Document Clustering….

Cluster Hypothesis:
By clustering, documents relevant to the same topics tend to be grouped together.

What can be Clustered?

• **Collection (Pre-retrieval)**
  – Reducing the search space to smaller subset -- not generally used due to expense in generating clusters.
  – Improving UI with displaying groups of topics -- have to label the clusters
    • Scatter-gather – the user selected clusters are merged and re-clustered

• **Result Set (Post-retrieval)**
  – Improving the ranking (re-ranking)
  – Utilizing in query refinement -- Relevance feedback
  – Improving UI to display clustered search results

• **Query**
  – Understanding the intent of a user query
  – Suggesting query to users (query suggestion/recommendation)

Document/Web Clustering

• **Input:** set of documents, [k clusters]
• **Output:** document assignments to clusters
• **Features**
  – Text – from document/snippet (words: single; phrase)
  – Link and anchor text
  – URL
  – Tag (social bookmarking websites allow users to tag documents)
  – ..........
• **Term weight** (tf, tf-idf, ...)
• **Distance measure:** Euclidian, Cosine, ...
• **Evaluation**
  – Manual -- difficult
  – Web directories
Result Set Clustering

• Clusters are generated online (during query processing)

Retrieved Result

• To improve efficiency, clusters may be generated from document snippets.
• Clusters for popular queries may be cached
• Clusters may be labeled into categories, providing the advantage of both query & category information for the search
• Clustering result set as a whole or per site
• Stemming can help due to limited result set
Cluster Labeling

• The goal is to create “meaningful” labels
• Approaches:
  – Manually (not a good idea)
  – Using already tagged documents (not always available)
  – Using external knowledge such as Wikipedia, etc.
  – Using each cluster’s data to determine label
    • Cluster’s Centroid terms/phrases -- frequency & importance
    • Title of document centroid or closest document to centroid can be used
  – Using also other clusters’ data to determine label
    • Cluster’s Hierarchical information (sibling/parent) of terms/phrases

Result Clustering Systems

• Northern Light (end of 90’s) -- used pre-defined categories
• Grouper (STC)
• Carrot
• CREDO
• WhatsOnWeb
• Vivisimo’s Clusty (acquired by Yippy): generated clusters and labels dynamically
• ………..etc.
Query Clustering Approach to Query Suggestion

- Exploit information on past users' queries
- Propose to a user a list of queries related to the one (or the ones, considering past queries in the same session/log) submitted
- Various approaches to consider both query terms and documents

Baeza-Yates et al. use a clustering approach
- A two tier approach
  - An offline component clusters past queries using query text along with the text of clicked URLs.
  - An online component that recommends queries based on an incoming query and using clusters generated in the offline mode
Query Clustering Approach to Query Suggestion

- **Offline component:**
  - Clustering algorithm operates over queries enriched by a selection of terms extracted from the documents pointed by the user clicked URLs.
  - Clusters computed by using an implementation of \( k \)-means
    - different values of \( k \)
    - \( SSE \) becomes even smaller by increasing \( k \)
  - Similarity between queries computed according to a vector-space approach
    - Vectors \( \vec{q} \) of \( n \) dimensions, one for each term

---

Query Clustering Approach to Query Suggestion

*Baeza-Yates et al. use a clustering approach (cont’d)*

- **Online component:**
  (I) given an input query the most representative (i.e. similar) cluster is found
    - each cluster has a natural representative, i.e. its centroid
  (II) ranking of the queries of the cluster, according to:
    - attractiveness of query answer, i.e. the fraction of the documents returned by the query that captured the attention of users (clicked documents)
    - similarity wrt the input query (the same distance used for clustering)
    - popularity of query, i.e. the frequency of the occurrences of queries
Clustering

- Automatically group related data into clusters.
- An unsupervised approach -- no training data is needed.

- A data object may belong to
  - only one cluster (Hard clustering)
  - overlapped clusters (Soft Clustering)

- Set of clusters may
  - relate to each other (Hierarchical clustering)
  - have no explicit structure between clusters (Flat clustering)

Considerations…

- Distance/similarity measures
  - Various; example: Cosine

- Number of clusters
  - Cardinality of a clustering (# of clusters)

- Objective functions
  - Evaluates the quality (structural properties) of clusters; often defined using distance/similarity measures

  - External quality measures such as: using annotated document set; existing directories; manual evaluation of documents
Distance/Similarity Measures

Euclidean Distance

\[ \text{dist}(d_i, d_j) = \sqrt{(|d_{t_i} - d_{t_j}|^2 + |d_{t_2} - d_{t_2}|^2 + \ldots + |d_{t_p} - d_{j_p}|^2)} \]

