1} * \mathcal{W}_{d,q} + \\
&\quad \mathcal{W}_{ab,1} * \mathcal{W}_{ab,q} + \mathcal{W}_{ad,1} * \mathcal{W}_{ad,q} + \mathcal{W}_{bd,1} * \mathcal{W}_{bd,q} + \\
&\quad \mathcal{W}_{abd,1} * \mathcal{W}_{abd,q}) \,/ |\vec{d_1}| \\
&= (4.75 * 1.58 + 2.44 * 1.22 + 2.00 * 1.00 + \\
&\quad 4.64 * 2.32 + 3.17 * 1.58 + 2.44 * 1.22 + \\
&\quad 4.64 * 2.32)/7.35 \\
&= 5.71
\end{aligned}
$$



d_1

Closed Termsets

- The concept of frequent termsets allows simplifying the ranking computation
- Yet, there are many frequent termsets in a large collection
  - The number of termsets to consider might be prohibitively high with large queries
- To resolve this problem, we can further restrict the ranking computation to a smaller number of termsets
- This can be accomplished by observing some properties of termsets such as the notion of **closure**

Closed Termsets

- The **closure of a termset** S_i is the set of all frequent termsets that co-occur with S_i in the same set of docs
- Given the closure of S_i, the largest termset in it is called a **closed termset** and is referred to as Φ_i
- We formalize, as follows
  - Let $D_i \subseteq C$ be the subset of all documents in which termset S_i occurs and is frequent
  - Let $S(D_i)$ be a set composed of the frequent termsets that occur in all documents in D_i and only in those

Closed Termsets

- Then, the closed termset S_{Φ_i} satisfies the following property

$$\nexists S_j \in S(D_i) \mid S_{\Phi_i} \subset S_j$$

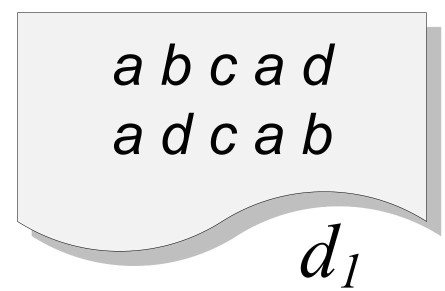
- Frequent and closed termsets for our example collection, considering a minimum threshold equal to 2

frequency(S_i)	frequent termset	closed termset
4	d	d
3	b, bd	bd
2	a, ad	ad
2	g, h, gh, ghd	ghd

Closed Termsets

- Closed termsets encapsulate smaller termsets occurring in the same set of documents
- Ranking of document d_1 with regard to the query $q = \{a, b, d, n\}$ (minimum frequency threshold of 2)

$$
\begin{aligned}
sim(d_1, q) &= (\mathcal{W}_{d,1} * \mathcal{W}_{d,q} + \mathcal{W}_{ab,1} * \mathcal{W}_{ab,q} + \mathcal{W}_{ad,1} * \mathcal{W}_{ad,q} + \\
&\quad \mathcal{W}_{bd,1} * \mathcal{W}_{bd,q} + \mathcal{W}_{abd,1} * \mathcal{W}_{abd,q})/|\vec{d_1}| \\
&= (2.00 * 1.00 + 4.64 * 2.32 + 3.17 * 1.58 + \\
&\quad 2.44 * 1.22 + 4.64 * 2.32)/7.35 \\
&= 4.28
\end{aligned}
$$



Closed Termsets

- Thus, if we restrict the ranking computation to closed termsets, we can expect a reduction in query time
- Smaller the number of closed termsets, sharper is the reduction in query processing time

Set Theoretic Models

Extended Boolean Model

Extended Boolean Model

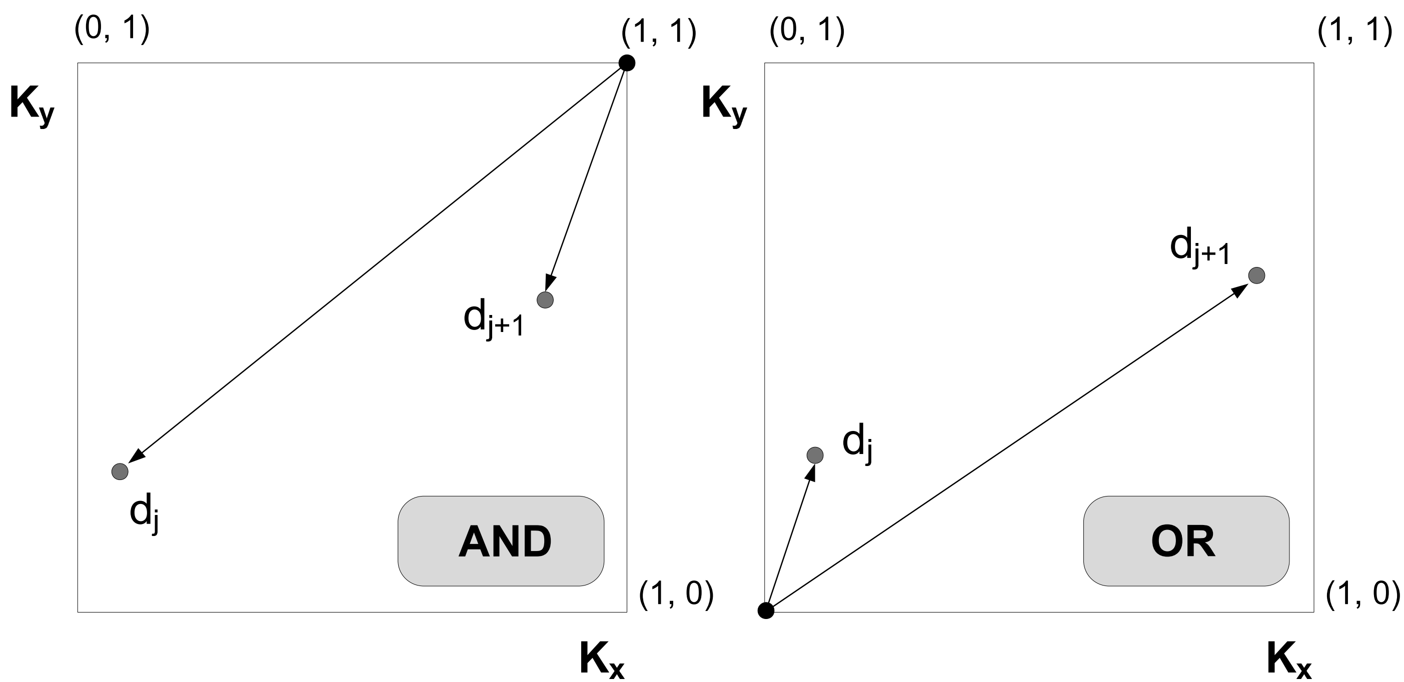
- In Boolean model, no **ranking** of the answer set is generated
- One alternative is to extend the Boolean model with the notions of **partial matching** and **term weighting**
- This strategy allows one to combine characteristics of the Vector model with properties of Boolean algebra

