Universität Konstanz

Label Hierarchy Inference in Property Graph Databases

Fabian Klopfer

Databases and Information Systems Chair

University of Konstanz, 18. August 2020

Running Example

Simple example, with no overlapping labels and a perfect hierarchy:

Node.name	Node.labels
Fernando’s	restaurant, italian
Arche	restaurant, vietnamese
Bangkok	restaurant, thai
CampusCafe	cafe, wifi
Endlicht	cafe, latenight
Pano	cafe, breakfast
Lago	shopping, mall
Seerhein Center	shopping, cheap
Seepark	Shopping, expensive

1.1 Motivation

- Implicit hierarchical structure in many data sets
- implicit in property graph
- no explicit representation in Neo4J or the property graph model

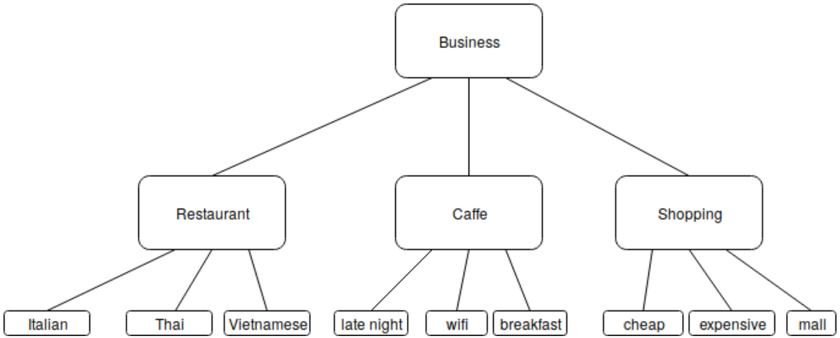
In practice it may help:

- Missing labels & other data impurities can be fixed
- Neo4J cardinality estimation can be improved
- and many others

However that’s not what is dealt with here

1.2 Problem Definition

Given: Set of labels $\{l_i, l_j, l_k, \ldots\}$
Wanted: Taxonomy/Hierarchy of labels



Node.name	Node.labels
Fernando's	restaurant, italian
Arche	restaurant, vietnamese
Bangkok	restaurant, thai
CampusCafe	cafe, wifi
Endlicht	cafe, latenight
Pano	cafe, breakfast
Lago	shopping, mall
Seerhein Center	shopping, cheap
Seepark	Shopping, expensive

2 Solutions: Hierarchical Clustering

2.1 Approaches

3 different approaches were considered:

