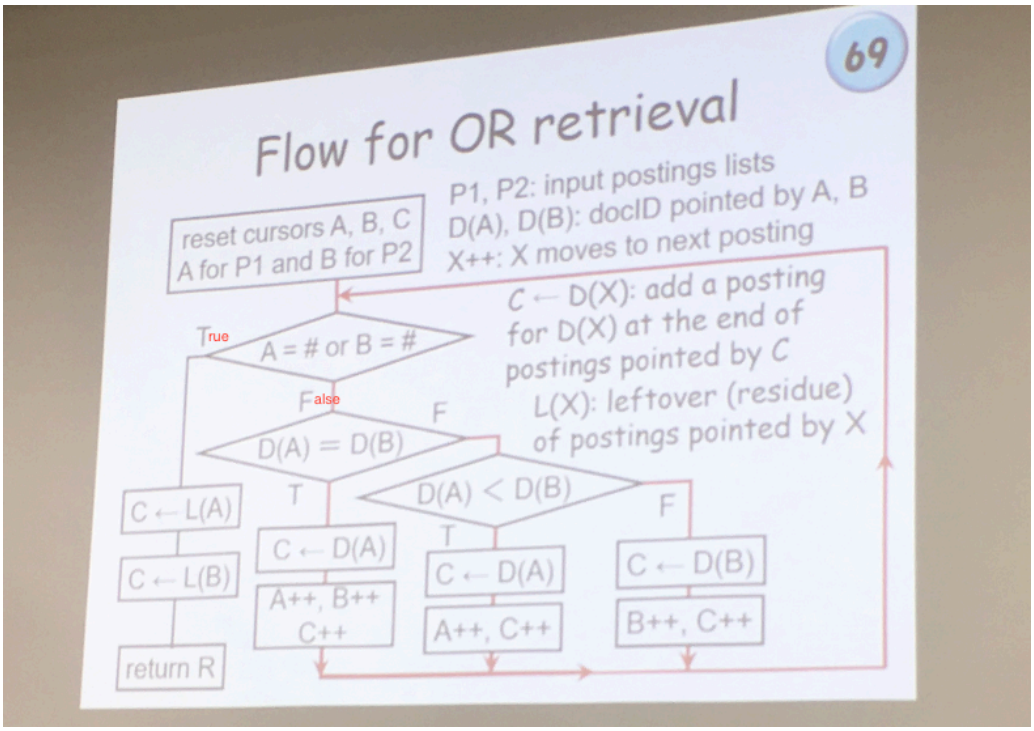
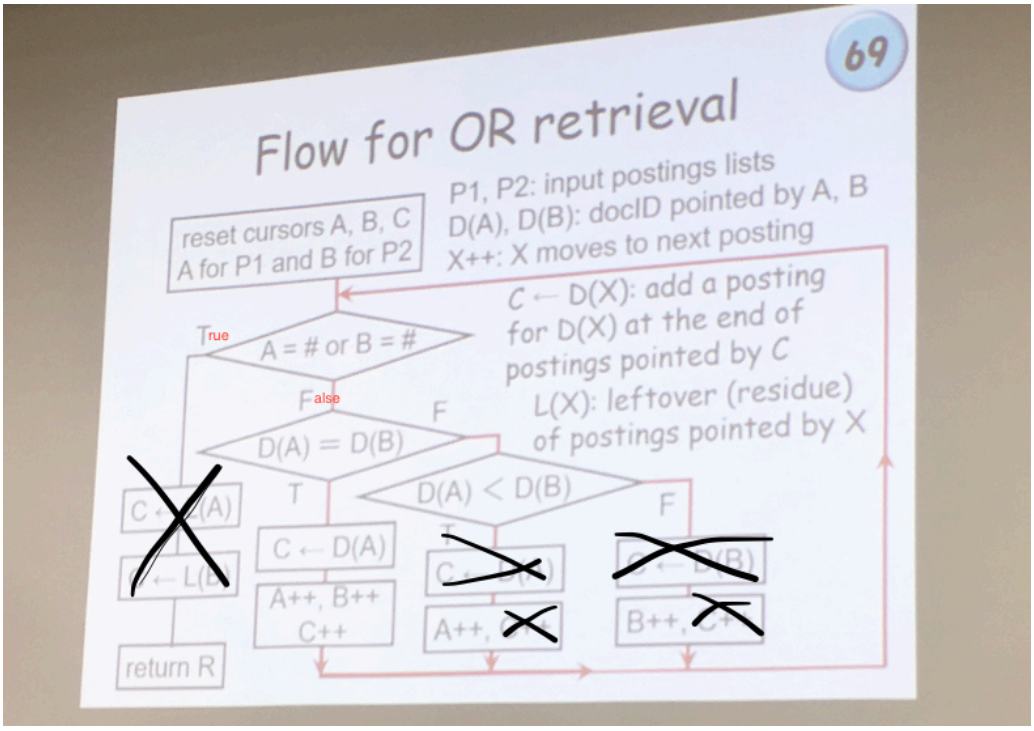
Day 6 & 7