The Idea

- Consider a conjunctive Boolean query given by $q = k_x \wedge k_y$
- For the boolean model, a doc that contains a single term of q is as irrelevant as a doc that contains none
- However, this **binary decision** criteria frequently is not in accordance with common sense
- An analogous reasoning applies when one considers purely disjunctive queries

The Idea

- When only two terms x and y are considered, we can plot queries and docs in a two-dimensional map



- A document d_j is positioned in this space through the adoption of weights $w_{x,j}$ and $w_{y,j}$

The Idea

- These weights can be computed as normalized tf-idf factors as follows

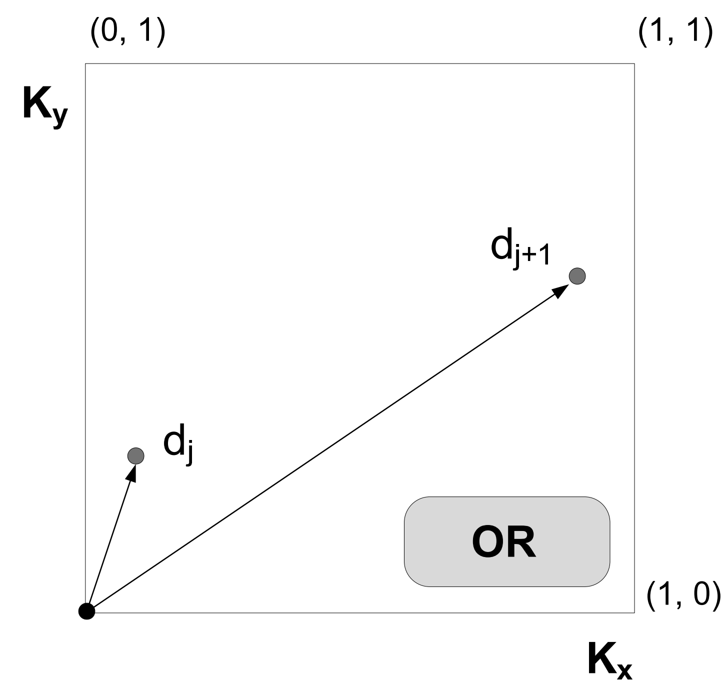
$$w_{x,j} = \frac{f_{x,j}}{max_x \;\; f_{x,j}} \times \frac{idf_x}{max_i \;\; idf_i}$$

- where
  - $f_{x,j}$ is the frequency of term k_x in document d_j
  - idf_i is the inverse document frequency of term k_i, as before
- To simplify notation, let
  - $w_{x,j} = x$ and $w_{y,j} = y$
  - $\vec{d}_j = (w_{x,j}, w_{y,j})$ as the point $d_j = (x, y)$

The Idea

- For a disjunctive query $q_{or} = k_x \vee k_y$, the point $(0,0)$ is the least interesting one
- This suggests taking the distance from $(0,0)$ as a measure of similarity

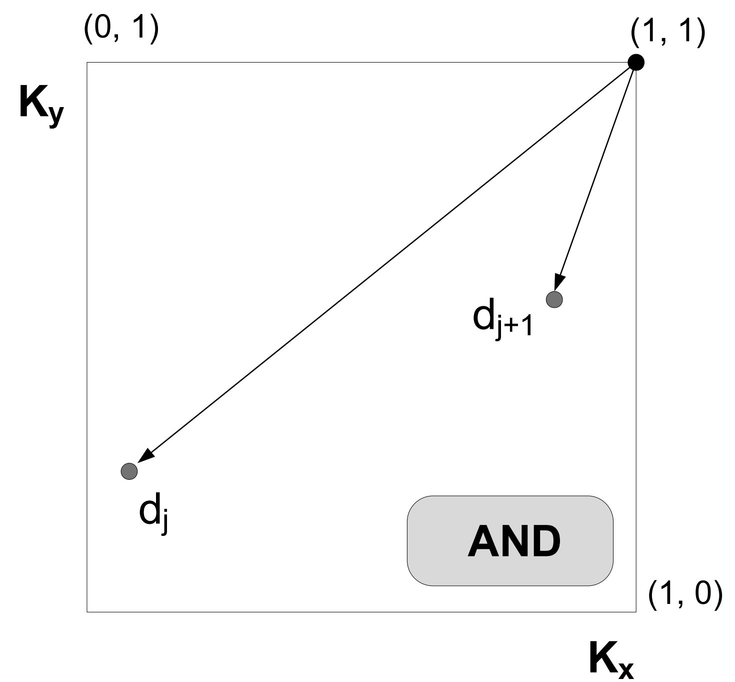
$$sim(q_{or}, d) = \sqrt{\frac{x^2 + y^2}{2}}$$



