Hierarchical clustering

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

We 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)

- Assumes a similarity measure for determining the similarity of two clusters (up to now: similarity of documents).
- We will look at four different cluster similarity measures.
- 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 forms a binary tree or hierarchy.
- The standard way of depicting this history is a dendrogram.

Divisive clustering

- Top-down (instead of bottom-up as in HAC)
- 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
Naive HAC algorithm

\textbf{SIMPLEHAC}(d_1, \ldots, d_N)

\begin{align*}
1 & \text{ for } n \leftarrow 1 \text{ to } N \\
2 & \text{ do for } i \leftarrow 1 \text{ to } N \\
3 & \quad \text{ do } C[n][i] \leftarrow \text{SIM}(d_n, d_i) \\
4 & \quad I[n] \leftarrow 1 \quad \text{(keeps track of active clusters)} \\
5 & \quad A \leftarrow [] \quad \text{(collects clustering as a sequence of merges)} \\
6 & \text{ for } k \leftarrow 1 \text{ to } N - 1 \\
7 & \quad \langle i, m \rangle \leftarrow \text{arg max}\{\langle i, m \rangle : i \neq m \land I[i] = 1 \land I[m] = 1\} \quad C[i][m] \\
8 & \quad A.\text{APPEND}((i, m)) \quad \text{(store merge)} \\
9 & \text{ for } j \leftarrow 1 \text{ to } N \\
10 & \quad C[i][j] \leftarrow \text{SIM}(i, m, j) \\
11 & \quad C[j][i] \leftarrow \text{SIM}(i, m, j) \\
12 & \quad I[m] \leftarrow 0 \quad \text{(deactivate cluster)} \\
13 & \text{return } A
\end{align*}

Computational complexity of the naive algorithm

- First, we compute the similarity of all $N \times N$ pairs of documents.
- Then, in each iteration:
  - 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

\begin{itemize}
\item Single-link: Maximum similarity
  - Maximum over all document pairs
\item Complete-link: Minimum similarity
  - Minimum over all document pairs
\item Centroid: Average “intersimilarity”
  - Average over all document pairs
  - This is equivalent to the similarity of the centroids.
\item Group-average: Average “intrasimilarity”
  - Average over all document pairs, including pairs of docs in the same cluster
\end{itemize}

Cluster similarity: Example
Single-link: Maximum similarity

Complete-link: Minimum similarity

Centroid: Average intersimilarity

Group average: Average intrasimilarity

\[ \text{inter similarity} = \text{similarity of two documents in different clusters} \]

\[ \text{intra similarity} = \text{similarity of any pair, including those that are in cluster 1 and those that are in cluster 2} \]
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:

\[
\text{SIM}(\omega_i, (\omega_{k_1} \cup \omega_{k_2})) = \max(\text{SIM}(\omega_i, \omega_{k_1}), \text{SIM}(\omega_i, \omega_{k_2}))
\]
Single-link: Chaining

- 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 radius of the cluster that we would get if we merged them.

Single-link clustering often produces long, straggly clusters. For most applications, these are undesirable.

What cluster structure after 10 mergers?
Complete link clustering: Example

Complete-link: Sensitivity to outliers

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.

What is the intuitively best 2-cluster clustering here? The complete-link clustering of this set. It's not intuitive. This shows that a single outlier can have a large effect on the final outcome of complete-link clustering. Coordinates:

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

Single-link vs. Complete link clustering

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.
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.
- Once we have merged two clusters, how do we update the similarity matrix?
- The above definition is inefficient (\(O(N^2)\)), but the definition is equivalent to computing the similarity of the centroids:

\[
\text{SIM-CENT}(\omega_i, \omega_j) = \bar{\mu}(\omega_i) \cdot \bar{\mu}(\omega_j)
\]

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

Centroid clustering: Example

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 \(\approx -3.5\).
Inversions

- Hierarchical clustering algorithms that allow are inferior.
- The rationale for hierarchical clustering is that at any given point, we’ve found the most coherent cluster of a given size.
- Intuitively: smaller clusters should be more coherent than larger clusters.
- An inversion contradicts this intuition: we have a large cluster that is more coherent than one of its subclusters.

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.