Flow of OR retrieval



Exercise

- The flow in the previous slide (OR retrieval) was used to obtain a document set, R, in response to query S = T2 OR T4
    - T2 -> D1, D2, D5, D7, D9 -> #
    - T4 -> D1, D3, D5, D6, D7, D8 -> #
- Q1: identify how many times the condition D(A) = D(B) was evaluated in the flow?
  Ans: 7 times
- Q2: modify the OR flow to realize AND
  Ans:



Optimizing Boolean queries

- If query contains more than 2 terms, the order of operations is important
- Use operations involving AND operators and shorter postings lists as soon as possible to reduce the candidate documents
- Keeping unpromising candidates decreases efficiency

Boolean operation

73

Boolean operation

+ OR	* AND	$\overline{\ }$ NOT

- Fill in blanks
- $0 + A = [A]$ $0 * A = [0]$ $1 + A = [1]$ $1 * A = [A]$
- $A + A = [A]$ $A * A = [A]$ $A + \overline{A} = [1]$ $A * \overline{A} = [0]$
- $A * (B + C) = ([A] * B) + (A * [C])$
- $\overline{A + B} = [\overline{A} * \overline{B}]$ $\overline{A * B} = [\overline{A} + \overline{B}]$
- Proof the following (may use above laws)

  [13] $A * (A + B) = A$

  [14] $(A + B) * (A + C) = A + B * C$

74

Answer

[1] A [2] 0 [3] 1 [4] A [5] A [6] A [7] 1 [8] 0

[9] A [10] C [11] $\overline{A} * \overline{B}$ [12] $\overline{A} + \overline{B}$

[13] $A * (A + B)$

$= A * A + A * B = A * 1 + A * B$

$= A * (1 + B) = A * 1 = A$

[14] $(A + B) * (A + C)$

$= A * A + A * C + A * B + B * C$

$= A + A * C + A * B + B * C$

$= A * (1 + C + B) + B * C = A + B * C$

The number of comparisons for postings lists can substantially be reduced by preprocessing

Term weighting: background

- In the best match model, degree of relevance for each doc D wrt Q, R(D,Q) is calculated
- R(D,Q) is accumulation of the weight of each term in D wrt Q
- Here, we have a fundamental question
  - Which terms represent topic of document?
  - Equivalent to how we modeled a document?

  -> a set of words w/o certain properties
- The topic of document is reprensented by a set of words (aka “Bag of Words” or “BOW”)

Term weighting: issues

- Wish to give a different weight to each term, depending on the importance for retrieval
- **TF*IDF** was proposed around the 1960s and has been used and developed in the IR
  - TF (term frequency)
  - IDF (inverse document frequency)
- Whereas the rationale has not been clearly explained, TF*IDF (especially) IDF, has been popular methods for other communities where target object is represented a vector of features

TF (term frequency)

- In principle, the more frequently appear, the more being important
  - TF(D_i, Tk): Frequency of Tk in D_i
- In practice, a modification is recommended
  - (Ex1) long documents are more likely a large terms -> normalized by document length
  - (Ex2) the value in linearly proportion to the frequency is too sensitive to a small change of the frequency and thus is unstable -> take logarithm of TF to make the increase being saturated for a large frequency