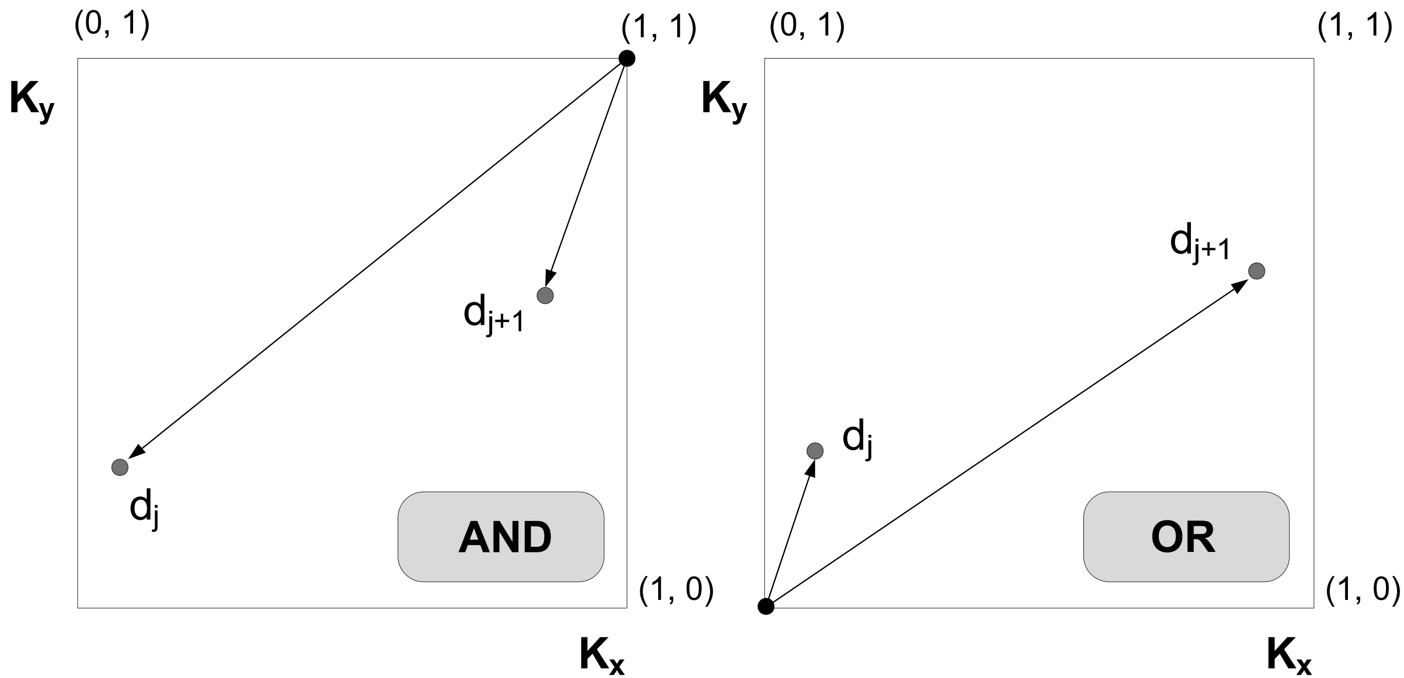
The Idea

- For a conjunctive query $q_{and} = k_x \wedge k_y$, the point $(1, 1)$ is the most interesting one
- This suggests taking the complement of the distance from the point $(1, 1)$ as a measure of similarity

$$sim(q_{and}, d) \quad = \quad 1 - \sqrt{\frac{(1-x)^2 + (1-y)^2}{2}}$$



The Idea



$$sim(q_{or}, d) = \sqrt{\frac{x^2 + y^2}{2}}$$

$$sim(q_{and}, d) = 1 - \sqrt{\frac{(1-x)^2 + (1-y)^2}{2}}$$

Generalizing the Idea

- We can extend the previous model to consider Euclidean distances in a t-dimensional space
- This can be done using *p-norms* which extend the notion of distance to include p-distances, where $1 \leq p \leq \infty$
- A generalized conjunctive query is given by
  - $q_{and} = k_1 \ \wedge^p \ k_2 \ \wedge^p \ \ldots \wedge^p k_m$
- A generalized disjunctive query is given by
  - $q_{or} = k_1 \ \vee^p \ k_2 \ \vee^p \ \ldots \vee^p k_m$

