

Query Operations

Berlin Chen 2003

Reference:

1. Modern Information Retrieval, chapter 5

Introduction

- Users have no detailed knowledge of
 - The collection makeup
 - The retrieval environment

} Difficult to formulate queries
- Scenario of (Web) IR
 1. An initial (naive) query posed to retrieve relevant docs
 2. Docs retrieved are examined for relevance and a new improved query formulation is constructed and posed again

↓

Expand the original query with new terms (query expansion) and reweight the terms in the expanded query (term weighting)

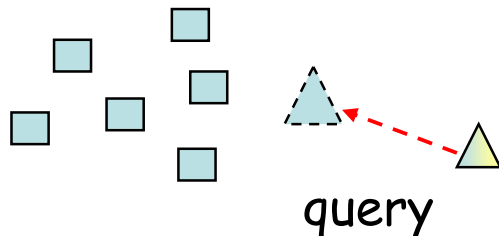
Query Reformulation

- Approaches through *query expansion (QE)* and *terming weighting*
 - *Feedback information from the user*
 - **Relevance feedback**
 - With vector, probabilistic models et al.
 - *Information derived from the set of documents initially retrieved* (called local set of documents)
 - **Local analysis**
 - Local clustering, local context analysis
 - *Global information derived from document collection*
 - **Global analysis**
 - Similar thesaurus or statistical thesaurus

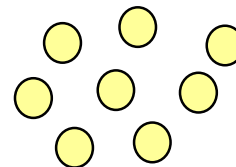
Relevance Feedback

- User (or Automatic) Relevance Feedback
 - The most popular query reformation strategy
- Process for user relevance feedback
 - A list of retrieved docs is presented
 - User or system exam them (e.g. the top 10 or 20 docs) and marked the relevant ones
 - Important terms are selected from the docs marked as relevant, and the importance of them are enhanced in the new query formulation

relevant docs



irrelevant docs



User Relevance Feedback

- Advantages
 - Shield users from details of query reformulation
 - User only have to provide a relevance judgment on docs
 - Break down the whole searching task into a sequence of small steps
 - Provide a controlled process designed to emphasize some terms (relevant ones) and de-emphasize others (non-relevant ones)

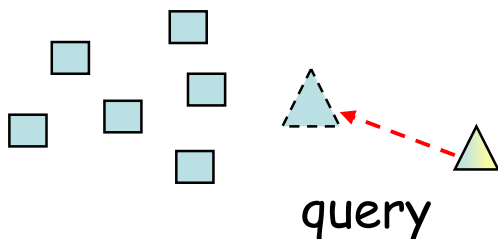
For **automatic relevance feedback**, the whole process is done in an implicit manner

Query Expansion and Term Reweighting for the Vector Model

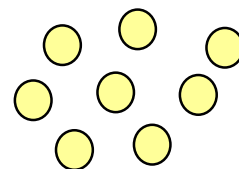
- **Assumptions**

- Relevant docs have term-weight vectors that resemble each other
- Non-relevant docs have term-weight vectors which are dissimilar from the ones for the relevant docs
- The reformulated query gets to closer to the term-weight vector space of relevant docs

relevant docs



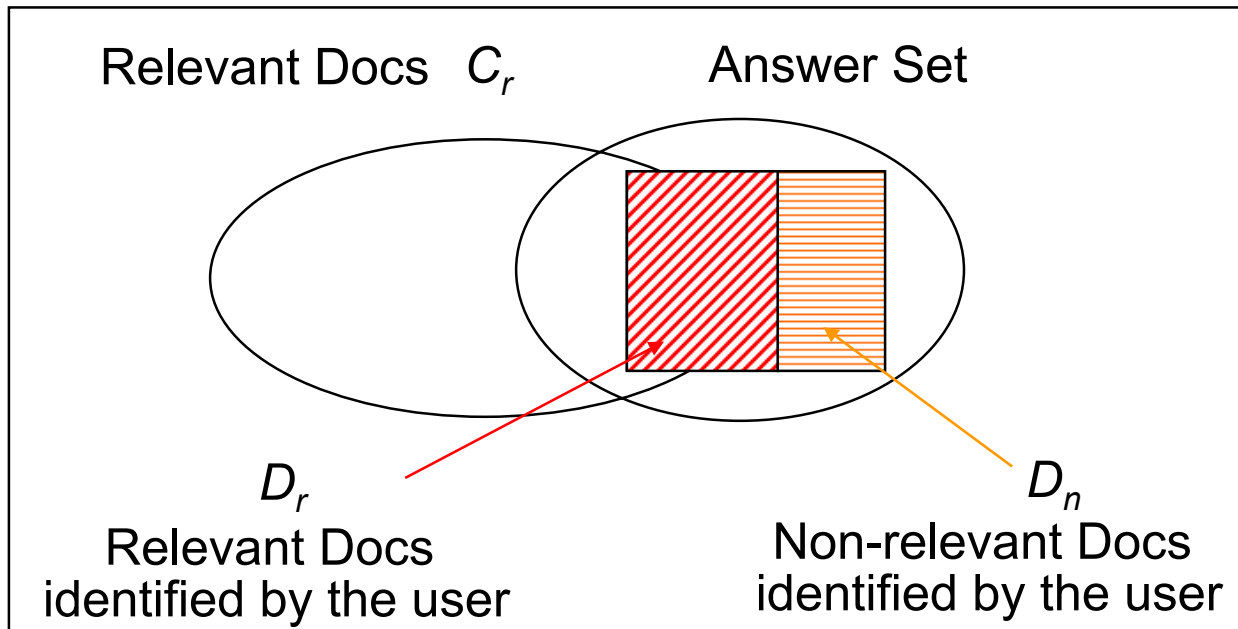
irrelevant docs



term-weight vectors

Query Expansion and Term Reweighting for the Vector Model

- **Terminology**



Doc Collection with size N

Query Expansion and Term Reweighting for the Vector Model

- **Optimal Condition**

- The complete set of relevant docs C_r to a given query q is known in advance

$$\vec{q}_{opt} = \frac{1}{|C_r|} \sum_{\forall \vec{d}_i \in C_r} \vec{d}_i - \frac{1}{N - |C_r|} \sum_{\forall \vec{d}_j \notin C_r} \vec{d}_j$$

