

Introduction to Information Retrieval

<http://informationretrieval.org>

IIR 17: Hierarchical Clustering

Hinrich Schütze

Institute for Natural Language Processing, Universität Stuttgart

2011-12-16

Overview

- 1 Introduction
- 2 Single-link/Complete-link
- 3 Centroid/GAAC
- 4 Labeling clusters
- 5 Variants

Take-away today

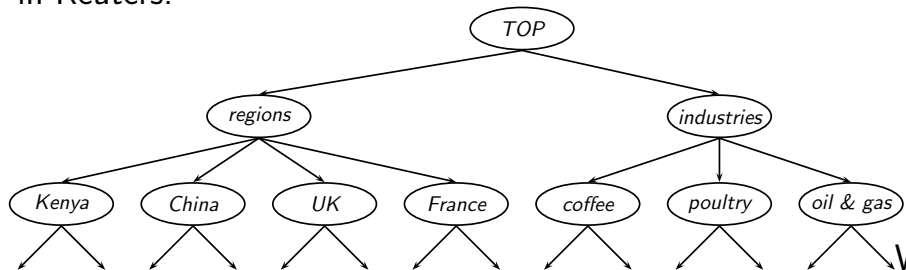
- Introduction to hierarchical clustering
- Single-link and complete-link clustering
- Centroid and group-average agglomerative clustering (GAAC)
- Bisecting K-means
- How to label clusters automatically □

Outline

- 1 Introduction
- 2 Single-link/Complete-link
- 3 Centroid/GAAC
- 4 Labeling clusters
- 5 Variants

Hierarchical clustering

Our goal in hierarchical clustering is to create a hierarchy like the one we saw earlier in Reuters:



want to create this hierarchy **automatically**. We can do this either **top-down** or **bottom-up**. The best known bottom-up method is **hierarchical agglomerative clustering**. □

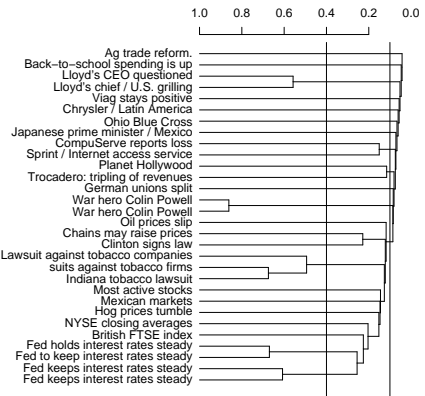
Hierarchical agglomerative clustering (HAC)

- HAC creates a hierarchy in the form of a binary tree.
- Assumes a similarity measure for determining the similarity of two clusters.
- Up to now, our similarity measures were for documents.
- We will look at four different cluster similarity measures. □

HAC: Basic algorithm

- Start with **each document in a separate cluster**
- Then **repeatedly merge** the two clusters that are most similar
- Until there is only one cluster.
- The history of merging is a hierarchy in the form of a binary tree.
- The standard way of depicting this history is a **dendrogram**. □

A dendrogram



- The history of mergers can be read off from bottom to top.
- The horizontal line of each merger tells us what the similarity of the merger was.
- We can cut the dendrogram at a particular point (e.g., at 0.1 or 0.4) to get a flat clustering.

Divisive clustering

- Divisive clustering is top-down.
- Alternative to HAC (which is bottom up).
- Divisive clustering:
 - Start with all docs in one big cluster
 - Then recursively split clusters
 - Eventually each node forms a cluster on its own.
- → Bisecting K -means at the end
- For now: HAC (= bottom-up)