Generalizing the Idea

- The query-document similarities are now given by

$$sim(q_{or}, d_j) = \left(\frac{x_1^p + x_2^p + \ldots + x_m^p}{m}\right)^{\frac{1}{p}}$$

$$sim(q_{and}, d_j) = 1 - \left(\frac{(1-x_1)^p + (1-x_2)^p + \ldots + (1-x_m)^p}{m}\right)^{\frac{1}{p}}$$

where each x_i stands for a weight $w_{i,d}$

- If $p = 1$ then (vector-like)
  - $sim(q_{or}, d_j) = sim(q_{and}, d_j) = \frac{x_1 + \ldots + x_m}{m}$
- If $p = \infty$ then (Fuzzy like)
  - $sim(q_{or}, d_j) = max(x_i)$
  - $sim(q_{and}, d_j) = min(x_i)$

Properties

- By varying p, we can make the model behave as a vector, as a fuzzy, or as an intermediary model
- The processing of more general queries is done by grouping the operators in a predefined order
- For instance, consider the query $q = (k_1 \wedge^p k_2) \vee^p k_3$
  - k_1 and k_2 are to be used as in a vectorial retrieval while the presence of k_3 is required
- The similarity $sim(q, d_j)$ is computed as

$$sim(q,d) = \left(\frac{\left(1 - \left(\frac{(1-x_1)^p + (1-x_2)^p}{2}\right)^{\frac{1}{p}}\right)^p + x_3^p}{2} \right)^{\frac{1}{p}}$$

Conclusions

- Model is quite powerful
- Properties are interesting and might be useful
- Computation is somewhat complex
- However, distributivity operation does not hold for ranking computation:
  - $q_1 = (k_1 \vee k_2) \wedge k_3$
  - $q_2 = (k_1 \wedge k_3) \vee (k_2 \wedge k_3)$
  - $sim(q_1, d_j) \neq sim(q_2, d_j)$

Set Theoretic Models

Fuzzy Set Model

Fuzzy Set Model

- Matching of a document to a query terms is approximate or vague
- This **vagueness** can be modeled using a fuzzy framework, as follows:
  - each query term defines a **fuzzy** set
  - each doc has a **degree of membership** in this set
- This interpretation provides the foundation for many IR models based on fuzzy theory
- In here, we discuss the model proposed by Ogawa, Morita, and Kobayashi

Fuzzy Set Theory

- Fuzzy set theory deals with the representation of classes whose boundaries are not well defined
- Key idea is to introduce the notion of a **degree of membership** associated with the elements of the class
- This degree of membership varies from 0 to 1 and allows modelling the notion of **marginal** membership
- Thus, membership is now a **gradual** notion, contrary to the crispy notion enforced by classic Boolean logic

Fuzzy Set Theory

- A fuzzy subset A of a universe of discourse U is characterized by a membership function

$$\mu_A : U \rightarrow [0,1]$$

- This function associates with each element u of U a number $\mu_A(u)$ in the interval $[0,1]$
- The three most commonly used operations on fuzzy sets are:
  - the complement of a fuzzy set
  - the union of two or more fuzzy sets
  - the intersection of two or more fuzzy sets

Fuzzy Set Theory

- Let,
  - U be the universe of discourse
  - A and B be two fuzzy subsets of U
  - $\overline{A}$ be the complement of A relative to U
  - u be an element of U
- Then,

$$\begin{aligned} \mu_{\overline{A}}(u) &= 1 - \mu_A(u) \\ \mu_{A \cup B}(u) &= max(\mu_A(u), \mu_B(u)) \\ \mu_{A \cap B}(u) &= min(\mu_A(u), \mu_B(u)) \end{aligned}$$

Fuzzy Information Retrieval

- Fuzzy sets are modeled based on a thesaurus, which defines term relationships
- A thesaurus can be constructed by defining a term-term correlation matrix C
- Each element of C defines a normalized correlation factor $c_{i,\ell}$ between two terms k_i and k_ℓ

$$c_{i,l} = \frac{n_{i,l}}{n_i + n_l - n_{i,l}}$$

where

- n_i: number of docs which contain k_i
- n_l: number of docs which contain k_l
- $n_{i,l}$: number of docs which contain both k_i and k_l

Fuzzy Information Retrieval

- We can use the term correlation matrix C to associate a fuzzy set with each index term k_i
- In this fuzzy set, a document d_j has a degree of membership $\mu_{i,j}$ given by

$$\mu_{i,j} = 1 - \prod_{k_l \, \in \, d_j} (1 - c_{i,l})$$

- The above expression computes an algebraic sum over all terms in d_j
- A document d_j belongs to the fuzzy set associated with k_i, if its own terms are associated with k_i