- **Problem:** the complete set of relevant docs C_r are not known a priori
 - **Solution:** formulate an initial query and incrementally change the initial query vector based on the known relevant/non-relevant docs
 - User or automatic judgments

Query Expansion and Term Reweighting for the Vector Model

- In Practice**

1. Standard_Rocchio

Rocchio 1965

$$\vec{q}_m = \alpha \cdot \vec{q} + \frac{\beta}{|D_r|} \cdot \sum_{\forall \vec{d}_i \in D_r} \vec{d}_i - \frac{\gamma}{|D_n|} \cdot \sum_{\forall \vec{d}_j \in D_n} \vec{d}_j$$

modified query \vec{q}_m (indicated by a red arrow)
initial/original query \vec{q} (indicated by a red arrow)

2. Ide_Regular

$$\vec{q}_m = \alpha \cdot \vec{q} + \beta \cdot \sum_{\forall \vec{d}_i \in D_r} \vec{d}_i - \gamma \cdot \sum_{\forall \vec{d}_j \in D_n} \vec{d}_j$$

3. Ide_Dec_Hi

The highest ranked non-relevant doc

$$\vec{q}_m = \alpha \cdot \vec{q} + \beta \cdot \sum_{\forall \vec{d}_i \in D_r} \vec{d}_i - \gamma \cdot \max_{\text{non-relevant}} (\vec{d}_j)$$

Query Expansion and Term Reweighting for the Vector Model

- **Some Observations**

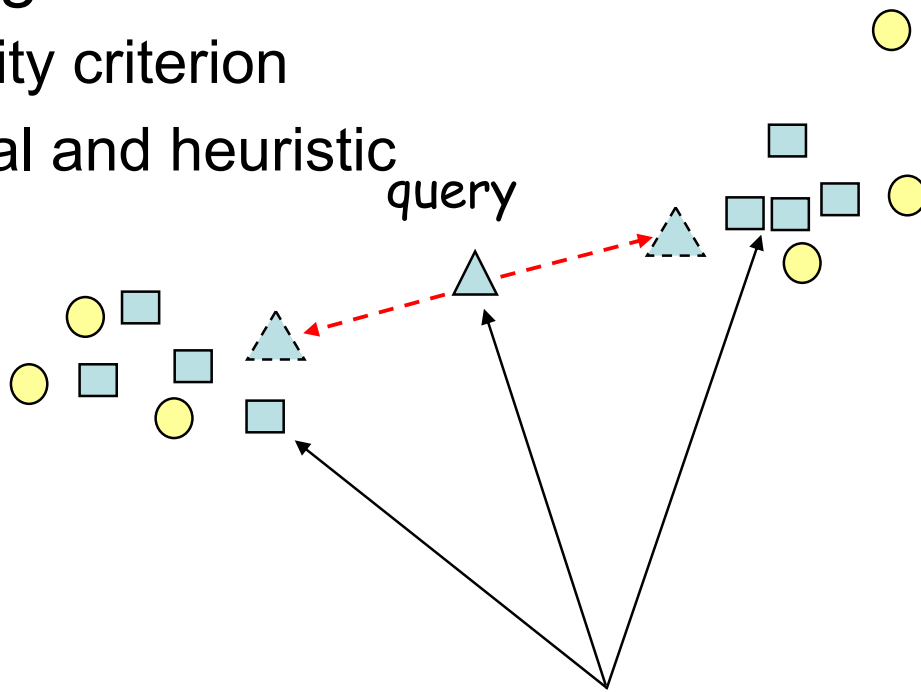
- Similar results were achieved for the above three approach (Dec-Hi slightly better in the past)
- Usually, constant β is bigger than γ (why?)

- **In Practice (cont.)**

- More about the constants
 - Rocchio, 1971: $\alpha = 1$
 - Ide, 1971: $\alpha = \beta = \gamma = 1$
 - **Positive feedback strategy:** $\gamma = 0$

Query Expansion and Term Reweighting for the Vector Model

- Advantages
 - Simple, good results
 - Modified term weights are computed directly from the retrieved docs
- Disadvantages
 - No optimality criterion
 - Empirical and heuristic



Term Reweighting for the Probabilistic Model

Roberston & Sparck Jones 1976

• Similarity Measure

$$\text{sim}(d_j, q) \approx \sum_{i=1}^t w_{i,q} \times w_{i,j} \times \left[\log \frac{P(k_i | R)}{1 - P(k_i | R)} + \log \frac{1 - P(k_i | \bar{R})}{P(k_i | \bar{R})} \right]$$

Binary weights (0 or 1) are used

prob. of observing term k_i in the set of relevant docs

• Initial Search (with some assumptions)

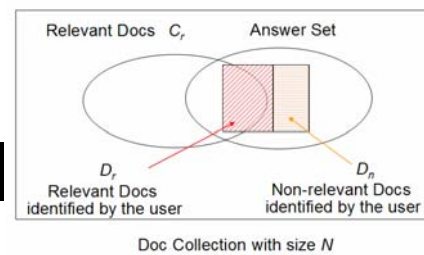
– $P(k_i | R) = 0.5$: is constant for all indexing terms

