Cut-based & divisive clustering

Pasi Fränti

17.3.2014

Speech & Image Processing Unit
School of Computing
University of Eastern Finland
Joensuu, FINLAND
Part I: Cut-based clustering
Cut-based clustering

- What is cut?
- Can we use graph theory in clustering?
- Is normalized-cut useful?
- Are cut-based algorithms efficient?
Clustering method

• Clustering **method** = defines the problem
• Clustering **algorithm** = solves the problem
• Problem defined as cost function
  – Goodness of one cluster
  – Similarity vs. distance
  – Global vs. local cost function (what is “cut”)
• Solution: algorithm to solve the problem
Cut-based clustering

• Usually assumes graph
• Based on concept of cut
• Includes implicit assumptions which are often:
  – No difference than clustering in vector space
  – Implies sub-optimal heuristics
  – Sometimes even false assumptions!
Cut-based clustering methods

- Minimum-spanning tree based clustering (single link)
- Split-and-merge (Lin&Chen TKDE 2005): Split the data set using K-means, then merge similar clusters based on Gaussian distribution cluster similarity.
- Split-and-merge (Li, Jiu, Cot, PR 2009): Splits data into a large number of subclusters, then remove and add prototypes until no change.
- DIVFRP (Zhong et al, PRL 2008): Dividing according to furthest point heuristic.
- Normalized-cut (Shi&Malik, PAMI-2000): Cut-based, minimizing the disassociation between the groups and maximizing the association within the groups.
- Ratio-Cut (Hagen&Kahng, 1992)
- Mcut (Ding et al, ICDM 2001)
- Max k-cut (Frieze&Jerrum 1997)
- Feng et al, PRL 2010. Particle Swarm Optimization for selecting the hyperplane.

Details to be added later…
Clustering a graph

But where we get this…?
Distance graph

Calculate from vector space!
Space complexity of graph

Distance graph

But…

Complete graph

$N \cdot (N-1)/2$ edges
$= \mathcal{O}(N^2)$
Minimum spanning tree (MST)

Distance graph

MST

Works with simple examples like this
Cut

Graph cut

Cost function is to maximize the weight of edges cut

Resulted clusters

This equals to minimizing the within cluster edge weights
Cut

Graph cut

Resulted clusters

Equivalent to minimizing MSE!
Stopping criterion
Ends up to a local minimum

Stops criterion
Ends up to a local minimum

Divisive
Agglomerative

Repeated K-means
RLS
Clustering method

![Graph showing MSE, DBI, and F-test across different number of clusters. The graph highlights the minimum point for each metric.](image)
Conclusions of “Cut”

- Cut \( \Rightarrow \) Same as partition
- Cut-based method \( \Rightarrow \) Empty concept
- Cut-based algorithm \( \Rightarrow \) Same as divisive
- Graph-based clustering \( \Rightarrow \) Flawed concept
- Clustering of graph \( \Rightarrow \) more relevant topic
Part II: Divisive algorithms
Divisive approach

Motivation
• Efficiency of divide-and-conquer approach
• Hierarchy of clusters as a result
• Useful when solving the number of clusters

Challenges
• Design problem 1: What cluster to split?
• Design problem 2: How to split?
• Sub-optimal local optimization at best
Split-based (divisive) clustering

\[ \text{Split}(X, M) \rightarrow C, P \]

\[ m \leftarrow 1; \]
\[ \text{REPEAT} \]
\[ \quad \text{Select cluster to be split;} \]
\[ \quad \text{Split the cluster;} \]
\[ \quad m \leftarrow m+1; \]
\[ \quad \text{UpdateData Structures;} \]
\[ \text{UNTIL } m = M; \]
Select cluster to be split

• Heuristic choices:
  – Cluster with highest variance (MSE)
  – Cluster with most skew distribution (3rd moment)

• Optimal choice:
  – Tentatively split all clusters
  – Select the one that decreases MSE most!

• Complexity of choice:
  – Heuristics take the time to compute the measure
  – Optimal choice takes only twice (2×) more time!!!
  – The measures can be stored, and only two new clusters appear at each step to be calculated.
Selection example

Biggest MSE…

… but dividing this decreases MSE more
Selection example

Only two new values need to be calculated
How to split

• Centroid methods:
  – Heuristic 1: Replace C by C-ε and C+ε
  – Heuristic 2: Two furthest vectors.
  – Heuristic 3: Two random vectors.

