Extremely randomized trees for clustering complex data

Forêts d'arbres aleatoires pour le clustering de donnés complèxes

AFIA-SFC: Recent advances on unsupervised learning

Kevin Dalleau*, **Miguel Couceiro**, Malika Smail-Tabbone

21 of September, 2021

Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France

* **Project RHU Fight-HF:** `https://anr.fr/ProjetIA-15-RHUS-0004`

Outline

1. Unsupervised Extremely randomized Trees
2. Empirical evaluation
3. Application: Graph clustering
4. Graph-Trees (GT)
5. Experiments
6. Discussion

Unsupervised classification

Unsupervised classification, a.k.a clustering:

- **Goal:** find homogeneous groups of unlabeled instances.
- Active field, with multiple types of approaches: centroid-based (k-means), density-based (DBSCAN), hierarchical clustering (HAC), etc.

Many algorithms rely on a distance metric between instances

- Large number of distances in the literature[1]

[1] M.M. & E. Deza. *Enciclopedia of distances* (3rd edition), Springer, 2014

Motivations and goals

Motivation I. Set of relevant and available distances depends on:

- characteristics of the data: continuous, categorical, ordinal, etc.
- chosen algorithm

Goal: Similarity measure agnostic to data types.

Motivation II. Preprocessing burdern in many practical cases:

- scaling issues
- correlation between variables
- missing values

Goal: Reduce the preprocessing burden.

Motivations and goals

Motivation I. Set of relevant and available distances depends on:

- characteristics of the data: continuous, categorical, ordinal, etc.
- chosen algorithm

Goal: Similarity measure agnostic to data types.

Motivation II. Preprocessing burdern in many practical cases:

- scaling issues
- correlation between variables
- missing values

Goal: Reduce the preprocessing burden.

Unsupervised Extremely randomized Trees

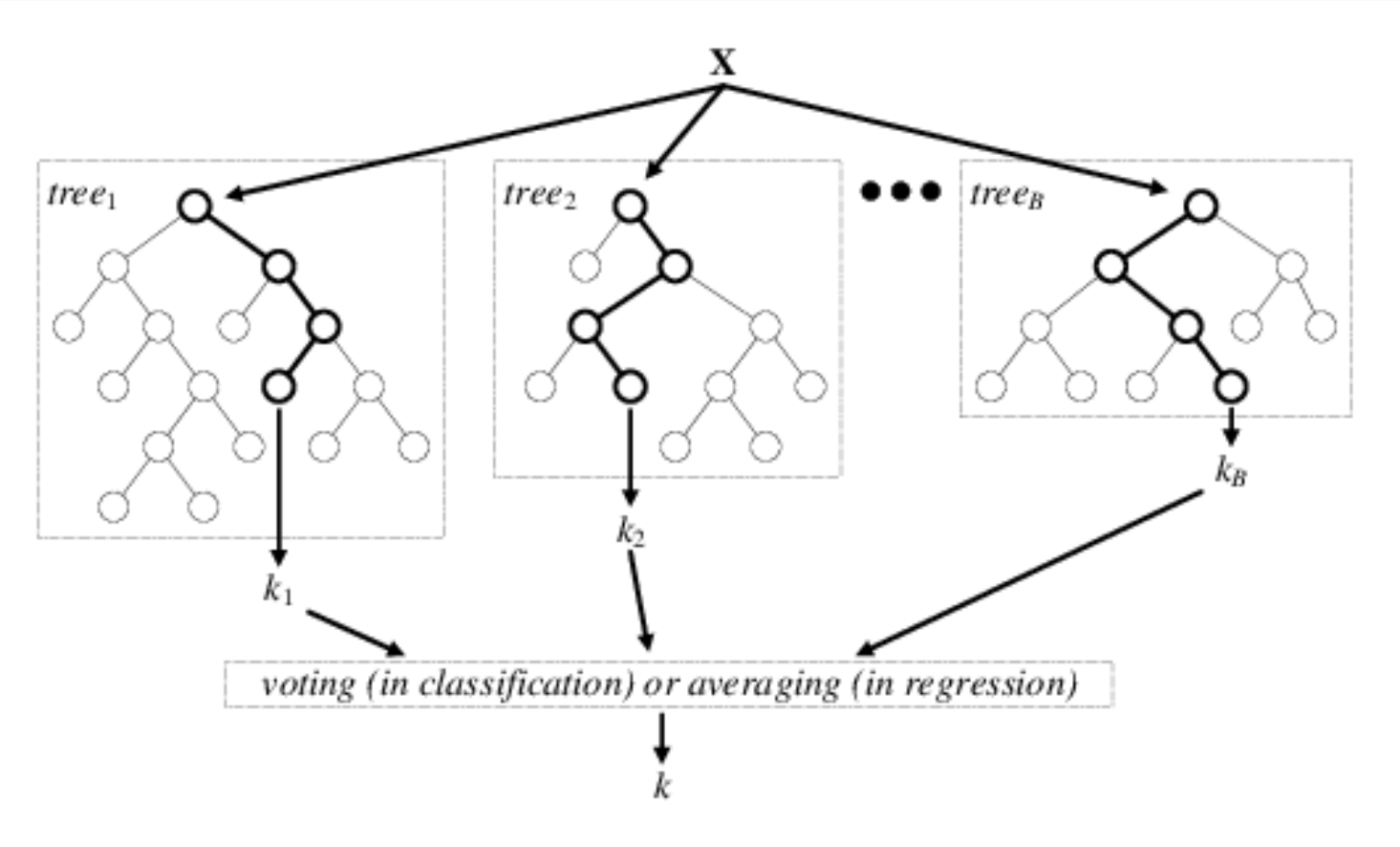
Unsupervised Random Forest (URF)

