Outline

Goal: ‘Find similar / interesting things’

- Intro to DB
- Indexing - similarity search
- Data Mining

Indexing - Detailed outline

- primary key indexing
- secondary key / multi-key indexing
- spatial access methods
- fractals
- text
  - multimedia
  - ...

Text - Detailed outline

- text
  - problem
  - full text scanning
  - inversion
  - signature files
  - clustering
  - information filtering and LSI

Vector Space Model and Clustering

- keyword queries (vs Boolean)
- each document: -> vector (HOW?)
- each query: -> vector
- search for ‘similar’ vectors

Vector Space Model and Clustering

- main idea:
  
  document
  "indexing"

  aaron  data  zoo

  ...data...

  V (= vocabulary size)
Vector Space Model and Clustering

Then, group nearby vectors together
- Q1: cluster search?
- Q2: cluster generation?

Two significant contributions
- ranked output
- relevance feedback

• cluster search: visit the (k) closest superclusters; continue recursively

• How? A: by adding the ‘good’ vectors and subtracting the ‘bad’ ones

• ranked output: easy!

• relevance feedback (brilliant idea) [Roccio’73]
Outline - detailed

• main idea
• cluster search
• cluster generation
• evaluation

Cluster generation

• Problem:
  – given N points in V dimensions,
  – group them

Cluster generation

We need
• Q1: document-to-document similarity
• Q2: document-to-cluster similarity

Cluster generation

Q1: document-to-document similarity
(recall: ‘bag of words’ representation)
• D1: \{‘data’, ‘retrieval’, ‘system’\}
• D2: \{‘lung’, ‘pulmonary’, ‘system’\}
• distance/similarity functions?

Cluster generation

A1: # of words in common
A2: ........ normalized by the vocabulary sizes
A3: .... etc

About the same performance - prevailing one:
cosine similarity
Cluster generation

cosine similarity:
\[
\text{similarity}(D1, D2) = \cos(\theta) = \frac{\sum(v_{1i} \cdot v_{2j})}{\|v_1\| \cdot \|v_2\|}
\]

Cluster generation

cosine similarity - observations:
• related to the Euclidean distance
• weights \( v_{ij} \) according to tf/idf

Cluster generation

\( \theta \)

Cluster generation

tf (‘term frequency’)
high, if the term appears very often in this document.

idf (‘inverse document frequency’)
penalizes ‘common’ words, that appear in almost every document

Cluster generation

We need
• Q1: document-to-document similarity
• Q2: document-to-cluster similarity

Cluster generation

• A1: min distance (‘single-link’)
• A2: max distance (‘all-link’)
• A3: avg distance
• A4: distance to centroid

Cluster generation

• A1: min distance (‘single-link’)
  – leads to elongated clusters
• A2: max distance (‘all-link’)
  – many, small, tight clusters
• A3: avg distance
  – in between the above
• A4: distance to centroid
  – fast to compute
Cluster generation

We have
- document-to-document similarity
- document-to-cluster similarity

Q: How to group documents into ‘natural’ clusters

Cluster generation

A: *many-many* algorithms - in two groups
[VanRijsbergen]:
- theoretically sound (O(N^2))
  – independent of the insertion order
- iterative (O(N), O(N log(N))

Cluster generation - ‘sound’ methods

- Approach#1: dendrograms - create a hierarchy (bottom up or top-down) - choose a cut-off (how?) and cut

Cluster generation - ‘sound’ methods

- Approach#2: min. some statistical criterion (eg., sum of squares from cluster centers)
  – like ‘k-means’
  – but how to decide ‘k’?

Cluster generation - ‘sound’ methods

- Approach#3: Graph theoretic [Zahn]:
  – build MST;
  – delete edges longer than 3* std of the local average

Result:
- why ‘3’?
- variations
- Complexity?
Cluster generation - ‘iterative’ methods

general outline:
• Choose ‘seeds’ (how?)
• assign each vector to its closest seed (possibly adjusting cluster centroid)
• possibly, re-assign some vectors to improve clusters
Fast and practical, but ‘unpredictable’

Cluster generation

one way to estimate # of clusters $k$: the ‘cover coefficient’ [Can+] ~ SVD

Outline - detailed

• main idea
• cluster search
• cluster generation
• evaluation

Evaluation

• Q: how to measure ‘goodness’ of one distance function vs another?
• A: ground truth (by humans) and
  – ‘precision’ and ‘recall’

Evaluation

• precision = (retrieved & relevant) / retrieved
  – 100% precision -> no false alarms
• recall = (retrieved & relevant)/ relevant
  – 100% recall -> no false dismissals
Evaluation

- compressing such a curve into a single number:
  - 11-point average precision
  - etc

References


References