Cosine

\[ \text{Sim}(d_i, d_j) = \frac{\sum_{k=1}^{t} d_{ik} \cdot d_{jk}}{\sqrt{\sum_{k=1}^{t} (d_{ik})^2 \cdot \sum_{k=1}^{t} (d_{jk})^2}} \]

Structural Properties of Clusters

• Good clusters have:
  – high intra-class similarity
  – low inter-class similarity

• Calculate the sum of squared error (Commonly done in K-means)
  – Goal is to minimize SSE (intra-cluster variance):

\[ SSE = \sum_{i=1}^{k} \sum_{p \in c_i} |p - m_i|^2 \]
Quality Measures

- Macro average precision -- measure the precision of each cluster (ratio of members that belong to that class label), and average over all clusters.

- Micro average precision -- precision over all elements in all clusters

- Accuracy: \( \frac{tp + tn}{tp + tn + fp + fn} \)

- F1 measure

Clustering Algorithms

- Hierarchical – A set of nested clusters are generated, represented as *dendrogram*.
  - Agglomerative (bottom-up) - a more common approach
  - Divisive (top-down)

- Partitioning (Flat Clustering) – no link (no overlapping) among the generated clusters
The **K-Means Clustering Method**

- A *Flat* clustering algorithm
- A *Hard* clustering
- A Partitioning (Iterative) Clustering
- Start with $k$ random cluster centroids and iteratively adjust (redistribute) until some termination condition is set.
- Number of cluster $k$ is an input in the algorithm. The outcome is $k$ clusters.

---

The **K-Means Clustering Method**

Pick $k$ documents as your initial $k$ clusters
Partition documents into $k$ closests cluster centroids (centroid: mean of document vectors; consider most significant terms to reduce the distance computations)
Re-calculate the centroid of each cluster
Re-distribute documents to clusters till a termination condition is met

- Relatively efficient: $O(tkn)$,
  - $n$: number of documents
  - $k$: number of clusters
  - $t$: number of iterations Normally, $k, t \ll n$
**The K-Means Clustering Method**

- Example

![Diagram](https://example.com)

© Jiawei Han and Micheline Kamber

---

**Limiting Random Initialization in K-Means**

Various methods, such as:

- *Various K* may be good candidates

- Take *sample* number of documents and perform *hierarchical clustering*, take them as *initial centroids*

- Select *more than k initial centroids* (choose the ones that are further away from each other)

- Perform clustering and *merge closer clusters*

- Try *various starting seeds* and pick the better choices
The *K-Means* Clustering Method

Re-calculating Centroid:
- Updating centroids after each iteration (all documents are assigned to clusters)
- Updating after each document is assigned.
  - More calculations
  - More order dependency

Termination Condition:
- A fixed number of iterations
- Reduction in re-distribution (no changes to centroids)
- Reduction in SSE
Effect of Outliers

- Outliers are documents that are far from other documents.

- Outlier documents create a singleton (cluster with only one member)

- Outliers should be removed and not picked as the initialization seed (centroid)

Evaluate Quality in K-Means

- Calculate the sum of squared error (Commonly done in K-means)
  - Goals to minimize SSE (intra-cluster variance):

\[ SSE = \sum_{i=1}^{k} \sum_{p \in c_i} |p - m_i|^2 \]
Hierarchical Agglomerative Clustering (HAC)

- Treats documents as singleton clusters, then merge pairs of clusters till reaching one big cluster of all documents.
- Any $k$ number of clusters may be picked at any level of the tree (using thresholds, e.g. SSE)
- Each element belongs to one cluster or to the superset cluster; but does not belong to more than one cluster.

Example

- Singletons A, D, E, and B are clustered.
Hierarchical Agglomerative

- Create NxN doc-doc similarity matrix
- Each document starts as a cluster of size one
- Do Until there is only one cluster
  - Combine the best two clusters based on cluster similarities using one of these criteria: single linkage, complete linkage, average linkage, centroid, Ward’s method.
  - Update the doc-doc matrix

- Note: Similarity is defined as vector space similarity (eg. Cosine) or Euclidian distance

Merging Criteria

- Various functions in computing the cluster similarity result in clusters with different characteristics.
- The goal is to minimize any of the following functions:
  - Single Link/MIN (minimum distance between documents of two clusters)
  - Complete Linkage/MAX (maximum distance between documents of two clusters)
  - Average Linkage (average of pair-wise distances)
  - Centroid (centroid distances)
  - Ward’s Method (intra-cluster variance)
**HAC’s Cluster Similarities**

