



Semi-supervised Consensus Clustering Based on Closed Patterns

Tianshu Yang, Nicolas Pasquier, Frédéric Precioso

► To cite this version:

Tianshu Yang, Nicolas Pasquier, Frédéric Precioso. Semi-supervised Consensus Clustering Based on Closed Patterns. Knowledge-Based Systems, 2022, Article 107599, 235, 10.1016/j.knosys.2021.107599 . hal-03402517

HAL Id: hal-03402517

<https://hal.science/hal-03402517>

Submitted on 5 Jan 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial 4.0 International License

Semi-supervised Consensus Clustering Based on Closed Patterns

Tianshu YANG^{a,b}, Nicolas PASQUIER^a and Frederic PRECIOSO^a

^aUniversité Côte d'Azur, CNRS, I3S, Sophia Antipolis, France

^bAmadeus, Sophia Antipolis, France

ARTICLE INFO

Keywords:

Semi-supervised learning
Semi-supervised clustering
Semi-supervised consensus clustering
Semi-supervised ensemble clustering
Frequent closed itemsets
Closed patterns


ABSTRACT

Semi-supervised consensus clustering, also called semi-supervised ensemble clustering, is a recently emerged technique that integrates prior knowledge into consensus clustering in order to improve the quality of the clustering result. In this article, we propose a novel semi-supervised consensus clustering algorithm extending the previous work on the MultiCons multiple consensus clustering approach. By using closed pattern mining technique, the proposed Semi-MultiCons algorithm manages to generate a recommended consensus solution with a relevant inferred number of clusters k based on ensemble members with different k and pairwise constraints. Compared with other semi-supervised and/or consensus clustering approaches, Semi-MultiCons does not require the number of generated clusters k as an input parameter, and is able to alleviate the widely reported negative effect related to the integration of constraints into clustering. The experimental results demonstrate that the proposed method outperforms state of the art semi-supervised consensus clustering algorithms.

1. Introduction

Clustering is the process of grouping data into partitions so that similar instances are most likely to be placed in the same partition [22]. Many clustering algorithms were proposed regards to different applications and data properties [1, 47, 68]. Each one of them has its own strengths as well as its own limitations [20, 30]. Therefore, more and more researchers focus on consensus clustering, also called clustering ensemble, since it provides a more stable, robust and accurate performance than any single clustering algorithm [9, 65]. Consensus clustering, also referred to as clustering ensemble, aims to enhance the clustering process by combining multiple clustering results generated from different clustering algorithms with different parameterizations [42, 55, 58]. However, as an unsupervised learning method, consensus clustering approaches cannot make use of supervision information, even if sometimes such information is available for some instances. This leads to the emergence of semi-supervised consensus clustering, also referred to as semi-supervised clustering ensemble, which aims at using such a prior knowledge in the consensus clustering to improve the quality of the final clustering.

In this paper, we propose a novel semi-supervised consensus clustering algorithm, named Semi-MultiCons, extending the previous work on the MultiCons consensus clustering approach [3]. Like described in [3], Semi-MultiCons uses closed pattern mining technique, based on frequent closed itemset mining technique, to generate multiple consensus clustering solutions with inferred k . Meanwhile, we developed a novel consensus function based on pairwise constraints and a strategy to select the most appropriate inferred k . We also optimized the implementation of [3] to reduce the time and space complexities of the closed pattern processing by the approach.

 yang@i3s.unice.fr (T. YANG); pasquier@i3s.unice.fr (N. PASQUIER); precioso@i3s.unice.fr (F. PRECIOSO)
ORCID(s):

The remaining of the paper is organized as follows. In Section 2, we report on related work on semi-supervised clustering, consensus clustering and semi-supervised consensus clustering. In Section 3, we present preliminary notions about prior knowledge representation and the MultiCons approach. Section 4 introduces and explains the novel Semi-MultiCons method. Section 5 presents and analyzes the experimental results. The conclusions and future work are stated in Section 6.

2. Related work

Many algorithms, based on different principles and formalisms for grouping instances of a dataset into clusters according to their similarity, have been proposed in the literature. The most recent approaches developed in this domain presented in this section intent to optimizing this process. Semi-supervised clustering methods aim at integrating available initial prior knowledge about the relevance of grouping or separating certain pairs of instances. Consensus clustering methods aim to combine the results of different clustering algorithms, based on different approaches, to generate a final clustering based on the most frequent aggregations between the initial clusterings. The semi-supervised consensus clustering approach combines these two methods in order to optimize the clustering result in terms of robustness and relevance of the result, depending on the initial information available in the application domain.

2.1. Semi-supervised clustering

Semi-supervised clustering incorporates prior knowledge such as class labels or pairwise constraints into traditional clustering methods to obtain better quality result [5, 46]. Semi-supervised clustering approaches can be classified into three types [6]: Constraint-based methods, distance-based methods and hybrid methods.

Constraint-based methods refer to algorithms that utilize supervision information to restrict the feasible solutions

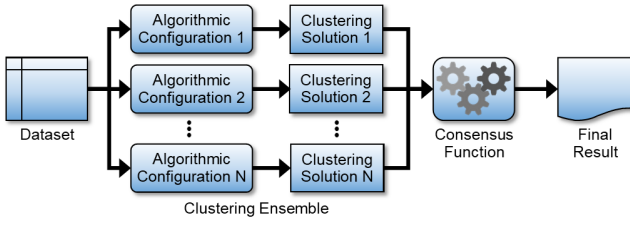


Figure 1: Overview of consensus clustering procedure. Different clustering algorithmic configurations are applied to the input dataset for creating the clustering ensemble to which is applied the consensus function for generating the final clustering result.

when assigning data to clusters, either directly by changing assignment strategy to prevent assignments that violate supervision information [60], or indirectly by penalizing and/or rewarding objective function if supervision information is violated and/or satisfied [13, 45]. With distance-based methods, supervision information is applied in distance learning. This distance can be a distance in the original data space [67], distance in low dimension feature space [52] or even kernel distance matrix [24]. Hybrid methods combine constraint-based methods and distance-based methods [6].

Semi-supervised clustering takes advantage of supervised information to improve the performance and guide the search to meet user preferences and domain knowledge. However, since most semi-supervised algorithms are iterative and sensitive to input order of data, they are less stable and robust.

2.2. Consensus clustering

Consensus clustering aims to address the limitations of single clustering approaches and to improve the robustness and quality of result by combining multiple clustering solutions generated from different clustering algorithmic configurations, that is different algorithms with different parameterizations, [55]. The problem can be described as follows. Let m denotes the number of clustering solutions. Φ represents the result sets of clustering solutions $\Phi = \{\gamma^1, \gamma^2, \dots, \gamma^m\}$, where $\gamma^i = \{C_1^i, C_2^i, \dots, C_k^i\}$ is the i^{th} clustering result with k clusters. C_j^i denotes the j^{th} cluster of the i^{th} solution. The goal is to find a consensus partition P which better reflects the relevant properties of each solution in Φ .

In general, consensus clustering methods consist of two stages. At the first stage, different clustering solutions Φ are generated from the same dataset. Then, in the second stage, a consensus function is applied on Φ to find the final consensus clustering [10]. Figure 1 shows an overview of the consensus clustering process.

There are various ensemble generation strategies and consensus function approaches. A clustering ensemble can be achieved by using different clustering algorithms, by random sub-spacing/sampling initial data [40], by using different number of clusters k parameter values, by repeated executions of a single clustering algorithm with several parameter sets [27], or by using any combination of

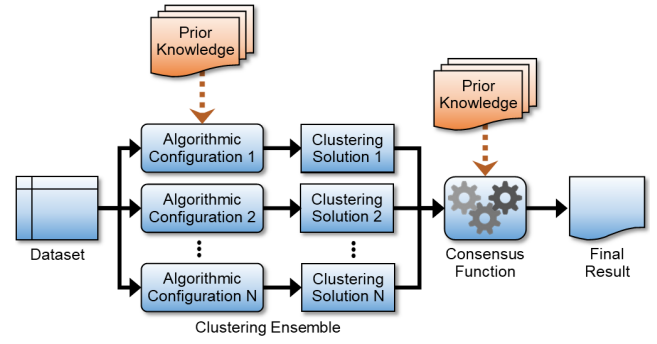


Figure 2: Overview of semi-supervised consensus clustering procedure. supervision information is used as prior knowledge either or both during the clustering ensemble creation and the generation of the final clustering result by the consensus function.

the aforementioned methods. Different consensus functions have been developed to combine the base clustering results, such as mixture models [54], voting-based approach [4], mutual information [53], co-association based approach [19], graph and hyper-graph based approaches [51] and genetic algorithms [34] among others.

Consensus clustering is proved to outperform single clustering method on stability and accuracy [56, 64]. However, as an unsupervised learning method, it is not designed to use any supervision information, even though sometimes a such information is available.

2.3. Semi-supervised consensus clustering

Considering the limitations of semi-supervised clustering and consensus clustering, it is natural to combine them, and thus semi-supervised consensus clustering emerged. Semi-supervised consensus clustering not only considers supervision information, but also integrates multiple clustering results into a unified consensus solution to improve the quality, stability and robustness of the final result [72].

Supervision information can be used in both steps of consensus clustering, as shown in Figure 2. It can be used in base clustering generation, which means, replace unsupervised clustering method with semi-supervised clustering method. In the consensus step, prior knowledge can be integrated in the consensus function to lead the consensus process. For example, instead of using all generated clusters and treating each of them without difference, the existence of supervision information makes it possible to assign different weight to clusters. Usually, clusters that violate supervision information are eliminated or assigned a lower weight.

Several semi-supervised consensus clustering approaches have been proposed during the last few years. In [74], prior knowledge is used to evaluate the quality of each base spectral clustering result. A confidence matrix of each cluster is then constructed based on it and is used with spectral clustering as consensus function. In [28], voting based consensus clustering is extended to semi-supervised context by replacing unsupervised base clusterings with semi-supervised base clusterings. Users are required to provide weights for each

semi-supervised clustering algorithm and for each cluster. These weights are engaged in a voting based consensus function to produce the final clustering. In [71], an improved COP-Kmeans (Constraint Partitioning K-means) algorithm [60] is proposed as base semi-supervised clustering method, and a new constrained self-organizing map approach is implemented as consensus function. In [64], a semi-supervised spectral clustering approach is used for base clusterings, while the authors' graph based consensus function, named Hybrid Bipartite Graph Function [18], remains unsupervised. In [39], a genetic algorithm based consensus function is extended by taking supervision information into consideration in the fitness function. In [73], cluster ensemble members are generated by a constraint propagation algorithm [33], which is a semi-supervised method. Prior knowledge is also used to evaluate and eliminate ensemble members, and only a subset of these ensemble members are taken into account in a graph based consensus function. In [66], prior knowledge is integrated in both the CHAMELEON unsupervised hierarchical clustering algorithm [30] to improve its result and in the co-association matrix used by the consensus function. In [72], instead of using all prior knowledge, different subsets of prior knowledge are assigned each to different ensemble members. Then, an adaptive weighting process associates each ensemble member with its weight, and the weighted normalized cut algorithm, that is a graph based consensus function, is adopted to generate the final result. A summary about this related work is shown in Table 1. Besides these approaches, recent reported methods seek to apply semi-supervised consensus learning to deep learning domain on various types of data and in many tasks. This is, for example, the approach taken in self-ensembling teacher-student method on graph-structured data [35] and co-training method in semantic segmentation task [36, 37].

In this paper, we propose a novel semi-supervised consensus clustering approach that integrates supervised information as pairwise constraints, similarly to the ideas in [71, 73], in both base clustering generation step and consensus function execution, like [28, 66, 71, 72, 73]. Unlike the other approaches which use base clusterings to estimate the relationships among instances, such as co-association matrix [66] or normalized cut graph [72, 73] based approaches, the Semi-MultiCons approach constructs a binary membership matrix, which eventually corresponds to a bipartite graph, as described in [64], representing the relationships between instances and ensemble clusters. The output of the Semi-MultiCons approach is a hierarchical structure similar to [66].

3. Preliminaries

The Semi-MultiCons approach relies on the use of partial background knowledge, represented as constraints of co-assignment to a cluster or of assignment to different clusters concerning a certain number of instances of the dataset, and the extension of the MultiCons approach for multiple

consensus clusterings based on the Galois closed set theory. These two central concepts are presented in this section.

3.1. Pairwise constraints

The prior knowledge integrated in semi-supervised clustering is also called constraints since this knowledge can be seen as constraints on how data should be grouped during the clustering process. Many types of constraints exist, including partial label constraints, pairwise constraints, capacity constraints [62], while the most popular one is pairwise constraints which describe true similarity between pairs of instances. Common pairwise constraints include must-link constraints and cannot-link constraints [59]:

- **Must-link constraints** imply that two instances must be assigned to the same cluster, or more generally, they are more likely to be similar with each other.
- **Cannot-link constraints** imply that two instances can not be assigned to the same cluster, or more generally, they are more likely to be dissimilar with each other.

Must-link constraints and cannot-link constraints are widely used in semi-supervised clustering algorithms [48, 59], in metric learning [63], in dimensionality reduction [11] and many other domains. In this paper, prior knowledge is provided in the form of must-link and cannot-link constraints.

3.2. MultiCons Approach

The MultiCons approach for multiple consensus clustering is presented in [2]. This approach is based on the frequent closed itemset framework [43] to efficiently discover closed patterns among the different base clustering solutions. Each closed pattern defines an agreement between the base clusterings in grouping a set of dataset instances. By dividing/merging these patterns into groups, MultiCons generates multiple consensus in a tree-like structure that helps understanding the clustering process and the data space subjacent natural structures.

In the following subsections, the MultiCons approach is described step by step and illustrated using a dataset $X = \{x_1, x_2, \dots, x_9\}$ of nine instances as a support example.