Shi et al.[2]: method to compute a *similarity* in unsupervised settings.

- Method based on RF: Unsupervised Random Forest (URF).
- RF: popular tree-based algorithm, extensively used.
- Ensemble method, combining decision trees in order to obtain better results in supervised learning tasks.

[2] Unsupervised learning with random forest predictors. Journal of Computational and Graphical Statistics, 15(1):118–138, 2006.

Random Forest



A.Verikas *et al.*, Electromyographic Patterns during Golf Swing: Activation Sequence Profiling and Prediction of Shot Effectiveness, Sensors, 2016.

Unsupervised Random Forest (URF)

Idea: once the forest constructed, run the training data down each tree.

1. All instances in the same leaf are considered similar.
2. Similarity measure: if two instances i and j are in the same leaf of a tree, the overall similarity between the two instances is increased by one.

Normalization: all values lie in $[0, 1]$.

How to build a decision-tree in an unsupervised setting ?
Answer: generation of synthetic instances.

URF-Synthetic instance generation

Two procedures to generate synthetic instances are presented in Shi *et al.*[3]

- addCl1: the synthetic instances are obtained by a random sampling from the observed distributions of variables.
- addCl2: random sampling in the hyper rectangle containing the observed instances.

[3] Unsupervised learning with random forest predictors. Journal of Computational and Graphical Statistics, 15(1):118–138, 2006.

addCl1: an example

Instance	Feature #1	Feature #2
1	5.1	3.5
2	7.0	3.2
3	6.4	2.8

Instance	Feature #1	Feature #2	Label
1	5.1	3.5	1
2	7.0	3.2	1
3	6.4	2.8	1

addCl1: an example

Instance	Feature #1	Feature #2
1	5.1	3.5
2	7.0	3.2
3	6.4	2.8

Instance	#1	Feature #2	Label
1	5.1	3.5	1
2	7.0	3.2	1
3	6.4	2.8	1
4	5.1	3.2	0

addCl1: an example

Instance	Feature #1	Feature #2
1	5.1	3.5
2	7.0	3.2
3	6.4	2.8

Instance	Feature #1	Feature #2	Label
1	5.1	3.5	1
2	7.0	3.2	1
3	6.4	2.8	1
4	5.1	3.2	0
5	6.4	3.5	0

addCI1: an example

Instance	Feature #1	Feature #2
1	5.1	3.5
2	7.0	3.2
3	6.4	2.8

Instance	Feature #1	Feature #2	Label
1	5.1	3.5	1
2	7.0	3.2	1
3	6.4	2.8	1
4	5.1	3.2	0
5	6.4	3.5	0
6	5.1	2.8	0

addCI2: an example

Instance	Feature #1	Feature #2
1	5.1	3.5
2	7.0	3.2
3	6.4	2.8

Feature #1 : $[5.1, 7.0]$
Feature #2 : $[2.8, 3.5]$

Instance	Feature #1	Feature #2	Label
1	5.1	3.5	1
2	7.0	3.2	1
3	6.4	2.8	1
4	5.5	2.9	0
5	6.7	3.1	0
6	5.9	3.4	0

Limitations

Successfully used in fields such as biology or image processing.