- **Single Link**
- **Complete Link**
- **Average Link**
- **Centroid**

**Example** *(Hierarchical Agglomerative)*

- Consider A, B, C, D, E as objects with the following similarities:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>-</td>
<td>2</td>
<td>7</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>B</td>
<td>2</td>
<td>-</td>
<td>9</td>
<td>11</td>
<td>14</td>
</tr>
<tr>
<td>C</td>
<td>7</td>
<td>9</td>
<td>-</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>D</td>
<td>9</td>
<td>11</td>
<td>4</td>
<td>-</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>4</td>
<td>14</td>
<td>8</td>
<td>2</td>
<td>-</td>
</tr>
</tbody>
</table>

  Highest pair is: E-B = 14
Example (Cont’d)

- So let’s cluster E and B. We now have the structure:

```
  A  C  D  E  B
  \   /   \   /  \
   BE
```

Example (Cont’d)

- Now we update the matrix:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>BE</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>-</td>
<td>2</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>BE</td>
<td>2</td>
<td>-</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>7</td>
<td>8</td>
<td>-</td>
<td>4</td>
</tr>
<tr>
<td>D</td>
<td>9</td>
<td>2</td>
<td>4</td>
<td>-</td>
</tr>
</tbody>
</table>
So let's cluster A and D. We now have the structure:

Example (Cont'd)

Now we update the matrix:

<table>
<thead>
<tr>
<th></th>
<th>AD</th>
<th>BE</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>AD</td>
<td>-</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>BE</td>
<td>2</td>
<td>-</td>
<td>8</td>
</tr>
<tr>
<td>C</td>
<td>4</td>
<td>8</td>
<td>-</td>
</tr>
</tbody>
</table>
Example (Cont’d)

• So let's cluster BE and C. At this point, there are only two nodes that have not been clustered, AD and BCE. We now cluster them.

Example (Cont’d)

• Now we update the similarity matrix.

<table>
<thead>
<tr>
<th></th>
<th>AD</th>
<th>BEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>AD</td>
<td>-2</td>
<td>2</td>
</tr>
<tr>
<td>BEC</td>
<td>2</td>
<td>-</td>
</tr>
</tbody>
</table>

At this point there is only one cluster.
Example (Cont’d)

- Now we have clustered everything.

How to do Query Processing

- Calculate the centroid of each cluster.
- Calculate the SC between the query vector and each cluster centroid.
- Pick the cluster with higher SC.
- Continue the process toward the leafs of the subtree of the cluster with higher SC.
Analysis

• Hierarchical clustering requires:
  – $O(n^2)$ to compute the doc-doc similarity matrix
  – One node is added during each round of clustering, thus $n$ steps
  – For each clustering step we must re-compute the DOC-DOC matrix. That is finding the “closest” is $O(n^2)$ plus re-computing the similarity in $O(n)$ steps. Thus:
    $O(n^2 + n)$
  – Thus, we have:
    $O(n^2) + O(n)(n^2 + n) = O(n^3)$
  (with an efficient implementation in some cases may accomplish finding the “closest” in $O(n \log n)$ steps; Thus:
    $O(n^2) + (n)(n \log n + n) = O(n^2 \log n)$

  Thus, very expensive!

Buckshot Clustering

• A hybrid approach (HAC & K-Means)

• To avoid building the DOC-DOC matrix:
  – Buckshot (building similarity matrix for a subset)

• Goal is to reduce run time to $O(kn)$ instead of $O(n^3)$ or $O(n^2 \log n)$ of HAC.
Buckshot Algorithm

• Randomly select $d$ documents where $d$ is $\sqrt{n}$ or $\sqrt{kn}$
• Cluster these using hierarchical clustering algorithm into $k$ clusters: $\sim O(\sqrt{n})^2$
• Compute the centroid of each of the $k$ clusters: $O(\sqrt{n})$
• Scan remaining documents and assign them to the closest of the $k$ clusters ($k$-means): $O(n - \sqrt{n})$
• Thus: $O(\sqrt{n})^2 + O(\sqrt{n}) + O(n - \sqrt{n}) \sim O(n)$

Summary

• Clustering provides users an overview of the contents of a document collection
• Commonly used in organizing search results
• Cluster labeling aims to make the clusters meaningful for users
• Can reduce the search space and improve efficiency, and potentially accuracy
• HAC is computationally expensive
• K-Means suits for clustering large data sets
• Difficulty in evaluating the quality of clusters