Naive HAC algorithm

```
SIMPLEHAC( $d_1, \dots, d_N$ )
1  for  $n \leftarrow 1$  to  $N$ 
2  do for  $i \leftarrow 1$  to  $N$ 
3      do  $C[n][i] \leftarrow \text{SIM}(d_n, d_i)$ 
4       $I[n] \leftarrow 1$  (keeps track of active clusters)
5   $A \leftarrow []$  (collects clustering as a sequence of merges)
6  for  $k \leftarrow 1$  to  $N - 1$ 
7      do  $\langle i, m \rangle \leftarrow \arg \max_{\{ \langle i, m \rangle : i \neq m \wedge I[i]=1 \wedge I[m]=1 \}} C[i][m]$ 
8           $A.\text{APPEND}(\langle i, m \rangle)$  (store merge)
9          for  $j \leftarrow 1$  to  $N$ 
10             do (use  $i$  as representative for  $\langle i, m \rangle$ )
11                  $C[i][j] \leftarrow \text{SIM}(\langle i, m \rangle, j)$ 
12                  $C[j][i] \leftarrow \text{SIM}(\langle i, m \rangle, j)$ 
13              $I[m] \leftarrow 0$  (deactivate cluster)
14  return  $A$ 
```

Computational complexity of the naive algorithm

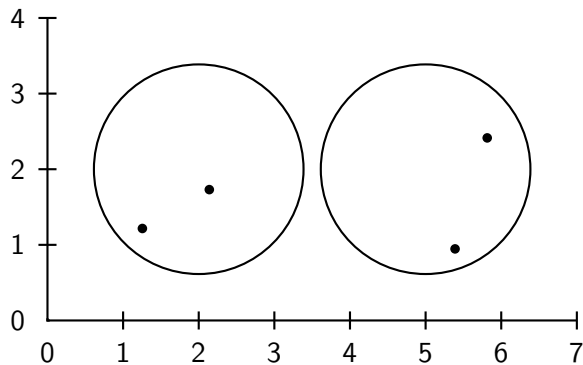
- First, we compute the similarity of all $N \times N$ pairs of documents.
- Then, in each of N iterations:
 - We scan the $O(N \times N)$ similarities to find the maximum similarity.
 - We merge the two clusters with maximum similarity.
 - We compute the similarity of the new cluster with all other (surviving) clusters.
- There are $O(N)$ iterations, each performing a $O(N \times N)$ “scan” operation.
- Overall complexity is $O(N^3)$.
- We'll look at more efficient algorithms later. □

Key question: How to define cluster similarity

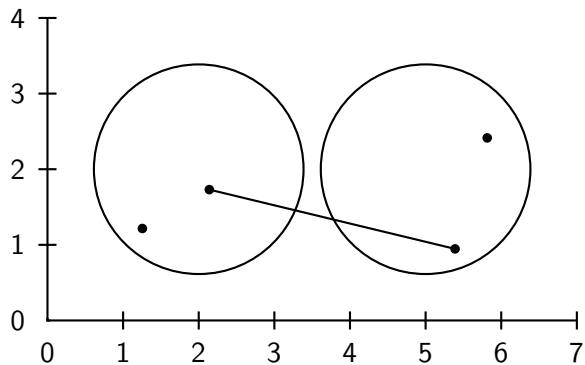
- Single-link: Maximum similarity
 - Maximum similarity of any two documents
- Complete-link: Minimum similarity
 - Minimum similarity of any two documents
- Centroid: Average “intersimilarity”
 - Average similarity of all document pairs (but excluding pairs of docs in the same cluster)
 - This is equivalent to the similarity of the centroids.
- Group-average: Average “intrasimilarity”
 - Average similarity of all document pairs, including pairs of docs in the same cluster



