

CS574 Language Technologies

Text Clustering

Claire Cardie and Thorsten Joachims
Cornell University

Some slides based on those of Yiming Yang (CMU);
some based on those of Ray Mooney (UTAustin)

Text clustering

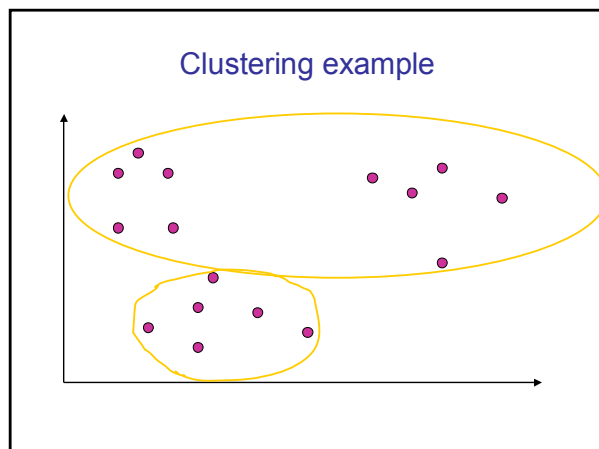
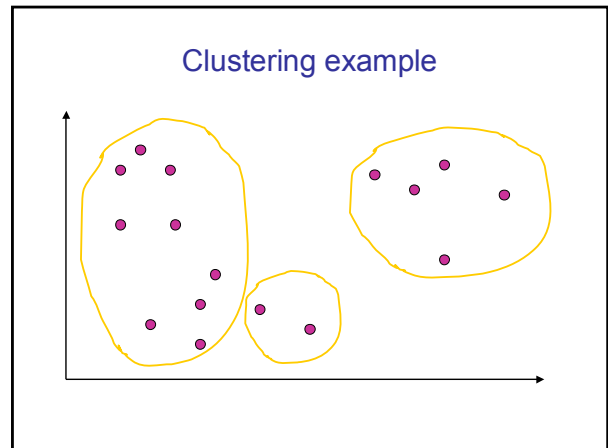
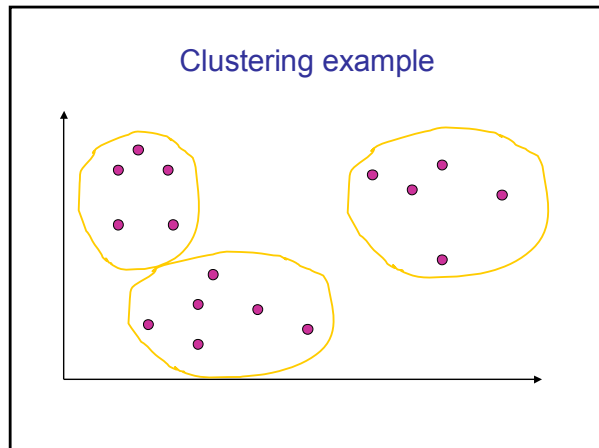
- Applications of clustering in IR
- Document clustering approaches
- Evaluation

Clustering

- Partition unlabeled examples into disjoint subsets of *clusters*, such that:
 - Examples within a cluster are very similar
 - Examples in different clusters are very different
- Discover new categories in an *unsupervised* manner (no sample category labels provided).

Introduction to document clustering

- Text categorization
 - *Supervised* method for partitioning documents into groups according to pre-defined categories
 - Requires labeled data for training
- Document clustering
 - *Unsupervised* method for partitioning documents into groups when no pre-defined categories/classes are available
 - Discovers new categories of document in an unsupervised manner

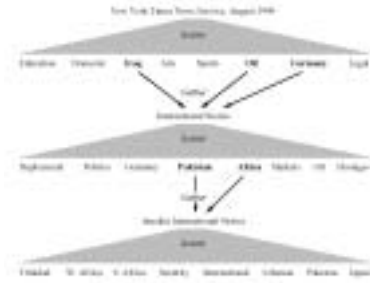


Applications of document clustering

- ┆ **Cluster retrieved documents**
 - ┆ to present more organized and understandable results to user
- ┆ **Cluster documents in collection (global analysis)**
 - ┆ during retrieval, add other documents in the same cluster as the initial retrieved documents to improve recall
- ┆ **Automated (or semi-automated) creation of document taxonomies**
 - ┆ e.g. Yahoo-style
- ┆ **Improve document representation**
 - ┆ e.g. probabilistic LSI [Hofmann SIGIR 98]

