CS630 Representing and Accessing Digital Information

Text Clustering

Thorsten Joachims
Cornell University

Based on slides from Prof. Claire Cardie, Prof. Ray Mooney, Prof. Yiming Yang

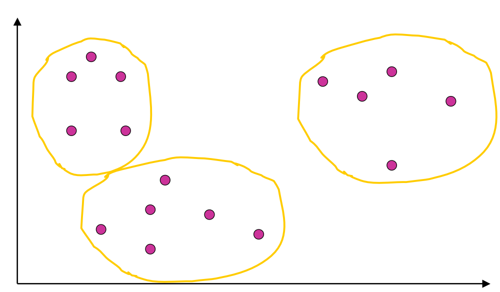
Introduction to Document Clustering

- **Text classification**
  - *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

Text Clustering

- **Clustering of Text**
  - Task definition
  - Application settings
- **Document clustering approaches**
  - Similarity measure
  - Clustering algorithm
    - Hierarchical agglomerative clustering
    - K-means
- **Evaluation**

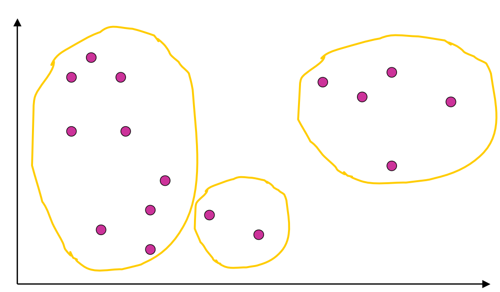
Clustering Example



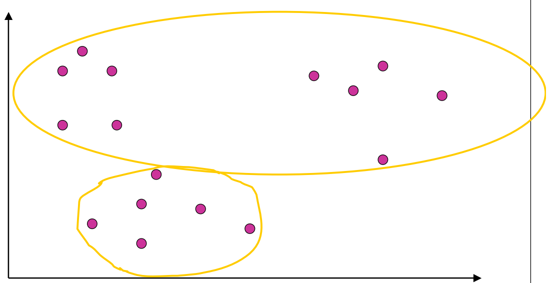
Clustering

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

Clustering Example



Clustering Example



Applications of Document Clustering

- **Event detection from news streams**
  - TDT = topic detection and tracking
  - TREC track beginning in late 1990's

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

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]

Text Clustering

- **Applications of clustering in IR**
- **Document clustering approaches**
  - Similarity measure
  - Clustering algorithm
    - Hierarchical agglomerative clustering
    - K-means
- **Evaluation**

Applications of Document Clustering

- **Cluster-based browsing: scatter/gather**

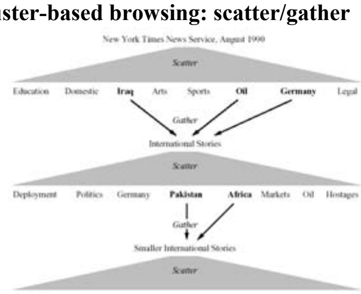


Figure 1: Illustration of Scatter/Gather

Cutting et al. 1992

Similarity (Distance) Measures

- **Euclidian distance (L_2 norm):**

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

- **L_1 norm:**

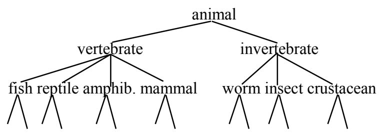
$$L_1(\vec{x},\vec{x}') = \sum_{i=1}^{m}|x_i - x_i'|$$

- **Cosine similarity:**

$$\cos(\vec{x},\vec{x}') = \frac{\vec{x} \bullet \vec{x}'}{|\vec{x}| \cdot |\vec{x}'|}$$

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.**

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.