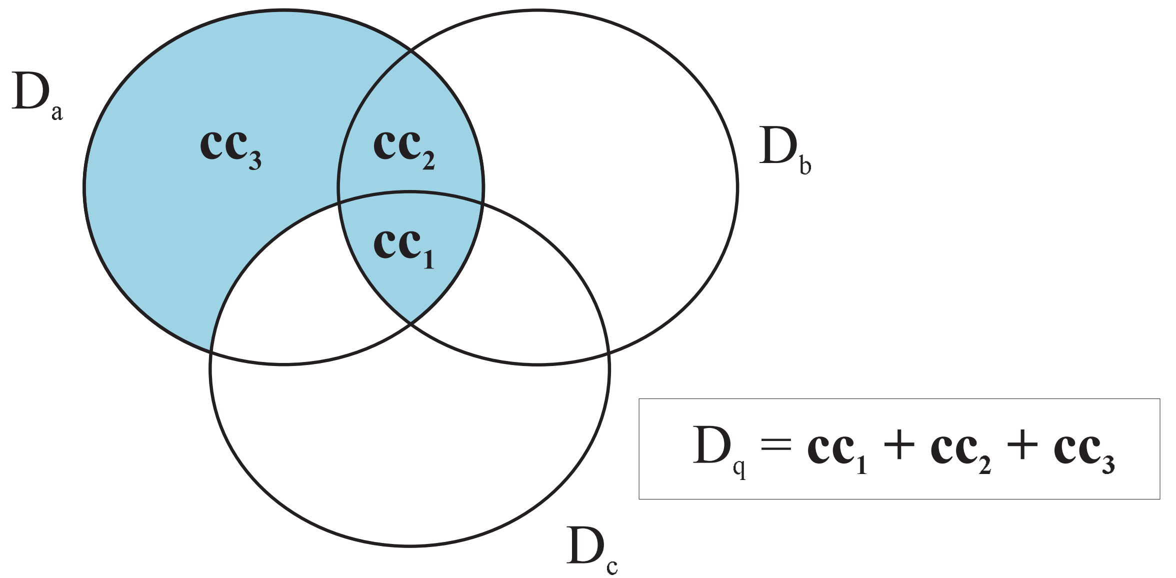
Fuzzy Information Retrieval

- If d_j contains a term k_l which is closely related to k_i, we have
  - $c_{i,l} \sim 1$
  - $\mu_{i,j} \sim 1$
  - and k_i is a good fuzzy index for d_j

$$\mu_{i,j} = 1 - \prod_{k_l \in d_j} (1 - c_{i,l})$$

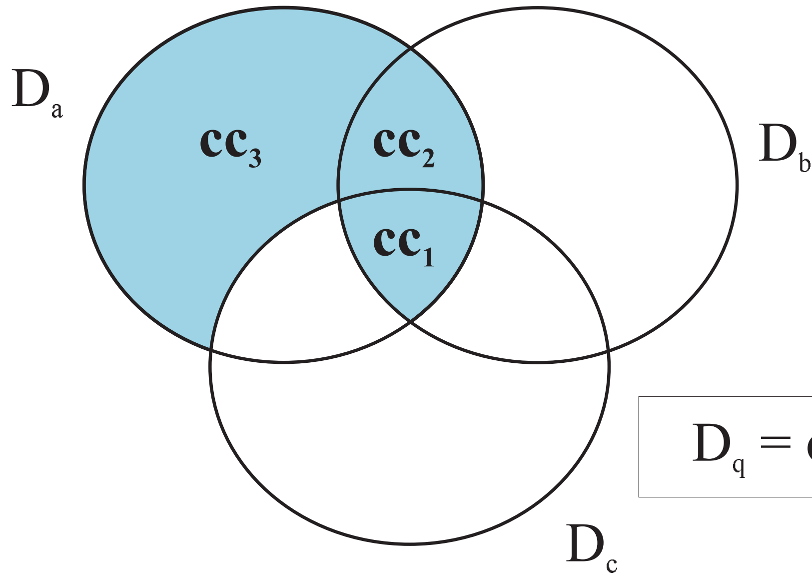
$\mu_{i,j}$: membership of doc d_j in fuzzy subset associated with k_i

Fuzzy IR: An Example



- Consider the query $q = k_a \ \wedge \ (k_b \ \vee \ \neg k_c)$
- Disjunct normal form of q is given by
  $\vec{q}_{dnf} = (1,1,1) + (1,1,0) + (1,0,0) = cc_1 + cc_2 + cc_3$
- Let D_a, D_b and D_c be the fuzzy set of documents associated to the terms k_a, k_b and k_a, respectively

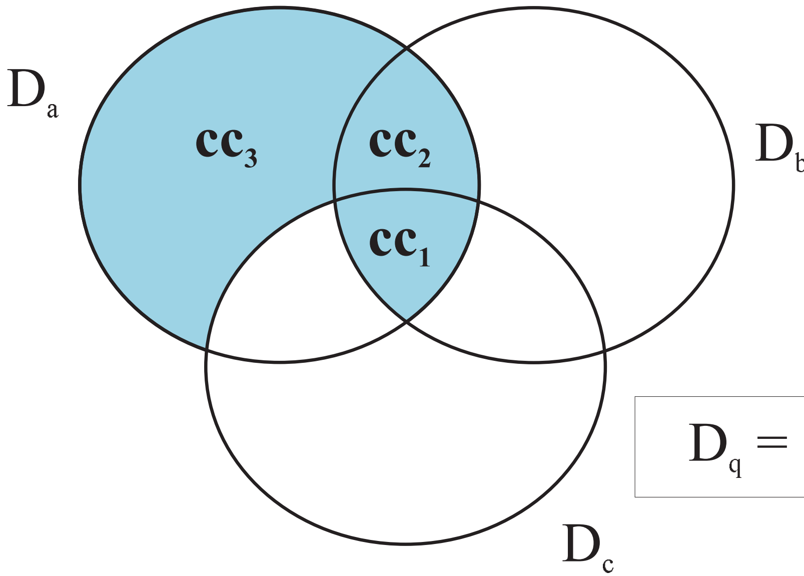
Fuzzy IR: An Example



$$D_q = \mathbf{cc_1} + \mathbf{cc_2} + \mathbf{cc_3}$$

- Let $\mu_{a,j}$, $\mu_{b,j}$, and $\mu_{c,j}$ be the degrees of memberships of document d_j in the fuzzy sets D_a, D_b, and D_c
- In this case, we have $cc_1 = \mu_{a,j}\mu_{b,j}\mu_{c,j}$, $cc_2 = \mu_{a,j}\mu_{b,j}(1 - \mu_{c,j})$, and $cc_3 = \mu_{a,j}(1 - \mu_{b,j})(1 - \mu_{c,j})$