– $P(k_i | \bar{R}) = \frac{n_i}{N}$: approx. by doc freq. of index terms

$$\Rightarrow \text{sim}(d_j, q) \approx \sum_{i=1}^t w_{i,q} \times w_{i,j} \times \left[\log \frac{0.5}{1 - 0.5} + \log \frac{1 - \frac{n_i}{N}}{\frac{n_i}{N}} \right]$$

$$= \sum_{i=1}^t w_{i,q} \times w_{i,j} \times \log \frac{N - n_i}{n_i}$$

Term Reweighting for the Probabilistic Model



• Relevance feedback (term reweighting alone)

$$P(k_i | R) = \frac{|D_{r,i}|}{|D_r|}$$

← Relevant docs containing term k_i
 ← Relevant docs

$$P(k_i | \bar{R}) = \frac{n_i - |D_{r,i}|}{N - |D_r|}$$



$$\text{sim}(d_j, q) \approx \sum_{i=1}^t w_{i,q} \times w_{i,j} \times \left[\log \frac{\frac{|D_{r,i}|}{|D_r|}}{1 - \frac{|D_{r,i}|}{|D_r|}} + \log \frac{1 - \frac{n_i - |D_{r,i}|}{N - |D_r|}}{\frac{n_i - |D_{r,i}|}{N - |D_r|}} \right]$$

$$= \sum_{i=1}^t w_{i,q} \times w_{i,j} \times \log \left[\frac{|D_{r,i}|}{|D_r| - |D_{r,i}|} \cdot \frac{N - |D_r| - n_i + |D_{r,i}|}{n_i - |D_{r,i}|} \right]$$

Approach 1

$$P(k_i | R) = \frac{|D_{r,i}| + 0.5}{|D_r| + 1}$$

$$P(k_i | \bar{R}) = \frac{n_i - |D_{r,i}| + 0.5}{N - |D_r| + 1}$$

Approach 2

$$P(k_i | R) = \frac{|D_{r,i}| + \frac{n_i}{N}}{|D_r| + 1}$$

$$P(k_i | \bar{R}) = \frac{n_i - |D_{r,i}| + \frac{n_i}{N}}{N - |D_r| + 1}$$

Term Reweighting for the Probabilistic Model

- Advantages
 - Feedback process is directly related to the derivation of new weights for query terms
 - The term reweighting is optimal under the assumptions of term independence and binary doc indexing
- Disadvantages
 - Document term weights are not taken into considered
 - Weights of terms in previous query formulations are disregarded
 - No query expansion is used
 - The same set of index terms in the original query is reweighted over and over again

A Variant of Probabilistic Term Reweighting

Croft 1983

<http://ciir.cs.umass.edu/>

- **Differences**
 - Distinct initial search assumptions
 - **Within-document frequency weight** included
- **Initial search** (assumptions)

$$\text{sim}(d_j, q) \propto \sum_{i=1}^t w_{i,q} w_{i,j} F_{i,j,q}$$

$$F_{i,j,q} = (C + idf_i) \bar{f}_{i,j} \quad \bar{f}_{i,j} = K + (1 + K) \frac{f_{i,j}}{\max(f_{i,j})}$$

~ Inversed document frequency

~ Term frequency

(normalized within document frequency)

- C and K are adjusted with respect to the doc collection

A Variant of Probabilistic Term Reweighting

- **Relevance feedback**

$$F_{i,j,q} = \left(C + \log \frac{P(k_i | R)}{1 - P(k_i | R)} + \log \frac{1 - P(k_i | \bar{R})}{P(k_i | \bar{R})} \right) \bar{f}_{i,j}$$

$$P(k_i | R) = \frac{|D_{r,i}| + 0.5}{|D_r| + 1}$$

$$P(k_i | \bar{R}) = \frac{n_i - |D_{r,i}| + 0.5}{N - |D_r| + 1}$$

A Variant of Probabilistic Term Reweighting

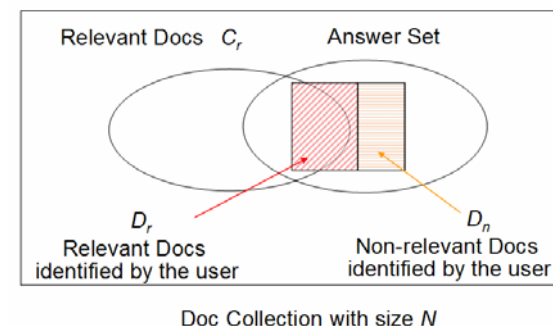
- Advantages
 - The within-doc frequencies are considered
 - A normalized version of these frequencies is adopted
 - Constants C and K are introduced for greater flexibility
- Disadvantages
 - More complex formulation
 - No query expansion (just reweighting of index terms)

Evaluation of relevance feedback Strategies

- Recall-precision figures of user reference feedback is unrealistic
 - Since the user has seen the docs during reference feedback
 - A significant part of the improvement results from the high ranker ranks assigned to the set R of docs

$$\vec{q}_m = \alpha \cdot \vec{q} + \frac{\beta}{|D_r|} \cdot \sum_{\forall \vec{d}_i \in D_r} \vec{d}_i - \frac{\gamma}{|D_n|} \cdot \sum_{\forall \vec{d}_j \in D_n} \vec{d}_j$$

↗ \vec{q}_m modified query ↖ \vec{q} original query



- The real gains in retrieval performance should be measured based on the docs **not seen** by the user yet

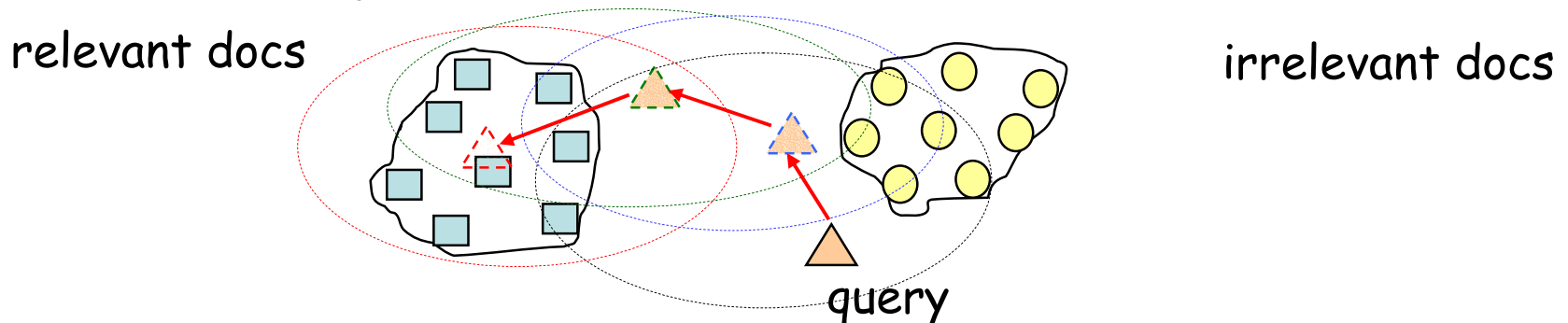
Evaluation of relevance feedback Strategies