3.2.1. Ensemble members

The clustering ensemble, consisting of several base clusterings, defines the search space in input of the approach. Different clustering algorithms with different parameter settings, defining each an algorithmic configuration, are used in this step to ensure diversity in the initial clustering solutions that will be combined by the consensus function. This diversity allows to discover intrinsic clusters in the data space, defined by the analyzed dataset, corresponding to different subspace structures. These data subspace structures can be discovered by different algorithmic approaches, such as centroid based, density based or hierarchical based clustering approaches for instance, with different parameter settings.

As a support example of the MultiCons process, we consider the five base clusterings in the clustering ensemble shown in Table 2 generated by applying five unsupervised clustering algorithms to dataset X .

Table 1

A summary on semi-supervised consensus clustering methods. For each referred method, the approach used for generating base clusterings and in the consensus function, as well as the steps involving the use of the supervision information are described.

Method	Base clustering	Consensus function	Steps involving supervision information
[74]	Spectral clustering	Spectral clustering	Consensus step
[28]	Semi-supervised clustering	Voting based approach	Generation step and consensus step
[71]	Improved COP-Means	COP-SOM	Generation step and consensus step
[64]	Semi-supervised spectral clustering	Hybrid Bipartite Graph Function	Generation step
[39]	Unsupervised clustering	Genetic algorithm	Consensus step
[73]	Constraint propagation	Normalized cut	Generation step and consensus step
[66]	Improved CHAMELEON	Co-association based approach	Generation step and consensus step
[72]	Constraint propagation	Weighted normalized cut	Generation step and consensus step

Table 2

Example of clustering ensemble members. Each of the five base clustering results γ_i , for i between 1 and 5, is represented by the instance sets of its clusters C_i^j .

Base clustering	List of instance sets of base clusters		
γ_1	$C_1^1 = \{x_1, x_2, x_3\}$	$C_1^2 = \{x_4, x_5, x_6, x_7, x_8, x_9\}$	
γ_2	$C_2^1 = \{x_1, x_2, x_3\}$	$C_2^2 = \{x_4, x_5, x_6, x_7, x_8, x_9\}$	
γ_3	$C_3^1 = \{x_1, x_2, x_3, x_4, x_5\}$	$C_3^2 = \{x_6, x_7\}$	$C_3^3 = \{x_8, x_9\}$
γ_4	$C_4^1 = \{x_1, x_2, x_3\}$	$C_4^2 = \{x_4, x_5, x_6, x_7\}$	$C_4^3 = \{x_8, x_9\}$
γ_5	$C_5^1 = \{x_1, x_2, x_3\}$	$C_5^2 = \{x_4, x_5, x_6, x_7\}$	$C_5^3 = \{x_8, x_9\}$

3.2.2. Binary membership matrix transformation

The representation of the clustering ensemble as a binary membership matrix aims at optimizing the efficiency of closed pattern mining from the ensemble. Each cluster of the base clusterings in the ensemble is then represented as a binary vector depicting the set of instances assigned to the cluster.

The binary membership matrix M for the support example in Table 2 is shown in Table 3. It represents relationships between instances and clusters in the clustering ensemble: Rows represent the finite set of instances and columns represent the finite set of clusters. A cell value $M[i, c] = 1$ in the i^{th} row and c^{th} column denotes that instance i belongs to cluster c , $M[i, c] = 0$ otherwise.

3.2.3. Closed pattern extraction

Closed patterns in a binary matrix, also called *Galois closed concepts*, each consist of two components: Its *extension* that is a closed set of rows, i.e., instances, and its *intension* that is a closed set of columns, i.e., attributes, of the binary matrix. These closed sets are defined according to two functions:

- The $f(R)$ function associates with a set of rows R all the columns related to every row $r \in R$ in the matrix.
- The $g(C)$ function associates with a set of columns C all the rows related to every column $c \in C$ in the matrix.

The couple of applications (f, g) is a *Galois connection*, and their compositions $h(C) = f(g(C))$ and $h'(R) = g(f(R))$

are *Galois closure operators*. A set of columns C is closed if $h(C) = C$ and a set of rows R is closed if $h'(R) = R$. The $h(C)$ and $h'(R)$ operators generate extensions and intensions, respectively, of closed patterns when applied to the binary matrix. A closed pattern thus represents a maximal, according to inclusion relation, set of columns related to a maximal set of rows in the matrix, that is all the columns in the closed pattern intension and all the rows in its extension are related in the binary matrix.

For efficiently extracting all closed patterns from large data matrices, since the number of columns is usually much lower than the number of rows in the dataset, closed pattern extensions are identified in the matrix using the $h(C)$ closure function. The corresponding intensions are implicitly generated during the calculation of the $h(C)$ closure from the matrix [41]. The search space of the algorithm is then the subset lattice of columns, where nodes represent column sets and edges between them represent inclusion relationships. In the subset lattice, each level contains all column sets of a given size, and the algorithm performs an iterative level-wise bottom-up traversal of the lattice, calculating the h closure of column sets for a level at each iteration [70]. The closure $h(C)$ of a column set C is computed by intersection of all columns $c \in C$.

Closed patterns are extracted from the binary membership matrix using the *apriori()* function of the *arules* R package [23] with the “closed frequent itemsets” parameter. A closed pattern is then a pair consisting of a cluster set and an instance set, such that all instances in the instance set belong to all clusters in the cluster set. Each denotes a

Table 3

Example binary membership matrix M . Each clusters C_i^j of clustering result γ_i in Table 2 is represented as a vertical binary vector indicating for each instance x_k , with k between 1 and 9, if it was assigned to the cluster by a value of 1, or not assigned to the cluster by a value of 0.

Instance	C_1^1	C_1^2	C_2^1	C_2^2	C_3^1	C_3^2	C_3^3	C_4^1	C_4^2	C_4^3	C_5^1	C_5^2	C_5^3
x_1	1	0	1	0	1	0	0	1	0	0	1	0	0
x_2	1	0	1	0	1	0	0	1	0	0	1	0	0
x_3	1	0	1	0	1	0	0	1	0	0	1	0	0
x_4	0	1	0	1	1	0	0	0	1	0	0	1	0
x_5	0	1	0	1	1	0	0	0	1	0	0	1	0
x_6	0	1	0	1	0	1	0	0	1	0	0	1	0
x_7	0	1	0	1	0	1	0	0	1	0	0	1	0
x_8	0	1	0	1	0	0	1	0	0	1	0	0	1
x_9	0	1	0	1	0	0	1	0	0	1	0	0	1

Table 4

Closed patterns extracted from example binary membership matrix M . The seven closed patterns extracted from the binary membership matrix in Table 3 are ordered in decreasing order of the size of their Cluster set L_i containing the list of clusters C_i^j that agree to group together the list of instances x_k in the Instance set.

Length ($ L_i $)	Cluster set L_i	Instance set
5	$\{C_1^2, C_2^2, C_3^1, C_4^2, C_5^2\}$	$\{x_4, x_5\}$
5	$\{C_1^2, C_2^2, C_3^2, C_4^2, C_5^2\}$	$\{x_6, x_7\}$
5	$\{C_1^2, C_2^2, C_3^3, C_4^3, C_5^3\}$	$\{x_8, x_9\}$
5	$\{C_1^1, C_2^1, C_3^1, C_4^1, C_5^1\}$	$\{x_1, x_2, x_3\}$
4	$\{C_1^2, C_2^2, C_4^2, C_5^2\}$	$\{x_4, x_5, x_6, x_7\}$
2	$\{C_1^2, C_2^2\}$	$\{x_4, x_5, x_6, x_7, x_8, x_9\}$
1	$\{C_3^1\}$	$\{x_1, x_2, x_3, x_4, x_5\}$

clustering pattern, that is an agreement between the base clusterings to group together the instances in the instance set. A closed pattern can be observed as a maximal rectangle of '1' in the binary membership matrix, which denotes a pair consisting of a row set and a column set, such that for every row i in the instance set and every column c in the cluster set, we have $M[i, c] = 1$.

This closed pattern based approach enables the processing of datasets with a very large number of instances N , as in contrast to most other consensus clustering approaches, it does not require the processing of a co-association matrix of size N^2 but only of a membership matrix of size $N \cdot M$, where M is the number of base clusters, with $M \ll N$, and regarding the demonstrated scalability properties of algorithms for extracting Galois closed sets [7, 41, 70].

For further processings, the length of a closed pattern is defined as the length of its cluster set, that is the number of base clusterings that agreed to group its instance set. Its instance set can also be seen as a cluster. The extracted closed patterns for the support example are shown in Table 4.

3.2.4. Consensus function

The consensus function in [2] generates a number L_{unique} of consensus clustering solutions, where L_{unique} is the number of unique values among the sizes $|L_i|$ of closed pattern

cluster sets. Concretely, we iterate a loop index l_i from the maximum value of $|L_i|$ to the minimum value of $|L_i|$ and generate one consensus solution S_{l_i} per iteration. Each consensus solution S_{l_i} is generated based on instance sets with $|L_i| = l_i$ and the consensus solution $S_{l_{i-1}}$ of the previous iteration. The first consensus solution is generated from instance sets of closed patterns with maximal length $|L_{\max(i)}|$. Since for each consensus solution the clusters must be disjoint, the consensus function determines to either merge or split two intersecting cluster sets X and Y using Jaccard similarity [29]:

$$s(X, Y) = \max \left(\frac{|X \cap Y|}{|X|}, \frac{|X \cap Y|}{|Y|} \right)$$

If $s(X, Y)$ is greater than the input merging threshold MT , usually set to 0.5 by default, then X and Y are merged. Otherwise the larger cluster among X and Y is split. The merging threshold MT is a parameter of the function that can be defined by the user to adapt the merging/splitting consensus creation process to the properties of the data in input. Experiments have shown that a value $MT = 0.5$ is the most adequate for most benchmark datasets, originating from different application contexts and with different data space structure properties, that have been tested. The iterative consensus process repeats until there is no intersection

among clusters. The pseudo-code of the consensus function of the MultiCons approach is detailed in Algorithm 1.

ALGORITHM 1

Consensus process of the MultiCons approach.

Input: Instance sets of closed patterns C_i with their length $|L_i|$, merging threshold $MT = 0.5$

Output: Multiple consensus solutions S_l

```

1:  $L_{unique} \leftarrow$  unique values in the list of pattern lengths  $|L_i|$ 
2:  $S_{l_0} \leftarrow \emptyset$ 
3: for  $l_i$  in  $L_{unique}$  do
4:    $S_{l_i} \leftarrow S_{l_{i-1}} \cup C_i$  which length  $|L_i| = l$ 
5:    $endFlag \leftarrow True$ 
6:   repeat
7:     for each pair of clusters  $(X, Y)$  in  $S_{l_i}$  do
8:       if  $X$  intersects with  $Y$  then
9:          $endFlag \leftarrow False$ 
10:        Calculate  $s(X, Y)$ 
11:        if  $s(X, Y) \geq MT$  then
12:          Merge  $X$  and  $Y$ 
13:        else
14:          Split larger cluster
15:        end if
16:      end if
17:    end for
18:  until  $endFlag = True$ 
19: end for

```

The execution of the closed pattern based consensus function for the support example is shown in Figure 3. Each iterations consider closed patterns in decreasing order of their cluster set size $|L_i|$. The iteration generates a consensus solution according to the clusters generated during the previous iteration and the considered closed patterns. The first consensus clustering solution consists of clusters corresponding to the maximal number of agreements between the base clusterings, that is the closed patterns with the largest cluster set size. For each iteration of the loop, the cluster sets X and Y considered, i.e., among the previously generated clusters and considered closed patterns, that are intersecting are underlined in the execution trace.

3.2.5. Hierarchical graphical representation

The generated consensus solutions are presented to the user in a hierarchical graphical representation, as shown in Figure 4 for the support example. Each level represents a generated consensus solution corresponding to a given minimal number of agreements between the base clusterings, that is the cluster set size of closed patterns considered during the iteration that generated the solution. A consensus solution that is generated by several successive iterations of the process is depicted only once in the graphical representation, with the associated number of times it was generated (not shown in Figure 4) as it denotes a higher stability of this solution, and thus a higher robustness of its constituting clusters.

A recommended consensus clustering solution is suggested to the user. This solution is selected based on both its highest similarity with the clustering ensemble and its stability in the consensus creation process. However, different clustering solutions can be chosen as the final result depending on requirements and constraints of the application

performed. The hierarchical graphical representation can also provide important information for choosing a clustering solution regarding its stability and clusters that are present in several successive clustering solutions.

3.2.6. Properties of the approach

The major advantages of MultiCons over other consensus clustering methods are about its easily readable hierarchical graphical representation, allowing the user to analyze the consensus building process and identify clusters of particular importance, and its ability to discover the number of intrinsic clusters in the dataset data space without explicitly specifying the number of generated clusters k . However, due to the fact that the consensus process needs to be repeated until all clusters are disjoint, evaluating the computational complexity of MultiCons is difficult, since it significantly relies on how many times the process is repeated. Nevertheless, considerations about the computational complexity of the Semi-MultiCons approach are provided in section 4.1, considering the algorithmic optimization introduced in the clustering pattern processing with the novel Semi-MultiCons semi-supervised consensus function.

4. Semi-MultiCons Approach

In the Semi-MultiCons approach, the implementation of the consensus function is optimized compared with the MultiCons approach. This optimization reduces the number of loops of the consensus cluster creation process from closed patterns. Novel constraints-based consensus function and selection method of the recommended final clustering solution are also introduced. These new algorithmic processes add the use of supervision information, represented as must-link and cannot-link constraints, for optimizing the relevance of the recommended consensus solution regarding available prior knowledge.

The workflow of the Semi-MultiCons approach is shown in Figure 5. The initial steps of Semi-MultiCons, that is the creation of the clustering ensemble, its transformation into a binary membership matrix and the extraction of closed patterns, are identical to MultiCons initial steps. These steps define the search space for the consensus function and thus, the generation of the consensus clustering hierarchical graphical representation.