Fuzzy IR: An Example



$D_q = \mathbf{cc_1} + \mathbf{cc_2} + \mathbf{cc_3}$

$$
\begin{aligned}
\mu_{q,j} &= \mu_{cc_1+cc_2+cc_3,j} \\
&= 1 - \prod_{i=1}^{3} (1 - \mu_{cc_i,j}) \\
&= 1 - (1 - \mu_{a,j}\mu_{b,j}\mu_{c,j}) \times \\
&\quad (1 - \mu_{a,j}\mu_{b,j}(1 - \mu_{c,j})) \times (1 - \mu_{a,j}(1 - \mu_{b,j})(1 - \mu_{c,j}))
\end{aligned}
$$

Conclusions

- Fuzzy IR models have been discussed mainly in the literature associated with fuzzy theory
- They provide an interesting framework which naturally embodies the notion of term dependencies
- Experiments with standard test collections are not available

Alternative Algebraic Models

Generalized Vector Model

Generalized Vector Model

- Classic models enforce independence of index terms
- For instance, in the Vector model
  - A set of term vectors $\{\vec{k}_1, \vec{k}_2, \ldots, \vec{k}_t\}$ are linearly independent
  - Frequently, this is interpreted as $\forall_{i,j} \Rightarrow \vec{k}_i \bullet \vec{k}_j = 0$
- In the generalized vector space model, two index term vectors might be non-orthogonal

Key Idea

- As before, let $w_{i,j}$ be the weight associated with $[k_i, d_j]$ and V = {k_1, k_2, ..., k_t} be the set of all terms
- If the $w_{i,j}$ weights are binary, all patterns of occurrence of terms within docs can be represented by minterms:

$$
\begin{array}{rcl}
 & & (k_1, k_2, k_3, \ldots, k_t) \\
m_1 & = & (0,\ 0,\ 0,\ \ \ldots,\ 0) \\
m_2 & = & (1,\ 0,\ 0,\ \ \ldots,\ 0) \\
m_3 & = & (0,\ 1,\ 0,\ \ \ldots,\ 0) \\
m_4 & = & (1,\ 1,\ 0,\ \ \ldots,\ 0) \\
 & \vdots & \\
m_{2^t} & = & (1,\ 1,\ 1,\ \ \ldots,\ 1)
\end{array}
$$

For instance, m_2 indicates documents in which solely the term k_1 occurs

Key Idea

- For any document d_j, there is a minterm m_r that includes exactly the terms that occur in the document
- Let us define the following set of minterm vectors $\vec{m}_r$,

$$
\begin{array}{rcl}
 & & 1, 2, \ldots, 2^t \\
\vec{m}_1 & = & (1, 0, \ldots, 0) \\
\vec{m}_2 & = & (0, 1, \ldots, 0) \\
 & \vdots & \\
\vec{m}_{2^t} & = & (0, 0, \ldots, 1)
\end{array}
$$

Notice that we can associate each unit vector $\vec{m}_r$ with a minterm m_r, and that $\vec{m}_i \bullet \vec{m}_j = 0$ for all $i \neq j$

Key Idea

- Pairwise orthogonality among the $\vec{m}_r$ vectors does not imply independence among the index terms
- On the contrary, index terms are now correlated by the $\vec{m}_r$ vectors
  - For instance, the vector $\vec{m}_4$ is associated with the minterm $m_4 = (1, 1, \ldots, 0)$
  - This minterm induces a dependency between terms k_1 and k_2
  - Thus, if such document exists in a collection, we say that the minterm m_4 is active
- The model adopts the idea that co-occurrence of terms induces dependencies among these terms

Forming the Term Vectors

- Let $on(i, m_r)$ return the weight $\{0, 1\}$ of the index term k_i in the minterm m_r
- The vector associated with the term k_i is computed as:

$$\vec{k}_i = \frac{\sum_{\forall r} on(i, m_r) \; c_{i,r} \; \vec{m}_r}{\sqrt{\sum_{\forall r} on(i, m_r) \; c_{i,r}^2}}$$

$$c_{i,r} = \sum_{d_j \mid c(d_j) = m_r} w_{i,j}$$

- Notice that for a collection of size N, only N minterms affect the ranking (and not 2^t)