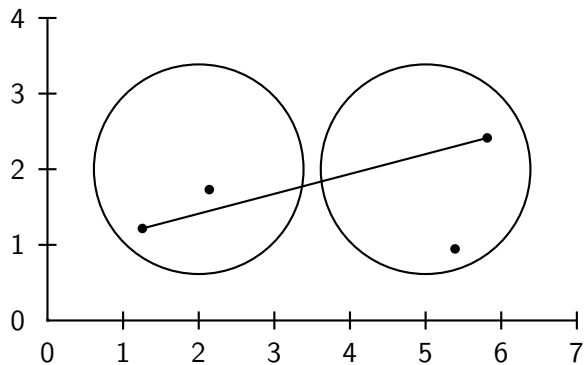
Cluster similarity: Example



Single-link: Maximum similarity

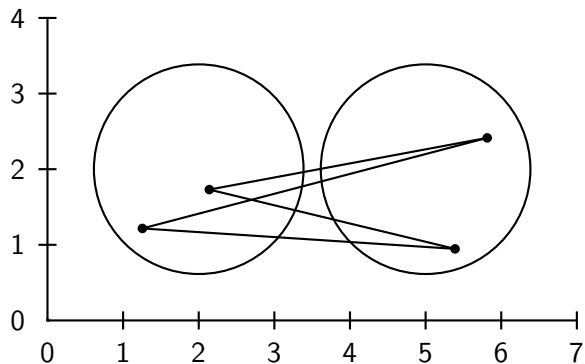


Complete-link: Minimum similarity



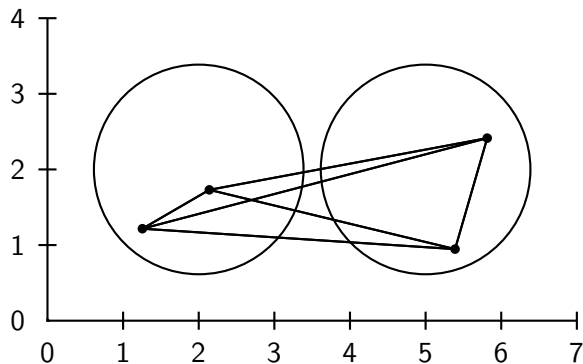
Centroid: Average intersimilarity

intersimilarity = similarity of two documents in different clusters

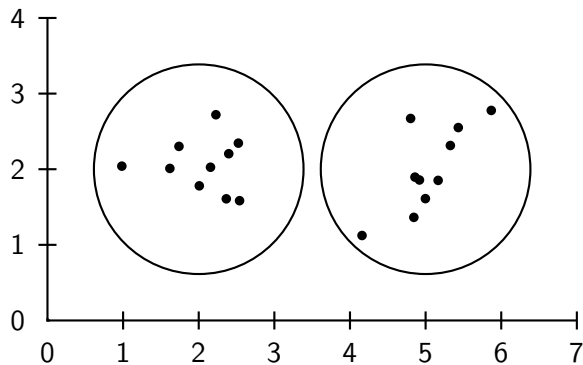


Group average: Average intrasimilarity

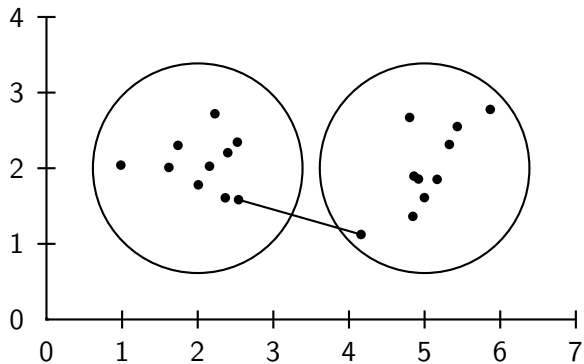
intrasimilarity = similarity of **any pair**, including cases where the two documents are in the same cluster