4.1. Implementation optimization

For each iteration l_i of the consensus process of MultiCons, the instance sets of closed patterns with $|L_i| = l_i$ are first combined with clusters of the previous consensus solution $S_{l_{i-1}}$. These two types of clustering patterns are compared during the l_i iteration to create the S_{l_i} consensus clustering solution. For this, each pair of instance sets and consensus clusters are enumerated until no intersection is detected. This results in inaccessible computational complexity evaluation due to the unknown number of loops required. However, since all clusters in $S_{l_{i-1}}$ are already disjoint, this overlapping check is not required for the Semi-MultiCons approach.

Size l_i	Set $S_{l_{i-1}} \cup C_i$ patterns with $ L_i = l_i$	Processing explanation
5	$S_0 = \emptyset$, patterns C_i with $ L_i = 5$: $\{\{x_1, x_2, x_3\}, \{x_4, x_5\}, \{x_6, x_7\}, \{x_8, x_9\}\}$ $\{\{x_1, x_2, x_3\}, \{x_4, x_5\}, \{x_6, x_7\}, \{x_8, x_9\}\}$	No intersection, generate S_5
4	$S_5 = \{\{x_1, x_2, x_3\}, \{x_4, x_5\}, \{x_6, x_7\}, \{x_8, x_9\}\}$, patterns C_i with $ L_i = 4$: $\{\{x_4, x_5, x_6, x_7\}\}$ $\{\{x_1, x_2, x_3\}, \{x_4, x_5\}, \{x_6, x_7\}, \{x_8, x_9\}, \{x_4, x_5, x_6, x_7\}\}$ $\{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7\}, \{x_6, x_7\}, \{x_8, x_9\}\}$ $\{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7\}, \{x_8, x_9\}\}$	$s(X, Y) = 1$, merge $s(X, Y) = 1$, merge No intersection, generate S_4
2	$S_4 = \{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7\}, \{x_8, x_9\}\}$, patterns C_i with $ L_i = 2$: $\{\{x_4, x_5, x_6, x_7, x_8, x_9\}\}$ $\{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7\}, \{x_8, x_9\}, \{x_4, x_5, x_6, x_7, x_8, x_9\}\}$ $\{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7, x_8, x_9\}, \{x_8, x_9\}\}$ $\{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7, x_8, x_9\}\}$	$s(X, Y) = 1$, merge $s(X, Y) = 1$, merge No intersection, generate S_2
1	$S_2 = \{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7, x_8, x_9\}\}$, C_i patterns with $ L_i = 1$: $\{\{x_1, x_2, x_3, x_4, x_5\}\}$ $\{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7, x_8, x_9\}, \{x_1, x_2, x_3, x_4, x_5\}\}$ $\{\{x_1, x_2, x_3, x_4, x_5\}, \{x_4, x_5, x_6, x_7, x_8, x_9\}\}$ $\{\{x_1, x_2, x_3, x_4, x_5\}, \{x_6, x_7, x_8, x_9\}\}$	$s(X, Y) = 1$, merge $s(X, Y) = 0.4$, split the larger cluster No intersection, generate S_1
End	$S_1 = \{\{x_1, x_2, x_3, x_4, x_5\}, \{x_6, x_7, x_8, x_9\}\}$	

Figure 3: Example MultiCons consensus process from closed patterns in Table 4. Closed patterns are processed in decreasing order of their cluster set size $|l_i|$, and the first line of each row shows the patterns considered during the l_i cluster set size iteration. The merging and splitting operations performed during each iteration, depending on underlined intersecting subsets of instances between created clusters and closed patterns, are then depicted. The resulting consensus clustering S_{l_i} of iteration l_i is shown on the last line of the row. The last row shows the S_1 consensus clustering at the top of the hierarchical graphical representation in Figure 4.

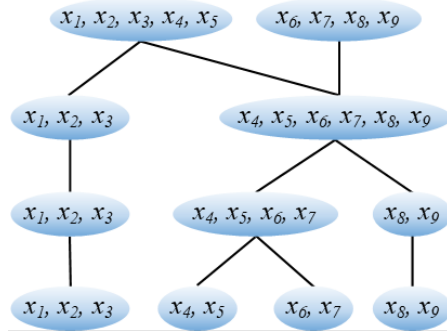


Figure 4: Hierarchical consensus clustering of the MultiCons approach. Each level in the hierarchical representation depicts a clustering of the dataset. Nodes represent instance sets of clusters and edges represent inclusion relationships between clusters, showing the successive groupings of instances corresponding each to a different number of agreements between clusterings of the base clustering ensemble.

The implementation is optimized by handling separately the closed patterns C_i with $|L_i| = l_i$ and already generated clusters in S_{l_i} , and is thus able to avoid the use of the repeat loop used in the MultiCons consensus function. For each closed pattern X in the set of patterns C_i with $|L_i| = l_i$, intersecting cluster sets Y in S_{l_i} are enumerated, and merged/split based on constrained consensus function. If the test comparing X and Y determines to merge, then X is updated to $X \cup Y$ and set Y is updated to \emptyset ; otherwise

the largest cluster set is modified to split the result. After enumerating all cluster sets Y , the resulting clustering pattern X is disjoint with all clusters in S_{l_i} . Cluster X is then added to S_{l_i} and the function continues to enumerate closed patterns C_i with $|L_i| = l_i$. The pseudo-code of the consensus generation process of Semi-MultiCons is given in Algorithm 2.

Assume that on average a closed pattern C_i contains n_c instances and a cluster in S_{l_i} contains n_s instances. The average number of loops performed is then $N/n_c \times N/n_s$. Intersection check and constraint check needs $O(k \times n_c \times n_s)$, resulting in an estimated complexity of $O(kN^2)$ on average for the Semi-MultiCons consensus process.

4.2. Constraint-based consensus generation

The novel Semi-MultiCons constraints-based consensus function makes use of supervision information represented in the form of must-link constraints and cannot-link constraints. The objective is to define a normalized score that evaluates how many constraints are satisfied or violated if the merge or split operation between clusters is performed.

Let's consider the following $g()$ function that defines a value representing existing constraints between two instance a and b of the dataset. $g(a, b) = 1$ and $g(a, b) = -1$ denotes respectively that instance a and instance b have a must-link or cannot-link constraint. Otherwise, if no constraint exists between a and b , we have $g(a, b) = 0$. Then, the score of

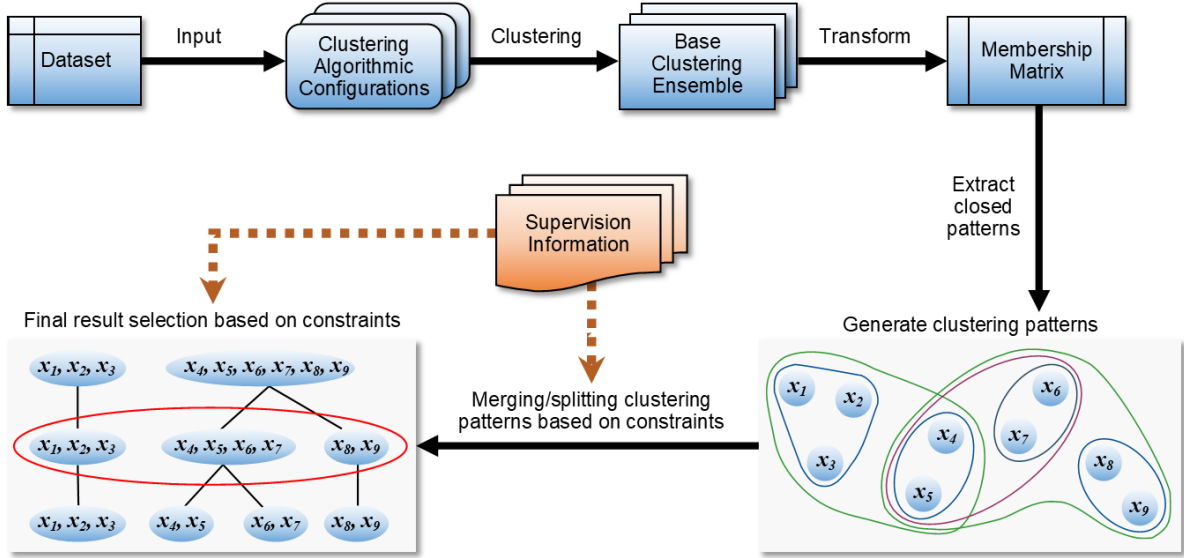


Figure 5: Workflow of the Semi-MultiCons approach. The dataset in input is processed by different clustering algorithmic configurations to create the clustering ensemble. The ensemble is transformed into a binary membership matrix from which closed patterns are extracted. These closed patterns are then processed by the Semi-MultiCons consensus function according to supervision information to generate the hierarchical consensus clustering result in which the final recommended clustering is identified.

ALGORITHM 2

Optimized consensus process of Semi-MultiCons.

Input: Instance sets of closed patterns C_i with their length $|L_i|$
Output: Multiple consensus solution S_i

```

1:  $L_{unique} \leftarrow$  unique values in the list of pattern lengths  $|L_i|$ 
2:  $S_{i0} \leftarrow \emptyset$ 
3: for  $l_i$  in  $L_{unique}$  do
4:    $S_{li} \leftarrow S_{li-1}$ 
5:    $C_{now} \leftarrow C_i$  with length  $L_i = l_i$ 
6:   for  $X$  in  $C_{now}$  do
7:     for  $Y$  in  $S_{li}$  do
8:       if Should merge according to constraint-based consensus function then
9:          $X \leftarrow X \cup Y$ 
10:         $Y \leftarrow \emptyset$ 
11:       else
12:         Larger cluster  $\leftarrow$  larger cluster -  $X \cap Y$ 
13:       end if
14:     end for
15:   Add  $X$  to  $S_{li}$ 
16: end for
17: end for
    
```

merging two clusters X and Y is defined as:

$$S_{merge} = \frac{\sum_{a \in X \setminus (X \cap Y)} \sum_{b \in Y \setminus (X \cap Y)} g(a, b)}{|(X \cup Y) \setminus (X \cap Y)|} \quad (1)$$

It represents how many must-link constraints per instance are satisfied if X and Y are merged. Similarly, the score of split $X \cap Y$ from X and of split $X \cap Y$ from Y are defined as follows:

$$S_{splitX} = \frac{- \sum_{a \in X \setminus (X \cap Y)} \sum_{b \in X \cap Y} g(a, b)}{|X|} \quad (2)$$

$$S_{splitY} = \frac{- \sum_{a \in Y \setminus (X \cap Y)} \sum_{b \in X \cap Y} g(a, b)}{|Y|} \quad (3)$$

If the three scores all equal to 0, it means that no supervision information on X and Y is available and the $s(X, Y)$ measure, as defined in Section 3.2.4, will be used. Otherwise, we will select the highest score to merge or split the clusters to comply with the objective to meet as many constraints as possible.

To illustrate the effect of the use of supervision information in the consensus generation process, let's consider the support example consisting of closed patterns in Table 4 with an additional cannot-link constraint between x_4 and x_8 . The resulting constraint-based consensus process of Semi-MultiCons is shown in Figure 6. For simplification of the presentation, the iterations before $l_i = 2$, i.e., for $l_i = 5$ and $l_i = 4$, that are not impacted by the additional cannot-link constraint are omitted since their results are the same as for the Multi-Cons approach.

4.3. Selection strategy

The Semi-MultiCons result for the support example, which execution is depicted in Table 6, is presented in the hierarchical graphical representation shown in Figure 7.

Duplicated consensus clustering solution are generated for iterations $l_i = 4$ and $l_i = 2$. This common solution is then represented only once in the final hierarchical graphical representation with its associated count of how many times this consensus clustering occurs. This count represents the frequency of the consensus solution among all possible solutions, denoting its stability in the consensus generation

Size l_i	$S_{l_{i-1}}$ (sets Y)	Set of patterns C_i with $ L_i = l_i$ (sets X)	Processing explanation
2	$\{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7\}, \{x_8, x_9\}\}$ $\{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7\}, \{x_8, x_9\}\}$ $\{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7\}, \emptyset\}$	$S_4 = \{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7\}, \{x_8, x_9\}\}$ $\{\{x_4, x_5, x_6, x_7, x_8, x_9\}\}$ $\{\{x_8, x_9\}\}$ $\{\{x_8, x_9\}\}$	$S_{merge} = 0, S_{splitY} = 0, S_{splitX} = 1/6$, split X No constraint, $s(X, Y) = 1$, merge to X No intersection, generate S_2
1	$\{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7\}, \{x_8, x_9\}\}$ $\{\emptyset, \{x_4, x_5, x_6, x_7\}, \{x_8, x_9\}\}$ $\{\emptyset, \emptyset, \{x_8, x_9\}\}$	$S_2 = \{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7\}, \{x_8, x_9\}\}$ $\{\{x_1, x_2, x_3, x_4, x_5\}\}$ $\{\{x_1, x_2, x_3, x_4, x_5\}\}$ $\{\{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}\}$	No constraint, $s(X, Y) = 1$, merge to X No constraint, $s(X, Y) = 0.5$, merge to X No intersection, generate S_1
End		$S_1 = \{\{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}, \{x_8, x_9\}\}$	

Figure 6: Example Semi-MultiCons constraint-based consensus process. Closed patterns are processed in decreasing order of their Cluster set size $|L_i|$. Iterations for cluster set size $l_i = 5$ and 4 are identical to the corresponding iterations in Figure 3. For $l_i = 2$ and 1, the first line of the corresponding row shows the cluster sets of closed patterns considered during the iteration. The merging and splitting operations performed during the iteration, depending on underlined intersecting subsets of instances between created clusters and closed patterns, and the supervision information constraints, are then depicted. The resulting consensus clustering S_2 and S_1 are shown on the last line of the rows. The last row shows the S_1 consensus clustering at the top of the hierarchical graphical representation in Figure 7.