1. Agglomerative clustering
2. Two-step clustering
3. Conceptual clustering

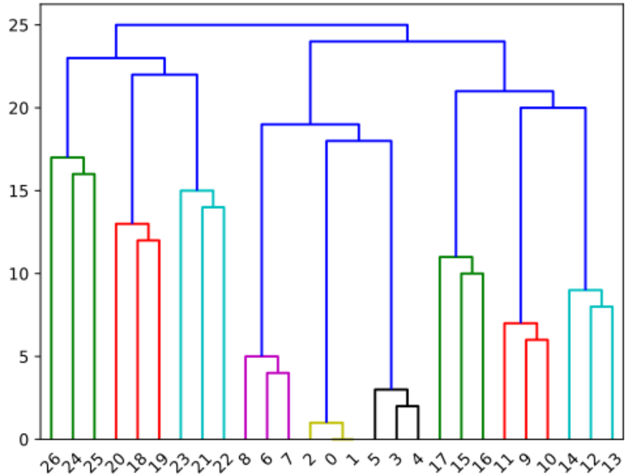
Common for 1 & 2

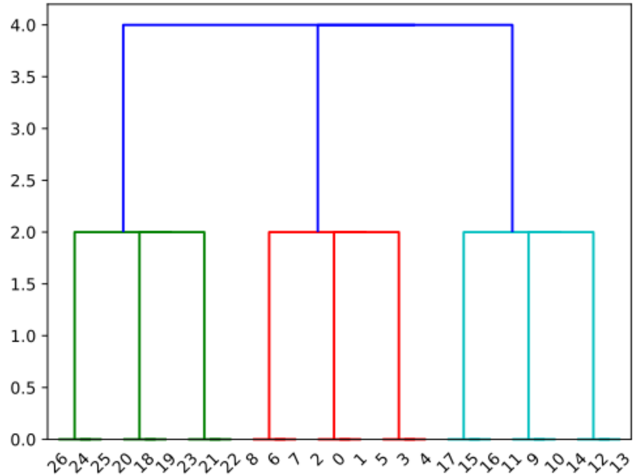
- Each data instance is set of labels
- distance measure is set similarity e.g. jaccard or l1 on vectorized representation
- Dendrograms got flattened

Node.name	Node.labels
Fernando's	restaurant, italian
Arche	restaurant, vietnamese
Bangkok	restaurant, thai
CampusCafe	cafe, wifi
Endlicht	cafe, latenight
Pano	cafe, breakfast
Lago	shopping, mall
Seerhein Center	shopping, cheap
Seepark	Shopping, expensive

Single Linkage Clustering

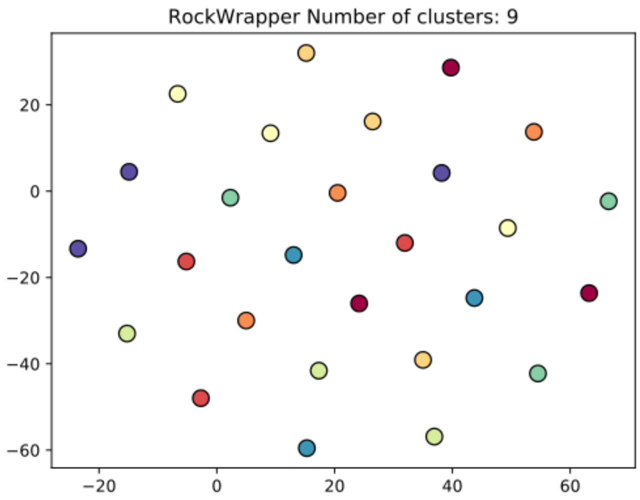
Merge the two clusters with the smallest distance per cluster/set of labels
Implementation of SciPy [1]
In the following: Enhanced plots and result trees: non-standard single linkage!

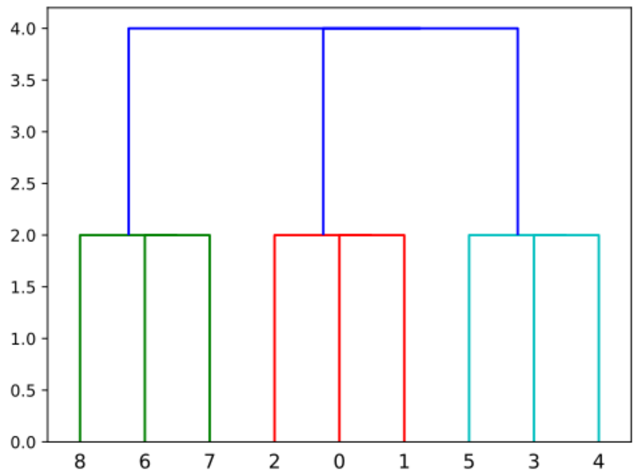




Two-step clustering

Idea: reduce number of observations by other clustering and then do hierarchical clustering.
Implementations as before, plus Scikit-Learn [2] and PyClustering [3].