- Recall-precision figures relative to the residual collection
 - Residual collection
 - The set of all docs minus the set of feedback docs provided by the user
 - Evaluate the retrieval performance of the modified query \vec{q}_m considering only the residual collection
 - The recall-precision figures for \vec{q}_m tend to be lower than the figures for the original query \vec{q}
 - It's OK ! If we just want to compare the performance of different relevance feedback strategies

Automatic Local/Global Analysis

- **Recall** - in user relevance feedback cycles
 - Top ranked docs separated into two classes
 - Relevant docs
 - Non-relevant docs
 - Terms in known relevant docs help describe a larger cluster of relevant docs
 - From a “**clustering**” perspective
 - Description of larger cluster of relevant docs is built iteratively **with assistance from the user**

Attar and Fraenkel 1977



Automatic Local/Global Analysis

- **Alternative approach:** automatically obtain the description for a large cluster of relevant docs
 - **Identify terms which are related to the query terms**
 - Synonyms
 - Stemming variations
 - Terms are close each other in context

陳水扁 總統 李登輝 總統府 秘書長 陳師孟 一邊一國...

連戰 宋楚瑜 國民黨 一個中國 ...

Automatic Local/Global Analysis

- Two strategies
 - Global analysis
 - All docs in collection are used to determine a global thesaurus-like structure for QE
 - Local analysis
 - Similar to relevance feedback but without user interference
 - Docs retrieved at query time are used to determine terms for QE
 - Local clustering, local context analysis

QE through Local Clustering

- QE through **Clustering**

- Build **global structures** such as **association matrices** to quantify term correlations
- Use the correlated terms for QE
- But not always effective in general collections

陳水扁 總統 呂秀蓮 綠色矽島 勇哥 吳淑珍 ...

陳水扁 視察 阿里山 小火車

- QE through **Local Clustering**

- Operate solely on the docs retrieved for the query
- Not suitable for Web search: time consuming
- Suitable for intranets

- Especially, as the assistance for search information in specialized doc collections like medical doc collections

QE through Local Clustering

- Definition
 - Stem
 - $V(s)$: a non-empty subset of words which are grammatical variants of each other
 - E.g. {polish, polishing, polished}
 - A canonical form s of $V(s)$ is called a **stem**
 - e.g., $s = \text{polish}$
 - For a given query
 - Local doc set D_i : the set of documents retrieved
 - local vocabulary V_i : the set of all distinct words (stems) in the local document set
 - S_i : the set of all distinct stem derived from V_i

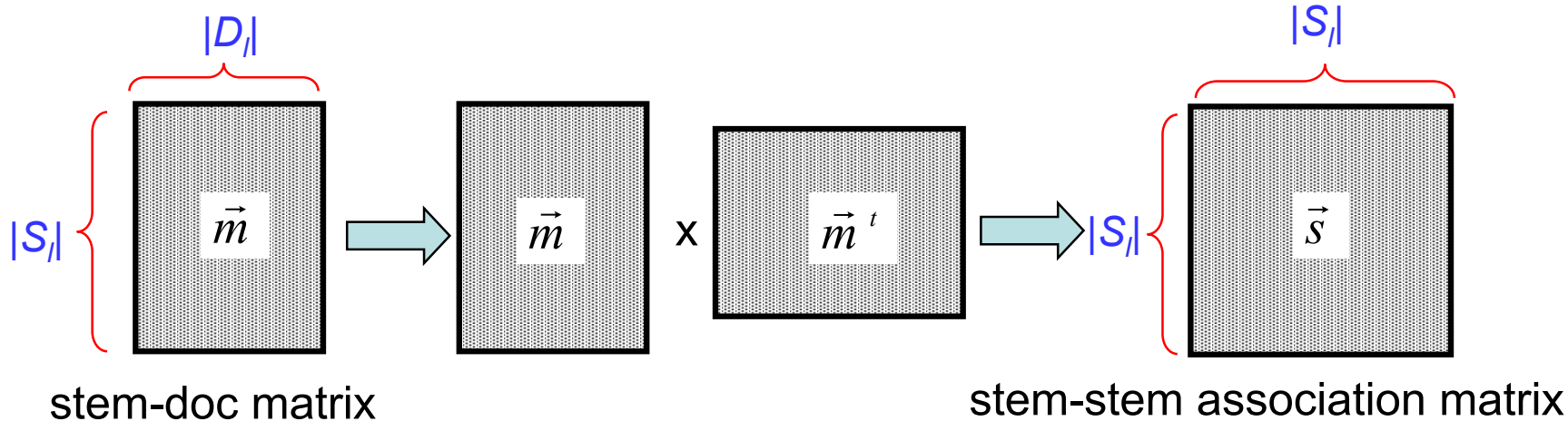
Strategies for Building Local Clusters

- **Association clusters**
 - Consider the **co-occurrence** of stems (terms) inside docs
- **Metric Clusters**
 - Consider the **distance** between two terms in a doc
- **Scalar Clusters**
 - Consider the **neighborhoods** of two terms
 - Do they have similar neighborhoods?