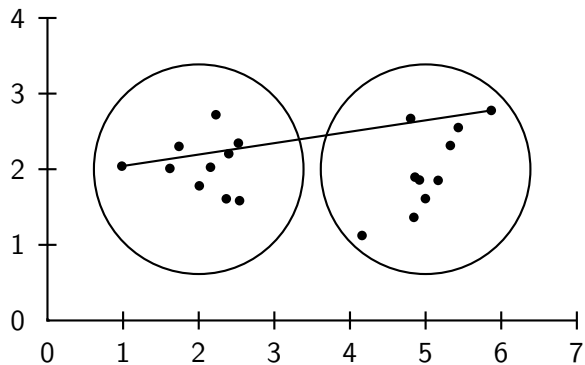
Cluster similarity: Larger Example



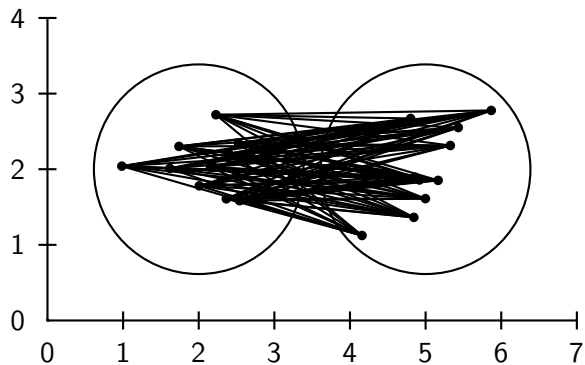
Single-link: Maximum similarity



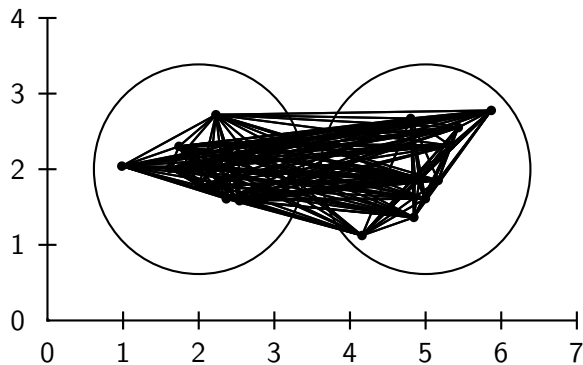
Complete-link: Minimum similarity



Centroid: Average intersimilarity



Group average: Average intrasimilarity



Outline

- 1 Introduction
- 2 Single-link/Complete-link
- 3 Centroid/GAAC
- 4 Labeling clusters
- 5 Variants

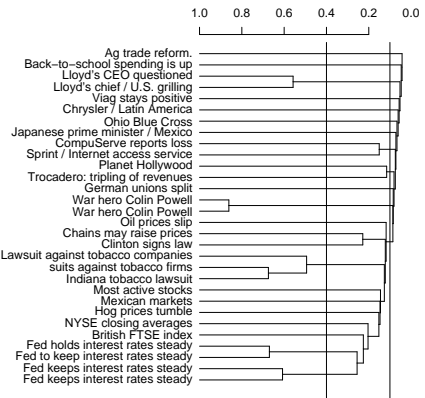
Single link HAC

- The similarity of two clusters is the **maximum** intersimilarity – the maximum similarity of a document from the first cluster and a document from the second cluster.
- Once we have merged two clusters, how do we update the similarity matrix?
- This is simple for single link:

$$\text{SIM}(\omega_i, (\omega_{k_1} \cup \omega_{k_2})) = \max(\text{SIM}(\omega_i, \omega_{k_1}), \text{SIM}(\omega_i, \omega_{k_2}))$$



This dendrogram was produced by single-link



- Notice: many small clusters (1 or 2 members) being added to the main cluster
- There is no balanced 2-cluster or 3-cluster clustering that can be derived by cutting the dendrogram. □

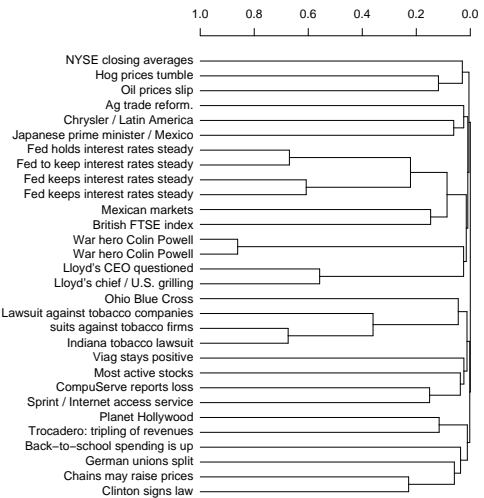
Complete link HAC

- The similarity of two clusters is the **minimum** intersimilarity – the minimum similarity of a document from the first cluster and a document from the second cluster.
- Once we have merged two clusters, how do we update the similarity matrix?
- Again, this is simple:

$$\text{SIM}(\omega_i, (\omega_{k_1} \cup \omega_{k_2})) = \min(\text{SIM}(\omega_i, \omega_{k_1}), \text{SIM}(\omega_i, \omega_{k_2}))$$