Conceptual clustering

Idea: Build a hierarchy of label sets/concepts with descriptions and integrate instances iteratively, splitting when too distinct

Comparable to decision trees, form of divisive clustering. Implementation of MacLellan et al. [4]

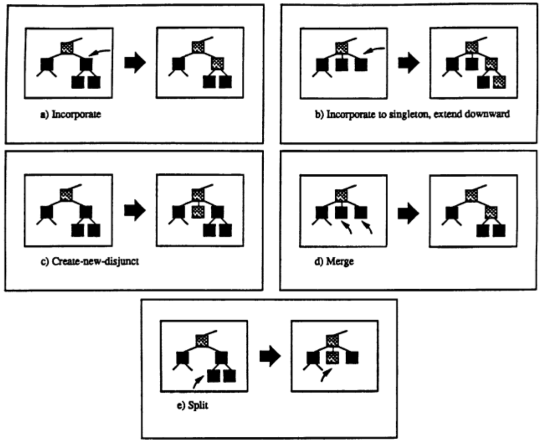


Figure 4. COBWEB's learning operators.

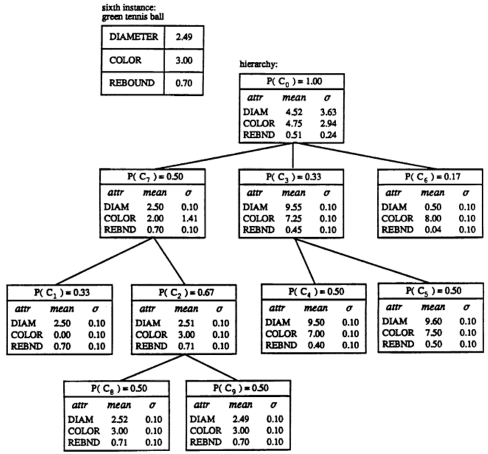


Figure 12. COBWEB hierarchy after the sixth instance of a ball.

3.1 Setup

noise $\equiv$ take a node and remove $\vee$ rename a label

```
"id":24,"labels":"l2, l22",
"id":25,"labels":"l22",
"id":26,"labels":"l1"
```

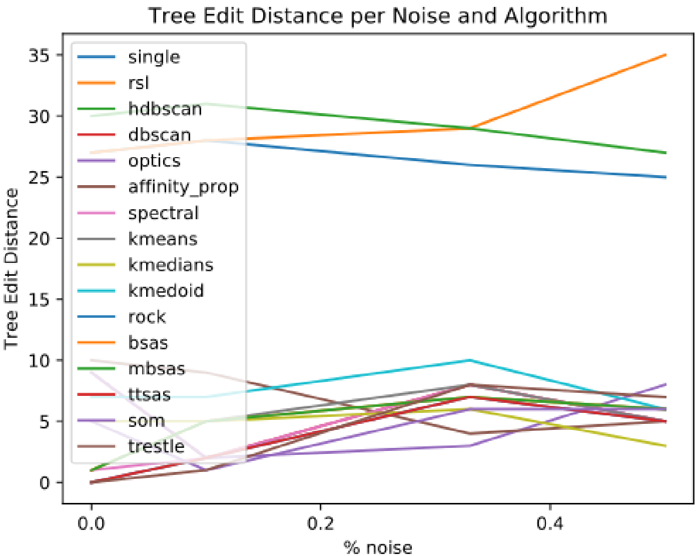
Run the algorithm on each of the variations:

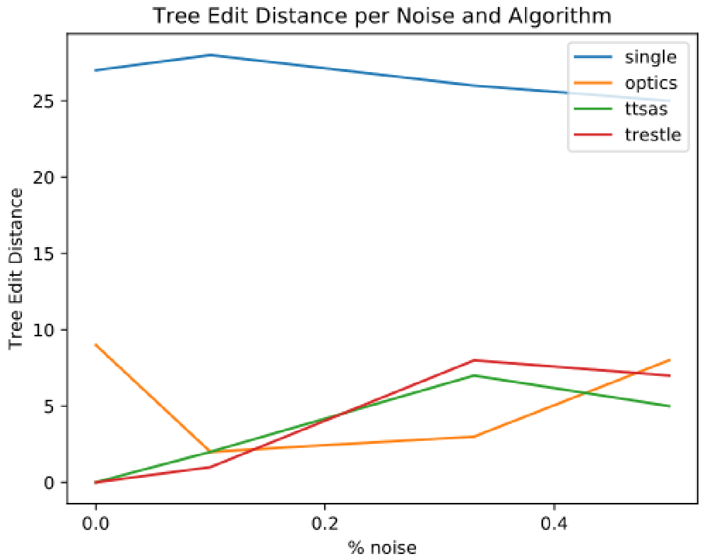
1. no noise
2. 10% noise
3. 33% noise
4. 50% noise