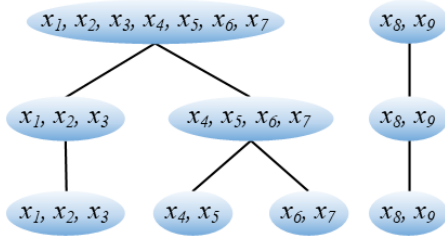


Figure 7: Hierarchical consensus clustering of the Semi-MultiCons approach. The three consensus clusterings generated from closed patterns and constraints are represented as a level in the hierarchical representation. The levels show the successive groupings of instances, each corresponding to a different number of agreements between clusterings of the ensemble, between instance sets of clusters represented as nodes. Edges represent inclusion relationships between clusters of different levels.

process, and thus the robustness of its constituting clusters regarding the search space in input. In the final hierarchical graphical representation for the support example shown in Figure 7, the frequencies of each level, from the bottom to the top of the hierarchy, are respectively 1, 2 and 1. The selection of the final recommended consensus solution to the user is based on both the number of satisfied must-link constraints and violated cannot-link constraints, and on the stability of the solution in the result and its similarity with the clustering ensemble. In the support example in Figure 7, the consensus clustering solution $\{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6, x_7\}, \{x_8, x_9\}\}$ that satisfies the cannot-link constraint between x_4 and x_8 and has a frequency equals to 2 is selected. This solution is the one that satisfies the largest number of constraints and has the highest frequency, as well as the highest similarity with the clustering ensemble, among the generated consensus clustering solutions.

5. Experimental results

An important difficulty in the application of most clustering and consensus clustering methods is the setting of the k parameter, that defines the number of clusters that will be generated. Indeed, an incorrect value for the k parameter may lead to a significant decrease in the relevance of the clustering result. Another difficulty, related to semi-supervised clustering usage of supervision information, is that integrating constraints sometimes lead to worse performance than using no constraints, which is a well-known potential *negative effect* reported in the literature [14, 61, 75]. During the experimental evaluation of the Semi-MultiCons approach, we address the following issues:

- Comparison between the inferred k value provided by Semi-MultiCons with the ground truth number of classes for classical benchmark datasets used to compare semi-supervised approaches.
- Comparison of performance between MultiCons, Semi-MultiCons and other semi-supervised clustering and/or consensus clustering approaches.
- Evaluation of the potential negative effect of constraints integration in the Semi-MultiCons process and comparison with other approaches.
- Study about scalability and complexity of the Semi-MultiCons approach for processing very large datasets.

The experimental settings, presented hereafter, were defined to address more specifically these central questions.

As detailed hereafter, the MPC-Kmeans single semi-supervised clustering approach is used for both comparison with Semi-MultiCons semi-supervised consensus clustering approach, and to generate base clusterings in the

clustering ensemble when comparing Semi-MultiCons with other semi-supervised consensus clustering approaches. The MPC-Kmeans approach [8] was selected since, to the best of our knowledge, it was reported in the literature as one of the most efficient single semi-supervised clustering approaches regarding result relevance. MPC-Kmeans is an hybrid method that combines constraint-based and distance-based methods. In [12], experiments reported in Table 3, representing NMI index score for UCI Machine Learning Repository datasets, show that MPC-Kmeans is actually the best performer among the CVQE, LCVQE, Kmeans and MPC-Kmeans algorithms, regardless execution times. Experimental results in [8] also show that MPC-Kmeans outperforms the PC-Kmeans and M-Kmeans semi-supervised clustering algorithms, and recently, in [75], MPC-Kmeans is used as baseline algorithm, and it is shown that it outperforms the Flexible CSP and COP-Kmeans algorithms.

As stated before for the MultiCons approach, ensemble diversity, that is the diversity among the base clusterings which constitute the search space for consensus clusters in ensemble methods, is a very important factor for the relevance of the consensus result. This is also the case for the clustering ensemble in the context of operational applications performed with the Semi-MultiCons semi-supervised approach. However, in the context of this article, only the MPC-Kmeans algorithm was used to create ensemble members since the objective is to assess and compare the results of the tested approaches with regards to the impact of pairwise constraints, and to ensure that no other factors related to the properties of the data space defined by the dataset and the single clustering algorithms used for creating base clusterings impact the results.

Experiments were conducted on a Dell server with 32 Intel Xeon 8 cores CPU E5-4620 2.20GHz processors and 529 GB main memory, running under CentOS operating system, and RAID 0 hard disk data storage virtualization technology for input/output performance optimization. An open source implementation in R of the Semi-MultiCons approach is available at <https://github.com/lazyCloud/semi-multicons> to facilitate the replication of the experiments.

Experimental datasets

Four classical benchmark datasets from the UCI Machine Learning Repository [16], namely the Iris, Wine, Zoo and Ecoli datasets, are used to evaluate the Semi-MultiCons semi-supervised ensemble clustering performance using the standard Normalized Mutual Information (NMI) [51], clustering accuracy (ACC) [69] and Purity [49] indexes. Besides these datasets, the MNIST dataset [32], that is very large regarding both its number of attributes and its number of instances, is used to evaluate the scalability and complexity of the Semi-MultiCons approach.

The potential negative effect of constraints was not studied for the MNIST dataset due to the limitations of the MPC-Kmeans approach for such a large dataset. For the negative effect experiment, repeated trials are necessary in order to get rid of potential randomness introduced by

Table 5

Benchmark dataset properties. For each of the five experimental datasets, its number of instance classes, number of attributes and number of instances are shown.

Dataset	Classes	Attribute	Number of instances
Iris	3	4	150
Wine	3	13	178
Zoo	7	17	101
Ecoli	8	8	336
MNIST	10	784	70000

constraint selection and/or random seeds. The processing of the MNIST dataset by the MPC-Kmeans requires around 10 hours to complete, which results in unacceptable time cost regarding the number of repeated trials required by the experiment. Therefore, the number of constraints for MNIST is fixed to 6000 to demonstrate the Semi-MultiCons ability to process large and challenging, regarding the number of instance classes, datasets.

Some basic facts about these datasets are stated in Table 5. Note that the Zoo and Ecoli datasets present an imbalanced class issue, with classes represented by very few instances. This can result in a number of intrinsic clusters in their data space corresponding to the number of major classes in the dataset, and thus, fewer than the total number of classes in the dataset. Both these five well-known benchmark datasets and the Normalized Mutual Information index measure were selected as they are the most popular in the domain, thus enabling to compare results with most unsupervised and semi-supervised clustering algorithms proposed in the literature.

Constraint generation

For all experiments, the must-link and cannot-link constraints are generated randomly from the classes of instances in the dataset. A pair of instances is added to the set of must-link constraints if these randomly chosen instances belong to the same. Otherwise, this pair of instances is added to the set of cannot-link constraints. This process is repeated until the number of must-link and cannot-link pairwise constraints required for the experiment is satisfied. For each experiment, all compared algorithms use exactly the same set of constraints.

Algorithmic approaches

Diverse criteria were considered for determining the best approaches to compare with Semi-MultiCons. These criteria consider in first place the quality of the clustering results, the efficiency and scalability of the approach regarding data size, the applicability of the approach to dataset containing heterogeneous and missing data, and the approach robustness to noise and outliers in the data. Considering theoretical and experimental results reported in the literature, and the availability and the results of implementation tests, the following semi-supervised clustering algorithmic approaches were selected: Third model (GV3) from [21], soft least

Table 6

Input parameters. For each of the five datasets, the range of values for the k parameter defining the number of clusters extracted in the base clusterings are shown.

Dataset	Range of values for k parameter
Iris	[2 – 6]
Wine	[2 – 6]
Zoo	[5 – 9]
Ecoli	[4 – 8]
MNIST	[8 – 12]

squares Euclidean consensus (DWH) [15], hard Euclidean consensus (HE) [26] and metric pairwise constrained K-means (MPC-Kmeans) [8]. Implementations of these approaches can be found in R packages *clue* [25] and *conclust* [57]. The original MultiCons method [3] was also used in the experiments to demonstrate the improvement offered by the Semi-MultiCons approach in the context of semi-supervised clustering.

Input parameters

For consensus clustering, algorithmic configurations defining the base clusterings are necessary to generate ensemble members. The range of values for the number of clusters k parameter used for the ensemble member generation are shown in Table 6. In operational applications of clustering, the number of clusters that are inherent to the data space properties is unknown, and a general idea is to estimate it based on a small sample set of data. However, this estimation for the k parameter can deviate from the number of intrinsic clusters. We therefore choose a range of values for the k parameter according to the approximate the number of intrinsic clusters in the dataset. The MPC-Kmeans algorithm was chosen as baseline for base clusterings to show the benefits of consensus process based on semi-supervised clustering algorithm.

Evaluation indexes

Normalized Mutual Information (NMI) [31], clustering accuracy (ACC) [69] and Purity [49] indexes are used to evaluate the quality of the resulting clusterings. For the Semi-MultiCons approach, while multiple consensus clustering solutions are generated, only the recommended solution is considered as the output. In the following experimental results, the evaluation index score and the inferred number of clusters k of the recommended solution are represented as S_r , that is the average evaluation index value obtained for Semi-MultiCons, and k_r , that is the average number of clusters generated for Semi-MultiCons for the repeated executions of each experiment. For other consensus clustering approaches that require k as an input parameter, the number of cluster k_b that reaches their best performance is used. To avoid bias in comparison, we also demonstrate the level of Semi-MultiCons which number of clusters is equal to k_b as reference. Essentially, a better performance means a greater

evaluation index value and an inferred k_r that is closer to the number of classes in the dataset.

Experiment settings

The number of constraints ranges from 0 to 150 for UCI Machine Learning Repository datasets. For each number of constraints, 30 different constraint sets were generated with 10 times repeated trials for each execution. For the MNIST dataset, due to the size of the dataset and the computational complexity of the MPC-Kmeans algorithm, the number of constraints was fixed to 6 000 and each consensus approach was run 10 times. Semi-MultiCons and other methods are guaranteed to access exactly the same constraints for each execution.

5.1. Comparison between inferred and real number of classes

The average inferred numbers of clusters k_r by Semi-MultiCons over all trials for each dataset are listed in Table 7. k_b represents the number of clusters defined by the k parameter in input for which the MPC-Kmeans algorithm achieves its best performance. Detailed results about MPC-Kmeans and Semi-MultiCons performances are given in Section 5.2. The real, i.e., ground truth, number of classes and the number of major classes in the dataset are also given as reference. The major classes refers to the class which contains at least 5% number of instances. As stated before, the Zoo and Ecoli datasets contain classes with only few instances and have therefore a number of intrinsic clusters that is lower than their total number of classes. We can thus observe that MPC-Kmeans does not always achieve its best performance when its k parameter value is equal to ground truth number of classes. The MPC-Kmeans approach trends to find large, balanced clusters in data, while the inferred k_r provided by Semi-MultiCons is much closer to the ground truth number of classes.

For the MNIST dataset, the largest k in input gives the best MPC-Kmeans performance. Semi-MultiCons also infers a large k_r , implying that the number of clusters in the data space may be larger than the number of classes in the dataset. Actually, several subspaces in the data space defined by the input dataset, that is intrinsic clusters in this data space, can correspond to the same class. This is the case if the class corresponds to different subgroups of instances, which means the class can be characterized by several distinct groups of instances in the data space. On the contrary, if some classes cannot be fully distinguished using the information provided by the dataset, several classes may belong to the same subspace of the data space, i.e., correspond to a unique intrinsic group of instances in the data space, and then the number of underlying clusters in the data space may be lower than the number of classes in the dataset. The multi-level structure of clusters in output of Semi-MultiCons can provide information about this property, by showing the successive merging and splitting operations performed, and help to automatically discover the appropriate number of clusters k for the dataset, that is to identify the most relevant

Table 7

Comparison between the inferred and ground truth numbers of classes. For each of the five datasets, the range of values for the k parameter, the average number of clusters k_r inferred by Semi-MultiCons, the number of clusters k_b for MPC-Kmeans to generate the best base clustering solution, the total number of classes and the number of major classes in the dataset are shown.

Dataset	Input k	Average k_r	k_b	Number of classes	Number of major classes
Iris	[2 – 6]	3.3	3	3	3
Wine	[2 – 6]	3.4	3	3	3
Zoo	[5 – 9]	5.5	5	7	5
Ecoli	[4 – 8]	5.7	5	8	5
MNIST	[8 – 12]	20	12	10	10

consensus, i.e., level in the hierarchy, among the consensuses in output.

5.2. Comparison with single semi-supervised clustering approaches

In this experiment, the results of the Semi-MultiCons semi-supervised consensus clustering approach and the MPC-Kmeans single clustering approach are compared in terms of the relevance of the consensus solution generated. As stated before, the NMI, ACC and Purity indexes are used to evaluate and compare the relevance of the generated clustering solutions. The number of pairwise constraints used was varied between 0 and 160 to evaluate their impact on the clustering result. The results of Semi-MultiCons and MPC-Kmeans for the Iris, Wine, Zoo and Ecoli datasets are presented in Figure 8. The best performance obtained for all trials of each execution is shown. We could note that for some experiments, the number of clusters obtained may be different from the number of classes in the dataset, due to the potential existence of several clusters defining a class in the data space as discussed in Section 5.1.

The three curves depicting the evaluation of MPC-Kmeans results correspond each to a different value for the input parameter k . The blue curve corresponds to the k_b value number of clusters, that generates the best result among all tested values in 6. The red curve corresponds to the $k_b - 1$ value, and the green curve corresponds to the $k_b + 1$ value. The yellow curve represents the S_r evaluation in the situation where the number of classes is unknown for Semi-MultiCons, that is the consensus clustering evaluated is the one automatically selected by the approach in the output hierarchy. The black curve represents the S_r evaluation when the number of classes k_b is given as input, that is the consensus clustering evaluated is the one with a number of clusters equal to k_b in the output hierarchy. We can clearly see from the curves that the performance increases with the number of constraints, proving that constraints are useful for improving the quality of the clustering.