- We measure the similarity of two clusters by computing the diameter of the cluster that we would get if we merged them. □

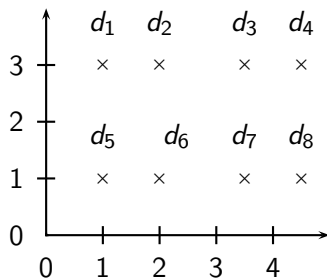
Complete-link dendrogram



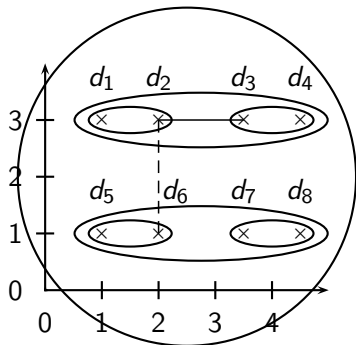
- Notice that this dendrogram is much more balanced than the single-link one.
- We can create a 2-cluster clustering with two clusters of about the same size.



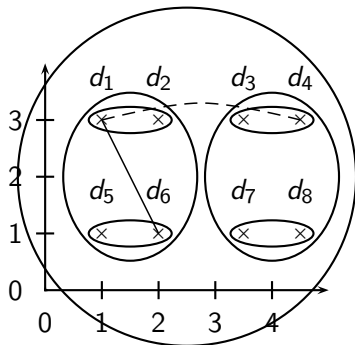
Exercise: Compute single and complete link clusterings



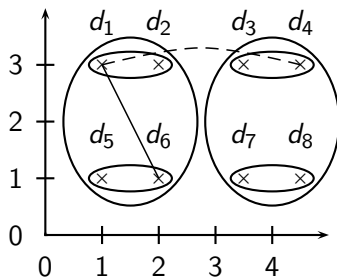
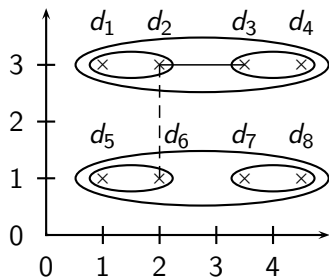
Single-link clustering



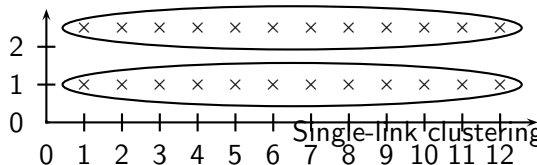
Complete link clustering



Single-link vs. Complete link clustering



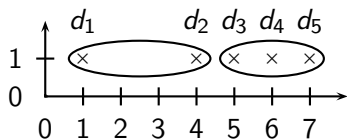
Single-link: Chaining



Single-link clustering often produces long,

straggly clusters. For most applications, these are undesirable. □

What 2-cluster clustering will complete-link produce?

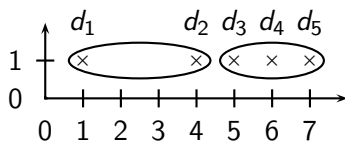


Coordinates:

$$1 + 2 \times \epsilon, 4, 5 + 2 \times \epsilon, 6, 7 - \epsilon.$$



Complete-link: Sensitivity to outliers



- The complete-link clustering of this set splits d_2 from its right neighbors – clearly undesirable.
- The reason is the outlier d_1 .
- This shows that a single outlier can negatively affect the outcome of complete-link clustering.
- Single-link clustering does better in this case. □

Outline

- 1 Introduction
- 2 Single-link/Complete-link
- 3 Centroid/GAAC
- 4 Labeling clusters
- 5 Variants