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.**

Single-Link Agglomerative Clustering

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

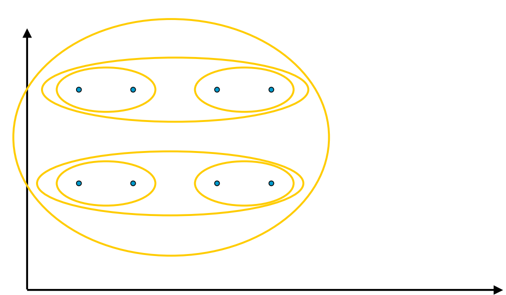
$$sim(c_i, c_j) = \max_{x \in c_i, y \in c_j} sim(x, y)$$

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

Hierarchical Agglomerative Clustering (HAC)

- **Assumes a *similarity function* for determining the similarity of two clusters.**
- **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.**
- **Basic algorithm:**
  - 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 Example



Complete Link Agglomerative Clustering

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

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

- **Makes more "tight," spherical clusters.**

Computing Cluster Similarity

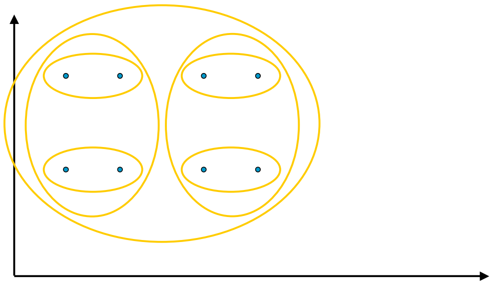
- **After merging c_i and c_j, the similarity of the resulting cluster to any other cluster, c_k, can be computed by:**
  - Single Link:

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

  - Complete Link:

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

Complete Link Example



Group Average Agglomerative Clustering

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

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

- **Compromise between single and complete link.**

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 the similarity to any other cluster must each be done in constant time.**

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:**

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

Non-Hierarchical Clustering

- **Single-pass clustering**
- **K-means clustering ("hard")**
- **Expectation maximization ("soft")**

Centroid-Based Clustering

- **Assumes instances are real-valued vectors.**
- **Clusters represented via *centroids* (i.e. mean of points in a cluster) c:**

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

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

Clustering Criterion

- **Evaluation function that assigns a (usually real-valued) value to a clustering**
  - Typically function of
    - within-cluster similarity and
    - between-cluster dissimilarity
- **Optimization**
  - Find clustering that maximizes the criterion
    - Global optimization (often intractable)
    - Greedy search
    - Approximation algorithms

K-Means Algorithm

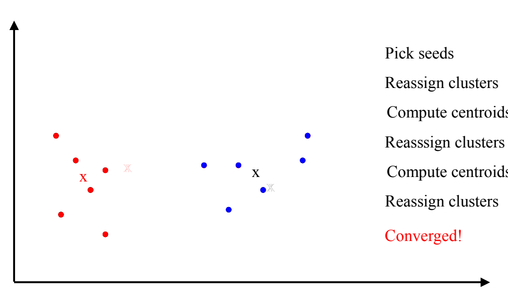
- Input: k = number of clusters, distance measure d
- Select k random instances $\{s_1, s_2, \ldots s_k\}$ as seeds.
- Until clustering converges or other stopping criterion:
  - For each instance x_i:
    - Assign x_i to the cluster c_j such that $d(x_i, s_j)$ is min.
  - For each cluster c_j *//update the centroid of each cluster*
    - $s_j = \mu(c_j)$

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 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.**

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.**

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.**

Clustering Applications in IR

- **Relevance feedback for query expansion**
  - Clustering of ***terms*** in top-ranked documents [Xu and Croft, 1996]
- **Scatter-Gather**
  - Clustering top-ranked documents to remove redundancy [Cutting et al., 1992]
- **Word clustering for text categorization**
  - Group similar words into equivalent classes [Baker and McCallum, 1998]
- **Co-Clustering**
  - Simultanously cluster words and documents [Dhillon, 2001]

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

- **Applications of clustering in IR**
- **Document clustering approaches**
  - Similarity measure
  - Clustering algorithm
    - Hierarchical agglomerative clustering
    - K-means
- **Evaluation**

Evaluation Methodologies

- **Ask end-users whether they like the clusters**
- **Let the "market" (e.g. the Internet) select the winner**
- **Measure the "tightness" or "purity" of clusters**
- **Use human-identified clusters to evaluate system-generated ones**
  - Ask humans to identify all of the clusters
  - Use the system to generate a set of clusters
  - Assign one system cluster to each human cluster
  - 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** **[Beeferman, 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]**