However: The method presents some limitations:

- The generation step is not computationally efficient.
- Bias induced by the generated instances.
- It is necessary to construct many forests with different synthetic instances and average their results.

Extremely Randomized Trees (ET)

P.Geurts *et al.*: Extremely Randomized Trees (ET) [4]

- Very similar to RF.
- Another randomization: split threshold selected partially/totally at random

Two important parameters:

1. K, the number of attributes to be randomly selected at each node.
2. n_{min} (*smoothing strength*), the minimum instance size for node split.

[4] Extremely randomized trees. Machine learning, 63(1):3–42, 2006.

Contributions

Following the tracks of Shi *et al.* of URF, we propose to use ET.

- Novel approach where the generation of synthetic cases is not necessary.
- addCl3: a method to generate synthetic labels and associate them to the observed instances.

Result: Unsupervised Extremely randomized Trees (UET)[5]

Randomization: numerical/ordinal **or** categorical variables

[5] K. Dalleau, M. Couceiro, M. Smaïl-Tabbone: Unsupervised Extremely Randomized Trees. PAKDD (3) 2018: 478-489

addCl3: example

Instance	Feature #1	Feature #2
1	5.1	3.5
2	7.0	3.2
3	6.4	2.8

Table 1: addCl3

Instance	Feature #1	Feature #2	Label
1	5.1	3.5	0
2	7.0	3.2	1
3	6.4	2.8	0

Algorithm

Algorithme 1 : Unsupervised Extremely Randomized Trees

Données : Observations O

K, n_{min}, n_{trees}

Résultat : Similarity matrix S

$D \longleftarrow addCl3(O)$;

$T \longleftarrow Build\_an\_extra\_tree\_ensemble(D)$ // *Here* $K = 1$;

$S = 0_{n_{obs}, n_{obs}}$ // *Initialization of a zero matrix of size* n_{obs} ;

pour $d_i \in D$ **faire**

  pour $d_j \in D$ **faire**

    $S_{i,j}$ = number of times the instances d_i and d_j fall in the same leaf node in each tree of T $= \{t_1, t_2, ..., t_M\}$;

  fin

fin

$S_{i,j} = \frac{S_{i,j}}{M}$;

Empirical evaluation

The procedure goes as follows:

1. A similarity matrix is constructed using UET.
2. This similarity matrix is transformed into a dissimilarity matrix using[6]:

$$DIS_{ij} = \sqrt{1 - SIM_{ij}}$$

3. An hierarchical agglomerative clustering (with average linkage) is performed using this distance matrix, with the relevant number of clusters for the labeled dataset.

This procedure is ran 10 times: For each clustering, Adjusted Rand Indices (ARI) are computed, and are compared using the Kruskal-Wallis test.

[6] Shi *et al.* Unsupervised learning with random forest predictors. Journal of Computational and Graphical Statistics, 15(1):118–138, 2006.

Parameter settings

First we evaluate the influence of the parameters on the results of UET:

- The number of trees (averaging strength) n_{trees}.
- The minimum number of instances to split n_{min}.

Dataset	# instances	# features	# labels
Iris	150	4	3
Wine	178	13	3
Wisconsin	699	9	2

Table 2: Properties of used datasets

Parameter tuning: n_{trees} and n_{min}

Observations:

- n_{trees}: no significant diff. in ARI for $n_{trees} > 50$ ($p > 0.1$ for all datasets)
- n_{min}: values between 20% and 30% of the number of instances seems to lead to the best results.
- UET fails with small values of n_{min}

Explanation: larger values of n_{min} are necessary with noisy data[7]

[7] P.Geurts *et al.*: Extremely randomized trees. Machine learning, 63(1):3–42, 2006.

Cluster discrimination

Question: is UET able to discriminate instances from different clusters?