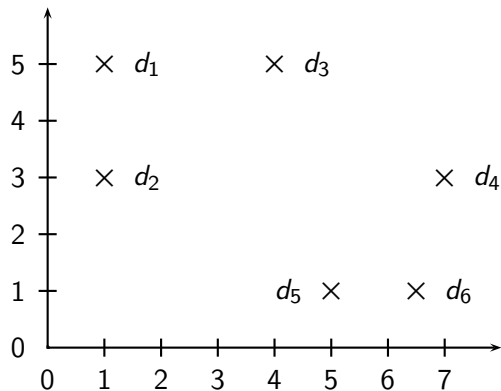
Centroid HAC

- The similarity of two clusters is the average intersimilarity – the average similarity of documents from the first cluster with documents from the second cluster.
- A naive implementation of this definition is inefficient ($O(N^2)$), but the definition is equivalent to **computing the similarity of the centroids**:

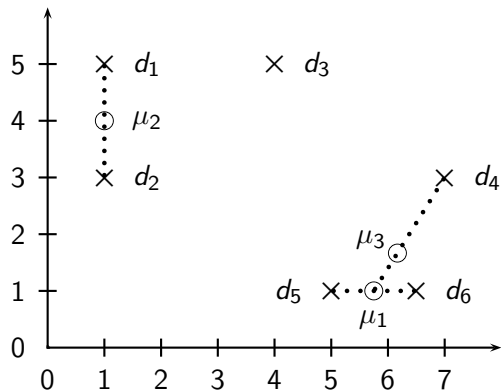
$$\text{SIM-CENT}(\omega_i, \omega_j) = \vec{\mu}(\omega_i) \cdot \vec{\mu}(\omega_j)$$

- Hence the name: centroid HAC
- Note: this is the dot product, not cosine similarity! □

Exercise: Compute centroid clustering

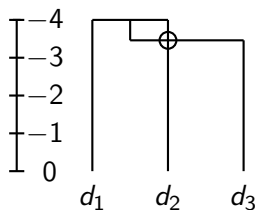
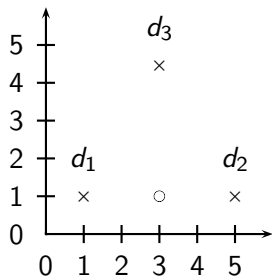


Centroid clustering



Inversion in centroid clustering

- In an inversion, the similarity **increases** during a merge sequence. Results in an “inverted” dendrogram.
- Below: Similarity of the first merger ($d_1 \cup d_2$) is -4.0 , similarity of second merger ($(d_1 \cup d_2) \cup d_3$) is ≈ -3.5 .



Inversions

- Hierarchical clustering algorithms that allow inversions are inferior.
- The rationale for hierarchical clustering is that at any given point, we've found the most coherent clustering for a given K .
- Intuitively: smaller clusterings should be more coherent than larger clusterings.
- An inversion contradicts this intuition: we have a large cluster that is more coherent than one of its subclusters.
- The fact that inversions can occur in centroid clustering is a reason not to use it. □

Group-average agglomerative clustering (GAAC)

- GAAC also has an “average-similarity” criterion, but does not have inversions.
- The similarity of two clusters is the average **intrasimilarity** – the average similarity of all document pairs (including those from the same cluster).
- But we exclude self-similarities. □

Group-average agglomerative clustering (GAAC)

- Again, a naive implementation is inefficient ($O(N^2)$) and there is an equivalent, more efficient, centroid-based definition:

$$\text{SIM-GA}(\omega_i, \omega_j) = \frac{1}{(N_i + N_j)(N_i + N_j - 1)} \left[\left(\sum_{d_m \in \omega_i \cup \omega_j} \vec{d}_m \right)^2 - (N_i + N_j) \right]$$

- Again, this is the dot product, not cosine similarity. □

Which HAC clustering should I use?

- Don't use centroid HAC because of inversions.
- In most cases: GAAC is best since it isn't subject to chaining and sensitivity to outliers.
- However, we can only use GAAC for vector representations.
- For other types of document representations (or if only pairwise similarities for documents are available): use complete-link.
- There are also some applications for single-link (e.g., duplicate detection in web search). □

Flat or hierarchical clustering?