Dependency between Index Terms

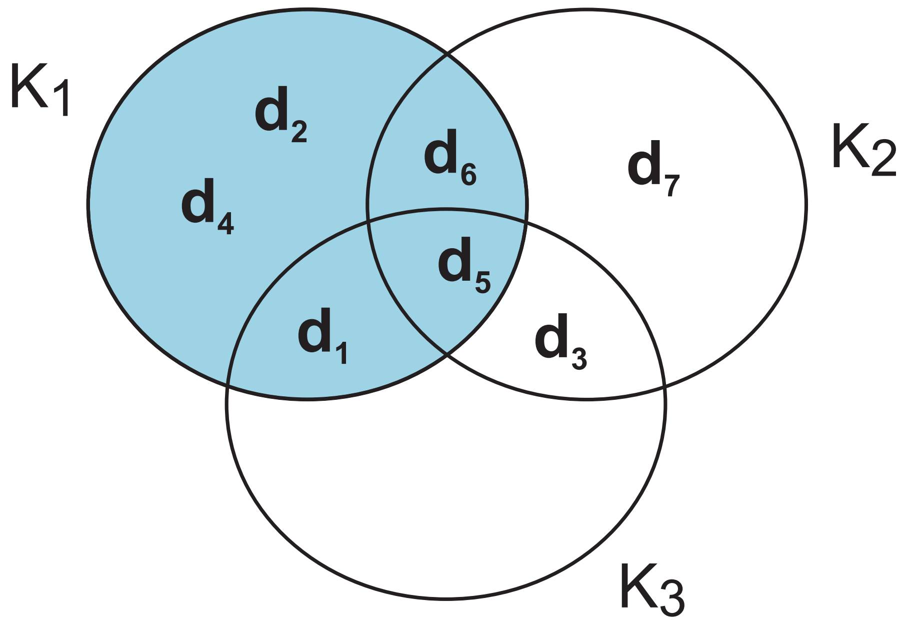
- A degree of correlation between the terms k_i and k_j can now be computed as:

$$\vec{k}_i \bullet \vec{k}_j = \sum_{\forall r} on(i, m_r) \times c_{i,r} \times on(j, m_r) \times c_{j,r}$$

- This degree of correlation sums up the dependencies between k_i and k_j induced by the docs in the collection

The Generalized Vector Model

- An Example



	K_1	K_2	K_3
d_1	2	0	1
d_2	1	0	0
d_3	0	1	3
d_4	2	0	0
d_5	1	2	4
d_6	1	2	0
d_7	0	5	0
q	1	2	3

Computation of $c_{i,r}$

	K_1	K_2	K_3
d_1	2	0	1
d_2	1	0	0
d_3	0	1	3
d_4	2	0	0
d_5	1	2	4
d_6	0	2	2
d_7	0	5	0
q	1	2	3

	K_1	K_2	K_3
$d_1 = m_6$	1	0	1
$d_2 = m_2$	1	0	0
$d_3 = m_7$	0	1	1
$d_4 = m_2$	1	0	0
$d_5 = m_8$	1	1	1
$d_6 = m_7$	0	1	1
$d_7 = m_3$	0	1	0
$q = m_8$	1	1	1

	$c_{1,r}$	$c_{2,r}$	$c_{3,r}$
m_1	0	0	0
m_2	3	0	0
m_3	0	5	0
m_4	0	0	0
m_5	0	0	0
m_6	2	0	1
m_7	0	3	5
m_8	1	2	4

Computation of $\overrightarrow{k_i}$

- $\overrightarrow{k_1} = \frac{(3m_2+2m_6+m_8)}{\sqrt{3^2+2^2+1^2}}$
- $\overrightarrow{k_2} = \frac{(5m_3+3m_7+2m_8)}{\sqrt{5+3+2}}$
- $\overrightarrow{k_3} = \frac{(1m_6+5m_7+4m_8)}{\sqrt{1+5+4}}$

	$c_{1,r}$	$c_{2,r}$	$c_{3,r}$
m_1	0	0	0
m_2	3	0	0
m_3	0	5	0
m_4	0	0	0
m_5	0	0	0
m_6	2	0	1
m_7	0	3	5
m_8	1	2	4

Computation of Document Vectors

- $\overrightarrow{d_1} = 2\overrightarrow{k_1} + \overrightarrow{k_3}$
- $\overrightarrow{d_2} = \overrightarrow{k_1}$
- $\overrightarrow{d_3} = \overrightarrow{k_2} + 3\overrightarrow{k_3}$
- $\overrightarrow{d_4} = 2\overrightarrow{k_1}$
- $\overrightarrow{d_5} = \overrightarrow{k_1} + 2\overrightarrow{k_2} + 4\overrightarrow{k_3}$
- $\overrightarrow{d_6} = 2\overrightarrow{k_2} + 2\overrightarrow{k_3}$
- $\overrightarrow{d_7} = 5\overrightarrow{k_2}$
- $\overrightarrow{q} = \overrightarrow{k_1} + 2\overrightarrow{k_2} + 3\overrightarrow{k_3}$

	K_1	K_2	K_3
d_1	2	0	1
d_2	1	0	0
d_3	0	1	3
d_4	2	0	0
d_5	1	2	4
d_6	0	2	2
d_7	0	5	0
q	1	2	3

Conclusions

- Model considers correlations among index terms
- Not clear in which situations it is superior to the standard Vector model
- Computation costs are higher
- Model does introduce interesting new ideas

Alternative Algebraic Models

Latent Semantic Indexing

Latent Semantic Indexing

- 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

Latent Semantic Indexing

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

Latent Semantic Indexing

- 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 $\mathbf{M} = [m_{ij}]$ be given by the matrix

	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 $\mathbf{D}^t$

Latent Semantic Indexing

- 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

Latent Ranking

- The user query can be modelled as a pseudo-document in the original $\mathbf{M}$ matrix
- Assume the query is modelled as the document numbered 0 in the $\mathbf{M}$ matrix
- The matrix $\mathbf{M}_s^T \cdot \mathbf{M}_s$ 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

Conclusions

- Latent semantic indexing provides an interesting conceptualization of the IR problem
- Thus, it has its value as a new theoretical framework
- From a practical point of view, the latent semantic indexing model has not yielded encouraging results

