Pattern Analysis and Machine Intelligence

Lecture Notes on Clustering (IV)
2012-2013

Davide Eynard

davide.eynard@usi.ch

Department of Electronics and Information

Politecnico di Milano

Course Schedule [*Tentative*]

Date	Topic
06/05/2012	Clustering I: Introduction, K-means
07/05/2012	Clustering II: K-M alternatives, Hierarchical, SOM
13/05/2012	Clustering III: Mixture of Gaussians, DBSCAN, J-P
14/05/2012	Clustering IV: Spectral Clustering + Text
20/05/2012	Clustering V: Evaluation Measures

Search Engines

How do search engines work?

- document retrieval and indexing
- query language that allows to search for Web pages that contain (or not) given words and phrases
- SE have their roots in *information retrieval* systems, which prepare a keyword index for the given corpus and respond to keyword queries with a ranked list of documents
- some queries:
  - docs containing the word “Java”
  - docs containing “Java” but not “coffee”
  - docs containing the phrase “Java Beans” and the word “API”
  - docs where “Java” and “island” occur in the same sentence

Search Engines - A naive approach

tid	did	pos
my	1	1
care	1	2
is	1	3
⋮		
new	2	8
care	2	9
won	2	10

1. `select did from POSTING where tid = 'java'`
2. ```
   (select did from POSTING where tid = 'java') except (select did
   from POSTING where tid = 'coffee')
   ```
3. ```
   with
     D_JAVA (did, pos) as (select did, pos from POSTING where tid = 'java'),
     D_BEANS(did, pos) as (select did, pos from POSTING where tid = 'beans'),
     D_JAVABEANS(did) as
       (select D_JAVA.did from D_JAVA, D_BEANS
         where D_JAVA.did = D_BEANS.did
         and D_JAVA.pos + 1 = D_BEANS.pos),
     D_API(did) as (select did from POSTING where tid = 'api'),
   (select did from D_JAVABEANS) union (select did from D_API)
   ```

Search Engines - Not naive at all

Can we always think about text search in terms of sets?

- The index.of approach
- The epanaleptical approach

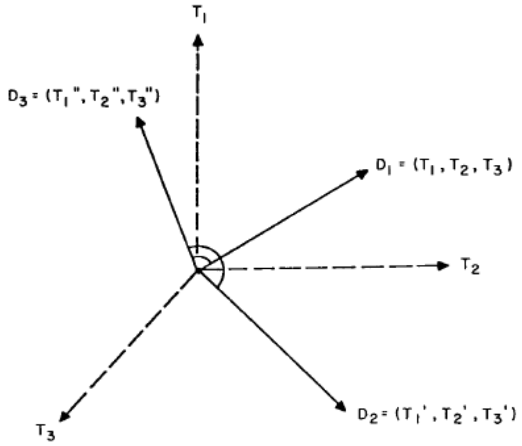
Main problems:

- Index creation, compression and update
- Stopwords and stemming
- Relevance ranking
  - recall vs precision
- Relevance feedback

Vector Space Model

VSM

In the Vector Space Model, documents are represented as vectors in a multidimensional Euclidean space.



VSM

The coordinate of document d in the direction corresponding to the term t is determined by two quantities:

- *Term Frequency* $TF(d, t)$: this is simply $n(d, t)$, the number of times term t occurs in document d, scaled to normalize document length
- *Inverse Document Frequency* $IDF(t)$: this is a weight factor used to scale down the coordinates of terms which occur in many documents

$$IDF(t) = log \frac{|D|}{1+|D_t|}$$

VSM

TF and IDF are combined into the complete vector-space model in the obvious way: the coordinate of document d in axis t is given by

$$d_t = TF(d,t)IDF(T)$$

Then, the *cosine distance* between the two vectors

$$v_{d1} = [w_{1,d1}, w_{2,d1}, \ldots, w_{N,d1}]^T$$
$$v_{d2} = [w_{1,d2}, w_{2,d2}, \ldots, w_{N,d2}]^T$$

is calculated as

$$cos\theta = \frac{v_1 \cdot v_2}{||v_1||||v_2||}$$

Note: one of the two vectors might be the query itself!

Limits of VSM

- Long documents are poorly represented, because they have poor similarity values (a small scalar product and a large dimensionality)
- Search keywords must precisely match document terms; word substrings might result in a "false positive" match
- Semantic sensitivity: documents with similar context but different term vocabulary won't be associated, resulting in a "false negative" match

Bibliography

- Salton, G., Wong, A., and Yang, C. S. (1975).
  A Vector Space Model for Automatic Indexing.

- The end