• Partition according to principal axis:
  – Calculate principal axis
  – Select dividing point along the axis
  – Divide by a hyperplane
  – Calculate centroids of the two sub-clusters
Splitting along principal axis
pseudo code

Step 1: Calculate the principal axis.
Step 2: Select a dividing point.
Step 3: Divide the points by a hyper plane.
Step 4: Calculate centroids of the new clusters.
Example of dividing
Step 2.1: Calculate projections on the principal axis.
Step 2.2: Sort vectors according to the projection.
Step 2.3: FOR each vector \( x_i \) DO:
  - Divide using \( x_i \) as dividing point.
  - Calculate distortion of subsets \( D_1 \) and \( D_2 \).
Step 2.4: Choose point minimizing \( D_1 + D_2 \).
Finding dividing point

• Calculating error for next dividing point:

\[ D' = D + \frac{n_1}{n_1 + 1} \cdot |c_1 - v_i|^2 - \frac{n_2}{n_2 - 1} \cdot |c_2 - v_i|^2 \]

• Update centroids:

\[ c_1' = \frac{n_1 c_1 + v_i}{n_1 + 1} \]

\[ c_2' = \frac{n_2 c_2 - v_i}{n_2 - 1} \]

Can be done in O(1) time!!!
Sub-optimality of the split

optimal partition boundary for 2-level clustering
Example of splitting process

2 clusters

3 clusters

Principal axis
Dividing hyperplane
Example of splitting process

4 clusters

5 clusters
Example of splitting process

6 clusters

7 clusters
Example of splitting process

8 clusters

9 clusters
Example of splitting process

10 clusters

11 clusters
Example of splitting process

12 clusters

13 clusters
Example of splitting process

14 clusters

15 clusters

MSE = 1.94
K-means refinement

Result directly after split:
MSE = 1.94

Result after re-partition:
MSE = 1.39

Result after K-means:
MSE = 1.33
Time complexity

Number of processed vectors, assuming that clusters are always split into two equal halves:

$$\sum n_i = N + \left( \frac{N}{2} + \frac{N}{2} \right) + \left( \frac{N}{4} + \frac{N}{4} + \frac{N}{4} + \frac{N}{4} \right) + \ldots + \left( \frac{N}{M/2} + \ldots + \frac{N}{M/2} \right)$$

$$= N + 2 \cdot \frac{N}{2} + 4 \cdot \frac{N}{4} + \ldots \frac{M}{2} \cdot \frac{N}{M/2} = O(N \cdot \log M)$$

Assuming unequal split to $n_{\text{max}}$ and $n_{\text{min}}$ sizes:

$$n_{\text{min}} \geq p \cdot n_{\text{max}}$$

$$N = n_1 + n_2 + \ldots + n_m \geq m \cdot n_{\text{min}} \geq m \cdot p \cdot n_{\text{max}}$$

$$\Leftrightarrow n_{\text{max}} \leq \frac{N}{p \cdot m}$$
Time complexity

Number of vectors processed:

\[
\sum n_i \leq \frac{N}{p} + \frac{N}{p \cdot 2} + \frac{N}{p \cdot 3} + \ldots + \frac{N}{p \cdot M}
\]

\[
= \sum_{m=1}^{M} \frac{N}{p \cdot m} = O(N \cdot \log M)
\]

At each step, sorting the vectors is bottleneck:

\[
T(N) = \sum n_i \log n_i \leq \sum_{m=1}^{M} \frac{N}{p \cdot m} \log \frac{N}{p \cdot m}
\]

\[
\leq \log \frac{N}{p} \cdot \sum_{m=1}^{M} \frac{N}{p \cdot m} = O(N \cdot \log N \cdot \log M)
\]
Comparison of results

**Birch**

![Comparison of results graph](image-url)
Conclusions

• Divisive algorithms are efficient
• Good quality clustering
• Several non-trivial design choices
• Selection of dividing axis can be improved!
References


