COBRA

A Fast and Simple Method for Active Clustering with Pairwise Constraints

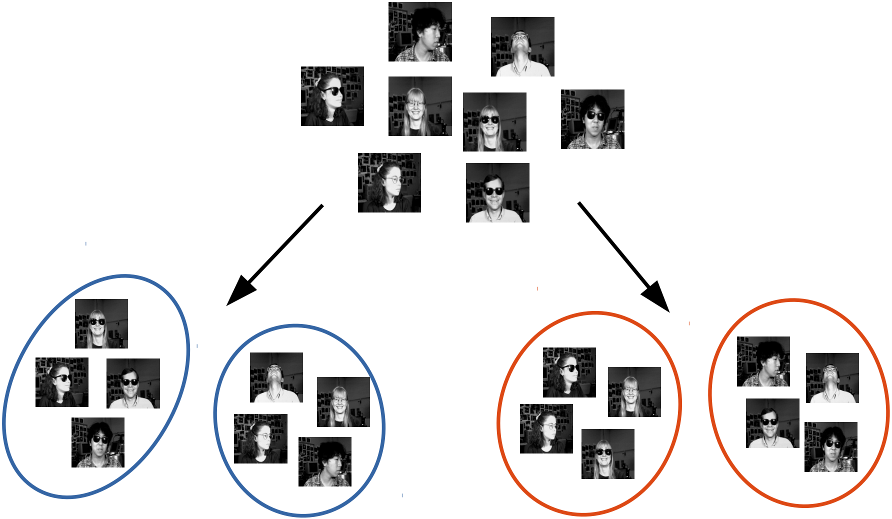
Toon Van Craenendonck, Sebastijan Dumančić and Hendrik Blockeel
{firstname.lastname}@kuleuven.be

Overview

Section 1

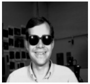
Problem setting

Problem: clustering is inherently subjective



Solution: obtain limited supervision from user

Semi-supervised clustering methods exploit pairwise constraints to produce clusterings that are more aligned with the user's preferences

query: Should  and  be in the same cluster?

We obtain a **must-link** constraint if the answer is yes, a **cannot-link** otherwise

By **actively** selecting informative pairwise queries, we aim to produce a good clustering using as little supervision as possible

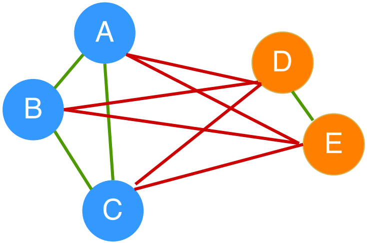
Section 2

COBRA: Constraint-based Repeated Aggregation

Most naive strategy: query all pairwise relations

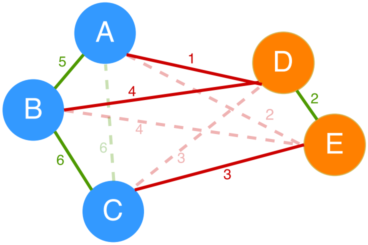
Given all pairwise relations, identifying clusters is trivial ☺

This requires $\binom{n}{2}$ queries ☹



Improvement 1: exploit transitivity and entailment

Query random pairs. Each time a new constraint is obtained, the constraint set is extended by applying entailment and transitivity.

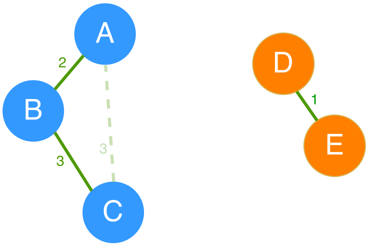


Entailment: $\texttt{cannot-link}(A, D) \wedge \texttt{must-link}(D, E) \Rightarrow \texttt{cannot-link}(A, E)$

Transitivity: $\texttt{must-link}(A, B) \wedge \texttt{must-link}(B, C) \Rightarrow \texttt{must-link}(A, C)$

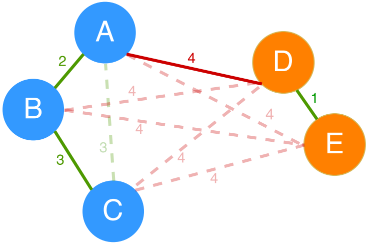
Improvement 2: querying closest pairs first

Query the closest pairs first. Each time a new constraint is obtained, the constraint set is extended by applying entailment and transitivity.