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: 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 Recap
2 Introduction
3 Single-link/Complete-link
4 Centroid/GAAC
5 Variants
6 Labeling clusters

Efficient single link clustering

\[
\text{SINGLELINKCLUSTERING}(d_1, \ldots, d_N)
\]

1. for \(n \leftarrow 1\) to \(N\)
2. do for \(i \leftarrow 1\) to \(N\)
3. do \(C[n][i].\text{sim} \leftarrow \text{SIM}(d_n, d_i)\)
4. \(C[n][i].\text{index} \leftarrow i\)
5. \(I[n] \leftarrow n\)
6. \(\text{NBM}[n] \leftarrow \arg \max_{X \in \{C[n][i]: i \neq n\}} X.\text{sim}\)
7. \(A \leftarrow []\)
8. for \(n \leftarrow 1\) to \(N - 1\)
9. do \(i_1 \leftarrow \arg \max_{\{j| j \neq i\}} \text{NBM}[i].\text{sim}\)
10. \(i_2 \leftarrow I[\text{NBM}[i].\text{index}]\)
11. \(A.\text{APPEND}((i_1, i_2))\)
12. for \(i \leftarrow 1\) to \(N\)
13. do if \(I[i] = i \land i \neq i_1 \land i \neq i_2\)
14. then \(C[i][i].\text{sim} \leftarrow \max(C[i][i].\text{sim}, C[i][i_1].\text{sim}, C[i][i_2].\text{sim})\)
15. if \(I[i] = i_2\)
16. then \(I[i] \leftarrow i_1\)
17. \(\text{NBM}[i] \leftarrow \arg \max_{X \in \{C[i][i]; I[i] = i \land i \neq i_1\}} X.\text{sim}\)
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

<table>
<thead>
<tr>
<th>Clustering algorithm</th>
<th>Combination similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-link</td>
<td>( \max(\text{sim}(\ell, k_1), \text{sim}(\ell, k_2)) )</td>
</tr>
<tr>
<td>Complete-link</td>
<td>( \min(\text{sim}(\ell, k_1), \text{sim}(\ell, k_2)) )</td>
</tr>
<tr>
<td>Centroid</td>
<td>( \frac{1}{N_m} \bar{v}<em>m \cdot \frac{1}{N</em>\ell} \bar{v}_\ell )</td>
</tr>
<tr>
<td>Group-average</td>
<td>( \frac{1}{(N_m + N_\ell)(N_m + N_\ell - 1)} [(\bar{v}<em>m + \bar{v}</em>\ell)^2 - (N_m + N_\ell)] )</td>
</tr>
</tbody>
</table>

Comparison of HAC algorithms

<table>
<thead>
<tr>
<th>Method</th>
<th>Combination similarity</th>
<th>Time Complexity</th>
<th>Optimal?</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-link</td>
<td>Max intersimilarity of any 2 docs</td>
<td>( \Theta(N^2) )</td>
<td>Yes</td>
<td>Chaining effect</td>
</tr>
<tr>
<td>Complete-link</td>
<td>Min intersimilarity of any 2 docs</td>
<td>( \Theta(N^2 \log N) )</td>
<td>No</td>
<td>Sensitive to outliers</td>
</tr>
<tr>
<td>Group-average</td>
<td>Average of all sims</td>
<td>( \Theta(N^2 \log N) )</td>
<td>No</td>
<td>Best choice for most applications</td>
</tr>
<tr>
<td>Centroid</td>
<td>Average intersimilarity</td>
<td>( \Theta(N^2 \log N) )</td>
<td>No</td>
<td>Inversions can occur</td>
</tr>
</tbody>
</table>

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 \( K \)
- Hierarchical clustering is often used to get \( K \) flat clusters. The hierarchy is then ignored.

\( K \)-means vs. HAC

- Consider running 2-means clustering on a corpus, each doc of which is from one of two different languages.
- What are the two clusters we would expect to see?
- Is HAC likely to produce results different from the above?
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

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

Outline

1. Recap
2. Introduction
3. Single-link/Complete-link
4. Centroid/GAAC
5. Variants
6. Labeling clusters
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”: “animal”, “car”, “operating system”
- How can we do this?

Discriminative labeling

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

Non-discriminative labeling

- Select terms or phrases based solely on information from the cluster itself
- 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.
<table>
<thead>
<tr>
<th># docs</th>
<th>centroid</th>
<th>mutual information</th>
<th>title</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>oil plant mexico production crude</td>
<td>plant oil production barrels crude bpd</td>
<td>MEXICO: Hurricane Dolly heads for Mexico coast</td>
</tr>
<tr>
<td></td>
<td>power000refinerygasbpd</td>
<td>mexico dolly capacitypetroleum</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>police security russian people military peace killed told groznycourt</td>
<td>police killed military security peace told troops forcesrebels people</td>
<td>RUSSIA: Russia's Lebed meets rebel chief in Chechnya</td>
</tr>
<tr>
<td>10</td>
<td>00 000 tonnes traders futures wheat prices centsseptember tonne</td>
<td>delivery traders futures tonne tonnes desk wheat prices 000 00</td>
<td>USA: Export Business Grain/oilseeds complex</td>
</tr>
</tbody>
</table>

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.