- For high efficiency, use flat clustering (or perhaps bisecting k -means)
- For deterministic results: HAC
- When a hierarchical structure is desired: hierarchical algorithm
- HAC also can be applied if K cannot be predetermined (can start without knowing K) □

Outline

- 1 Introduction
- 2 Single-link/Complete-link
- 3 Centroid/GAAC
- 4 Labeling clusters**
- 5 Variants

Major issue in clustering – labeling

- After a clustering algorithm finds a set of clusters: how can they be useful to the end user?
- We need a pithy label for each cluster.
- For example, in search result clustering for “jaguar”, The labels of the three clusters could be “animal”, “car”, and “operating system” .
- Topic of this section: How can we automatically find good labels for clusters? □

Exercise

- Come up with an algorithm for labeling clusters
- Input: a set of documents, partitioned into K clusters (flat clustering)
- Output: A label for each cluster
- Part of the exercise: What types of labels should we consider?
Words?

Discriminative labeling

- To label cluster ω , compare ω with all other clusters
- Find terms or phrases that distinguish ω from the other clusters
- We can use any of the feature selection criteria we introduced in text classification to identify discriminating terms: mutual information, χ^2 and frequency.
- (but the latter is actually not discriminative) □

Non-discriminative labeling

- Select terms or phrases based solely on information from the cluster itself
 - E.g., select terms with high weights in the centroid (if we are using a vector space model)
- Non-discriminative methods sometimes select frequent terms that do not distinguish clusters.
- For example, MONDAY, TUESDAY, ... in newspaper text □

Using titles for labeling clusters

- Terms and phrases are hard to scan and condense into a holistic idea of what the cluster is about.
- Alternative: titles
- For example, the titles of two or three documents that are closest to the centroid.
- Titles are easier to scan than a list of phrases. □

Cluster labeling: Example

	# docs	labeling method		
		centroid	mutual information	title
4	622	oil plant mexico production crude power000refinerygas bpd	plant oil production barrels crude bpd mexico dolly capaci- typetroleum	MEXICO: Hurricane Dolly heads for Mex- ico coast
9	1017	police security rus- sian people military peace killed told groznicourt	police killed military security peace told troops forcesrebels people	RUSSIA: Russia's Lebed meets rebel chief in Chechnya
10	1259	00 000 tonnes traders futures wheat prices centsseptember tonne	delivery traders fu- tures tonne tonnes desk wheat prices 000 00	USA: Export Business - Grain/oilseeds com- plex

- Three methods: most prominent terms in centroid, differential labeling using MI, title of doc closest to centroid
- All three methods do a pretty good job. □

Outline

- 1 Introduction
- 2 Single-link/Complete-link
- 3 Centroid/GAAC
- 4 Labeling clusters
- 5 Variants

Bisecting K -means: A top-down algorithm

- Start with all documents in one cluster
- Split the cluster into 2 using K -means
- Of the clusters produced so far, select one to split (e.g. select the largest one)
- Repeat until we have produced the desired number of clusters



Bisecting K -means

```
BISECTINGKMEANS( $d_1, \dots, d_N$ )  
1  $\omega_0 \leftarrow \{\vec{d}_1, \dots, \vec{d}_N\}$   
2  $leaves \leftarrow \{\omega_0\}$   
3 for  $k \leftarrow 1$  to  $K - 1$   
4 do  $\omega_k \leftarrow \text{PICKCLUSTERFROM}(leaves)$   
5      $\{\omega_i, \omega_j\} \leftarrow \text{KMEANS}(\omega_k, 2)$   
6      $leaves \leftarrow leaves \setminus \{\omega_k\} \cup \{\omega_i, \omega_j\}$   
7 return  $leaves$ 
```

Bisecting K -means

- If we don't generate a complete hierarchy, then a top-down algorithm like bisecting K -means is **much more efficient** than HAC algorithms.
- But bisecting K -means is not deterministic.
- There are deterministic versions of bisecting K -means (see resources at the end), but they are much less efficient. □