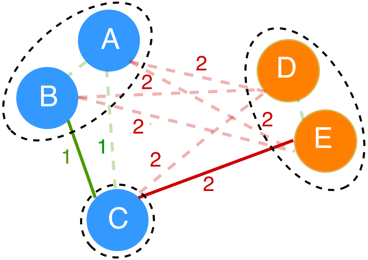
Improvement 2: querying closest pairs first

Query the closest pairs first. Each time a new constraint is obtained, the constraint set is extended by applying entailment and transitivity.



Improvement 3: introduce super-instances

Introduce **super-instances**: small local regions in the data that are assumed to be grouped together in all potential clusterings



COBRA (for Constraint-based Repeated Aggregagation):

1. Construct super-instances
2. Aggregate these super-instances into clusters by repeatedly querying pairwise relations between them

COBRA: Constraint-based Repeated Aggregation

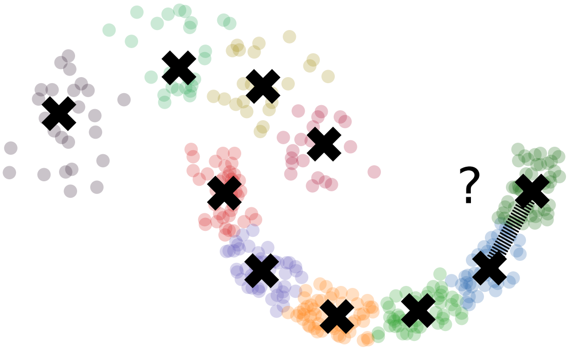
Constraint-based Repeated Aggregation

Require: D: a dataset, N_S: the number of super-instances
Ensure: a clustering of D

Construct N_S super-instances by over-clustering D using K-means
Initially, each (partial) cluster consists of a single super-instance
while the clustering changed **do**
  Let L be the list of all pairs of partial clusters between which the relation is not known yet, sorted by their pairwise distance
  for $P_1, P_2 \in L$ **do**
    Query the relation between partial clusters P_1 and P_2
    if a must-link relation is obtained **then**
      merge P_1 and P_2 into a new partial cluster
      break
    end if
  end for
end while
return the current clustering

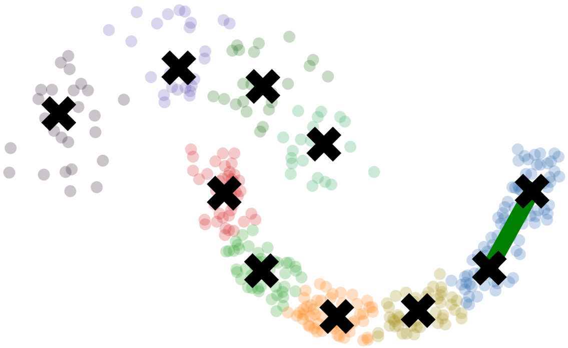
COBRA: Constraint-based Repeated Aggregation

1. Construct super-instances
   - over-cluster the data using K-means
2. Aggregate these super-instances into clusters
   - by querying the pairwise relations between their medoids



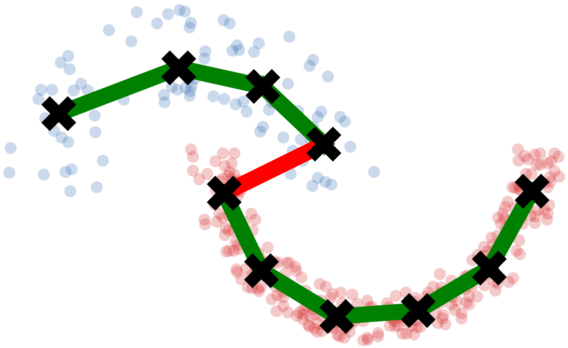
COBRA: Constraint-based Repeated Aggregation