Strategies for Building Local Clusters

- **Association clusters**

- Based on the **co-occurrence** of stems (terms) inside docs
 - Assumption: stems co-occurring frequently inside docs have a **synonymy** association
- An association matrix with $|S_i|$ rows and $|D_i|$ columns
 - Each entry $f_{s_i,j}$ the frequency of a stem s_i in a doc d_j



Strategies for Building Local Clusters

- **Association clusters**

- Each entry in the stem-stem association matrix stands for **the correlation factor** between two stems

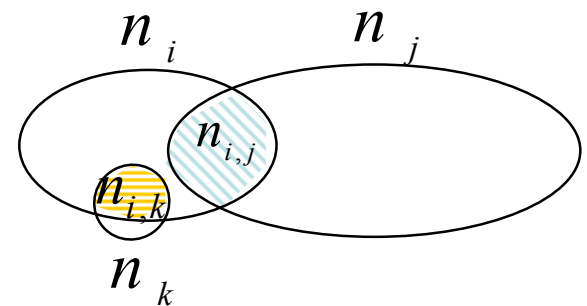
$$C_{u,v} = \sum_{d_j \in D_l} f_{s_{u,j}} \times f_{s_{v,j}}$$

- The unnormalized form

$$S_{u,v} = C_{u,v}$$

- The normalized form (**ranged from 0 to 1**)

$$S_{u,v} = \frac{C_{u,v}}{C_{u,u} + C_{v,v} - C_{u,v}}$$



Tanimoto coefficient

Strategies for Building Local Clusters

- **Association clusters**

- The u -th row in the association matrix stands all the associations for the stem s_u
- A **local association cluster** $S_u(m)$
 - Defined as a set of stems s_v ($v \neq u$) with their respective values $s_{u,v}$ being the **top m** ones in the u -th row of the association matrix
- Given a query, only the association clusters of query terms are calculated
 - The stems (terms) belong to the association clusters are selected and added the query formulation

Strategies for Building Local Clusters

- **Association clusters**

- Other measures for term association

- Dice coefficient

$$S_{u,v} = \frac{2 \times c_{u,v}}{c_{u,u} + c_{v,v}}$$

- Mutual information

$$s_{u,v} = MI(k_u, k_v) = \log \frac{P(k_u, k_v)}{P(k_u)P(k_v)} = \log \frac{\frac{n_{u,v}}{N}}{\frac{n_u}{N} \times \frac{n_v}{N}}$$

Strategies for Building Local Clusters

- **Metric Clusters**

- Take into consideration the **distance** between two terms in a doc while computing their correlation factor

$$C_{u,v} = \sum_{k_i \in V(s_u)} \sum_{k_j \in V(s_v)} \frac{1}{r(k_i, k_j)}$$

no. of words between k_i and k_j in the same doc

$r(k_i, k_j) = \infty$ if k_i and k_j are in distinct docs

- The entry of **local stem-stem metric correlation** matrix \vec{s} can be expressed as

- The unnormalized form

$$S_{u,v} = C_{u,v}$$

- The normalized form

$$S_{u,v} = \frac{C_{u,v}}{|V(s_u)| \times |V(s_v)|}$$

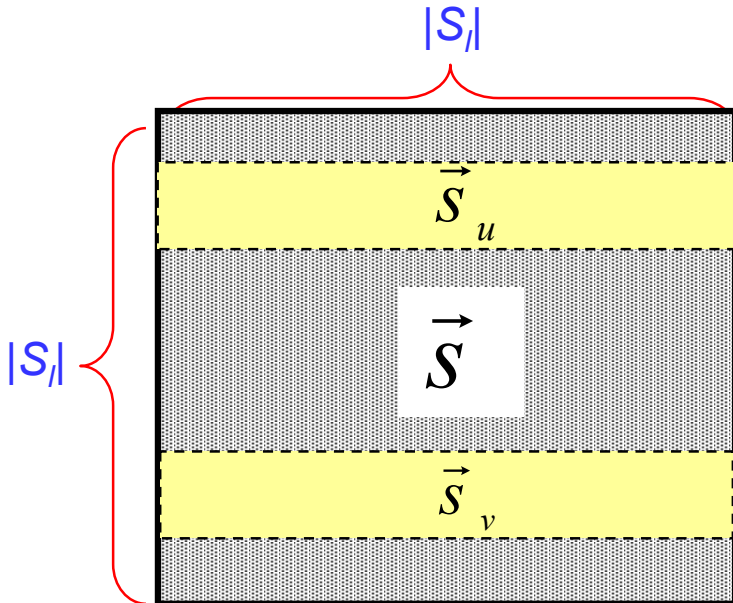
The local association clusters of stems can be similarly defined

ranged from 0 to 1

Strategies for Building Local Clusters

- **Scalar Clusters**

- **Idea:** two stems (terms) with similar neighborhoods have some synonymy relationship
- Derive the synonymy relationship between two stems by comparing the sets $S_u(m)$ and $S_v(m)$



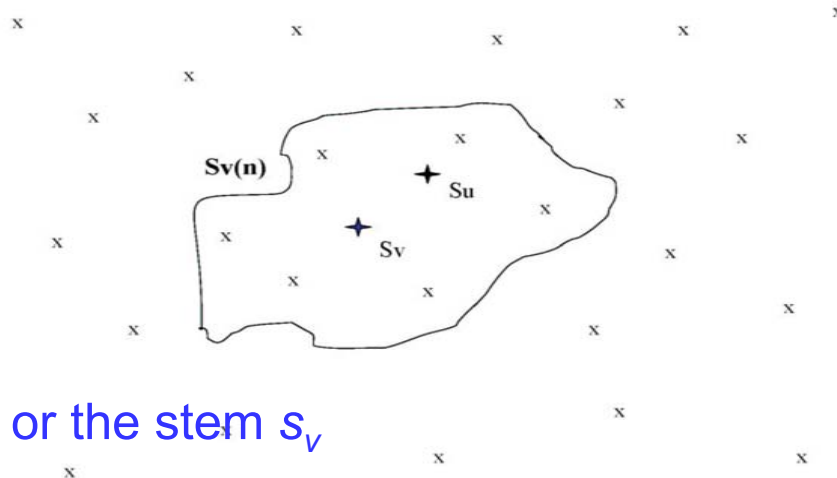
$$\longrightarrow S_{u,v} = \frac{\vec{s}_u \cdot \vec{s}_v}{|\vec{s}_u| \times |\vec{s}_v|}$$