Considering the MPC-Kmeans approach, we observe that its performance strongly depends on the value of its input parameter k . If this value does not fit well with the number of clusters in the dataset, MPC-Kmeans fails to give good performance. Also, for imbalanced Zoo and Ecoli dataset, MPC-Kmeans achieves its best performance when

k does not correspond to the number of classes, showing the difficulties MPC-Kmeans faces when the number of clusters is large. In contrast, the Semi-MultiCons approach is able to give a comparable or better performance when the number of clusters k is given as input, as shown by the black curves, for all three evaluation indexes. Moreover, without knowing k , Semi-MultiCons is still able to give good performance, close to those obtained with known k , as shown by the yellow curves, especially when k is large.

5.3. Comparison with semi-supervised consensus clustering approaches

This experiment compares the results of the Semi-MultiCons, the MultiCons and other baseline semi-supervised consensus clustering approaches in terms of relevance of the consensus solution generated. Experimental results for the Semi-MultiCons, MultiCons, DWH, GV3 and HE approaches for the Iris, Wine, Zoo and Ecoli datasets are presented in Figure 9. The height of the curves represents the value of the three evaluation indexes for the clustering generated when varying the number of pairwise constraints between 0 and 160 as shown on the horizontal axis. As in all the subsequent experiments, all the tested semi-supervised approaches use exactly the same information in input, that is the set of base clusterings in the clustering ensemble and the set of pairwise constraints between instances.

Clearly, Semi-MultiCons reaches comparable or better performance when the number of clusters k_b is used to choose the final clustering result in the output hierarchy, for all three evaluation indexes. It also outperforms the original MultiCons approach, demonstrating the positive effect of the proposed constraint-based consensus process.

Particularly, the original MultiCons approach outperforms Semi-MultiCons with known k on Purity evaluation index for Zoo dataset. However, high Purity index can be easily achieved when the number of clusters is large. Combining the results for the NMI and ACC evaluation indexes, the Semi-MultiCons approach still performs better than the original MultiCons approach. We can note that even without explicitly knowing the k_b value, Semi-MultiCons is able to generate a solution reaching a good performance, overall close to the best solution. For the three other approaches, results are similar for the four datasets: The best solutions are generated by the GV3 algorithm, the lower performer

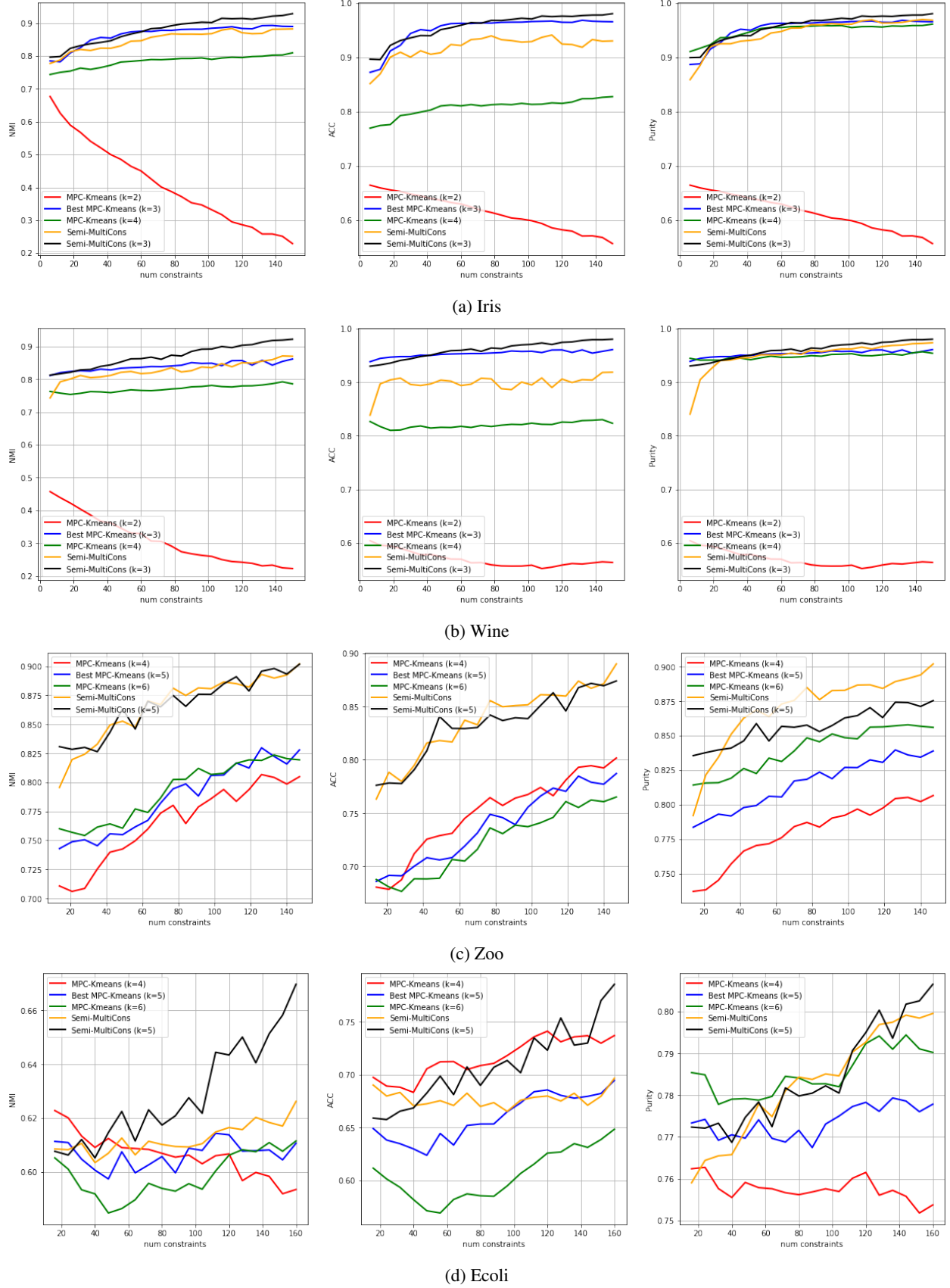


Figure 8: Comparison between Semi-MultiCons and MPC-Kmeans approaches. The curves illustrate the best clustering result obtained by the MCP-Kmeans single clustering approach and the Semi-MultiCons consensus clustering approach for each of the Iris (a), Wine (b), Zoo (c) and Ecoli (d) benchmark datasets. The horizontal axis shows the number of pairwise constraints used during the execution and the vertical axis shows the evaluation index value (NMI, ACC or Purity) of the clustering solution.

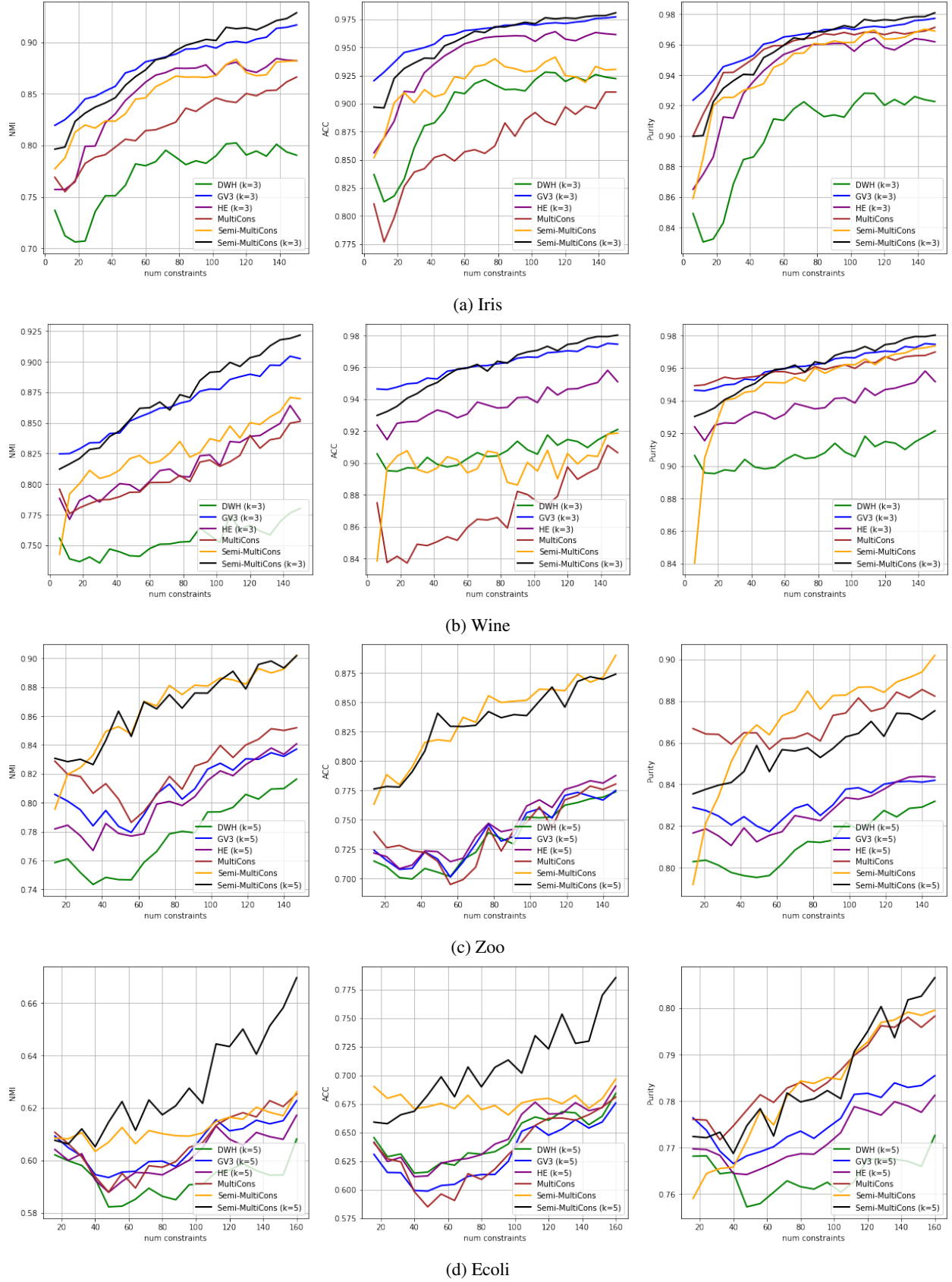


Figure 9: Comparison between the Semi-MultiCons, MultiCons and other baseline semi-supervised consensus clustering approaches. The curves illustrate the best clustering result obtained by the DWH, GV3, HE, MultiCons and Semi-MultiCons approaches for each of the Iris (a), Wine (b), Zoo (c) and Ecoli (d) benchmark datasets. The horizontal axis shows the number of pairwise constraints used during the execution and the vertical axis shows the evaluation index value, for the NMI, ACC or Purity evaluation indexes, of the clustering solution generated.

solutions are generated by the DWH algorithm, and the HE algorithm generates solutions with an evaluation that is intermediate between those of GV3 and DWH algorithms. During this experiment, the GV3 algorithm is the only approach with performances that are comparable with Semi-MultiCons. However, Semi-MultiCons has better properties than GV3 regarding efficiency in time and space, i.e., number of operations performed and memory usage, as shown by the scalability and complexity analysis presented in the following sections.

5.4. Analysis about negative effect

A potential *negative effect* issue of semi-supervised clustering methods was largely reported in the literature [14, 61, 75]. This issue relates to the use of pairwise constraints as supervision information in the clustering process that sometimes leads to performance, in terms of quality of the clustering result, that are worse than using no constraint. However, most semi-supervised consensus clustering algorithms were only evaluated by average performance, thus not highlighting this potential issue.

The importance of the negative effect of using constraints is evaluated by the fraction of times that unconstrained clustering produces better results than constrained clustering. For comparison of results, the unconstrained version is defined by the performance of the K-means approach, i.e., equivalent to using no constraints for performing unsupervised MPC-Kmeans, with an input parameter k equals to the optimal number of clusters k_b . The percentage values shown represent the proportion of experiments where the clustering solution generated by the constrained algorithm underperforms the result of K-means approach with parameter k equals to k_b .

The results of the experiment are shown in Figure 10. The horizontal axis shows the number of pairwise constraints used during the execution and the vertical axis shows the fraction of times that the algorithm produced worse performance than the baseline K-means clustering with k_b as input parameter. The black curve represents the evaluation of the Semi-MultiCons consensus solution which number of clusters is equal to the optimal number of clusters k_b . The yellow curve represents the negative effect ratio of S_r evaluation in the situation where the number of classes is unknown for Semi-MultiCons. The brown curve represents the negative effect ratio of the MultiCons approach. The four remaining curves represent the evaluation of the semi-supervised results of the MPC-Kmeans, GV3, HE and DWH algorithms with a k input parameter equals to k_b .

We can see from the results in the figure that for all algorithms, the negative effect decreases when the number of constraints increases, implying that extending the size of the constraint set can be a possible solution to fight against the negative effect. Under the same condition, where the optimal number of clusters is provided, Semi-MultiCons highly reduces the occurrence of negative effect, compared to other approaches, for most datasets as illustrated by the black curve. When the optimal number of clusters is not

Table 8

Performance on the MNIST dataset. Comparison of the semi-supervised single and consensus clustering approaches on the MNIST dataset of 70 000 instances. Results show the relevance of the clustering solution evaluated with the NMI, ACC and Purity indexes and execution times in seconds.

Algorithm	NMI	ACC	Purity	Time (s)
Semi-MultiCons	0.8024	0.7735	0.7839	268.39
MultiCons	0.7721	0.7092	0.7445	30.81
MPC-Kmeans	0.7706	0.7025	0.7428	15677.64
HE	0.7603	0.6861	0.7197	0.99
DWH	0.7529	0.6873	0.7193	0.24

provided, Semi-MultiCons alleviates negative effect as well, especially for datasets where the number of clusters is large. For the Iris dataset, GV3 and Semi-MultiCons even reach 0% negative effect ratio when size of constraint set is large. A specific feature of the Iris dataset, compared to other benchmark datasets used, is that this dataset is perfectly balanced regarding the number of instances in each class.