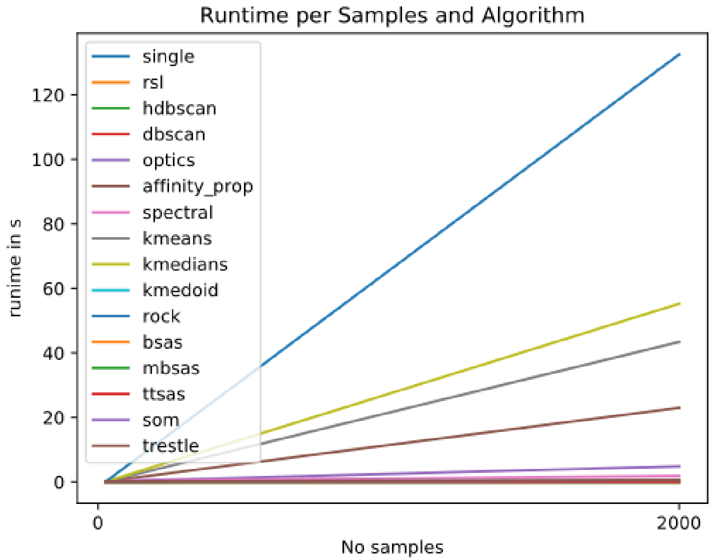
Convert the output into a bracket tree

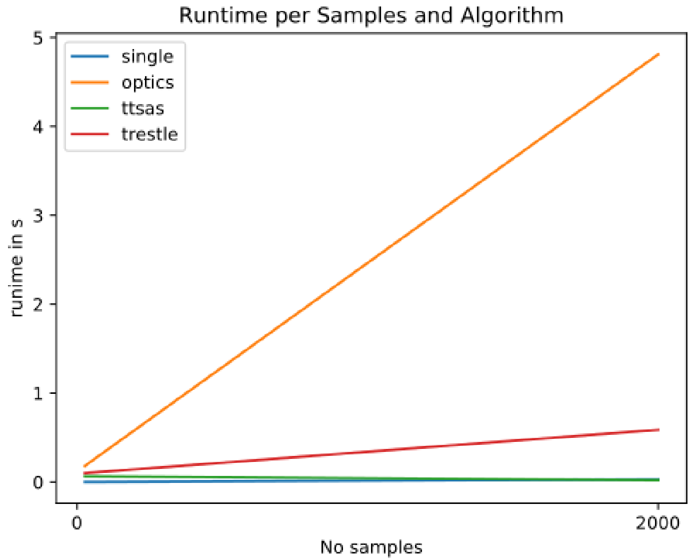
Metric for how much resulting hierarchy deviates from perfect: Tree Edit Distance [5] [6]