- 3 generated datasets of 1000 instances: two without any cluster structure (*NoC4* and *NoC5*), and one with a cluster structure (*C4*)
- 20 runs of UET: 20 similarity matrices
- Comparison of the mean difference $\bar{\Delta}$ between
  1. the mean intracluster similarity μ_{intra}
  2. the mean intercluster similarity μ_{inter}

Dataset	$\bar{\Delta}$	σ
NoC4	0.00042	0.00003
NoC50	0.00007	0.00003
C4	0.68417	0.00341

Table 3: Mean difference between intercluster and intracluster similarities in different settings, on synthetic datasets.

Cluster discrimination

Question: is UET able to discriminate instances from different clusters?

- 3 generated datasets of 1000 instances: two without any cluster structure (*NoC4* and *NoC5*), and one with a cluster structure (*C4*)
- 20 runs of UET: 20 similarity matrices
- Comparison of the mean difference $\bar{\Delta}$ between
  1. the mean intracluster similarity μ_{intra}
  2. the mean intercluster similarity μ_{inter}

Dataset	$\bar{\Delta}$	σ
NoC4	0.00042	0.00003
NoC50	0.00007	0.00003
C4	0.68417	0.00341

Table 3: Mean difference between intercluster and intracluster similarities in different settings, on synthetic datasets.

Benchmarking

We then assessed UET on benchmark datasets:

- Comparison of Normalized Mutual Information (NMI) scores with the values presented in H. Elghazel *et al.* [8].
- Comparison of ARI obtained with UET and URF.
- UET computed with $n_{trees} = 50$, $n_{min} = \lceil \frac{n_{instances}}{3} \rceil$.

[8] H.Elghazel and A.Aussem, Feature selection for unsupervised learning using random cluster ensembles, Data Mining, 2010

Benchmarking

Dataset	# instances	# features	# labels
Iris	150	4	3
Wine	178	13	3
Wisconsin	699	9	2
Lung	32	56	3
Breast tissue	106	9	6
Isolet	1559	617	26
Pima	768	8	2
Parkinson	195	22	2
Ionosphere	351	34	2
Segmentation	2310	19	7

Table 4: Datasets used for comparison

Benchmarking

Comparative evaluation with the results from Elghazel *et al.* [9].

Dataset	UET - **NMI**	Literature - **NMI**
Wisconsin	**78.33** $\pm$ 3.25	73.61 $\pm$ 0.00
Lung	**29.98** $\pm$ 6.17	22.51 $\pm$ 5.58
Breast tissue	**74.48** $\pm$ 2.92	51.18 $\pm$ 1.38
Isolet	61.22 $\pm$ 1.47	**69.83** $\pm$ 1.74
Parkinson	**25.50** $\pm$ 6.14	**23.35** $\pm$ 0.19
Ionosphere	**13.47** $\pm$ 1.11	**12.62** $\pm$ 2.37
Segmentation	**69.62** $\pm$ 2.14	60.73 $\pm$ 1.71

[9] Feature selection for unsupervised learning using random cluster ensembles, Data Mining (ICDM), 2010

Benchmarking

Dataset	UET (**ARI** - Time (s))	URF (**ARI** - Time (s))
Wisconsin	**87.13** - **128.42 s**	**82.92** - 968.71 s
Lung	**23.24** - **5.23 s**	6.52 - 86.93 s
Breast tissue	**58.85** - **9.15 s**	39.05 - 99.40 s
Isolet	**28.04** - **692.82 s**	* - * s
Parkinson	**25.21** - **16.27 s**	12.68 - 279.30 s
Ionosphere	6.04 - **39.13 s**	7.28 - 727.30 s

Table 5: Comparative evaluation between URF and UET

What about the preprocessing tasks we mentioned earlier ?

Datasets

Here we used two datasets freely available in *Scikit-learn*

- blob500: 500 instances, 5 features and 3 blob shaped clusters
- moon500: 500 instances, 2 features, 2 moon-shaped clusters

Question: robustness to variable transformations & correlations

Robustness to monotone transformation of variables

Why:[10]

- Robustness to change in scales
- Robustness to outliers

Procedure:

- computation of $\bar{\Delta}$ on the original data
- multiplication or addition of n column of the dataset by a scalar (drawn from $\mathcal{U}(2, 100)$)
- computation of new $\bar{\Delta}$

[10] J. Friedman, T. Hastie, and R. Tibshirani. The elements of statistical learning, volume 1. Springer series in statistics New York, 2001.