Applications of document clustering

Cluster-based browsing: scatter/gather



Cutting et al. 1992

Applications of document clustering

Event detection from news streams

- ñ TDT = topic detection and tracking
- ñ TREC track beginning in late 1990s

TDT1 corpus: CNN & Reuters news stories, Jan-Feb 1995

size	top-ranking words per cluster
330	republ clinton congress hous amend
217	simpson o presecut trial jury
98	israel palestin gaza peac arafat
97	japan kobe earthquake quak toky
93	russian chhech chechny grozn yeltsin

Text clustering

Applications of clustering in IR

Document clustering approaches

- ñ Standard clustering techniques
 - Y Similarity measure
 - Y Clustering criterion
 - Y Clustering algorithm
 - ñ Non-hierarchical
 - ñ Hierarchical

Evaluation

Similarity (distance) measures

Euclidian distance (L_2 norm):

$$L_2(\vec{x}, \vec{y}) = \sum_{i=1}^m (x_i - y_i)^2$$

L_1 norm:

$$L_1(\vec{x}, \vec{y}) = \sum_{i=1}^m |x_i - y_i|$$

Cosine similarity (transform to a distance by subtracting from 1):

$$\frac{\vec{x} \cdot \vec{y}}{|\vec{x}| \cdot |\vec{y}|}$$

Cluster similarity

- **How to compute similarity of two clusters each possibly containing multiple instances?**

- **Single link:** Similarity of two most similar members.
- **Complete link:** Similarity of two least similar members.
- **Group average:** Average similarity between members.

Clustering criterion

- **Evaluation function that assigns a (usually real-valued) value to a clustering**
 - Maximize product of within-cluster similarity and between-cluster dissimilarity
- **No optimization --- just greedy search**

Non-hierarchical clustering

- **Single-pass clustering**
- **K-means clustering** (i hardi)
- **Expectation maximization** (i softi)

Single-pass clustering

- **Set the initial set S of clusters to be empty.**
- **Pick the next document d at random (or following a given order)**
 - Treat d as a new cluster with only one member
- **Compare d to all clusters in S :**
 - If the **similarity** between d and any cluster in S is above a (pre-defined) threshold,
 - Then merge d with the closest cluster in S ;
 - Else add d to S .
- **Repeat steps 2 and 3 until all documents are processed.**

Complexity:

K-means clustering

- Must provide the number of desired clusters, k .
- Randomly choose k instances as *seeds*, one per cluster.
- Form initial clusters based on these seeds.
- Iterate, repeatedly reallocating instances to different clusters to improve the overall clustering.
- Stop when clustering converges or after a fixed number of iterations.

K-means

- Assumes instances are real-valued vectors.
- Clusters represented via *centroids*, *center of gravity*, or mean of points in a cluster, c :

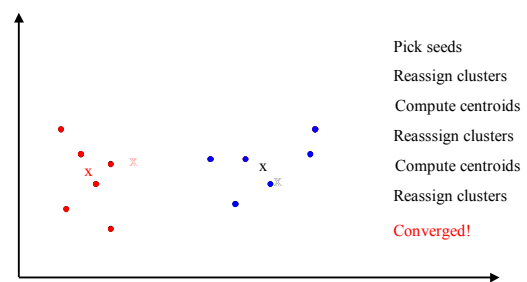
$$\bar{\mu}(c) = \frac{1}{|c|} \sum_{\vec{x} \in c} \vec{x}$$

- Reassignment of instances to clusters is based on *distance* to the current cluster centroids.