5.5. Analysis about convergence

Since the proposed constraint-based consensus process is a dynamic process with splitting/merging operations, we evaluate the convergence of Semi-MultiCons in this experiment, to illustrate the trend of the cost regards to the number of iterations. The objective of the novel constraints-based consensus function, as explained in 4.2, is to meet as many constraints as possible. The cost is hereby defined as the percentage of unsatisfied pairwise constraints. It is calculated for each of the successive levels of the hierarchy generated by Semi-MultiCons, from bottom to top. These successive levels from bottom to top eventually correspond the number of iterations, as each level represents a consensus solution generated based on the previous level and the considered closed patterns, as stated in Figure 3 and Figure 6.

Results presented in Figure 11 show that for all the four datasets, the cost moves continuously towards a minima, with a decreasing trend as the number of iteration increases, proving the convergence of the Semi-MultiCons approach.

5.6. Performance on the MNIST dataset

This experiment aims to compare the performances of the different semi-supervised clustering approaches in terms of execution times and applicability regarding memory usage. The MNIST benchmark dataset, containing 70 000 instances, is used for these performance tests. The number of pairwise constraints is fixed to 6 000 and each algorithm is run 10 times.

Results for the compared approaches are given in Table 8. These results present both the average quality of the clustering in output and execution times for each algorithm. Since the GV3 algorithm runs out of memory for such a large dataset, that exceeds its capacity regarding memory usage, its performances are not presented. Note that for this experiment, the optimal number of clusters is not provided

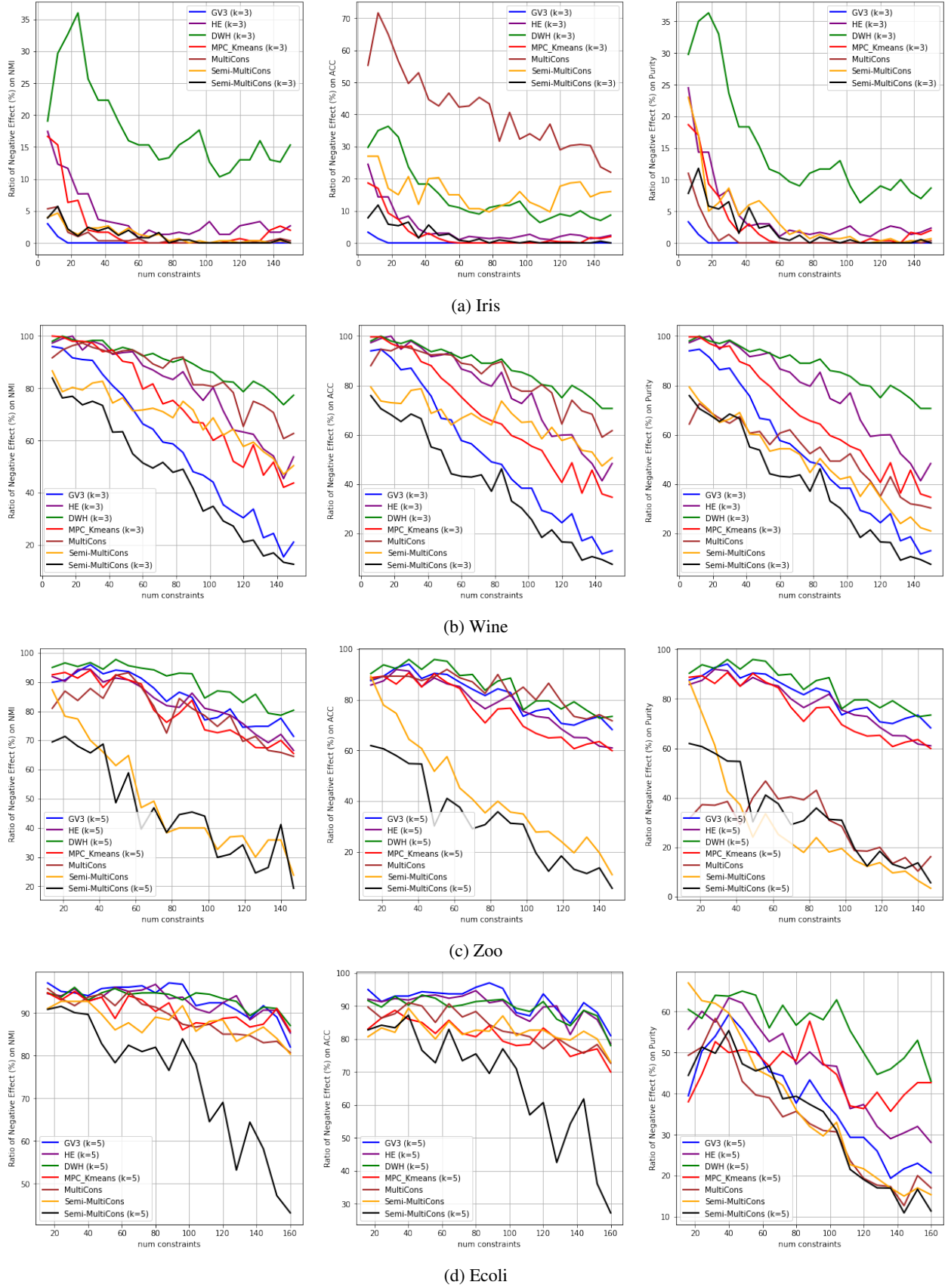


Figure 10: Ratio of negative effect. Evaluation of the negative effect of pairwise constraints for semi-supervised clustering in terms of the fraction of times that the algorithm produced worse performance than the baseline K-means clustering with optimal number of clusters as input parameter. The horizontal axis shows the number of pairwise constraints used during the execution and the vertical axis shows the ratio of negative effect in percentage, regards to the NMI, ACC and Purity evaluation indexes.

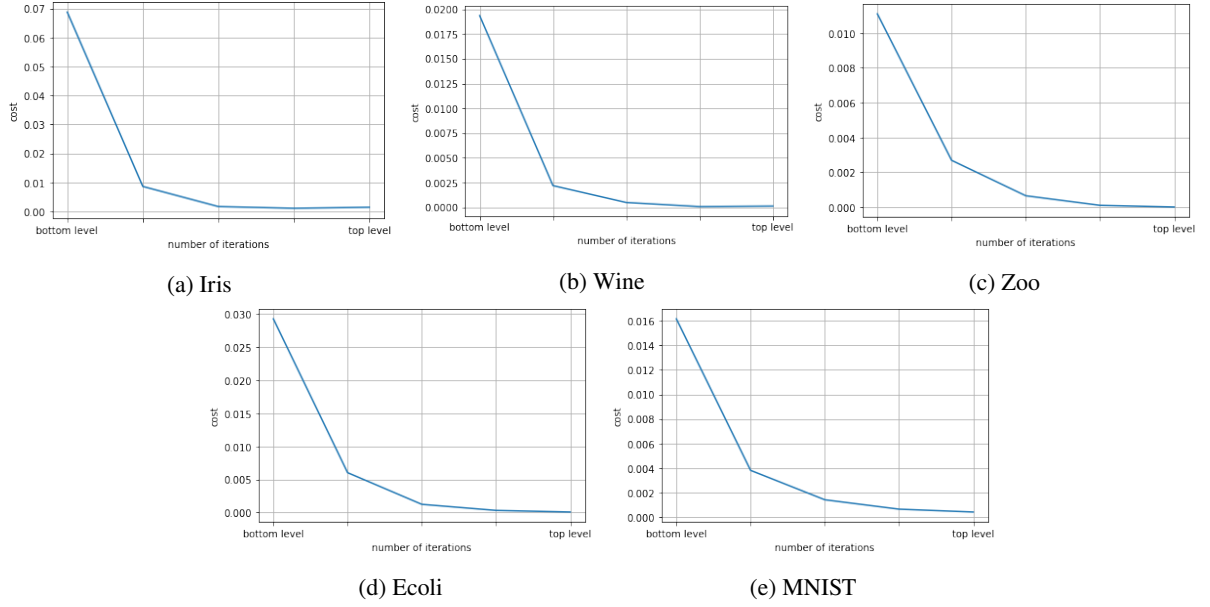


Figure 11: Convergence of Semi-MultiCons. Evaluation of the convergence of Semi-MultiCons in terms of the percentage of unsatisfied pairwise constraints that is represented as the cost. The horizontal axis shows the successive levels of the generated hierarchy from bottom to top, which eventually corresponds to the number of iterations, and the vertical axis shows the cost as the percentage of unsatisfied pairwise constraints.

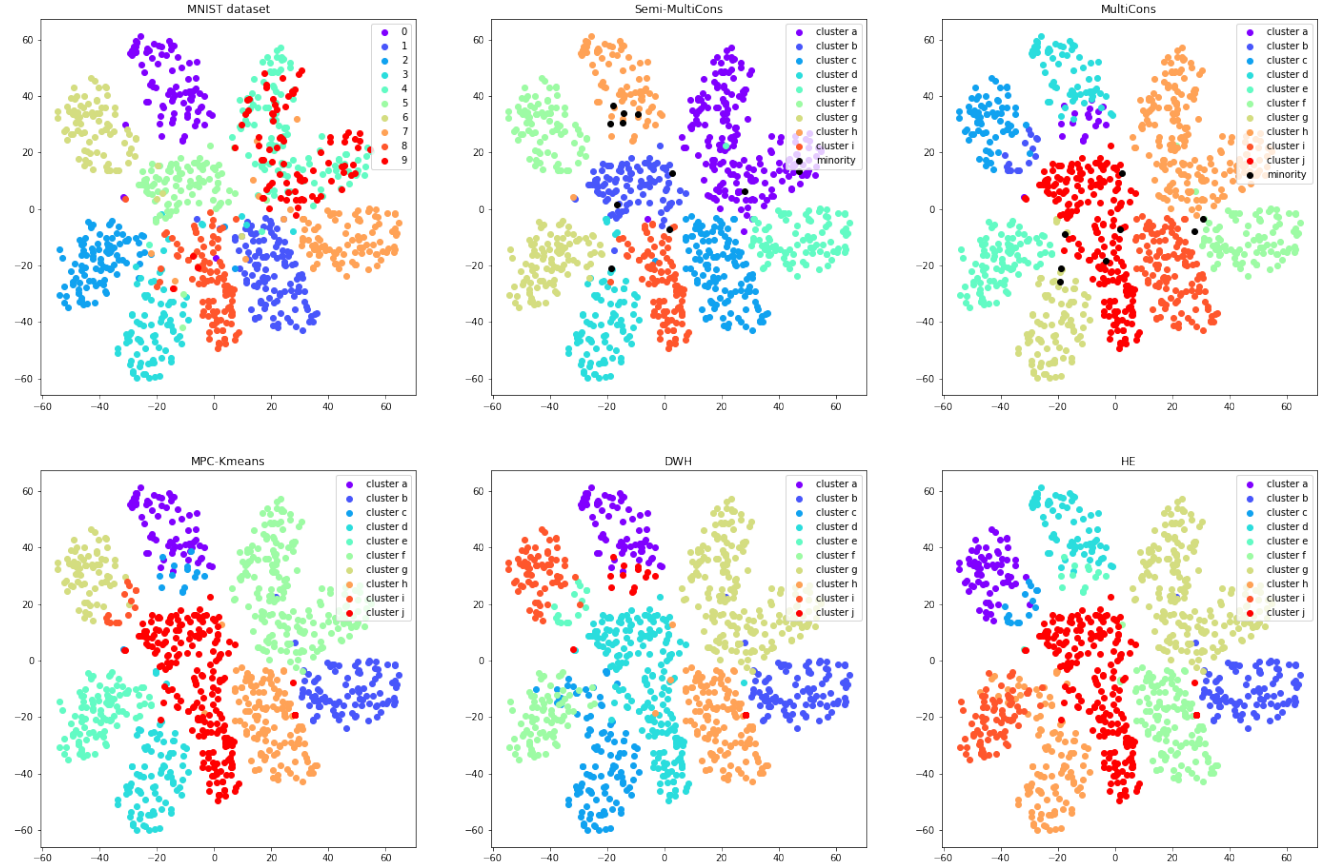


Figure 12: Representations of the clustering results of the semi-supervised single and consensus clustering approaches on the MNIST dataset of 70 000 instances using the t-SNE visualization. Figures show the clustering results for 1000 sample instances in the latent space. The horizontal and vertical axis represent the latent space and the colors represent the clustering results.

as parameter k to Semi-MultiCons. We can see that among the five compared approaches, the DWH and HE approaches have the lowest execution times. However, as observed before, their performance in terms of relevance of the clustering result is clearly lower compared to Semi-MultiCons and GV3 for datasets from the UCI Machine Learning Repository. The MPC-Kmeans approach requires very important execution times compared to all other approaches. Overall, we can find that the Semi-MultiCons approach is able to both handle large and challenging datasets, and provide a relevant clustering result even when the optimal number of clusters or classes is unknown.

The t-Distributed Stochastic Neighbor Embedding (t-SNE) visualization [38] of the clustering results for the compared approaches are demonstrated in Figure 12. The implementation in Scikit-learn Python package [44] was used, with initialization method set to PCA [50] and parameter perplexity set to 40, that is the same initialization setting as [38]. The t-SNE visualization is trained on the entire MNIST dataset with 70 000 instances. However, to better visualize the clustering results, only 1 000 randomly sampled instances are displayed. For reasons of clarity and readability, the minority clusters which contain less than 1.5% percentage of the total number of instances in the MNIST dataset, are represented as one cluster in black. We can observe that in the latent space, the digits 4 and 9 are difficult to be recognized and all the five approaches fail to separate them. Compared to other approaches, the Semi-MultiCons has better performance on digits 5 and 8 as it is the only approach that can partition them into two clusters. The HE and DWH approaches have the worst performance, since they under-perform the other approaches in terms of the digits 2 and 3. The t-SNE visualization gives a straightforward and complementary illustration to better understand and evaluate the clustering results presented in Table 8.