Robustness to monotone transformation of variables

Operation	Number of variables	$\bar{\Delta}$	σ
Multiplication	0	0.2981	0.0044
Multiplication	1	0.2991	0.0029
Multiplication	2	0.2992	0.0036
Addition	0	0.2987	0.0037
Addition	1	0.2976	0.0045
Addition	2	0.2970	0.0035

Table 6: Influence of a multiplication or addition by a scalar on $\bar{\Delta}$ (moon500)

Resistance to monotone transformation of variables

Operation	Number of variables	$\bar{\Delta}$	σ
Multiplication	0	0.3283	0.0072
Multiplication	1	0.3297	0.0060
Multiplication	2	0.3285	0.0067
Addition	0	0.3250	0.0053
Addition	1	0.3296	0.0046
Addition	2	0.3267	0.0059

Table 7: Influence of a multiplication or addition by a scalar on $\bar{\Delta}$ (blob500)

Behaviour w.r.t correlated variables

Procedure:

- *blob500* dataset
- replacement of each column by a random linear combination of another
- $\bar{\Delta}$ and σ computation.

Bahviour w.r.t correlated variables

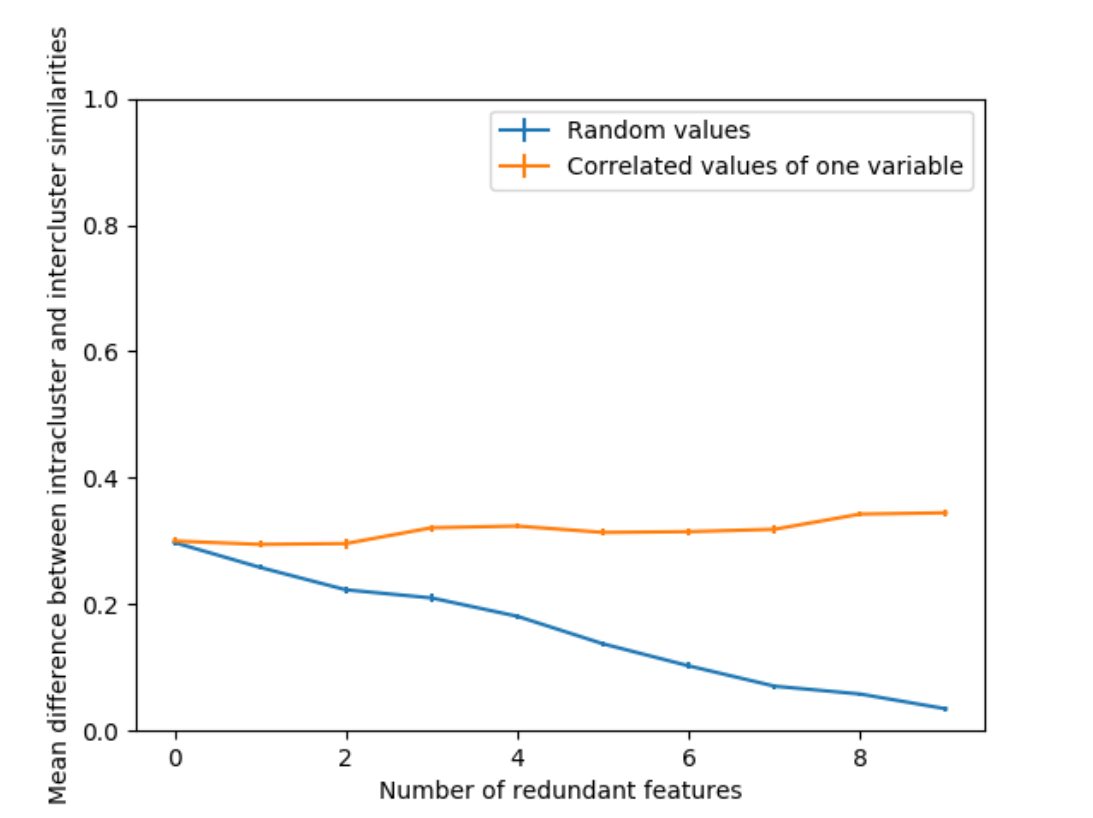


Figure 1: Change of difference between mean intracluster and mean intercluster similarities when (i) changing features by linear combinations of other features and (ii) changing features by random values. The x axis represents the number of features modified by the procedure.