3.2 Results

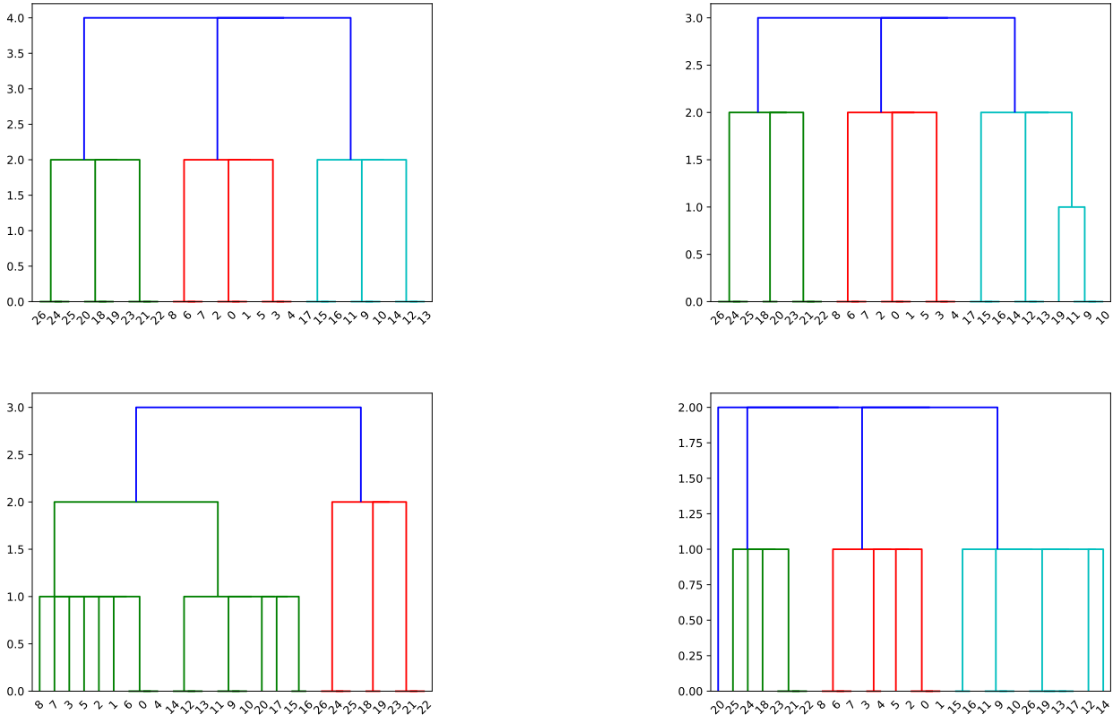

Tree Edit Distance per Noise and Algorithm
single
rsl
hdbscan
dbscan
optics
affinity_prop
spectral
kmeans
kmedians
kmedoid
rock
bsas
mbsas
ttsas
som
trestle
Tree Edit Distance
% noise
0
5
10
15
20
25
30
35
0.0
0.2
0.4


Tree Edit Distance per Noise and Algorithm
single
optics
ttsas
trestle
Tree Edit Distance
% noise
0
5
10
15
20
25
0.0
0.2
0.4