IDF (inverse document frequency)

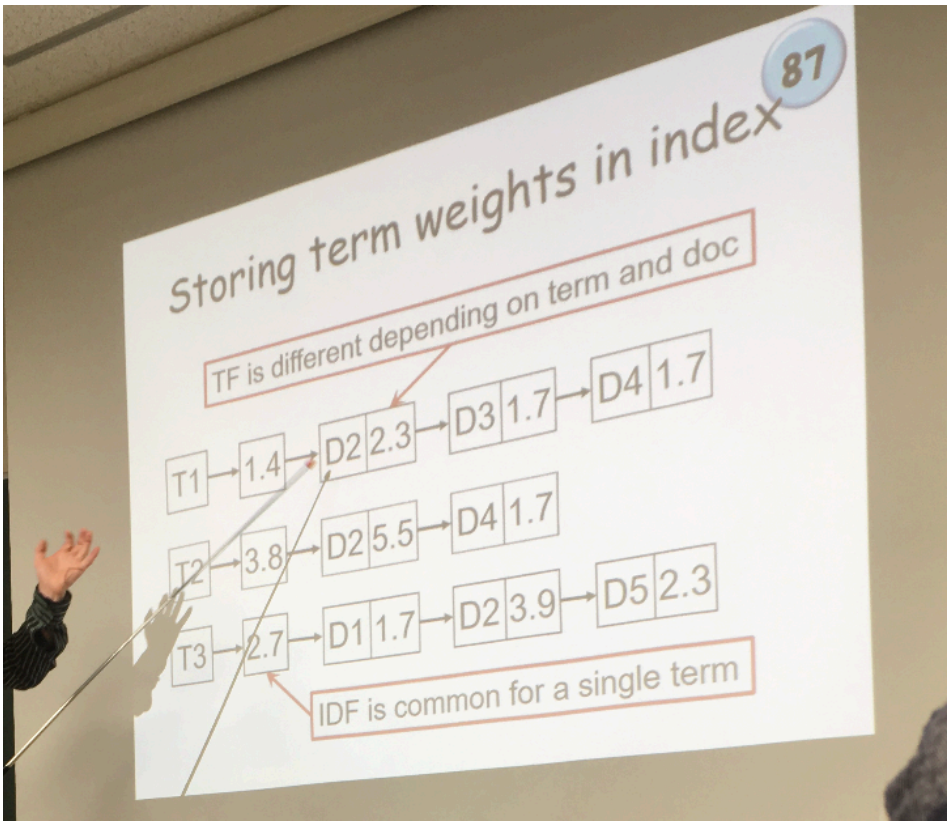
- in principle, terms that appear in a lot of documents is not associated with specificity
  - E.g. “the” is everywhere
- IDF for term T
  - N: # of documents in collection
  - DF(T): # of documents that contain T
  - **IDF(T) = log (N/DF(T) +1)**
  - +1 is to prevent IDF from being 0
- IDF is proper to each term, unlike TF

Why is IDF such a formula?

- Plausible explanation would be:
  - DF(T)/N is a probability that a randomly selected document D from the collection contains term T