Conclusion

What has been presented:

- A novel stochastic method to compute similarities using decision trees.
- Extension of URF by using extremely randomized trees as a base estimator.
- With no need for instance generation.

Conclusion:

- Essentially one parameter influenced the results: n_{min} (smoothing).
- Explanation: higher values n_{min} give better results under noise.

Advantages and perspectives

Advantages of UET:

1. Synthetic data generation is no longer necessary.
2. 1.5 to more than 10 times faster than URF in our experiments.
3. Adaptability to complex data: **attributed graphs**

Application: Graph clustering

Graph clustering

What is a graph ?

- $G = (V, E)$, V set of vertices and E a set of edges (pairs of vertices).
- Graphs can be attributed: vertices/edges endowed with an attribute tuple.

Goal of graph clustering:

- Two types: *between* and *within* graphs.
- Within graphs: find a partition of sets of *related* vertices in a graph.
- Related: connected by many edges w.r.t. vertices from other clusters.
- **Vertex-attributed graphs:** attribute homogeneity taken into account

Application of UET?

- A tree-based method for computing vertex (dis)similarities.
- Bridging the gap between random decision trees and graph clustering.
- Handles vertex attributes by building forests with different tree types.

Graph clustering

What is a graph ?

- $G = (V, E)$, V set of vertices and E a set of edges (pairs of vertices).
- Graphs can be attributed: vertices/edges endowed with an attribute tuple.

Goal of graph clustering:

- Two types: *between* and *within* graphs.
- Within graphs: find a partition of sets of *related* vertices in a graph.
- Related: connected by many edges w.r.t. vertices from other clusters.
- **Vertex-attributed graphs:** attribute homogeneity taken into account

Application of UET?

- A tree-based method for computing vertex (dis)similarities.
- Bridging the gap between random decision trees and graph clustering.
- Handles vertex attributes by building forests with different tree types.

Graph-Trees (GT)

Tree-based distances: field with recent interesting developments.

- Shi *et al.* [11]: method to compute distances between samples in unsupervised settings.
- Dalleau *et al.* [12]: extension using Extremely Randomized Trees, with better performance.
- **Ting et al.** [13]: mass-based distance using isolation forests.

[11] Unsupervised learning with random forest predictors. Journal of Computational and Graphical Statistics, 2006.
[12] Unsupervised Extra Trees: a stochastic approach to compute similarities in heterogeneous data. International Journal of Data Science and Analytics, Springer Verlag, 2020
[13] Overcoming key weaknesses of distance-based neighbourhood methods using a data dependent dissimilarity measure. Proceedings of the 22nd ACM SIGKDD (2016)

idea: use a hierarchical partitioning of the original space into non-overlapping and non-empty regions H_i's

- $R(x, y|H_i)$ be the smallest local region covering x and y w.r.t. H.

Mass-based dissimilarity: estimated by a number t of models is

$$m_e(x, y|D) = \frac{1}{t}\sum_{i=1}^{t} \tilde{P}(R(x, y|H_i))$$

where $\tilde{P}(R) = \frac{1}{|D|}\sum_{z\in D} 1(z \in R)$.