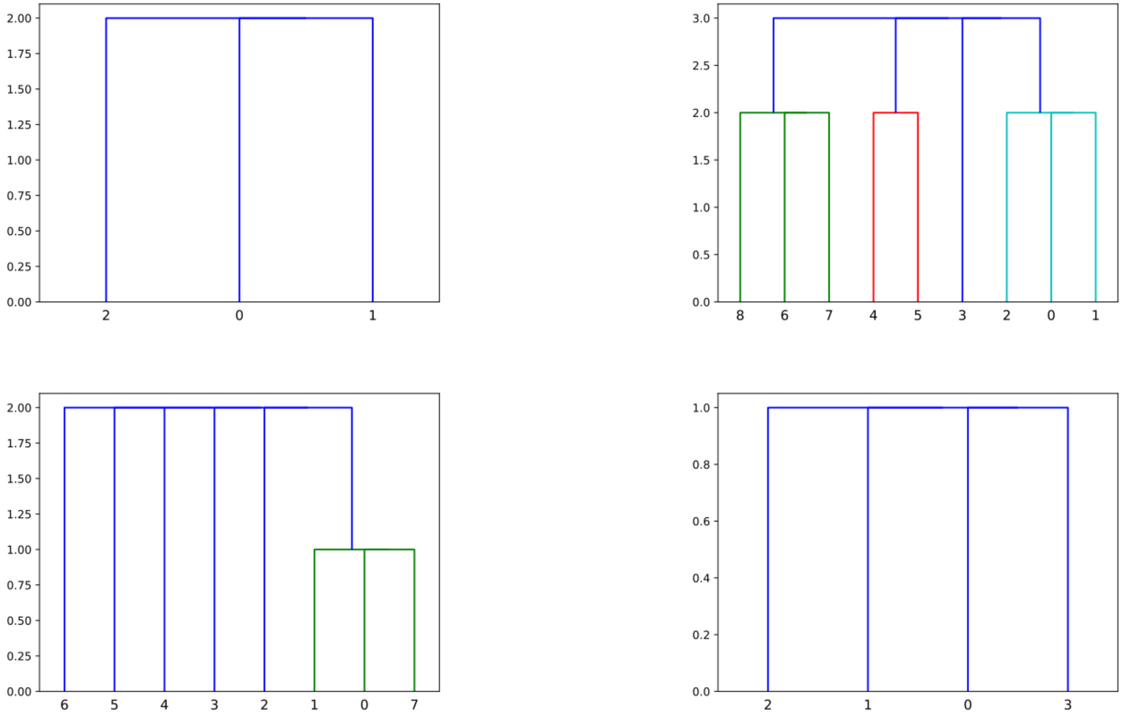

Runtime per Samples and Algorithm
single
rsl
hdbscan
dbscan
optics
affinity_prop
spectral
kmeans
kmedians
kmedoid
rock
bsas
mbsas
ttsas
som
trestle
runime in s
0
20
40
60
80
100
120
0
2000
No samples


Runtime per Samples and Algorithm
single
optics
ttsas
trestle
runime in s
0
1
2
3
4
5
0
2000
No samples

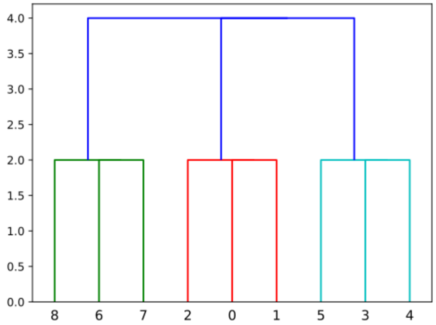
Agglomerative Clustering: Single Linkage