↓

Derive a new scalar association matrix

QE through Local Clustering

- Iterative Search Formulation
 - “**neighbor**”: a stem s_u belongs to a cluster associated to another term s_v is said to be a neighbor of s_v
 - Not necessarily synonyms in the grammatical sense
 - Stems belonging to clusters associated to the query stems (terms) can be used to expand the original query



stems s_u as a neighbor of the stem s_v

QE through Local Clustering

- Iterative Search Formulation
 - Query expansion
 - For each stem $s_v \in q$, select m neighbors stems from the cluster $S_v(m)$ and add them to the query
 - The additional neighbor stems will retrieve new relevant docs
 - The impact of normalized or unnormalized clusters
 - Unnormalized: group stems with high frequency
 - Normalized: group rare stems
 - Union of them provides a better representation of stem (term) correlations

Local Context Analysis

- Local Analysis

Calculation of term correlations at query time

- Based on the set of docs retrieved for the original query
- Based on term (stem) correlation inside docs
- Terms are neighbors of **each query terms** are used to expand the query

- Global Analysis

Pre-calculation of term correlations

- Based on the whole doc collection
- The thesaurus for term relationships are built by considering small contexts (e.g. passages) and phrase structures instead of the context of the whole doc
- Terms closest to **the whole query** are selected for query expansion

Local context analysis combines features from both

Local Context Analysis

Xu and Croft 1996

- Operations of local context analysis
 - **Document concepts**: Noun groups from retrieved docs as the units for QE instead of single keywords
 - **Concepts** selected from the top ranked passages (instead of docs) based on their co-occurrence with the whole set of query terms (no stemming)

QE through Local Context Analysis

- The operations can be further described in three steps
 - Retrieve the top n ranked passages using the original query (a doc is segmented into several passages)
 - For each concept c in the top ranked passages, the similarity $sim(q,c)$ between the whole query q and the concept c is computed using a variant of *tf-idf* ranking
 - The top m ranked concepts are added to the original query q
 - Each concept is assigned a weight $1-0.9 \times i/m$ (i : the position in rank)
 - Original query terms are stressed by a weight of 2

QE through Local Context Analysis

- The similarity between a concept and a query

$$sim(q, c) = \prod_{k_i \in q} \left(\delta + \frac{\log(f(c, k_i) \times idf_c)}{\log n} \right)$$

δ → Set to 0.1 to avoid zero
 idf_c → emphasize the infrequent terms
 n → the no. of top ranked passages considered

$$f(c, k_i) = \sum_{j=1}^n pf_{i,j} \times pf_{c,j}$$

$$idf_c = \max \left(1, \frac{\log_{10} N / np_c}{5} \right)$$

the no. of passages in the collection

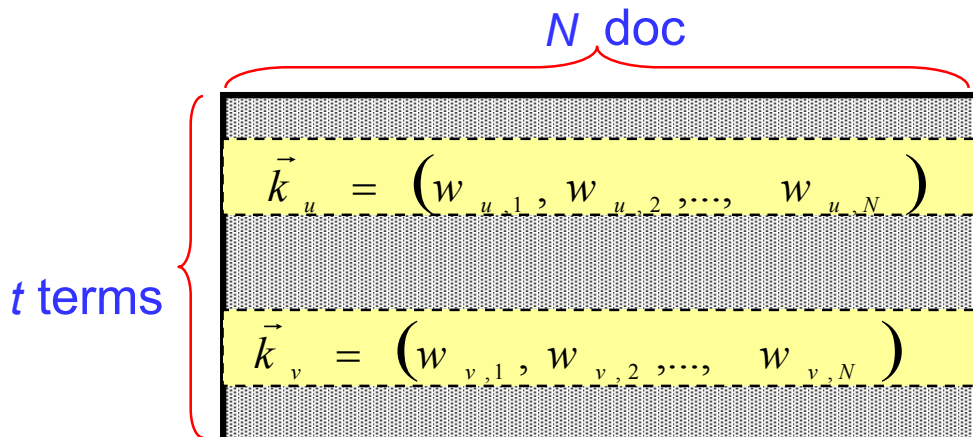
$$idf_i = \max \left(1, \frac{\log_{10} N / np_i}{5} \right)$$

the no. of passages containing concept c

QE based on a Similarity Thesaurus

Qiu and Frei 1993

- Belongs to Global Analysis
- How to construct the similarity thesaurus
 - Term to term relationships rather than term co-occurrences are considered
- How to select term for query expansion
 - Terms for query expansion are selected based on their similarity to the whole query rather the similarities to individual terms



Docs are interpreted as indexing elements here

- Doc frequency within the term vector
- Inverse term frequency

term-doc matrix

QE based on a Similarity Thesaurus

- Definition

- $f_{u,j}$: the frequency of term k_u in document d_j
- t_j : the number of distinct index terms in document d_j
- Inverse term frequency

$$itf_j = \log \frac{t}{t_j} \quad (\text{doc containing more distinct terms is less important})$$

- The weight associated with each entry in the term-doc matrix

$$w_{u,j} = \frac{\left(0.5 + 0.5 \frac{f_{u,j}}{\max_j f_{u,j}} \right) \times itf_j}{\sqrt{\sum_{l=1}^N \left[\left(0.5 + 0.5 \frac{f_{u,l}}{\max_l f_{u,l}} \right) \times itf_l \right]^2}}$$

The importance of the doc d_j to a term k_u

Let term vector have a unit norm

QE based on a Similarity Thesaurus

- The relationship between two terms k_u and k_v

$$c_{u,v} = \vec{k}_u \cdot \vec{k}_v = \sum_{\forall d_j} w_{u,j} \times w_{v,j}$$

is just a cosine measure

ranged from 0 to 1

- The vector representations are normalized
- The computation is computationally expensive
 - There may be several hundred thousands of docs

QE based on a Similarity Thesaurus

Concept-based QE

- Steps for QE based on a similarity thesaurus

1. Represent the query in the term-concept space

$$\vec{q} = \sum_{k_u \in q} w_{u,q} \times \vec{k}_u$$

2. Based on the global thesaurus, compute a similarity between the each term k_v and the whole query q

$$\text{sim}(q, k_v) = \left(\sum_{k_u \in q} w_{u,q} \times \vec{k}_u \right) \cdot \vec{k}_v = \sum_{k_u \in q} w_{u,q} \times c_{u,v}$$