1. Construct super-instances
   - over-cluster the data using K-means
2. Aggregate these super-instances into clusters
   - by querying the pairwise relations between their medoids



COBRA: Constraint-based Repeated Aggregation

1. Construct super-instances
   - over-cluster the data using K-means
2. Aggregate these super-instances into clusters
   - by querying the pairwise relations between their medoids



Section 3

Related work

Existing work on active semi-supervised clustering

- Existing semi-supervised clustering methods
  - modify the clustering objective/procedure of an unsupervised algorithm (e.g. COP-Kmeans, COSC, FOSC-OpticsDend, ...)
  - or learn a metric, which is then used in an unsupervised algorithm (e.g. Xing et al. , ITML, ...)

$\updownarrow$

COBRA is not a direct extension of an existing unsupervised algorithm

- Can use a separate active selection component that is typically based on **uncertainty sampling** (e.g. MinMax, NPU, ...)

$\updownarrow$

COBRA is inherently active

Section 4

Experiments

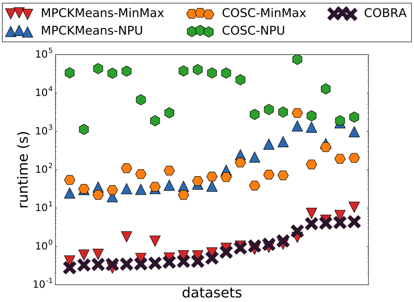
Experiments

- Classification datasets: classes are assumed to represent the clustering of interest
- 21 clustering tasks, evaluated by computing ARI in 5-fold cross-validation
- Comparing COBRA to state-of-the-art competitors MPCKMeans and COSC, both combined with the MinMax (MM) and NPU active selection strategies

Average ranks for quality
(* denotes statistical significance)

25 super-instances		100 super-instances	
COBRA	2.43	COBRA	2.52
MPCK-NPU	3.00	COSC-NPU*	2.98
MPCK-MM	3.07	MPCK-NPU*	3.00
COSC-MM*	3.12	MPCK-MM*	3.19
COSC-NPU*	3.40	COSC-MM*	3.31

Runtimes on 21 clustering tasks



Section 5

Conclusion

Conclusion

We introduce COBRA, a method for **active semi-supervised clustering**.

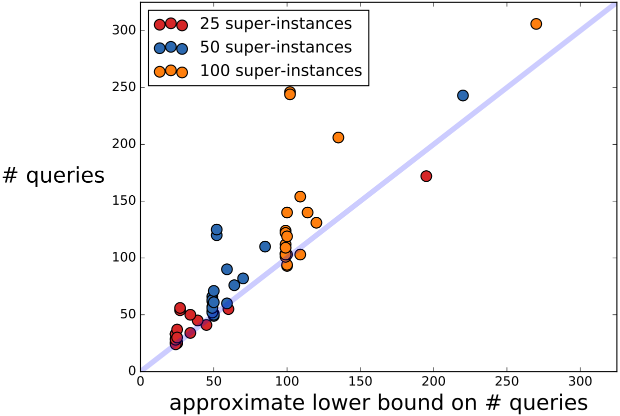
COBRA first over-clusters the data into super-instances, and then merges these super-instances into clusters based on pairwise constraints.

COBRA

+ produces high quality clusterings, compared to competitors
+ is fast, as it relies on a single run of K-means
+ does not require knowing the number of clusters beforehand
- does require setting the number of super-instances
- does not always produce high quality intermediate clusterings

Implementation available at `https://dtai.cs.kuleuven.be/software/cobra/`

Number of queries



Performance for increasing number of super-instances