K-means algorithm

Let d be the distance measure between instances.
 Select k random instances $\{s_1, s_2, \dots, s_k\}$ as seeds.
 Until clustering converges or other stopping criterion:
 For each instance x_i :
 Assign x_i to the cluster c_j such that $d(x_i, s_j)$ is minimal.
 //Update the seeds to be the centroid of each cluster
 For each cluster c_j
 $s_j = \mu(c_j)$

K-means example (k=2)



Time complexity

- ┆ Assume computing distance between two instances is $O(m)$ where m is the dimensionality of the vectors.
- ┆ Reassigning clusters for n points: $O(kn)$ distance computations, or $O(knm)$.
- ┆ Computing centroids: Each instance gets added once to some centroid: $O(nm)$.
- ┆ Assume these two steps are each done once for i iterations: $O(iknm)$.
- ┆ Linear in all relevant factors, assuming a fixed number of iterations, more efficient than $O(n^2)$ HAC.

Seed choice

- ┆ Results can vary based on random seed selection.
- ┆ Some seeds can result in poor convergence rate, or convergence to sub-optimal clusterings.
- ┆ Select good seeds using a heuristic or the results of another method.

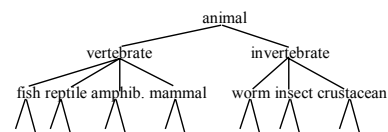
Text clustering

- ┆ Applications of clustering in IR
- ┆ Document clustering approaches
 - Standard clustering techniques
 - ┆ Similarity measure
 - ┆ Clustering criterion
 - ┆ Clustering algorithm
 - Non-hierarchical
 - Hierarchical
- ┆ Evaluation



Hierarchical clustering

- ┆ Build a tree-based hierarchical taxonomy from a set of unlabeled examples.



- ┆ Recursive application of a standard clustering algorithm can produce a hierarchical clustering.

Agglomerative vs. divisive clustering

- **Agglomerative (bottom-up)** methods start with each example in its own cluster and iteratively combine them to form larger and larger clusters.
- **Divisive (top-down)** separate all examples immediately into clusters.

Hierarchical agglomerative clustering (HAC)

- Assumes a **similarity function** for determining the similarity of two instances.
- Starts with all instances in a separate cluster and then repeatedly joins the two clusters that are most similar until there is only one cluster.
- The history of merging forms a binary tree or hierarchy.

Start with all instances in their own cluster.

Until there is only one cluster:

Among the current clusters, determine the two clusters, c_i and c_j , that are most similar.

Replace c_i and c_j with a single cluster $c_i \cup c_j$

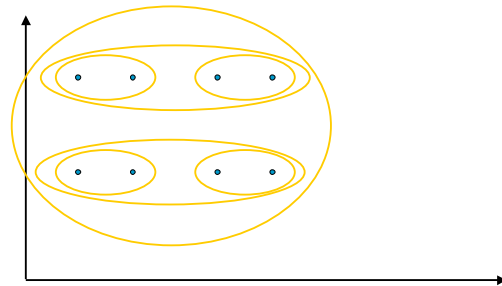
Single link agglomerative clustering

- When computing cluster similarity, use maximum similarity of pairs:

$$\text{sim}(c_i, c_j) = \max_{x \in c_i, y \in c_j} \text{sim}(x, y)$$

- Can result in "straggly" (long and thin) clusters due to chaining effect.

Single link example



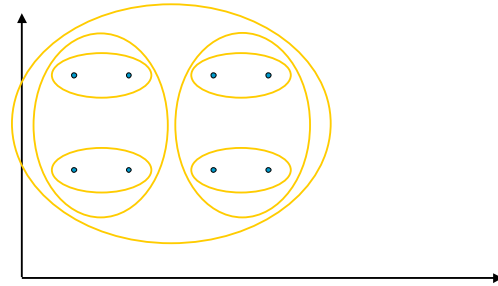
Complete link agglomerative clustering

- When computing cluster similarity, use minimum similarity of pairs:

$$\text{sim}(c_i, c_j) = \min_{x \in c_i, y \in c_j} \text{sim}(x, y)$$

- Makes more "tight" spherical clusters that are typically preferable.