Alternative Algebraic Models

Neural Network Model

Neural Network Model

- Classic IR:
  - Terms are used to index documents and queries
  - Retrieval is based on index term matching
- Motivation:
  - Neural networks are known to be good pattern matchers

Neural Network Model

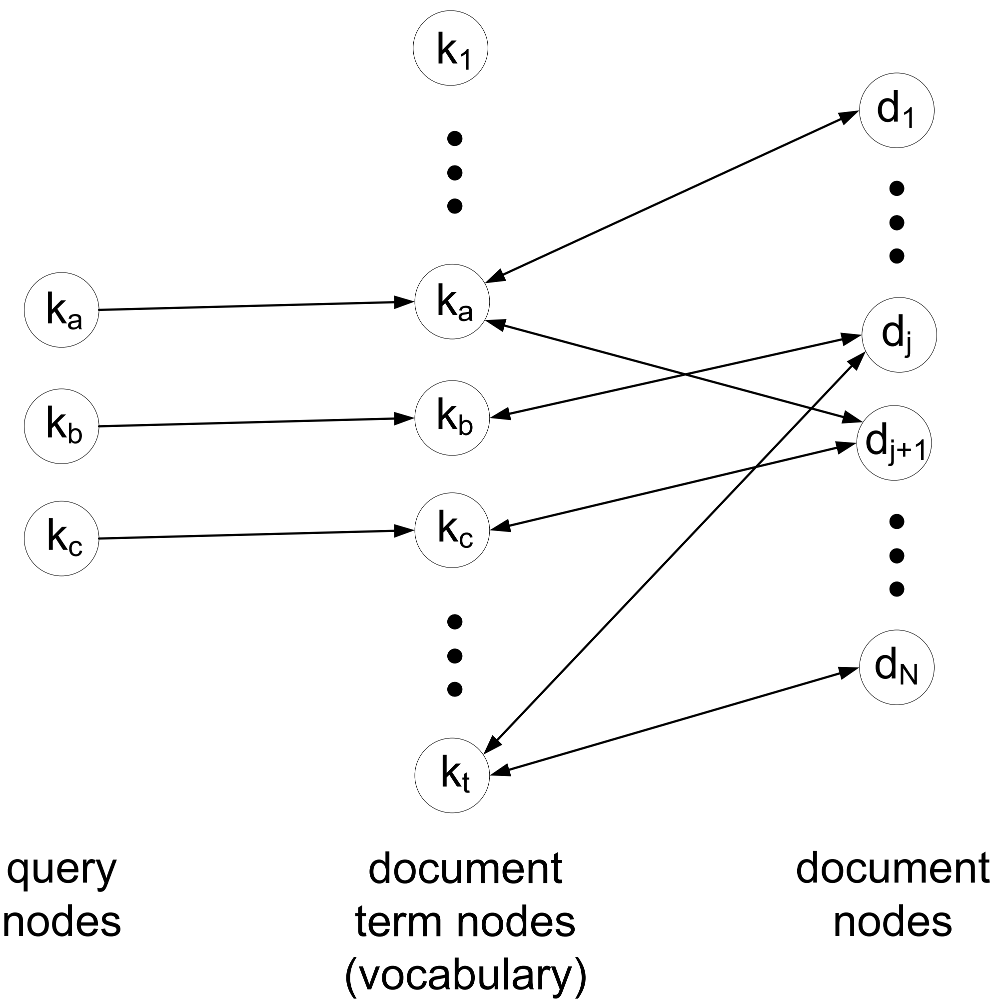
- The human brain is composed of billions of neurons
- Each neuron can be viewed as a small processing unit
- A neuron is stimulated by input signals and emits output signals in reaction
- A chain reaction of propagating signals is called a **spread activation process**
- As a result of spread activation,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

Neural Network Model

- 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 for IR

- A neural network model for information retrieval



Neural Network for IR

- Three layers network: one for the query terms, one for the document terms, and a third one for the documents
- Signals propagate across the network
- First level of propagation:
  - Query terms issue the first signals
  - These signals propagate across the network to reach the document nodes
- Second level of propagation:
  - Document nodes might themselves generate new signals which affect the document term nodes
  - Document term nodes might respond with new signals of their own

Quantifying Signal Propagation

- Normalize signal strength (MAX = 1)
- Query terms emit initial signal equal to 1
- Weight associated with an edge from a query term node k_i to a document term node k_i:

$$\overline{w}_{i,q} = \frac{w_{i,q}}{\sqrt{\sum_{i=1}^{t} \quad w_{i,q}^2}}$$

- Weight associated with an edge from a document term node k_i to a document node d_j:

$$\overline{w}_{i,j} = \frac{w_{i,j}}{\sqrt{\sum_{i=1}^{t} \quad w_{i,j}^2}}$$

Quantifying Signal Propagation

- After the first level of signal propagation, the activation level of a document node d_j is given by:

$$\sum_{i=1}^{t} \overline{w}_{i,q} \, \overline{w}_{i,j} = \frac{\sum_{i=1}^{t} \; w_{i,q} \, w_{i,j}}{\sqrt{\sum_{i=1}^{t} \; w_{i,q}^2} \times \sqrt{\sum_{i=1}^{t} \; w_{i,j}^2}}$$

  which is exactly the ranking of the Vector model

- New signals might be exchanged among document term nodes and document nodes

- A minimum threshold should be enforced to avoid spurious signal generation

Conclusions

- Model provides an interesting formulation of the IR problem
- Model has not been tested extensively
- It is not clear the improvements that the model might provide