5.7. Computational complexity study

This study aims to evaluate the efficiency of the Semi-MultiCons approach and compare it with MultiCons and other semi-supervised clustering approaches. During the first experiment of this study, the execution times of Semi-MultiCons, MultiCons and the four other semi-supervised clustering approaches used in the experiments are compared. Experimental results are presented in Figure 13. Note that for the consensus-based approaches, the execution time of the ensemble member generation is not considered. The curves depict the execution times of the six approaches while varying the number of pairwise constraints used during the execution. We can see that DWH and HE are the most efficient approaches for the four datasets of the UCI Machine Learning Repository. They have close execution times and their curves often overlap each other in the figure. We can also observe that MultiCons and Semi-MultiCons execution times are systematically lower than those of MPC-Kmeans and GV3 for all the four datasets. The Semi-MultiCons execution times are slightly higher than those of MultiCons

due to the fact that it integrates constraints during consensus process.

During the second experiment of this study, Semi-MultiCons was applied to different samples of the MNIST dataset. Seven samples, containing from 10 000 instances for the smallest to 70 000 instances for the largest, were generated from the MNIST dataset. The number of pairwise constraints is fixed to 6 000, and 10 trials are performed for each execution. Figure 14 presents the average execution times for each of the seven executions. The curve shows that the scalability property of Semi-MultiCons is linear in the number of instances for generating the output consensus clustering hierarchy. In Section 4.1, the complexity of Semi-MultiCons was estimated as proportional to the squared number of instances N in the dataset. The results show that the time complexity is close to this estimate of the computational complexity of Semi-MultiCons.

6. Conclusion

In this article, we present a new semi-supervised consensus clustering approach named Semi-MultiCons. During the development of this approach, we optimized the implementation of the initial MultiCons approach for multiple consensus clustering [3], and we designed a new iterative constraint-based consensus function to integrate supervision information into the consensus clustering process. The experiments conducted on four reference benchmark datasets from the UCI Machine Learning Repository and on the MNIST large dataset have led to the following conclusions:

- The prior knowledge, i.e., supervision information, represented as pairwise constraints between instances is useful for improving the quality of clustering.
- Semi-MultiCons manages to infer the correct number of clusters in output, while processing base clustering ensemble members with different numbers of clusters.
- Semi-MultiCons is able to generate a clustering solution with comparable or better relevance compared to single semi-supervised clustering and consensus clustering approaches without explicitly knowing the optimal number of clusters k .
- Semi-MultiCons is able to process datasets with large number of instances contrarily to several other approaches, and has a linear scalability in the number of dataset instances.
- Semi-MultiCons has the ability to solve operational clustering problems encountered when data is imbalanced, the number of clusters is large and/or the number of clusters is ambiguous or unknown.

In the general case of operational applications, where the intrinsic number of clusters in the data space is unknown, the Semi-MultiCons approach can be used as a pre-processing step to contribute to the discovery of the appropriate number

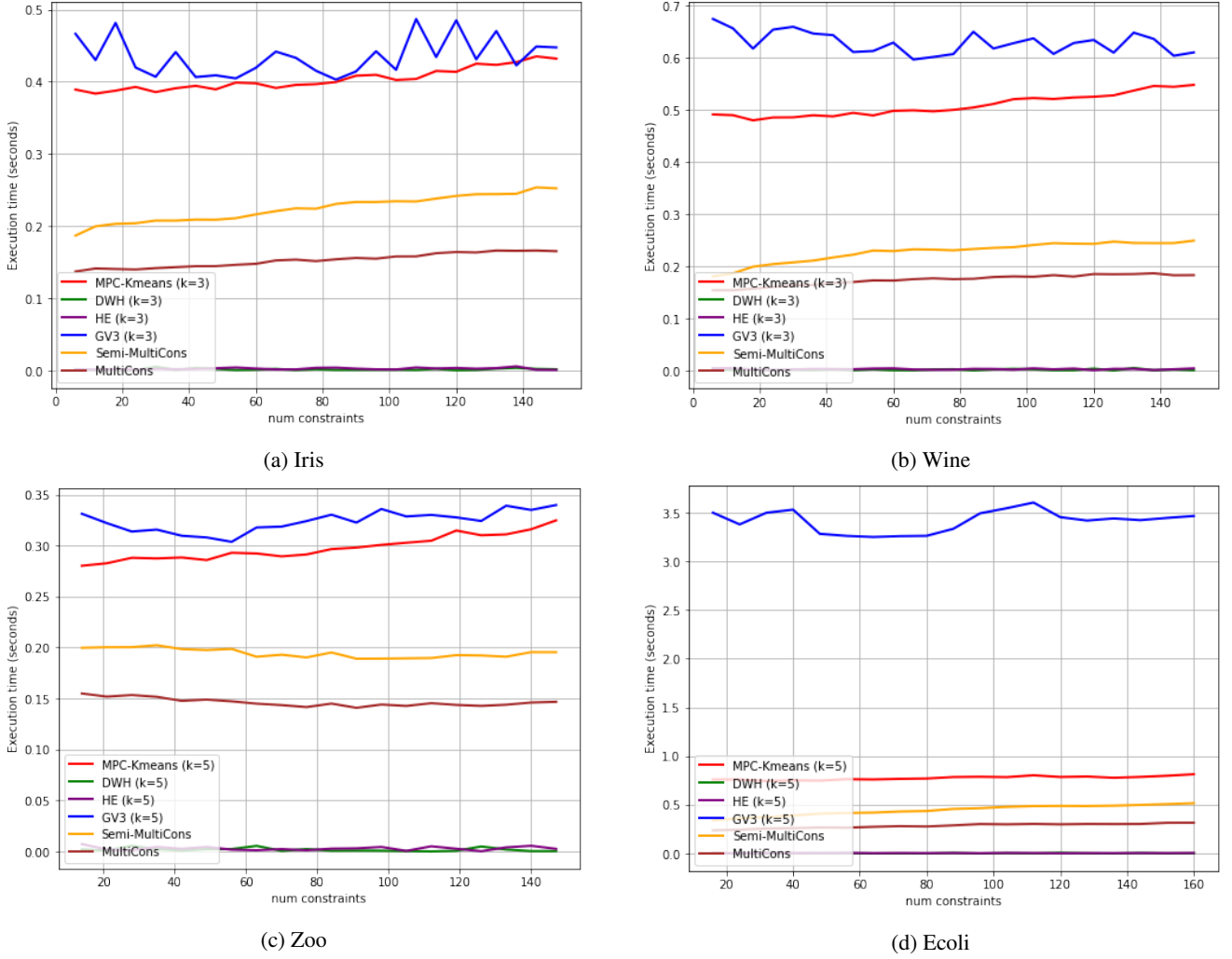


Figure 13: Execution times. The curves show the execution times of the MPC-Kmeans, DWH, GV3, HE, MultiCons and Semi-MultiCons clustering approaches for each of the Iris (a), Wine (b), Zoo (c) and Ecoli (d) benchmark datasets. The horizontal axis shows the number of pairwise constraints used during the execution and the vertical axis shows the number of seconds required by the approach to generate the output clustering solution.

of clusters, as well as a consensus clustering method to achieve better clustering quality.

Although Semi-MultiCons is more time consuming than the most efficient baseline semi-supervised consensus clustering approaches like HE and DWH, its execution times remain acceptable even for large and complex datasets such as the MNIST benchmark dataset. Also, the hierarchical output of Semi-MultiCons can offer significant improvements, compared to such approaches which generate a single clustering solution, regarding the application performed. For example, it can provide information to better understand the properties of the data space through the visualisation of successive mergers of sets of instances represented in the ConsTree. Future plans to improve the efficiency of the approach in terms of time complexity encompass the design of new algorithmic solutions to:

- Extract clustering patterns, such as generated by the MultiCons approach, directly from the binary membership matrix, by merging the iterative processes of closed pattern extraction and consensus cluster generation by the consensus function.
- Integrate supervised information represented as pairwise constraints between instances in the closed pattern mining phase from the binary membership matrix, by the definition of a new constraints-based closure function.
- Extract constraints-based clustering patterns, such as generated by the Semi-MultiCons approach, directly from the binary membership matrix with the new constraints-based closure function.

Among perspectives of further work, the first is the investigation of the effect of noise on performances of the

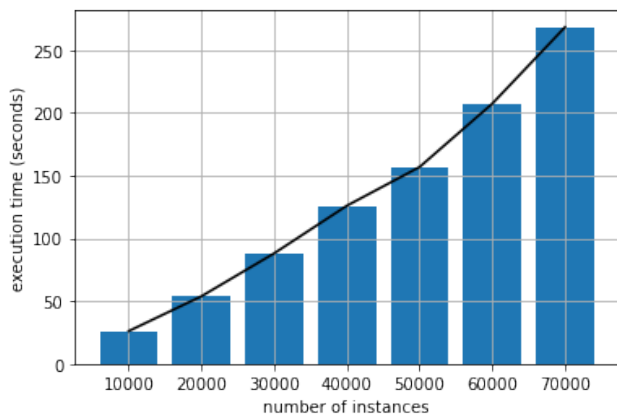


Figure 14: Execution times of Semi-MultiCons for the MNIST dataset. Each bar in the diagram represents the execution time in seconds for the MNIST sample which number of instances is represented on the horizontal axis.

approach. The second is the utilization of the Semi-MultiCons consensus function as a loss function, as the proposed constraints-based consensus function demonstrates an appropriate convergence, to apply it in a deep learning pipeline architecture [17]. As a matter of fact, several deep learning approaches recently reported in the literature, such as for example [75], integrate pairwise constraints in the objective function of deep neural networks.

Acknowledgment

This project was carried out as part of the IDEX UCA^{JEDI} MC² joint project between Amadeus and the Université Côte d'Azur. This work has been supported by the French government, through the UCA^{JEDI} Investments in the Future project managed by the National Research Agency (ANR) with the reference number ANR-15-IDEX-01.

References

- [1] Ahmad, A., Khan, S.S., 2019. Survey of state-of-the-art mixed data clustering algorithms. *IEEE Access* 7, 31883–31902. doi:10.1109/ACCESS.2019.2903568.
- [2] Al-Najdi, A., 2016. A closed patterns-based approach to the consensus clustering problem. Ph.D. thesis. Université Côte d'Azur. URL: <https://tel.archives-ouvertes.fr/tel-01478626>.
- [3] Al-Najdi, A., Pasquier, N., Precioso, F., 2016. Multiple consensus clustering by iterative merging/splitting of clustering patterns, in: *Proceedings of the MLDM International Conference on Machine Learning and Data Mining in Pattern Recognition*, Springer. pp. 790–804. doi:10.1007/978-3-319-41920-6_60.
- [4] Ayad, H.G., Kamel, M.S., 2010. On voting-based consensus of cluster ensembles. *Pattern Recognition* 43, 1943–1953. doi:10.1016/j.patcog.2009.11.012.
- [5] Bair, E., 2013. Semi-supervised clustering methods. *WIREs Computational Statistics* 5, 349–361. doi:https://doi.org/10.1002/wics.1270.
- [6] Basu, S., Bilenko, M., Mooney, R.J., 2004. A probabilistic framework for semi-supervised clustering, in: *Proceedings of the 10th SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM. pp. 59–68. doi:10.1145/1014052.1014062.
- [7] Bertet, K., Demko, C., Viaud, J.F., Guérin, C., 2018. Lattices, closures systems and implication bases: A survey of structural aspects and algorithms. *Theoretical Computer Science* 743, 93–109. doi:https://doi.org/10.1016/j.tcs.2016.11.021.
- [8] Bilenko, M., Basu, S., Mooney, R.J., 2004. Integrating constraints and metric learning in semi-supervised clustering, in: *Proceedings of the 21st ICML International Conference on Machine Learning*, ACM. pp. 81–88. doi:10.1145/1015330.1015360.
- [9] Boongoen, T., Iam-On, N., 2018a. Cluster ensembles: A survey of approaches with recent extensions and applications. *Computer Science Review* 28, 1–25. doi:10.1016/j.cosrev.2018.01.003.
- [10] Boongoen, T., Iam-On, N., 2018b. Cluster ensembles: A survey of approaches with recent extensions and applications. *Computer Science Review* 28, 1–25. doi:10.1016/j.cosrev.2018.01.003.
- [11] Chen, S., Zhang, D., 2011. Semisupervised dimensionality reduction with pairwise constraints for hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters* 8, 369–373. doi:10.1109/LGRS.2010.2076407.
- [12] Covões, T.F., Hruschka, E.R., Ghosh, J., 2013. A study of K-Means-based algorithms for constrained clustering. *Intelligent Data Analysis* 17, 485–505. doi:10.3233/IDA-130590.
- [13] Davidson, I., Ravi, S., 2005. Clustering with constraints: Feasibility issues and the k-means algorithm, in: *Proceedings of the SIAM International Conference on Data Mining*, SIAM. pp. 138–149. doi:10.1137/1.9781611972757.13.
- [14] Davidson, I., Wagstaff, K.L., Basu, S., 2006. Measuring constraint-set utility for partitioning clustering algorithms, in: *Proceedings of the 10th PKDD European Conference on Principles of Data Mining and Knowledge Discovery*, Springer. pp. 115–126. doi:10.1007/11871637_15.
- [15] Dimitriadou, E., Weingessel, A., Hornik, K., 2002. A combination scheme for fuzzy clustering. *International Journal of Pattern Recognition and Artificial Intelligence* 16, 901–912. doi:10.1142/S0218001402002052.
- [16] Dua, D., Graff, C., 2017. UCI Machine Learning Repository, University of California, Irvine. URL: <http://archive.ics.uci.edu/ml>.
- [17] El-Amir, H., Hamdy, M., 2020. A tour through the deep learning pipeline, in: *Deep Learning Pipeline*. Springer, pp. 57–84. doi:10.1007/978-1-4842-5349-6_3.
- [18] Fern, X.Z., Brodley, C.E., 2004. Solving cluster ensemble problems by bipartite graph partitioning, in: *Proceedings of the 21st ICML International Conference on Machine learning*, ACM. p. 36. doi:10.1145/1015330.1015414.
- [19] Fred, A., 2001. Finding consistent clusters in data partitions, in: *Proceedings of the International Workshop on Multiple Classifier Systems*, Springer. pp. 309–318. doi:10.1007/3-540-48219-9_31.
- [20] Ghosh, J., Acharya, A., 2016. A survey of consensus clustering, in: *Handbook of Cluster Analysis*. Chapman & Hall/CRC. chapter 22, pp. 497–518. doi:10.1201/b19706-28.
- [21] Gordon, A., Vichi, M., 2001. Fuzzy partition models for fitting a set of partitions. *Psychometrika* 66, 229–247. doi:10.1007/BF02294837.
- [22] Grira, N., Crucianu, M., Boujemaa, N., 2004. Unsupervised and semi-supervised clustering: A brief survey. A review of machine learning techniques for processing multimedia content 1, 9–16. URL: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.59.4074>.
- [23] Hahsler, M., Chelluboina, S., Hornik, K., Buchta, C., 2011. The arules R-package ecosystem: Analyzing interesting patterns from large transaction data sets. *Journal of Machine Learning Research* 12, 2021–2025. URL: <https://dl.acm.org/doi/10.5555/1953048.2021064>.
- [24] Hoi, S.C., Jin, R., Lyu, M.R., 2007. Learning nonparametric kernel matrices from pairwise constraints, in: *Proceedings of the 24th ICML International Conference on Machine Learning*, ACM. pp. 361–368. doi:10.1145/1273496.1273542.
- [25] Hornik, K., 2005. A CLUE for CLUster Ensembles. *Journal of Statistical Software* 14, 1–25. doi:10.18637/jss.v014.i12.
- [26] Hornik, K., Böhm, W., 2008. Hard and soft Euclidean consensus partitions, in: *Data Analysis, Machine Learning and Applications*. Springer, pp. 147–154. doi:10.1007/978-3-540-78246-9_18.