Complete link example



Computational complexity of HAC

- In the first iteration, all HAC methods need to compute similarity of all pairs of n individual instances which is $O(n^2)$.
- In each of the subsequent $n-2$ merging iterations, it must compute the distance between the most recently created cluster and all other existing clusters.
- In order to maintain an overall $O(n^2)$ performance, computing similarity to each other cluster must be done in constant time.

Computing cluster similarity

- After merging c_i and c_j , the similarity of the resulting cluster to any other cluster, c_k , can be computed by:

Single Link:

$$\text{sim}((c_i \cup c_j), c_k) = \max(\text{sim}(c_i, c_k), \text{sim}(c_j, c_k))$$

Complete Link:

$$\text{sim}((c_i \cup c_j), c_k) = \min(\text{sim}(c_i, c_k), \text{sim}(c_j, c_k))$$

Group average agglomerative clustering

- Use average similarity across all pairs within the merged cluster to measure the similarity of two clusters.

$$\text{sim}(c_i, c_j) = \frac{1}{|c_i \cup c_j|(|c_i \cup c_j| - 1)} \sum_{\vec{x} \in (c_i \cup c_j)} \sum_{\vec{y} \in (c_i \cup c_j); \vec{y} \neq \vec{x}} \text{sim}(\vec{x}, \vec{y})$$

- Compromise between single and complete link.

Computing group average similarity

- Assume cosine similarity and normalized vectors with unit length.
- Always maintain sum of vectors in each cluster.

$$\vec{s}(c_j) = \sum_{\vec{x} \in c_j} \vec{x}$$

- Compute similarity of clusters in constant time:

$$\text{sim}(c_i, c_j) = \frac{(\vec{s}(c_i) + \vec{s}(c_j)) \bullet (\vec{s}(c_i) + \vec{s}(c_j)) - (|c_i| + |c_j|)}{(|c_i| + |c_j|)(|c_i| + |c_j| - 1)}$$

Buckshot algorithm

- Combines HAC and K-means clustering.
- First randomly take a sample of instances of size $\sqrt{n}$
- Run group-average HAC on this sample, which takes only $O(n)$ time.
- Use the results of HAC as initial seeds for K-means.
- Overall algorithm is $O(n)$ and avoids problems of bad seed selection.

Text clustering

- HAC and K-means have been applied to text in a straightforward way.
- Typically use *normalized*, TF/IDF-weighted vectors and cosine similarity.
- Optimize computations for sparse vectors.

Clustering applications in IR

- ┆ **Relevance feedback for query expansion**
 - Clustering of *terms* in top-ranked documents
 - Local context analysis [Xu and Croft, SIGIR 1996]
 - ┆ Retrieve top n ranked *passages* using the original query q
 - ┆ For each concept c in the top-ranked passages, compute
 - $\text{sim}(q, c)$ using a variant of tf-idf ranking
 - concepts are one-, two-, or three-word noun groups
 - ┆ Add top m ranked concepts to q with weight $1 - 0.9x/m$ where x is the position of c in ranking. (Original query terms assigned high weights.)
- ┆ **Word clustering for text categorization**

Text clustering

- ┆ Applications of clustering in IR
- ┆ Document clustering approaches
- ➡ Evaluation

Evaluation methodologies

- ┆ Let the *market* (e.g. the Internet) select the winner
- ┆ Ask end-users whether they like the clusters
- ┆ Measure the *tightness* or *purity* of clusters
- ┆ Use human-identified clusters to evaluate system-generated ones
 - Ask humans to identify all of the clusters (possible??)
 - Use the system to generate a set of clusters (any number)
 - Assign one system cluster to each human cluster (how?)
 - Compute recall/precision/F/error/etc. for each pair of system/human clusters
 - Average the selected score over all clusters

Task-oriented evaluations---indirect

- ┆ Clustering of retrieved documents [Hearst & Pedersen, SIGIR 1996]
- ┆ Distributional word clustering for text categorization [Baker & McCallum, SIGIR 1998]
- ┆ Query clustering for recommendation systems [Beaulieu, SIGKDD 2000]
- ┆ Document clustering for novelty detection, i.e. first story detection in TDT [Yang et al. SIGIR 1998]
- ┆ Question clustering for QA [Harabagiu et al. SIGIR 2001]