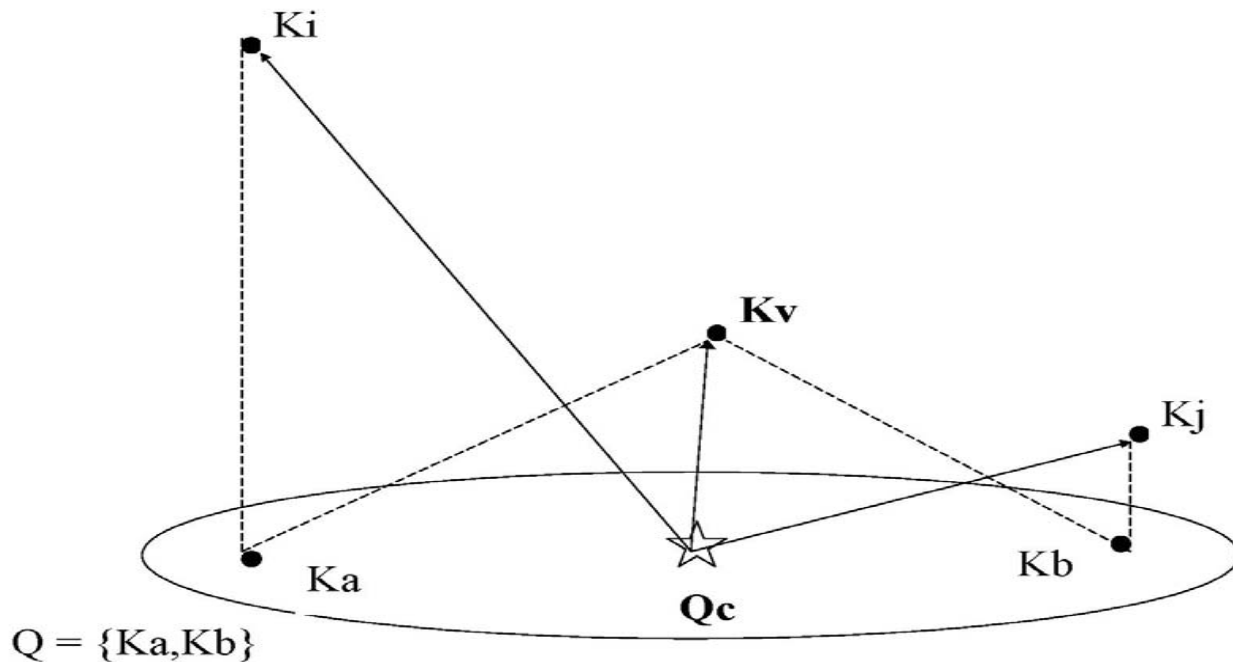
3. Expand the query with the top r ranked terms according to $\text{sim}(q, k_v)$

- The weight assigned to the expansion term

$$w_{v,q'} = \frac{\text{sim}(q, k_v)}{\sum_{k_u \in q} w_{u,q}} = \frac{\sum_{k_u \in q} w_{u,q} \times c_{u,v}}{\sum_{k_u \in q} w_{u,q}}$$

QE based on a Similarity Thesaurus

- The term k_v selected for query expansion might be quite close to the whole query while its distances to individual query terms are larger



QE based on a Similarity Thesaurus

- The similarity between query and doc measured in the term-concept space

- Doc is first represented in the **term-concept space**

$$\vec{d}_j = \sum_{k_v \in d_j} w_{v,j} \times \vec{k}_v$$

- Similarity measure

$$\text{sim}(q, d_j) \propto \sum_{k_v \in d_j} \sum_{k_u \in q} w_{v,j} \times w_{u,q} \times c_{u,v}$$

- Analogous to the formula for query-doc similarity in the generalized vector space model

- Differences

- » Weight computation

- » Only the top r ranked terms are used here

QE based on a Statistical Thesaurus

- Belongs to Global Analysis
- Global thesaurus is composed of classes which group correlated terms in the context of the whole collection
- Such correlated terms can then be used to expand the original user query
 - The terms selected must be low frequency terms
 - With high discrimination values

QE based on a Statistical Thesaurus

- However, it is difficult to cluster low frequency terms
 - To circumvent this problem, we cluster docs into classes instead and use the low frequency terms in these docs to define our thesaurus classes
 - This algorithm must produce small and tight clusters
 - Depend on the cluster algorithm

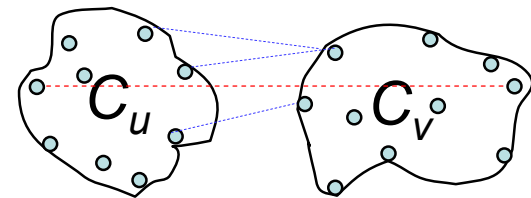
QE based on a Statistical Thesaurus

- Complete Link Algorithm

- Place each doc in a distinct cluster
- Compute the similarity between all pairs of clusters
- Determine the pair of clusters $[C_u, C_v]$ with the highest inter-cluster similarity (using the cosine formula)
- Merge the clusters C_u and C_v
- Verify a stop criterion. If this criterion is not met then go back to step 2
- Return a hierarchy of clusters