- [27] Iam-On, N., Boongoen, T., Garrett, S., 2008. Refining pairwise similarity matrix for cluster ensemble problem with cluster relations, in: Proceedings of the DS International Conference on Discovery Science, Springer. pp. 222–233. doi:10.1007/978-3-540-88411-8_22.
- [28] Iqbal, A.M., Moh'd, A., Khan, Z., 2012. Semi-supervised clustering ensemble by voting. arXiv preprint arXiv:1208.4138 URL: <https://arxiv.org/abs/1208.4138>.
- [29] Jaccard, P., 1912. The distribution of the flora in the alpine zone. New Phytologist 11, 37–50. doi:10.1111/j.1469-8137.1912.tb05611.x.
- [30] Karypis, G., Han, E.H.S., Kumar, V., 1999. Chameleon: Hierarchical clustering using dynamic modeling. Computer 8, 68–75. doi:10.1109/2.781637.
- [31] Kvålseth, T., 2017. On normalized mutual information: Measure derivations and properties. Entropy 19, 631. doi:10.3390/e19110631.
- [32] LeCun, Y., Cortes, C., Burges, C., 2010. MNIST Handwritten Digit Database. URL: <http://yann.lecun.com/exdb/mnist/>.
- [33] Lu, Z., Peng, Y., 2013. Exhaustive and efficient constraint propagation: A graph-based learning approach and its applications. International Journal of Computer Vision 103, 306–325. doi:10.1007/s11263-012-0602-z.
- [34] Luo, H., Jing, F., Xie, X., 2006. Combining multiple clusterings using information theory based genetic algorithm, in: Proceedings of the IEEE International Conference on Computational Intelligence and Security, IEEE. pp. 84–89. doi:10.1109/ICCIAS.2006.294095.
- [35] Luo, Y., Ji, R., Guan, T., Yu, J., Liu, P., Yang, Y., 2020. Every node counts: Self-ensembling graph convolutional networks for semi-supervised learning. Pattern Recognition 106, 107451. doi:10.1016/j.patcog.2020.107451.
- [36] Luo, Y., Liu, P., Zheng, L., Guan, T., Yu, J., Yang, Y., 2021. Category-level adversarial adaptation for semantic segmentation using purified features. IEEE Transactions on Pattern Analysis and Machine Intelligence doi:10.1109/TPAMI.2021.3064379.
- [37] Luo, Y., Zheng, L., Guan, T., Yu, J., Yang, Y., 2019. Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision and Pattern Recognition, pp. 2507–2516. doi:10.1109/CVPR.2019.00261.
- [38] Van der Maaten, L., Hinton, G., 2008. Visualizing data using t-sne. Journal of Machine Learning Research 9. URL: <https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf>.
- [39] Mahmood, A., Li, T., Yang, Y., Wang, H., Afzal, M., 2015. Semi-supervised evolutionary ensembles for web video categorization. Knowledge-Based Systems 76, 53–66.
- [40] Minaei-Bidgoli, B., Parvin, H., Alinejad-Rokny, H., Alizadeh, H., Punch, W.F., 2014. Effects of resampling method and adaptation on clustering ensemble efficacy. Artificial Intelligence Review 41, 27–48. doi:10.1007/s10462-011-9295-x.
- [41] Mondal, K.C., Pasquier, N., Mukhopadhyay, A., Maulik, U., Bandhopadhyay, S., 2012. A new approach for association rule mining and bi-clustering using formal concept analysis, in: Proceedings of the 8th MLDM International Conference on Machine Learning and Data Mining in Pattern Recognition, Springer. pp. 86–101. doi:10.1007/978-3-642-31537-4_8.
- [42] Nguyen, N., Caruana, R., 2007. Consensus clusterings, in: Proceedings of the 7th IEEE ICDM International Conference on Data Mining, IEEE. pp. 607–612. doi:10.1109/ICDM.2007.73.
- [43] Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L., 1999. Efficient mining of association rules using closed itemset lattices. Information Systems 24, 25–46. doi:10.1016/S0306-4379(99)00003-4.
- [44] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830. URL: <https://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf>.
- [45] Pelleg, D., Baras, D., 2007. K-means with large and noisy constraint sets, in: Proceedings of the ECML European Conference on Machine Learning, Springer. pp. 674–682. doi:10.1007/978-3-540-74958-5_67.
- [46] Qin, Y., Ding, S., Wang, L., Wang, Y., 2019. Research progress on semi-supervised clustering. Cognitive Computation 11, 599–612. doi:10.1007/s12559-019-09664-w.
- [47] Rodriguez, M.Z., Comin, C.H., Casanova, D., Bruno, O.M., Amancio, D.R., Costa, L.d.F., Rodrigues, F.A., 2019. Clustering algorithms: A comparative approach. PLoS ONE 14. doi:10.1371/journal.pone.0210236. e0210236.
- [48] Ruiz, C., Spiliopoulou, M., Menasalvas, E., 2010. Density-based semi-supervised clustering. Data Mining and Knowledge Discovery 21, 345–370. doi:10.1007/s10618-009-0157-y.
- [49] Schütze, H., Manning, C.D., Raghavan, P., 2008. Introduction to Information Retrieval. volume 39. Cambridge University Press Cambridge.
- [50] Smith, L.I., 2002. A Tutorial on Principal Components Analysis. Technical Report. University of Otago, New Zealand. URL: <http://www.cs.otago.ac.nz/research/publications/OUCS-2002-12.pdf>.
- [51] Strehl, A., Ghosh, J., 2002. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. Journal of Machine Learning Research 3, 583–617. URL: <https://www.jmlr.org/papers/volume3/strehl02a/strehl02a.pdf>.
- [52] Tang, W., Xiong, H., Zhong, S., Wu, J., 2007. Enhancing semi-supervised clustering: a feature projection perspective, in: Proceedings of the 13th SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM. pp. 707–716. doi:10.1145/1281192.1281268.
- [53] Topchy, A., Jain, A.K., Punch, W., 2003. Combining multiple weak clusterings, in: Proceedings of the 3rd IEEE ICDM International Conference on Data Mining, IEEE. pp. 331–338. doi:10.1109/ICDM.2003.1250937.
- [54] Topchy, A., Jain, A.K., Punch, W., 2004a. A mixture model for clustering ensembles, in: Proceedings of the SIAM International Conference on Data Mining, SIAM. pp. 379–390. doi:10.1137/1.9781611972740.35.
- [55] Topchy, A., Jain, A.K., Punch, W., 2005. Clustering ensembles: Models of consensus and weak partitions. IEEE Transactions on Pattern Analysis and Machine Intelligence 27, 1866–1881. doi:10.1109/TPAMI.2005.237.
- [56] Topchy, A.P., Law, M.H.C., Jain, A.K., Fred, A.L.N., 2004b. Analysis of consensus partition in cluster ensemble, in: Proceedings of the 4th IEEE ICDM International Conference on Data Mining, IEEE. pp. 225–232. doi:10.1109/ICDM.2004.10100.
- [57] Tran, K.H., Nguyen, M.D., 2016. conclust: Pairwise constraints clustering. URL: <https://CRAN.R-project.org/package=conclust>. R package version 1.1.
- [58] Vega-Pons, S., Ruiz-Shulcloper, J., 2011. A survey of clustering ensemble algorithms. International Journal of Pattern Recognition and Artificial Intelligence 25, 337–372. doi:10.1142/S0218001411008683.
- [59] Wagstaff, K., Cardie, C., 2000. Clustering with instance-level constraints, in: Proceedings of the 7th International Conference on Machine Learning, ACM. pp. 1103–1110. URL: <https://dl.acm.org/doi/10.5555/645529.658275>.
- [60] Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S., 2001. Constrained k-means clustering with background knowledge, in: Proceedings of the 18th ICML International Conference on Machine Learning, pp. 577–584. URL: <https://dl.acm.org/doi/10.5555/645530.655669>.
- [61] Wagstaff, K.L., Basu, S., Davidson, I., 2006. When is constrained clustering beneficial, and why?, in: Proceedings of the 21st Conference on Artificial Intelligence and 18th Innovative Applications of Artificial Intelligence Conference, American Association for Artificial Intelligence. URL: <https://www.aaai.org/Papers/AAAI/2006/AAAI06-384.pdf>.
- [62] Wagstaff, K.L., Cardie, C., 2002. Intelligent clustering with instance-level constraints. Cornell University USA. URL: <https://dl.acm.org/doi/10.5555/935860>.
- [63] Wang, F., 2011. Semisupervised metric learning by maximizing constraint margin. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 41, 931–939. doi:10.1109/TSMCB.2010.2101593.

- [64] Wang, Y., Pan, Y., 2014. Semi-supervised consensus clustering for gene expression data analysis. *BioData Mining* 7, 7. doi:10.1186/1756-0381-7-7.
- [65] Wu, X., Ma, T., Cao, J., Tian, Y., Alabdulkarim, A., 2018. A comparative study of clustering ensemble algorithms. *Computers & Electrical Engineering* 68, 603–615. doi:10.1016/j.compeleceng.2018.05.005.
- [66] Xiao, W., Yang, Y., Wang, H., Li, T., Xing, H., 2016. Semi-supervised hierarchical clustering ensemble and its application. *Neurocomputing* 173, 1362–1376. doi:10.1016/j.neucom.2015.09.009.
- [67] Xing, E.P., Jordan, M.I., Russell, S.J., Ng, A.Y., 2003. Distance metric learning, with application to clustering with side-information, in: *Advances in Neural Information Processing Systems*, pp. 521–528. URL: <https://dl.acm.org/doi/10.5555/2968618.2968683>.
- [68] Xu, D., Tian, Y., 2015. A comprehensive survey of clustering algorithms. *Annals of Data Science* 2, 165–193. doi:10.1007/s40745-015-0040-1.
- [69] Xu, W., Liu, X., Gong, Y., 2003. Document clustering based on non-negative matrix factorization, in: *Proceedings of the 26th ACM SIGIR International Conference on Research and Development in Information Retrieval*, pp. 267–273. doi:10.1145/860435.860485.
- [70] Yahia, S.B., Hamrouni, T., Nguifo, E.M., 2006. Frequent closed itemset based algorithms: A thorough structural and analytical survey. *SIGKDD Explorations Newsletter* 8, 93–104. doi:10.1145/1147234.1147248.
- [71] Yang, Y., Tan, W., Li, T., Ruan, D., 2012. Consensus clustering based on constrained self-organizing map and improved Cop-Kmeans ensemble in intelligent decision support systems. *Knowledge-Based Systems* 32, 101–115. doi:10.1016/j.knosys.2011.08.011.
- [72] Yu, Z., Luo, P., Liu, J., Wong, H.S., You, J., Han, G., Zhang, J., 2018. Semi-supervised ensemble clustering based on selected constraint projection. *IEEE Transactions on Knowledge and Data Engineering* 30, 2394–2407. doi:10.1109/TKDE.2018.2818729.
- [73] Yu, Z., Luo, P., You, J., Wong, H.S., Leung, H., Wu, S., Zhang, J., Han, G., 2016. Incremental semi-supervised clustering ensemble for high dimensional data clustering. *IEEE Transactions on Knowledge and Data Engineering* 28, 701–714. doi:10.1109/TKDE.2015.2499200.
- [74] Yu, Z., Wong, H.S., You, J., Yang, Q., Liao, H., 2011. Knowledge based cluster ensemble for cancer discovery from biomolecular data. *IEEE Transactions on Nanobioscience* 10, 76–85. doi:10.1109/TNB.2011.2144997.
- [75] Zhang, H., Basu, S., Davidson, I., 2019. A framework for deep constrained clustering – Algorithms and advances, in: *Proceedings of the ECML PKDD International Conference on Machine Learning and Knowledge Discovery in Databases*, Springer. pp. 57–72. doi:10.1007/978-3-030-46150-8_4.