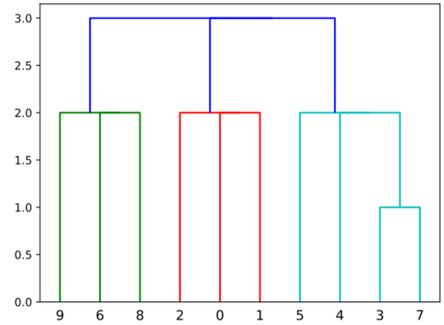


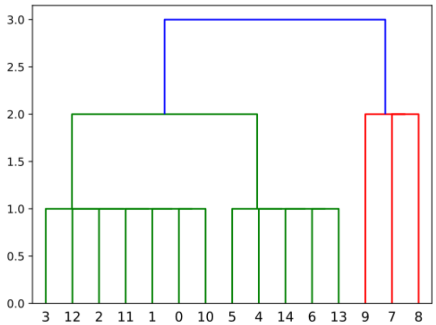
Two-Step Clustering: OPTICS

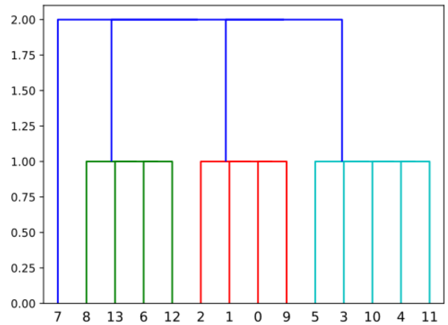


Two-Step Clustering: TTSAS

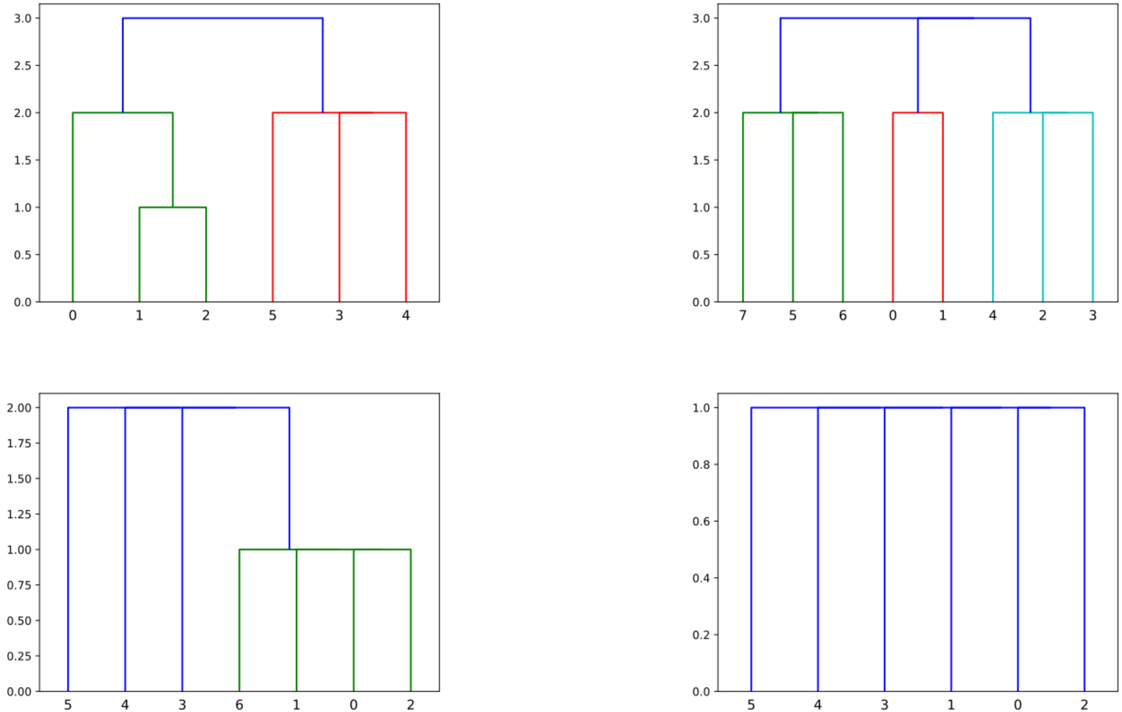




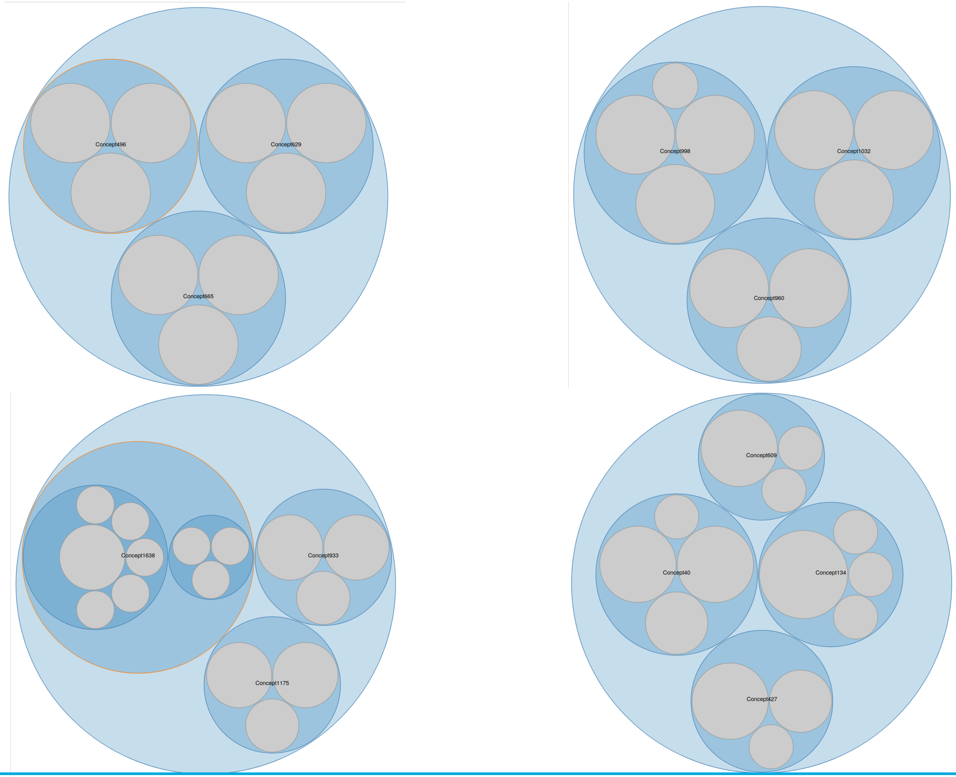




Two-Step Clustering: SOM



Conceptual Clustering: Trestle



Approaches I & II

P1: One merge per level $\Rightarrow$ not a proper tree with levels

P2: Merges are always between two clusters $\Rightarrow$ Flattening

P3: Breaks down immediately when introducing noise

P4: First step clustering algorithms are highly dependent on hyper-parameter tuning:
$\Rightarrow$ Randomized parameter search

Conceptual Clustering

- Outlier detection
- Preprocessing e.g. list flattening
- weighting of attributes (reliable vs. noisy)

What to achieve with thesis

Improve/extend the conceptual clustering framework:

- leverage graph edges/relationships to make algorithm more robust:
    - neighbourhood
    - ego net
    - recursive feature extraction
- deal with outliers
- cut the hierarchy tree in robust only sub-trees

References

E. Jones, T. Oliphant, P. Peterson u. a., *SciPy: Open source scientific tools for Python*, [Online; accessed <today>], 2001. Adresse: http://www.scipy.org/.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot und E. Duchesnay, „Scikit-learn: Machine Learning in Python“, *Journal of Machine Learning Research*, Jg. 12, S. 2825–2830, 2011.