- Similarity between two clusters is defined as

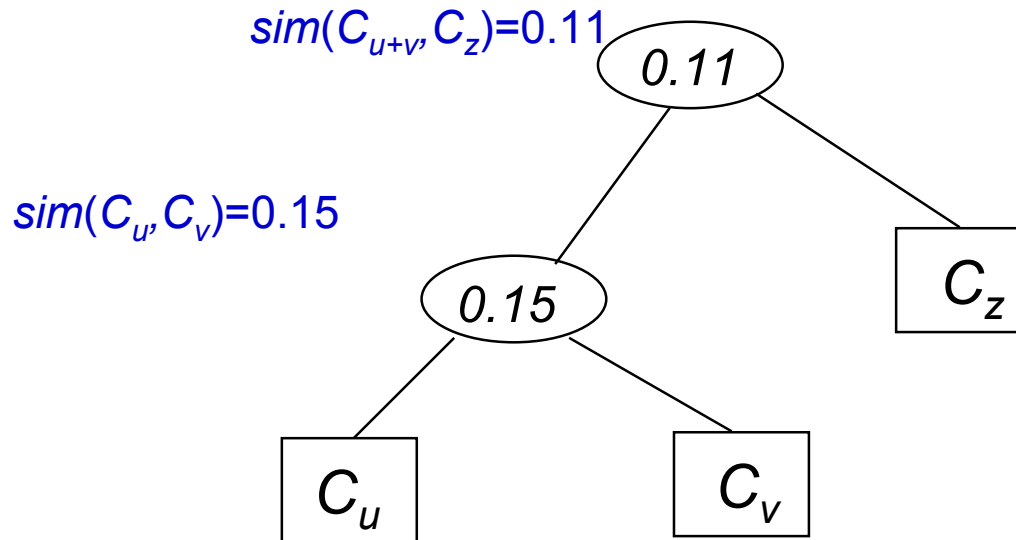
- The minimum of similarities between all pairs of inter-cluster docs



Cosine formula of the vector model is used

QE based on a Statistical Thesaurus

- Example: hierarchy of three clusters



- Higher level clusters represent a looser grouping
 - Similarities decrease as moving up in the hierarchy

QE based on a Statistical Thesaurus

- Given the doc cluster hierarchy for the whole collection, the *terms* that compose each class of the global thesaurus are selected as follows
 - Three parameters obtained from the user
 - *TC*: Threshold class
 - *NDC*: Number of docs in class
 - *MIDF*: Minimum inverse doc frequency

QE based on a Statistical Thesaurus

- Use the parameter TC as threshold value for determining the doc clusters that will be used to generate thesaurus classes
 - It has to be surpassed by $\text{sim}(C_u, C_v)$ if the docs in the clusters C_u and C_v are to be selected as sources of terms for a thesaurus class
- Use the parameter NDC as a limit on the size of clusters (number of docs) to be considered
 - A low value of NDC might restrict the selection to the smaller clusters

QE based on a Statistical Thesaurus

- Consider the set of docs in each doc cluster pre-selected above
 - Only **the lower frequency terms** are used as sources of terms for the thesaurus classes
 - The parameter *MIDF* defines the minimum value of **inverse doc frequency** for any term which is selected to participate in a thesaurus class
- Given the thesaurus classes have been built, they can be to query expansion

QE based on a Statistical Thesaurus

- Example

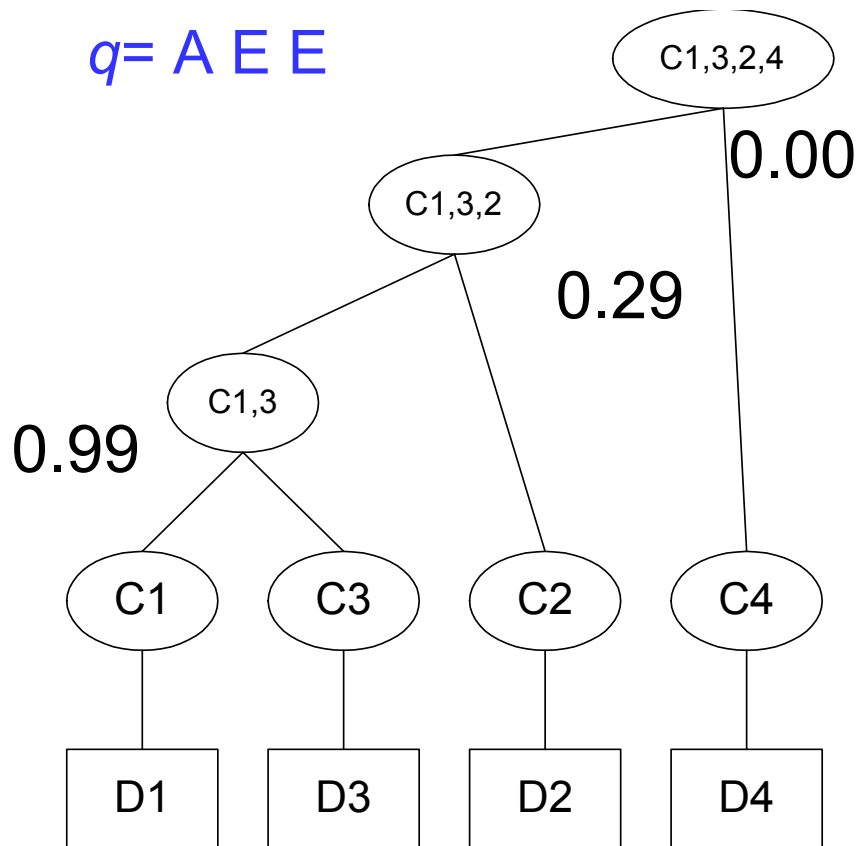
Doc1 = D, D, A, B, C, A, B, C
 Doc2 = E, C, E, A, A, D
 Doc3 = D, C, B, B, D, A, B, C, A
 Doc4 = A

sim(1,3) = 0.99
 sim(1,2) = 0.40
 sim(2,3) = 0.29
 sim(4,1) = 0.00
 sim(4,2) = 0.00
 sim(4,3) = 0.00

cosine formula
 with *tf-idf* weighting

idf A = 0.0
 idf B = 0.3
 idf C = 0.12
 idf D = 0.12
 idf E = 0.60

$q = A E E$



- $TC = 0.90$ $NDC = 2.00$ $MIDF = 0.2$

$q' = A B E E$

QE based on a Statistical Thesaurus

- Problems
 - Initialization of parameters TC, NDC and MIDF
 - TC depends on the collection
 - Inspection of the cluster hierarchy is almost always necessary for assisting with the setting of TC
 - A high value of TC might yield classes with too few terms
 - While a low value of TC yields too few classes

Trends and Research Issues

- Visual display
 - Graphical interfaces (2D or 3D) for relevance feedback
- Utilization of local and global analysis techniques to the Web environments
 - Alleviate the computational burden imposed on the search engine