Example: Tree-based mass-estimation

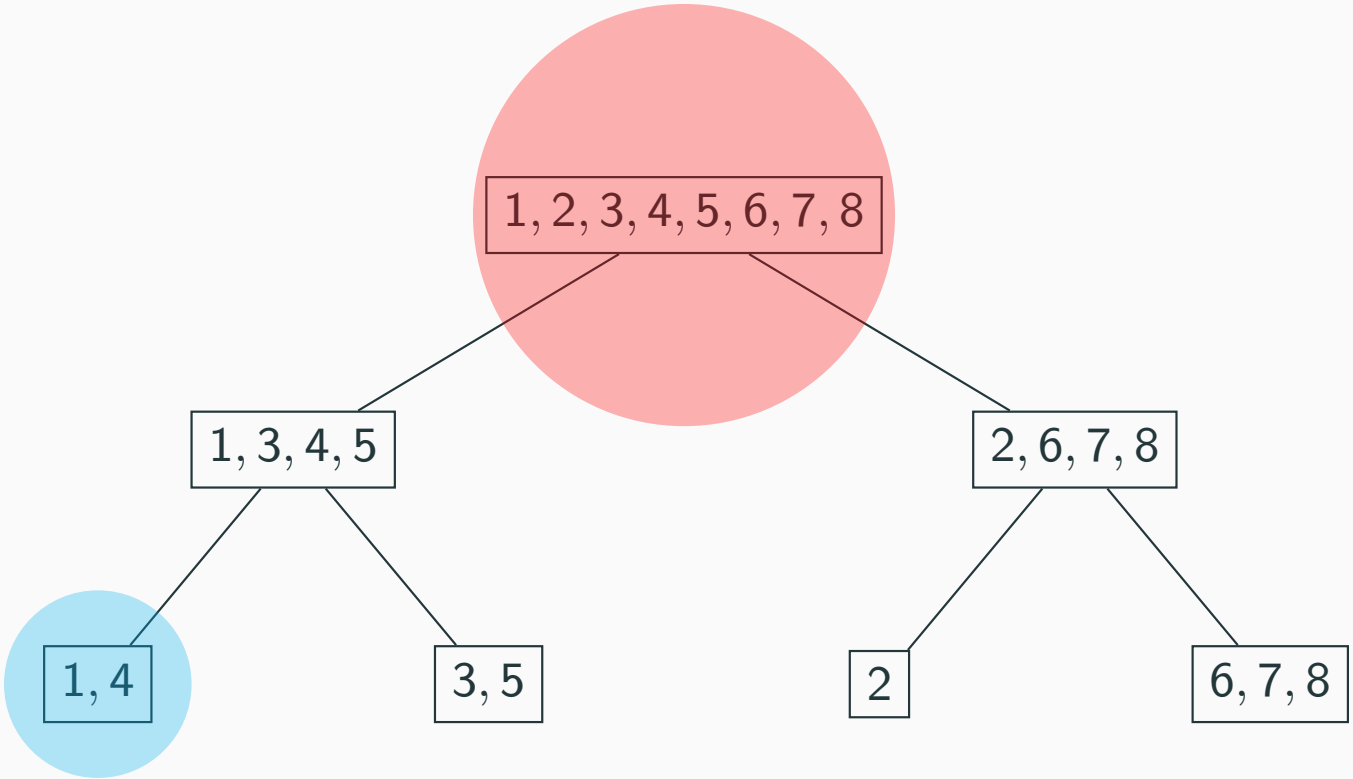


Figure 2: Ex. of partitioning of 8 instances in non-overlapping non-empty regions using a random tree structure: $m_e(1,4) = \frac{1}{8}(2) = 0.25$, and $m_e(1,8) = \frac{1}{8}(8) = 1$.

Proposed approach: Graph Trees

Idea of Graph Trees (GT)[14]:

1. Compute several partitions of the vertices using random trees,
2. Compute a dissimilarity measure between the vertices using the partitions.

How are the partitions of vertices obtained?

- The root node of each tree contains all the vertices of the graph.
- At each node, a split is performed. Split:
  1. A vertex v_1 is randomly sampled from that node
  2. Each vertex v_k that share an edge with v_1 form the left child node
  3. While all other vertices from the parent node form the right child node
- The growth is stopped when a stopping criterion is met.

[14] `https://gitlab.inria.fr/kdalleau/graphtrees/`

Proposed approach: Graph Trees

Idea of Graph Trees (GT)[14]:

1. Compute several partitions of the vertices using random trees,
2. Compute a dissimilarity measure between the vertices using the partitions.

How are the partitions of vertices obtained?

- The root node of each tree contains all the vertices of the graph.
- At each node, a split is performed. Split:
  1. A vertex v_1 is randomly sampled from that node
  2. Each vertex v_k that share an edge with v_1 form the left child node
  3. While all other vertices from the parent node form the right child node
- The growth is stopped when a stopping criterion is met.

[14] `https://gitlab.inria.fr/kdalleau/graphtrees/`

Graph forests

It is possible to build forests with different types of trees (*Graph forests*):

1. Graph trees that specialize on the graph structure
2. Trees that specialize on the attribute space.

In our case: Unsupervised Extremely randomized Trees (UET).