Efficient single link clustering

```
SINGLELINKCLUSTERING( $d_1, \dots, d_N, K$ )
1  for  $n \leftarrow 1$  to  $N$ 
2  do for  $i \leftarrow 1$  to  $N$ 
3      do  $C[n][i].sim \leftarrow SIM(d_n, d_i)$ 
4           $C[n][i].index \leftarrow i$ 
5       $I[n] \leftarrow n$ 
6       $NBM[n] \leftarrow \arg \max_{X \in \{C[n][i]: n \neq i\}} X.sim$ 
7   $A \leftarrow []$ 
8  for  $n \leftarrow 1$  to  $N - 1$ 
9  do  $i_1 \leftarrow \arg \max_{\{i: I[i]=i\}} NBM[i].sim$ 
10      $i_2 \leftarrow I[NBM[i_1].index]$ 
11      $A.APPEND(\langle i_1, i_2 \rangle)$ 
12     for  $i \leftarrow 1$  to  $N$ 
13     do if  $I[i] = i \wedge i \neq i_1 \wedge i \neq i_2$ 
14         then  $C[i_1][i].sim \leftarrow C[i][i_1].sim \leftarrow \max(C[i_1][i].sim, C[i_2][i].sim)$ 
15         if  $I[i] = i_2$ 
16             then  $I[i] \leftarrow i_1$ 
17      $NBM[i_1] \leftarrow \arg \max_{X \in \{C[i_1][i]: I[i]=i \wedge i \neq i_1\}} X.sim$ 
18  return  $A$ 
```

Time complexity of HAC

- The single-link algorithm we just saw is $O(N^2)$.
- Much more efficient than the $O(N^3)$ algorithm we looked at earlier!
- There is no known $O(N^2)$ algorithm for complete-link, centroid and GAAC.
- Best time complexity for these three is $O(N^2 \log N)$: See book.
- In practice: little difference between $O(N^2 \log N)$ and $O(N^2)$.



Combination similarities of the four algorithms

clustering algorithm	$\text{sim}(\ell, k_1, k_2)$
single-link	$\max(\text{sim}(\ell, k_1), \text{sim}(\ell, k_2))$
complete-link	$\min(\text{sim}(\ell, k_1), \text{sim}(\ell, k_2))$
centroid	$(\frac{1}{N_m} \vec{v}_m) \cdot (\frac{1}{N_\ell} \vec{v}_\ell)$
group-average	$\frac{1}{(N_m + N_\ell)(N_m + N_\ell - 1)} [(\vec{v}_m + \vec{v}_\ell)^2 - (N_m + N_\ell)]$ □

Comparison of HAC algorithms

method	combination similarity	time compl.	optimal?	comment
single-link	max intersimilarity of any 2 docs	$\Theta(N^2)$	yes	chaining effect
complete-link	min intersimilarity of any 2 docs	$\Theta(N^2 \log N)$	no	sensitive to outliers
group-average	average of all sims	$\Theta(N^2 \log N)$	no	best choice for most applications
centroid	average intersimilarity	$\Theta(N^2 \log N)$	no	inversions can occur □

What to do with the hierarchy?

- Use as is (e.g., for browsing as in Yahoo hierarchy)
- Cut at a predetermined threshold
- Cut to get a predetermined number of clusters K
 - Ignores hierarchy below and above cutting line.



Take-away today

- Introduction to hierarchical clustering
- Single-link and complete-link clustering
- Centroid and group-average agglomerative clustering (GAAC)
- Bisecting K-means
- How to label clusters automatically □

Resources

- Chapter 17 of IIR
- Resources at <http://ifnlp.org/ir>
 - Columbia Newsblaster (a precursor of Google News): McKeown et al. (2002)
 - Bisecting K -means clustering: Steinbach et al. (2000)
 - PDDP (similar to bisecting K -means; deterministic, but also less efficient): Saravesi and Boley (2004) □