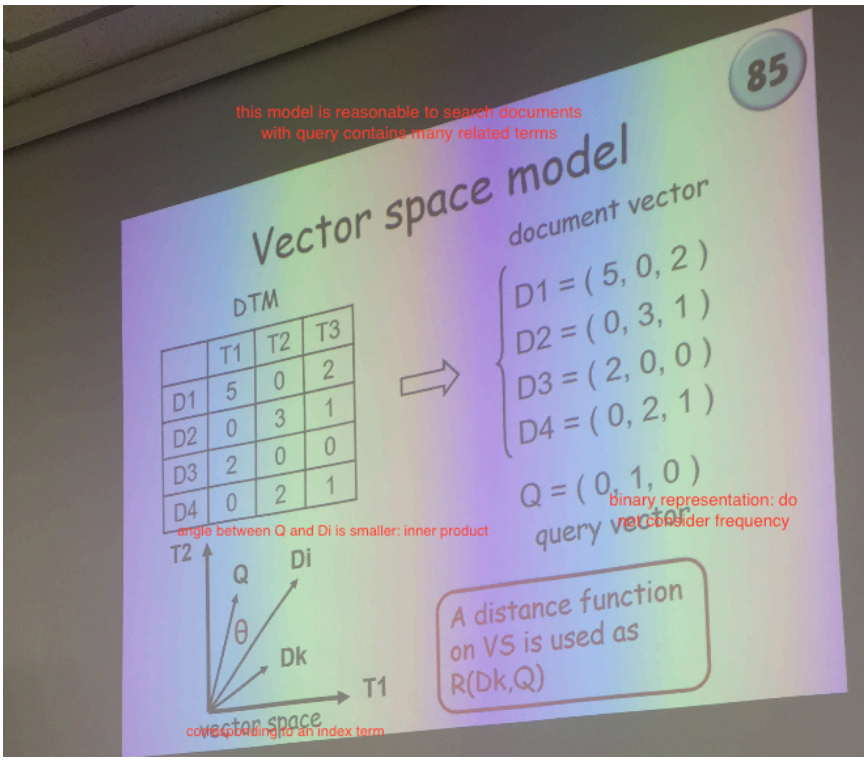
- The information content of the message “D contains T” is
  -log (DF(T)/N) = log (N/DF(T))
- IDF is usually effective whereas TF is rather harmful depending on the formula

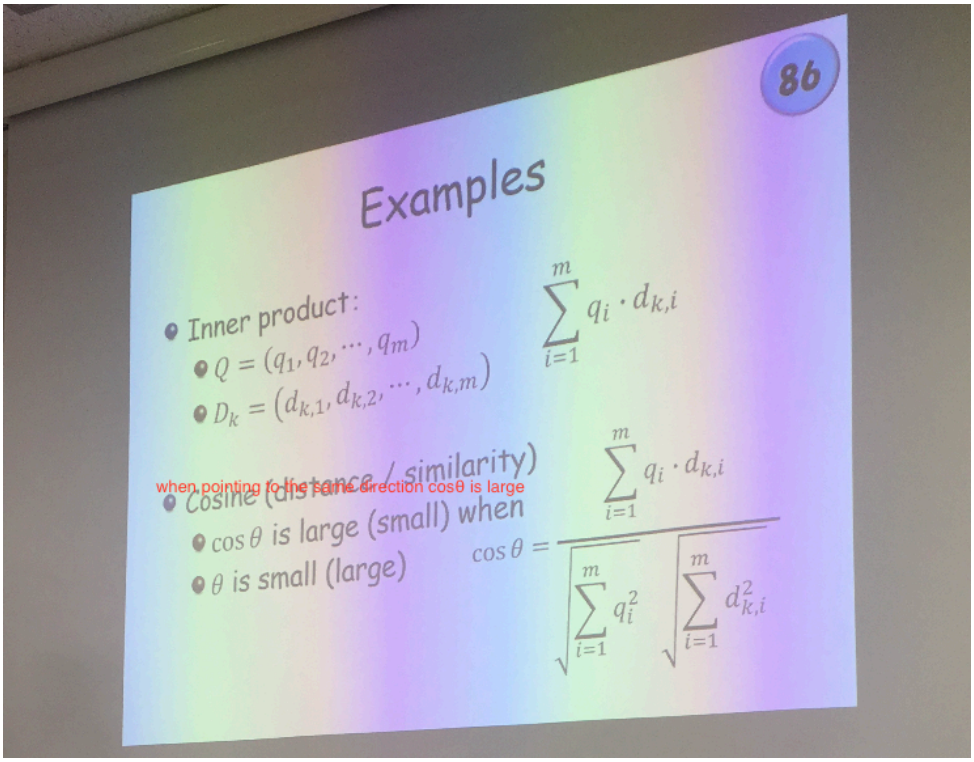
Storing term weights in index
T1 -> 1.4 -> D1: 2.3 (TF is depending on term and doc)-> D3: 1.7 -> D4: 1.7
T2 -> 3.8 -> D2: 5.5 -> D4: 1.7
T3 -> 2.7 (IDF is common for a single term) -> D1: 1.7 -> D2: 3.9 -> D5 :2.3