Aggregation of the (dis)similarities obtained with the different types of trees.

Experiments

Experiments on simple graphs

First evaluation: simple graphs with no attributes

1. Distance matrices using GT, with $n_{trees} = 200$
2. k-means on the points obtained using t-SNE [15] on the distance matrix
3. $\rightarrow$ NMI [16]

The process repeated 20 times.

[15] t-Distributed Stochastic Neighbor Embedding
[16] Normalized Mutual Information

Experiments on simple graphs

Dataset	# vertices	# edges	Average degree	# clusters
Football	115	1226	10.66	10
Email-Eu-Core	1005	25571	33.24	42
Polbooks	105	441	8.40	3
SBM3	450	65994	293.307	3

Table 8: Datasets used for the evaluation of GT clustering on simple graphs

Dataset	Graph-trees	Louvain[17]	MCL[18]
Football	**0.923 (0.007)**	**0.924 (0.000)**	0.879 (0.015)
Email-Eu-Core	**0.649 (0.008)**	0.428 (0.000)	0.589 (0.012)
Polbooks	0.524 (0.012)	0.521 (0.000)	**0.544 (0.02)**
SBM3	**0.998 (0.005)**	0.684 (0.000)	0.846 (0.000)

Table 9: Comparison of NMI on benchmark graph datasets. Best in boldface

[17] Blondel *et al.*. Fast unfolding of communities in large networks. J. statistical mechanics : theory and experiment, 2008(10) :P10008, 2008

[18] Markov Cluster Algo. S. M. Van Dongen.Graph clustering by flow simulation. PhD thesis, 2000.

Experiments on attributed graphs[19] [20]

Dataset	# vertices	# edges	# attributes	# clusters
Parliament	451	11646	108	7
HVR	307	6526	6	2
Lawyers	71	575	70	2
WebKB	877	1480	1703	4

Table 10: Datasets used for the evaluation of GT clustering on attributed graphs

Dataset	NMI GT+UET	NMI Literature
HVR	**1.00 (0.000)**	0.89
Parliament	0.65 (0.039)	**0.78**
Lawyers	0.12	**0.66**
WebKB	**0.999 (0,002)**	0.995 (0,002)

Table 11: Comparison of clusterings using GT. Best results from Bojchevski *et al.*, and Maekawa *et al.* on WebKB. Best results are indicated in boldface.

[19] Bojchevski *et al.* Bayesian Robust Attributed Graph Clustering: Joint Learning of Partial Anomalies and Group Structure, 2018

[20] Maekawa *et al.*: Non-linear Attributed Graph Clustering by Symmetric NMF with PU Learn.2018

Experiments on attributed graphs

Dataset	GT	GT+UET	Ground truth
HVR	**0.15**	**0.15**	**0.15**
Parliament	**0.46**	0.15	0.20
Lawyers	**0.27**	0.23	**0.26**
WebKB	**0.74**	0.70	**0.74**

Table 12: Results using the dissimilarities from UET and the labels (ground truth). Best results are indicated in boldface.

Discussion

Discussion

- Method based on the construction of random trees to compute similarities between graph vertices.
- Competitive with state of the art methods in terms of quality of clustering on non-attributed graphs.
- Computing forests of GT and other trees that specialize in other types of input data: possible to compute dissimilarities between vertices in attributed graphs.

Discussion

Graph forests using UET for the attribute trees seems promising:

- Less preprocessing, can manage mixed types attributes *out of the box*.
- **Some control:** importance of the vertex attributes, choice of aggregation method between the graph trees and the attribute trees
- **Real life application:** Project RHU Fight-HF[21] [22]

However

- **Empirical evaluation:** quality that varies greatly between the datasets.
- Choice to consider the attribute space: guided by the distribution of the variables or a visualization of the embeddings ?

[21] `https://anr.fr/ProjetIA-15-RHUS-0004`

[22] Preud'Homme *et al.* Head-to-head comparison of clustering methods for heterogeneous data: a simulation-driven benchmark. Scientific Reports, Nature Publishing Group, 2021, 11 (1), pp.4202.

Merci de votre attention!

Questions?