A. Novikov, „PyClustering: Data Mining Library“, *Journal of Open Source Software*, Jg. 4, Nr. 36, S. 1230, 2019. DOI: 10.21105/joss.01230. Adresse: https://doi.org/10.21105/joss.01230.

C. MacLellan, E. Harpstead, V. Aleven und K. Koedinger, „TRESTLE: A Model of Concept Formation in Structured Domains“, *Advances in Cognitive Systems*, Jg. 4, 2016.

M. Pawlik und N. Augsten, „RTED: a robust algorithm for the tree edit distance“, *Proceedings of the VLDB Endowment*, Jg. 5, Nr. 4, S. 334–345, 2011.

——, „Tree edit distance: Robust and memory-efficient“, *Information Systems*, Jg. 56, S. 157–173, 2016.

R. J. Campello, D. Moulavi und J. Sander, „Density-based clustering based on hierarchical density estimates“, in *Pacific-Asia conference on knowledge discovery and data mining*, Springer, 2013, S. 160–172.

L. McInnes, J. Healy und S. Astels, „hdbscan: Hierarchical density based clustering.“, *J. Open Source Software*, Jg. 2, Nr. 11, S. 205, 2017.

K. McKusick und K. Thompson, „Cobweb/3: A portable implementation“, , 1990.

J. H. Gennari, P. Langley und D. Fisher, „Models of incremental concept formation“, *Artificial intelligence*, Jg. 40, Nr. 1-3, S. 11–61, 1989.