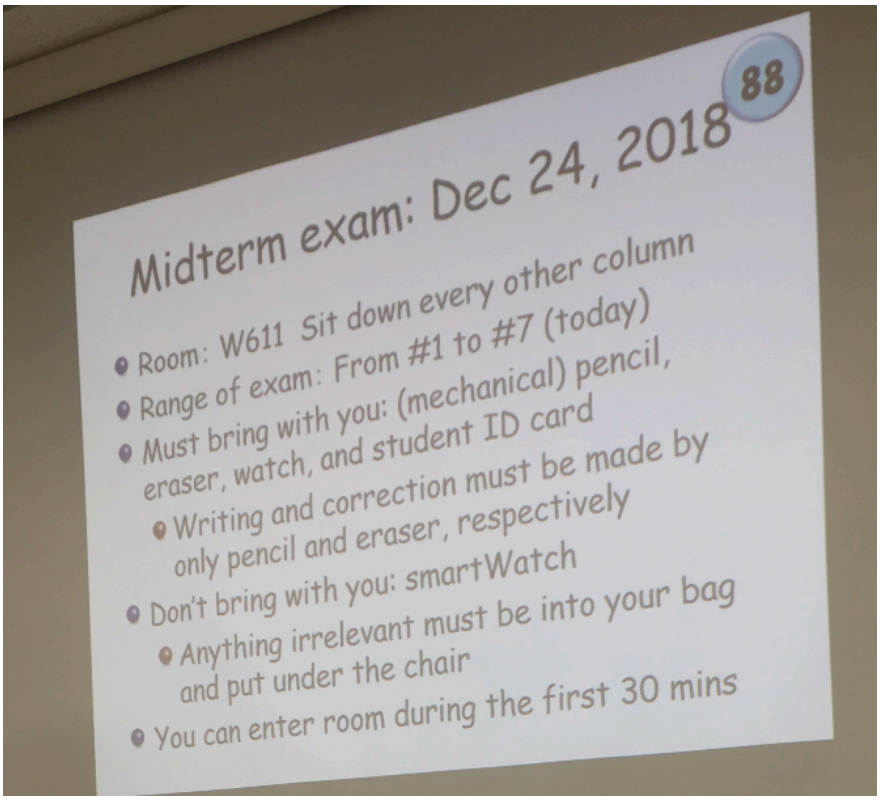


Discussion
- Pros and Cons for the Boolean and Best match IR match

Vector space model


85
this model is reasonable to search documents
with query contains many related terms
Vector space model
document vector
DTM
T1 T2 T3
D1 5 0 2
D2 0 3 1
D3 2 0 0
D4 0 2 1
D1 = (5, 0, 2)
D2 = (0, 3, 1)
D3 = (2, 0, 0)
D4 = (0, 2, 1)
Q = (0, 1, 0)
query vector
binary representation: do not consider frequency
angle between Q and Di is smaller: inner product
T2
Q
Di
θ
Dk
T1
vector space
corresponding to an index term
A distance function
on VS is used as
R(Dk,Q)


86
Examples
Inner product:
$Q = (q_1, q_2, \cdots, q_m)$
$D_k = (d_{k,1}, d_{k,2}, \cdots, d_{k,m})$
$\sum_{i=1}^{m} q_i \cdot d_{k,i}$
Cosine (distance / similarity)
when pointing to the same direction cosθ is large
$\cos\theta$ is large (small) when
θ is small (large)
$\cos\theta = \frac{\sum_{i=1}^{m} q_i \cdot d_{k,i}}{\sqrt{\sum_{i=1}^{m} q_i^2}\sqrt{\sum_{i=1}^{m} d_{k,i}^2}}$



https://nlp.stanford.edu/IR-book/essir2011/